Randomized Prediction Games for Adversarial Machine Learning

Samuel Rota Bulò, *Member, IEEE*, Battista Biggio, *Member, IEEE*, Ignazio Pillai, Marcello Pelillo, *Fellow, IEEE*, and Fabio Roli, *Fellow, IEEE*

***Abstract*— In spam and malware detection, attackers exploit randomization to obfuscate malicious data and increase their chances of evading detection at test time, e.g., malware code is typically obfuscated using random strings or byte sequences to hide known exploits. Interestingly, randomization has also been proposed to improve security of learning algorithms against evasion attacks, as it results in hiding information about the classifier to the attacker. Recent work has proposed game-theoretical formulations to learn secure classifiers, by simulating different evasion attacks and modifying the classification function accordingly. However, both the classification function and the simulated data manipulations have been modeled in a deterministic manner, without accounting for any form of randomization. In this paper, we overcome this limitation by proposing a randomized prediction game, namely, a noncooperative game-theoretic formulation in which the classifier and the attacker make randomized strategy selections according to some probability distribution defined over the respective strategy set. We show that our approach allows one to improve the tradeoff between attack detection and false alarms with respect to the state-of-the-art secure classifiers, even against attacks that are different from those hypothesized during design, on application examples including handwritten digit recognition, spam, and malware detection.**

***Index Terms*— Adversarial learning, computer security, evasion attacks, game theory, pattern classification, randomization.**

I. Introduction

MACHINE-LEARNING algorithms have been increasingly adopted in adversarial settings, such as spam, malware, and intrusion detection. However, such algorithms are not designed to operate against intelligent and adaptive attackers, thus making them inherently vulnerable to carefully crafted attacks. Evaluating security of machine learning against such attacks and devising suitable countermeasures are two among the main open issues under investigation in the field of adversarial machine learning [1]–[11]. In this paper, we focus on the issue of designing secure classification algorithms against evasion attacks, i.e., attacks in which malicious samples are manipulated at test time to evade detection. This is a typical setting, e.g., in spam filtering, where spammers manipulate the content of spam emails to get them past the antispam filters [1], [2], [12]–[14], or in malware detection, where hackers obfuscate malicious software (malware, for short) to evade detection of either known or zero-day exploits [8], [9], [15], [16]. Although out of the scope of this paper, it is worth mentioning here another pertinent attack scenario, referred to as classifier poisoning. Under this setting, the attacker can manipulate the training data to mislead classifier learning and cause a denial of service, e.g., by increasing the number of misclassified samples [6], [7], [17]–[20].

To date, several authors have addressed the problem of designing secure learning algorithms to mitigate the impact of evasion attacks [1], [6], [10], [11], [21]–[27] (see Section VII for further details). The underlying rationale of such approaches is to learn a classification function that accounts for potential malicious data manipulations at test time. To this end, the interactions between the classifier and the attacker are modeled as a game in which the attacker manipulates data to evade detection, while the classification function is modified to classify them correctly. This essentially amounts to incorporating knowledge of the attack strategy into the learning algorithm. However, both the classification function and the simulated data manipulations have been modeled in a deterministic manner, without accounting for any form of randomization.

Randomization is often used by attackers to increase their chances of evading detection, e.g., malware code is typically obfuscated using random strings or byte sequences to hide known exploits, and spam often contains bogus text randomly taken from English dictionaries to reduce the spamminess of the overall message. Surprisingly, randomization has also been proposed to improve classifier security against evasion attacks [3], [6], [28]. It has been shown that randomizing the learning algorithm may effectively hide information about the classification function to the attacker, requiring her to select a less effective attack (manipulation) strategy. In practice, the fact that the adversary may not know the classification function exactly (i.e., in a deterministic sense) decreases her (expected) payoff on each attack sample; i.e., to achieve the same expected evasion rate attained in the deterministic case, the attacker has to increase the number of modifications made to the attack samples [28].

