One-and-a-half-class Multiple Classifier Systems for Secure Learning against Evasion Attacks at Test Time

Battista Biggio[1], Igino Corona[1], Zhi-Min He[2], Patrick P. K. Chan[2], Giorgio Giacinto[1], Daniel S. Yeung[2], and Fabio Roli[1]

[1] Department of Electrical and Electronic Engineering, University of Cagliari
Piazza d'Armi, 09123 Cagliari, Italy
{battista.biggio,igino.corona,giacinto,roli}@diee.unica.it
[2] School of Computer Science and Engineering, South China University of Technology, China
{zhiminhe,patrickchan,danyeung}@ieee.org

Abstract. Pattern classifiers have been widely used in adversarial settings like spam and malware detection, although they have not been originally designed to cope with intelligent attackers that manipulate data at test time to evade detection. While a number of adversary-aware learning algorithms have been proposed, they are computationally demanding and aim to counter specific kinds of adversarial data manipulation. In this work, we overcome these limitations by proposing a multiple classifier system capable of improving security against evasion attacks at test time by learning a decision function that more tightly encloses the legitimate samples in feature space, without significantly compromising accuracy in the absence of attack. Since we combine a set of one-class and two-class classifiers to this end, we name our approach one-and-a-half-class (1.5C) classification. Our proposal is general and it can be used to improve the security of any classifier against evasion attacks at test time, as shown by the reported experiments on spam and malware detection.

1 Introduction

Pattern recognition systems have been largely employed in security-sensitive settings like biometric identity recognition, intrusion and malware detection, spam filtering, web-page ranking and network protocol verification, to discriminate between legitimate and malicious samples. However, these applications are characterized by the presence of intelligent adversaries who can deliberately attack the classifier by carefully manipulating malicious data at test time to evade detection. From the learning perspective, this means that the underlying class-conditional probability distribution of the malicious data may change from training to test time, *i.e.*, it is subject to an *adversarial* drift [1,2,3,4,5,6,7,8].

Accordingly, pattern classifiers are often characterized by an unsatisfying *trade-off* between accuracy and security against *evasion at test time*, especially in high-dimensional feature spaces. While two-class classifiers may achieve high accuracy in the absence of attack, they can be evaded by malicious samples that are sufficiently different from the training data. This is due to the fact that these classifiers minimize the classification *risk* (or error) over the training data, assuming a stationary distribution. Therefore, it does not make any significant difference in terms of risk if regions of the feature space which are not densely populated by training data are classified as legitimate or malicious [9,10,11]. On the other hand, one-class classifiers (trained on legitimate data) have been exploited in security applications exactly to detect these outlying, anomalous attacks. However, one-class classifiers may exhibit a significantly lower accuracy in the absence of attack (in particular, in high-dimensional feature spaces), as they do not exploit any information on the (available) malicious training data [10].

Motivated by the complementarity of the aforementioned approaches, in this work we propose a Multiple Classifier System (MCS) that combines two-class and one-class classifiers to achieve a better trade-off between accuracy and security against evasion. For this reason, we name it one-and-a-half-class (1.5C) MCS (Sect. 3). Our MCS is able to learn a more *secure* decision function by providing a tighter enclosing of the legitimate data in feature space, while also exploiting information from the available malicious data to retain high accuracy in the absence of attack. Compared to other secure learning techniques [12,13,14], we do not make any specific assumption on the kind of adversarial drift that may occur at test time, but rather only *agnostically* assume that malicious data may appear everywhere in feature space at test time with a non-null probability. This also allow us to reduce the computational complexity during the training phase.

To better motivate our proposal, we provide a simplified analysis of how the classification *risk* changes in the presence of evasion attacks (Sect. 2). Then, we evaluate the security of our approach in a fair, well-principled way, exploiting a recently-proposed framework to design well-crafted evasion attacks against each given classifier, including the proposed MCS, assuming perfect and limited knowledge of the targeted system [6,7,8] (Sect. 4). We finally evaluate the soundness of our approach on two real-world application examples involving spam and malware detection (Sect. 5), and conclude the paper by discussing related work (Sect. 6) and future research directions (Sect. 7).

2 A Simplified Risk Analysis Under Evasion Attacks

In this section, we provide an analysis of the evasion setting under the risk minimization framework [15]. In the classical setting, it is assumed that an underlying probability distribution $p(\boldsymbol{x}, y)$ generates data samples $\boldsymbol{x} \in \mathcal{X}$ along with their class labels $y \in \mathcal{Y}$, and risk minimization amounts to finding an hypothesis $f : \mathcal{X} \mapsto \mathcal{Y}$ that minimizes the expected risk (or loss) $\ell\left(y, f(\boldsymbol{x})\right)$ over p, *i.e.*, $f = \arg\min_{f'} R(f') = \mathbb{E}_{\boldsymbol{x}, y \sim p}\{\ell\left(y, f'(\boldsymbol{x})\right)\}$. For instance, if ℓ is the 0-1 loss, $R(f)$ corresponds to the minimum classification error, being f the optimal hypothesis that would be obtained if p were known.[3]

[3] Typically, the underlying process p is not known, and we are only given a finite set of samples ideally drawn from it. Then, the task of learning amounts to minimizing

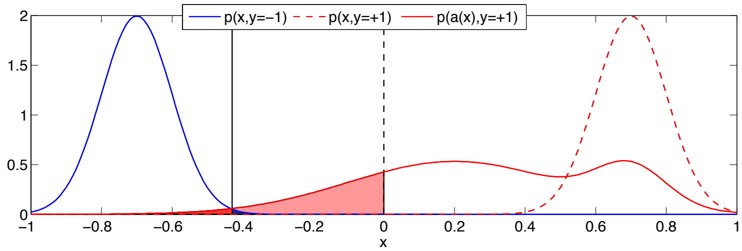

Fig. 1. Classification risk under evasion, in a one-dimensional feature space. The optimal hypothesis f learned on $p(x, y)$ classifies a sample as malicious if $x \geq 0$. Its decision boundary, achieving perfect separation between the two classes, is shown as a dashed black line. Under attack, the malicious distribution changes as $p(a(x), y = +1)$, and this causes an increase of the evasion rate given by the light and dark red areas. The decision boundary of the optimal function h trained after attack is depicted as a solid black line. It trades a much smaller evasion rate (dark red area) for a slightly higher false positive rate (blue area).

In the evasion setting, the malicious data may change at test time, while the legitimate samples can be considered stationary, *i.e.*, not affected by adversarial drift. This behavior can be modeled with a function $a : \mathcal{X} \mapsto \mathcal{X}$ that represents how the attacker modifies the malicious samples drawn from p at test time. The additional risk incurred by f at test time can be thus written as:

$$R_{\mathrm{ts}}(f) - R_{\mathrm{tr}}(f) = \mathbb{E}_{\boldsymbol{x},y}\{\ell(y, f(a(\boldsymbol{x}))) - \ell(y, f(\boldsymbol{x}))\}\,, \tag{1}$$

where $R_{\mathrm{tr}}(f)$ and $R_{\mathrm{ts}}(f)$ respectively represent the risk incurred by f before and after the attack. As the legitimate data is not affected by the function $a(\boldsymbol{x})$, the above difference is not null only for the malicious class. Thus, if ℓ is the 0-1 loss, it corresponds to the increase in the evasion rate at test time (see Fig. 1).

Now, assume that $f^\star = \arg\min_{f' \in \mathcal{F}} R_{\mathrm{ts}}(f')$ is the optimal hypothesis on the test data, including the manipulated attacks. Then, the difference of the risk incurred at test time by using f instead of $f^\star$ is:

$$R_{\mathrm{ts}}(f) - R_{\mathrm{ts}}(f^\star) = \mathbb{E}_{\boldsymbol{x},y}\{\ell(y, f(a(\boldsymbol{x}))) - \ell(y, f^\star(a(\boldsymbol{x})))\}\,. \tag{2}$$

If ℓ is the 0-1 loss, this amounts to the variation of the classification error between f and $f^\star$, computed on the manipulated test data.

As shown in Fig. 1, one may thus decide to trade a slightly higher number of misclassified legitimate samples for a significantly reduced evasion rate (*i.e.*, fraction of misclassified malicious samples after attack), improving security at the expense of slightly worsening accuracy in the absence of attack. To this end, decision boundaries that more tightly enclose the legitimate class should be designed by classifying regions for which $p(\boldsymbol{x}) \approx 0$ as malicious. This should allow for improving security under attack by reducing the feasible attack space

a trade-off between the *empirical* risk computed on such set and a regularization term (or a restricted class of functions) to avoid overfitting [15].

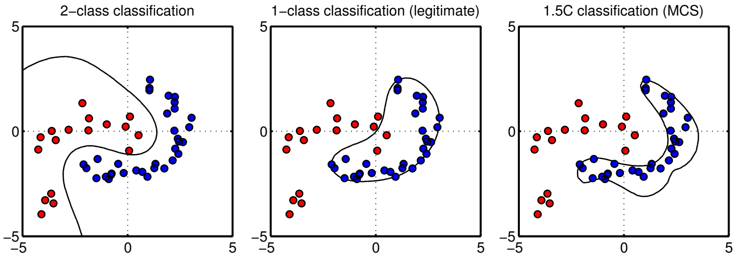

Fig. 2. 1.5C classification on two-dimensional toy data. Legitimate (malicious) training samples are shown as blue (red) points, and decision boundaries as solid black lines. *Left*: Two-class classification yields high accuracy in the absence of attack, but it can be evaded by evasion samples that are sufficiently different from the training data (*e.g.*, in the top-left corner). *Middle*: One-class classification (on the legitimate class) may worsen accuracy in the absence of attack, but improves security by enclosing the legitimate class. *Right*: 1.5C classification retains the advantages of both approaches: security is improved by enclosing the legitimate data, without significantly affecting accuracy in the absence of attack.

(*i.e.*, regions of the feature space classified as legitimate), without significantly increasing the rate of misclassified legitimate samples. Notably, this can be a relevant problem especially in high-dimensional feature spaces, where regions that are classified as legitimate despite $p(\boldsymbol{x}) \approx 0$ may be significantly wider.

3 Secure 1.5C Classification with MCSs

The analysis reported in the previous section suggests that two-class and one-class classifiers can be considered as complementary techniques, usually characterized by different challenges, especially in high-dimensional feature spaces, and a different trade-off between accuracy and security against evasion at test time. Towards *enhancing* this trade-off, we propose an MCS architecture where a two-class classifier is combined with two one-class classifiers, learned respectively on legitimate and malicious data: the two-class classifier should allow for high accuracy in the absence of attack, while the two one-class classifiers should enable the detection of evasion attacks that are (expected to be) significantly different from the training samples of either class. To combine the given classifiers in a *secure* way, *i.e.*, learning a decision function that encloses the legitimate data, we use a further one-class classifier trained on the outputs of the three base classifiers using only legitimate data. The trade-off between accuracy and security exhibited by one-class and two-class classification is exemplified in Fig. 2, along with an example of 1.5C classification overcoming the limitations of both approaches. The architecture of the proposed MCSs is depicted in Fig. 3.

4 Classifier Evasion

In this section, we consider a simplified version of the evasion attack algorithm proposed in [8], described in terms of the attack framework originally defined

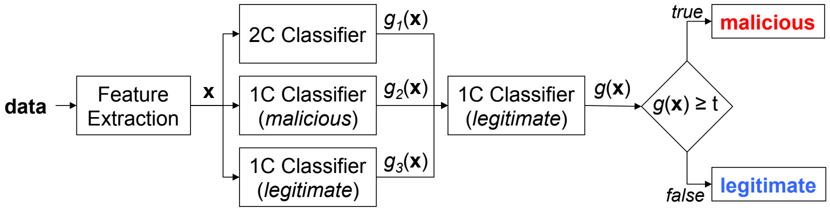

Fig. 3. 1.5C MCS: (i) features are extracted from raw data; (ii) the two-class and one-class classifiers assign scores $g_i(\boldsymbol{x}) \in \mathbb{R}$, $i = 1, 2, 3$, to the feature vector $\boldsymbol{x}$ (assuming that higher scores are given to the malicious class); (iii) these scores are combined by the one-class combiner, providing an aggregated score $g(\boldsymbol{x})$; (iv) $g(\boldsymbol{x})$ is then compared against a decision threshold t to make the final decision.

in [6], *i.e.*, by clearly stating the attacker's goal, knowledge of the system, and capability of manipulating the input data.

Attacker's Goal. In an evasion attack, the attacker's goal is to have malicious samples misclassified as legitimate at test time.

Attacker's Knowledge. The attacker may have different levels of knowledge about (i) the training data, (ii) the feature representation, and (iii) the learning model (including knowledge of the classifier's parameters after training, and feedback on its decisions on input samples). As in previous work [8,6], we consider here *perfect knowledge* (PK) and *limited knowledge* (LK) attacks. In the PK case, the attacker is assumed to know all the system details, including the trained classifier's parameters (*e.g.*, the weights assigned by a linear classifier to each feature). Although this may not be very realistic in practice, performing a security evaluation of the system under this setting allows one to assess the worst performance degradation that may be incurred by the system under attack. In the LK case, the attacker does not have access to the training data, but can collect surrogate data ideally sampled from the same distribution, and use feedback on the classifier's decisions to re-label such samples. The feature representation and the learning model (but not the trained classifier's parameters) are known. Under this setting, the attacker can learn a *surrogate* classifier $\hat{g}(\boldsymbol{x})$ on the re-labeled surrogate data, to approximate the discriminant function $g(\boldsymbol{x})$ of the targeted classifier.

Attacker's Capability. In an evasion attack, the attacker is assumed to be able to modify malicious data at test time, according to application-specific constraints, while she can neither access nor modify the classifier's training data.

Attack Strategy. Similarly to [8], and according to the previously-described attack scenario, an optimal evasion strategy can be thus formulated as:

$$\begin{aligned} \boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \quad & \hat{g}(\boldsymbol{x})\,, \\ \text{s.t.} \quad & d(\boldsymbol{x}, \boldsymbol{x}_a) \leq d_{\max}\,, \end{aligned} \tag{3}$$

where the attacker's goal is to find a set of feasible manipulations to the initial malicious sample $\boldsymbol{x}_a$ to obtain a camouflaged sample $\boldsymbol{x}^*$ that minimizes the

Algorithm 1 Evasion Attack

Input: $\boldsymbol{x}_a$: the malicious sample; $\boldsymbol{x}^{(0)}$: the initial location of the attack sample; t: the gradient step size; ϵ: a small positive constant.

Output: $\boldsymbol{x}'$: the evasion attack sample.

```
$i \leftarrow 0$
repeat
    $i \leftarrow i + 1$
    $\boldsymbol{x}^{(i)} \leftarrow \boldsymbol{x}^{(i-1)} - t\nabla g(\boldsymbol{x}^{(i-1)})$
    if $d(\boldsymbol{x}^{(i)}, \boldsymbol{x}_a) > d_{\max}$ or other constraints are violated then
        Project $\boldsymbol{x}^{(i)}$ onto the feasible domain
    end if
until $|g(\boldsymbol{x}^{(i)}) - g(\boldsymbol{x}^{(i-1)})| < \epsilon$
return $\boldsymbol{x}' = \boldsymbol{x}^{(i)}$
```

(surrogate) classifier's discriminant function $\hat{g}(\boldsymbol{x})$. Note that, without loss of generality, we are assuming here that the classification function is $f(\boldsymbol{x}) = (g(\boldsymbol{x}))_+$, where $(a)_+ = +1$ if $a \geq 0$, and -1 otherwise, and malicious samples should be assigned higher (positive) values of $g(\boldsymbol{x})$ to be correctly classified.

The feasible domain for $\boldsymbol{x}$ is defined in terms of constraints on the manipulation of the feature values of the malicious samples. For some classes of features, a simple distance metric can be adopted. For instance, if one considers the presence or absence of a given word in an email as a feature, then the ℓ_1-norm between two feature vectors amounts to counting how many words are different between the two emails. Therefore, as also done in previous work [1,2,6,8,5,16], we express this constraint as $d(\boldsymbol{x}, \boldsymbol{x}_a) \leq d_{\max}$, highlighting that only a maximum number of modifications $d_{\max}$ to each sample are allowed. Varying the parameter $d_{\max}$ is fundamental to properly understand system security under attack. In fact, as we will show in our experiments, more secure classifiers will exhibit a lower decrease in the detection rate of malicious samples as $d_{\max}$ increases (*i.e.*, as a larger amount of manipulations are performed to the malicious samples). Additional constraints to Problem (3) may be also considered, depending on the specific feature representation; *e.g.*, if features are real-valued and normalized in $[0, 1]^d$, one may consider an additional box constraint given as $\boldsymbol{x} \in [0, 1]^d$.

The solution of Problem (3) clearly depends on the kind of discriminant function and on the given set of constraints. In [8], a general solution for *nonlinear*, differentiable discriminant functions was proposed, based on a simple gradient-descent algorithm. It is worth remarking that an additional component to the objective function was also considered in that work, to drive the attack point during the descent towards regions of the feature space more densely populated by legitimate samples, with the goal of increasing the probability of evading detection (especially when attacking the surrogate classifier). In this work, we disregard this component, as it is possible to substantially obtain the same effect by running the evasion attack algorithm (given as Algorithm 1) from distinct initialization points, eventually retaining the attack sample that achieves the minimum value of the objective.

4.1 Gradient Computation

The gradients required to evaluate classifier security using Algorithm 1 are given below, for the classifiers considered in this work: linear and nonlinear Support Vector Machines (SVMs), and the proposed 1.5C MCS. Note that this approach can be easily applied to any classifier, provided that its discriminant function $g(\boldsymbol{x})$ is at least subdifferentiable; otherwise, one may use a different optimization strategy to solve Problem (3), or approximate the non-differentiable $g(\boldsymbol{x})$ with a *surrogate* classifier having a differentiable discriminant function [8].

Linear classifiers. For linear discriminant functions $g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$, with feature weights $\mathbf{w} \in \mathbb{R}^d$ and bias $b \in \mathbb{R}$, the gradient is simply $\nabla g(\mathbf{x}) = \mathbf{w}$.

Nonlinear SVMs. For kernelized (and one-class) SVMs, the discriminant function is $g(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + b$, where $\boldsymbol{\alpha}$ and b are the SVM parameters learned during training, $k(\mathbf{x}, \mathbf{x}_i)$ is the kernel function, and $\{\boldsymbol{x}_i, y_i\}_{i=1}^{n}$ are the training samples and their labels [15]. The gradient is $\nabla g(\mathbf{x}) = \sum_i \alpha_i y_i \nabla k(\mathbf{x}, \mathbf{x}_i)$ and, thus, the feasibility of our approach depends on the kernel gradient $\nabla k(\mathbf{x}, \mathbf{x}_i)$, which is computable for many numeric kernels; *e.g.*, for the RBF kernel $k(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}_i\|^2\right)$, it is $\nabla k(\mathbf{x}, \mathbf{x}_i) = -2\gamma \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}_i\|^2\right)(\mathbf{x} - \mathbf{x}_i)$.

1.5C MCS. In this case, the gradient will depend on the particular choice of each of the component classifiers g_1, g_2, g_3 and on the one-class combiner $g(\boldsymbol{x})$. To provide an example, we assume here that a one-class SVM with the RBF kernel is used to combine g_1, g_2, g_3. In this case, the discriminant function can be written as $g(\boldsymbol{x}) = \sum_i \alpha_i \exp\left(-\gamma \|\boldsymbol{z}(\boldsymbol{x}) - \boldsymbol{z}(\boldsymbol{x}_i)\|^2\right) + b$, where $\boldsymbol{z}(\boldsymbol{x}) = [g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), g_3(\boldsymbol{x})]^\top$. The corresponding gradient is thus given as:

$$\nabla g(\boldsymbol{x}) = -2\gamma \sum_i \alpha_i e^{-\gamma \|\boldsymbol{z}(\boldsymbol{x}) - \boldsymbol{z}(\boldsymbol{x}_i)\|^2} (\boldsymbol{z}(\boldsymbol{x}) - \boldsymbol{z}(\boldsymbol{x}_i))^\top \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}, \tag{4}$$

where $\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} = [\nabla g_1(\boldsymbol{x}), \nabla g_2(\boldsymbol{x}), \nabla g_3(\boldsymbol{x})]^\top \in \mathbb{R}^{3 \times d}$, and d is the feature set size.

Descent in discrete spaces. In discrete spaces, it is not possible to follow the gradient-descent direction precisely, as it may map the given sample to a set of non-admissible feature values. The gradient-descent direction can be nevertheless used as a search heuristic, as follows. Starting from the current sample $\boldsymbol{x}$, one may generate a set of candidate neighboring points by perturbing only those features of the current sample which correspond to the maximum absolute values of the gradient, one at a time, in the correct direction. Eventually, one should update the current sample to the neighbor that attained the minimum value of the objective function, and iterate until convergence.

5 Experiments

In this section, we report an experimental evaluation of our proposal on two distinct security applications, *i.e.*, spam filtering and PDF malware detection. Our goal is to investigate whether, and to what extent, the proposed 1.5C MCS is able to combine the advantages of two-class and one-class classification to yield a better trade-off between accuracy and security. To this end, we adopt a

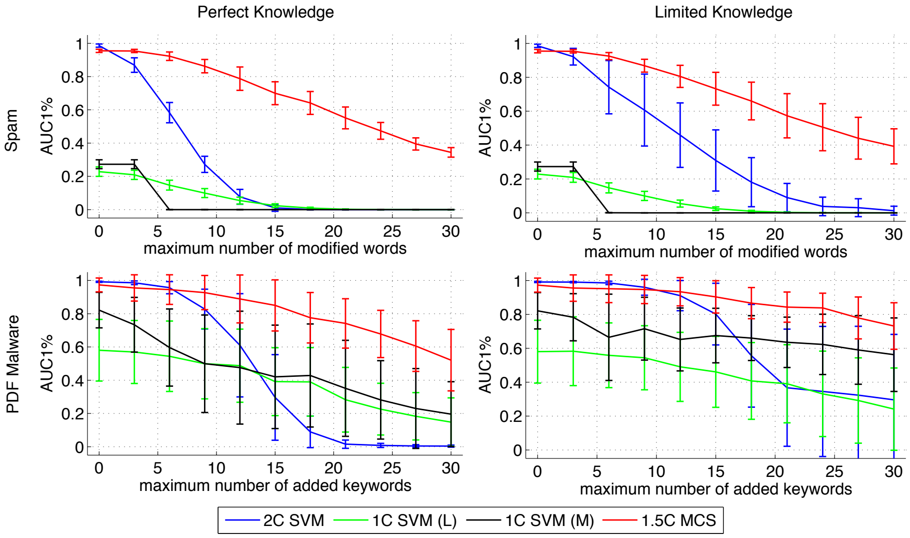

Fig. 4. Average $\mathrm{AUC}_{1\%}$ values (with standard deviation) for spam filtering (top row) and PDF malware detection (bottom row), attained by two-class (2C) SVMs, one-class (1C) SVMs trained on legitimate (L) and malicious (M) data, and the 1.5C MCS, against PK (first column) and LK (second column) attacks. In each plot, the $\mathrm{AUC}_{1\%}$ under attack is evaluated against an increasing maximum number of modifications to the malicious data, *i.e.*, number of modified words in each spam, and number of added keywords to each malicious PDFs.

common experimental setup, involving the following classifiers: (*i*) a two-class SVM, with the linear kernel for the spam data, and the RBF kernel for the PDF data; (*ii*-*iii*) two one-class SVMs with the RBF kernel, trained respectively on legitimate and malicious samples; and (*iv*) a one-class SVM with the RBF kernel to combine the former three classifiers in our 1.5C MCS (see Sect. 3).

We simulate attackers that have perfect (PK) and limited (LK) knowledge of the targeted system (see Sect. 4). In the LK case, we assume that the attacker collects a surrogate training set whose size is only 20% of the size of the data used to learn the targeted classifier. The reason is that, in practice, attackers may only perform a limited number of queries to the targeted classifier to recover the classification labels of such samples, and train the surrogate classifier.

We evaluate classifier performance by averaging the true positive (TP) rate, *i.e.*, the fraction of correctly-classified malicious samples, achieved when the false positive (FP) rate, *i.e.*, fraction of misclassified legitimate samples, is below 1%, which corresponds to the so-called Area Under the ROC Curve (AUC) at 1% FP rate. The reason is that, in these applications, false positives are more harmful than false negatives, and thus, they should be kept very low [18,6]. Our results are averaged over five repetitions, in which different training-test splits are considered. In particular, in each repetition, 50% of the data is randomly chosen for training (TR), while the rest is used for testing (TS).

5.1 Spam Filtering

Spam filtering is one of the most popular application examples considered in adversarial learning [1,2,17,18,6]. As in previous work, we consider the widely-used bag-of-words feature representation, in which an email is described as a set of binary features, indicating the presence (1) or absence (0) of a given word. Due to the high dimensionality of the corresponding feature space, linear classifiers have been often considered [17,19,18] and implemented in real anti-spam filters, including *e.g.* SpamAssassin and SpamBayes.[4] Therefore, in this investigation we implement the two-class classifier as a linear SVM. Popular attacks in spam filtering include modifications to the email's content. Attackers may obfuscate *bad* words (which typically appear in spam but not in legitimate emails) through misspelling (*e.g.*, changing "cheap" as "che4p"), and add *good* words (which typically appear in legitimate emails but not in spam). The corresponding effect is to change features from 1 to 0, or viceversa.

Experimental Setup. We use the benchmark TREC 2007 email corpus [20], which consists of 25,220 ham and 50,199 spam emails. The first 5,000 emails are adopted in this experiment. Each email is represented by a feature vector using a tokenization method of SpamAssassin. We use a feature selection method based on information gain [21], reducing the number of features from more than 20,000 to 500, to reduce computational complexity. All the malicious samples in TS are manipulated according to PK and LK evasion attacks. In the latter case, only 500 samples have been used as the surrogate training data available to the attacker. To evaluate classifier security against an increasing maximum number of modifiable words in each spam, we set $d(\boldsymbol{x}, \boldsymbol{x}_a) = \|\boldsymbol{x} - \boldsymbol{x}_a\|_1 \leq d_{\max}$ as the constraint in Problem (3). In this case, $d_{\max}$ is exactly the maximum number of modifiable words (*i.e.*, feature values) in each spam. The parameters $C, \gamma \in \{2^{-10}, 2^{-9}, \ldots, 2^{10}\}$ of each SVM are optimized through a 5-fold cross-validation on TR to minimize the classification error.

Results. The average $\text{AUC}_{1\%}$ values, along with their standard deviations, are shown in Fig. 4 (top row), against an increasing maximum number of modified words in each spam. In the absence of attack, the accuracy of the 1.5C MCS and the two-class (2C) SVM is similar, and very high, while for one-class (1C) SVMs it is very low. The reason is that we are not only working at extremely low FP rates, but also in high-dimensional spaces; thus, detecting a high number of attacks for these techniques becomes extremely challenging. Under attack, *i.e.*, where the number of modified words is greater than zero, the $\text{AUC}_{1\%}$ of the 2C SVMs drops significantly, in both the PK and the LK settings. Conversely, the performance of our 1.5C MCS decreases more gracefully, retaining a higher detection rate against evasion attempts, and thus, a higher security.

5.2 PDF Malware Detection

Another well-known application investigated in adversarial learning is PDF malware detection. PDF files are characterized by a hierarchy of interconnected ob-

[4] `http://spamassassin.apache.org/`, `http://spambayes.sourceforge.net/`

jects including keywords and data streams; *e.g.*, the keyword "PageLayout" characterizes an object describing the format of the corresponding page. Their flexible structure allows for embedding different kinds of content, including JavaScript and binary programs. This makes PDF files suitable vectors to spread malware infections, as malware can be easily embedded into them. In this experiment, we use the feature representation adopted in [22,8], in which each feature represents the number of occurrences of a given keyword in a PDF file. In this case, evasion attacks consist of manipulating the set of keywords present in the PDF file at test time. However, since embedded objects can be easily added to, but not removed from PDF files without corrupting their structure, as in [22,8] we assume that the number of occurrences of each keyword can be only increased. This can be implemented by adding the constraint $\boldsymbol{x}_a \leq \boldsymbol{x}$ (where inequality has to be fulfilled by each feature value) to Problem (3).

Experimental Setup. We use 2,000 samples from the PDF malware dataset in [22,8], represented by 114 distinct keywords (*i.e.*, features). Each feature value (*i.e.*, the number of occurrences of the corresponding keyword) is normalized in $[0, 1]$ by dividing its value by 100 (which is also set as the maximum value of the occurrences of each keyword). The SVMs' parameters are set as in the previous experiments on spam filtering. Classifier security is evaluated against an increasing maximum number of added keywords. To this end, we set $d(\boldsymbol{x}, \boldsymbol{x}_a) = 100\|\boldsymbol{x} - \boldsymbol{x}_a\|_1 \leq d_{\max}$ as the constraint in Problem (3), similarly to the experiment on spam filtering. Together with the constraint on feature addition $\boldsymbol{x}_a \leq \boldsymbol{x}$, this let $d_{\max}$ correspond to the maximum number of keywords that can be added to each malicious PDF file.

Results. As for the spam filtering case, results are reported in terms of $\text{AUC}_{1\%}$ values in Fig. 4 (bottom row). Similar conclusions can be drawn with respect to the results obtained for the spam filtering case, except that here, 1C SVMs perform slightly better in the absence of attack, and exhibit higher detection rates under attack than those exhibited by the 2C SVMs (for both PK and LK attacks). This means that, for a sufficiently high number of added keywords, 1C SVMs are more secure than 2C SVMs. Nevertheless, the proposed 1.5C MCS significantly outperforms the competing approaches under attack, while only performing slightly worse than the 2C SVM in the absence of attack. Worth noting, the detection rate of the 1.5C MCS decreases more gracefully in this case with the increase of the maximum number of added keywords, since the feature values of malicious samples can only be incremented.

6 Related Work

While many methods have been proposed to counter evasion attacks [1,12,14,25], they are computationally demanding and make specific assumptions on how attackers should manipulate data at test time to evade detection (see also [3,4,5,6] and references therein, for a more detailed review). Inspired by the seminal work in [1], recent work has shown that game theory can be used to model interactions between the classification system and the attacker in non-zero-sum games, yielding more secure classifiers [14]. However, satisfying conditions for the uniqueness of the equilibrium not only require the classifier's and the attacker's objectives to fulfill specific conditions, but also computationally demanding algorithms to find suitable solutions. A slightly different line of work [12,13,25] aims to learn secure classifiers (and SVMs, in particular), by minimizing a modified version of the loss function which accounts for worst-case manipulations to data at test time, including feature deletion, insertion, and rescaling. This also results in a computationally intensive training phase.

Other work has proposed countermeasures to evasion based on well-motivated heuristics, *e.g.*, that feature weights assigned by a linear classifier should be more evenly spread among the features, to require modifying more feature values to evade detection [23,18]. Linear classifiers exhibiting the aforementioned behavior have been implemented efficiently using MCSs, by averaging an ensemble of linear classifiers [23,26,27,18]. In the present work, we have shown that MCSs can also be used to learn secure, nonlinear classifiers, in an efficient manner.

7 Conclusions and Future Work

While pattern classifiers have been widely used in adversarial applications, they are often characterized by an unsatisfying trade-off between accuracy and security against evasion, especially in high-dimensional spaces: two-class classifiers may yield high accuracy but they are potentially insecure, while one-class classification may improve security at the expense of a lower accuracy in the absence of attack. Motivated by the complementarity of these approaches, we proposed a 1.5C MCS that achieves a better *trade-off* between accuracy and security. Our proposal is *general*, it is *agnostic* to the kind of adversarial drift that may occur at test time, and can be used to improve the security of *any classifier* against evasion attacks, as shown by the reported experiments on spam and malware detection. Interesting future research directions include providing a formal, well-principled characterization of the trade-off between accuracy and security, as well as its relationship with the dimensionality of the feature space, and investigating whether the proposed 1.5C MCS can also be capable to deal with the presence of *poisoning* attacks in the training data [3,4,5,6,7].

Acknowledgments. This work has been partly supported by the projects CRP-18293 and CRP-59872, both funded by Regione Autonoma della Sardegna, L.R. 7/2007, respectively with Bando 2009 and Bando 2012.

References

1. Dalvi, N., Domingos, P., Mausam, Sanghai, S., Verma, D.: Adversarial classification. In: 10th Int'l Conf. on Knowl. Disc. and Data Mining, ACM (2004) 99–108
2. Lowd, D., Meek, C.: Adversarial learning. In: 11th Int'l Conf. on Knowl. Disc. and Data Mining, ACM (2005) 641–647
3. Barreno, M., Nelson, B., Sears, R., Joseph, A.D., Tygar, J.D.: Can machine learning be secure? In: ASIA CCS, ACM (2006) 16–25
4. Barreno, M., Nelson, B., Joseph, A., Tygar, J.: The security of machine learning. Mach. Learning **81** (2010) 121–148

5. Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B., Tygar, J.D.: Adversarial machine learning. In: 4th Workshop Artificial Intell. and Sec., ACM (2011) 43–57
6. Biggio, B., Fumera, G., Roli, F.: Sec. evaluation of pattern classifiers under attack. IEEE T. on Knowl. and Data Eng. **26**(4) (2014) 984–996
7. Biggio, B., Fumera, G., Roli, F.: Patt. recognition systems under attack: Design issues and research challenges. Int'l J. Patt. Recogn. Artif. Intell. **28**(7) (2014)
8. Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., Roli, F.: Evasion attacks against machine learning at test time. In Blockeel, H., et al., eds.: ECML PKDD, Part III. Vol. 8190 of LNCS, Springer (2013) 387–402
9. Gori, M., Scarselli, F.: Are multilayer perceptrons adequate for pattern recognition and verification? IEEE TPAMI **20**(11) (1998) 1121–1132
10. Tax, D.M.J.: One-class classification. PhD Thesis (2001)
11. Biggio, B., Fumera, G., Roli, F.: Design of robust classifiers for adversarial environments. In: IEEE Int'l Conf. on Systems, Man, and Cybernetics (2011) 977–982
12. Globerson, A., Roweis, S.: Nightmare at test time: robust learning by feature deletion. In: 23rd ICML. Vol. 148., ACM (2006) 353–360
13. Teo, C.H., Globerson, A., Roweis, S., Smola, A.: Convex learning with invariances. In Platt, J., et al., eds.: NIPS 20. MIT Press (2008) 1489–1496
14. Brückner, M., Kanzow, C., Scheffer, T.: Static prediction games for adversarial learning problems. J. Mach. Learn. Res. **13** (2012) 2617–2654
15. Vapnik, V.N.: The nature of statistical learning theory. Springer-Verlag (1995)
16. Nelson, B., Rubinstein, B.I., Huang, L., Joseph, A.D., Lee, S.J., Rao, S., Tygar, J.D.: Query strategies for evading convex-inducing classifiers. J. Mach. Learn. Res. **13** (2012) 1293–1332
17. Nelson, B., Barreno, M., Chi, F.J., Joseph, A.D., Rubinstein, B.I.P., Saini, U., Sutton, C., Tygar, J.D., Xia, K.: Exploiting machine learning to subvert your spam filter. In: Large-Scale Exploits and Emergent Threats, USENIX (2008) 1–9
18. Biggio, B., Fumera, G., Roli, F.: Multiple classifier systems for robust classifier design in adversarial environments. Int'l J. Mach. Learn. Cyb. **1**(1) (2010) 27–41
19. Jorgensen, Z., Zhou, Y., Inge, M.: A multiple instance learning strategy for combating good word attacks on spam filters. J. Mach. Learn. Res. **9** (2008) 1115–1146
20. Cormack, G.V.: Trec 2007 spam track overview. In Voorhees, E.M., Buckland, L.P., eds.: TREC. Volume Special Publication 500-274., NIST (2007)
21. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. **34** (2002) 1–47
22. Maiorca, D., Corona, I., Giacinto, G.: Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious PDF files detection. In: 8th ASIA CCS, ACM (2013) 119–130
23. Kolcz, A., Teo, C.H.: Feature weighting for improved classifier robustness. In: 6th Conf. on Email and Anti-Spam (2009)
24. Sutton, C., Sindelar, M., McCallum, A.: Feature bagging: Preventing weight undertraining in structured discriminative learning. Tech. Report, IR-402, University of Massachusetts (2005)
25. Zhou, Y., Kantarcioglu, M., Thuraisingham, B., Xi, B.: Adversarial support vector machine learning. In: 18th Int'l Conf. on Knowl. Disc. and Data Mining. ACM (2012) 1059–1067
26. Biggio, B., Fumera, G., Roli, F.: Multiple classifier systems for adversarial classification tasks. In Benediktsson, J.A., Kittler, J., Roli, F., eds.: 8th MCS. Vol. 5519 of LNCS, Springer (2009) 132–141
27. Biggio, B., Fumera, G., Roli, F.: Multiple classifier systems under attack. In Gayar, N.E., Kittler, J., Roli, F., eds.: 9th MCS. Vol. 5997 of LNCS, Springer (2010) 74–83