Motivated by the aforementioned facts, in this paper, we generalize static prediction games, i.e., the game-theoretical formulation proposed in [10] and [11], to account for randomized classifiers and data manipulation strategies. For this

Manuscript received April 9, 2016; accepted July 14, 2016.

S. Rota Bulò is with ICT-Tev, Fondazione Bruno Kessler, Trento 38123, Italy (e-mail: rotabulo@fbk.eu).

B. Biggio, I. Pillai, and F. Roli are with the Department of Electrical and Electronic Engineering, University of Cagliari, Cagliari 09123, Italy (e-mail: battista.biggio@diee.unica.it; pillai@diee.unica.it; roli@diee.unica.it).

M. Pelillo is with the Dipartimento di Scienze Ambientali, Informatica e Statistica, Ca' Foscari University of Venice, Venice 30123, Italy (e-mail: pelillo@dsi.unive.it).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2593488

reason, we refer to our game as a randomized prediction game. A randomize prediction game is a noncooperative game between a randomized learner and a randomized attacker (also called data generator), where the player's strategies are replaced with probability distributions defined over the respective strategy sets. Our goal is twofold. We do not only aim to assess whether randomization helps achieving a better tradeoff in terms of false alarms and attack detection (with respect to the state-of-the-art secure classifiers), but also whether our approach remains more secure against attacks that are different from those hypothesized during design. In fact, given that our game considers randomized players, it is reasonable to expect that it may be more robust to potential deviations from its original hypotheses about the players' strategies.

This paper is structured as follows. Randomized prediction games are presented in Section II, where sufficient conditions for the existence and uniqueness of a Nash equilibrium in these games are also given. In Section III, we focus on a specific game instance involving a linear support vector machine (SVM) learner, for which we provide an effective method to find an equilibrium by overcoming some computational problems. We discuss how to enable the use of nonlinear (kernelized) SVMs in Section IV. In Section V, we report a simple example to explain how the proposed methods enforce security in adversarial settings. Related work is discussed in Section VII. In Section VI, we empirically validate the soundness of the proposed approach on an handwritten digit recognition task, and on adversarial application examples involving spam filtering and malware detection in PDF files. Notably, to evaluate robustness of our approach and the state-of-the-art secure classification algorithms, we also consider attacks that deviate from the models hypothesized during classifier design. Finally, in Section VIII, we summarize our contributions and sketch potential directions for future work.

II. Randomized Prediction Games

Consider an adversarial learning setting involving two actors: a data generator and a learner.[1] The data generator produces at training time a set $\hat{\mathcal{D}} = \{\hat{\boldsymbol{x}}_i, y_i\}_{i=1}^{\mathsf{n}} \subseteq \mathcal{X} \times \mathcal{Y}$ of n training samples, sampled from an unknown probability distribution. Sets $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output spaces of the learning task, respectively. At test time, the data generator modifies the samples in $\hat{\mathcal{D}}$ to form a new data set $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{Y}$, reflecting a test distribution, which differs in general from the training distribution and it is not available at training time. We assume binary learners, i.e., $\mathcal{Y} = \{-1, +1\}$, and we also assume that the data transformation process leaves the labels of the samples in $\hat{\mathcal{D}}$ unchanged, i.e., $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{\mathsf{n}}$. Hence, a perturbed data set will simply be represented in terms of a tuple $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\mathsf{n}}) \in \mathcal{X}^{\mathsf{n}}$, each element being the perturbation of the original input sample $\hat{\boldsymbol{x}}_i$, while we implicitly assume the label to remain y_i. The role of the learner is to classify samples $\boldsymbol{x} \in \mathcal{X}$ according to the prediction function $h(\boldsymbol{x}) = \text{sign}[f(\boldsymbol{x}; \boldsymbol{w})]$, which is expressed in terms of a linear generalized decision function $f(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^\top \phi(\boldsymbol{x})$, where $\boldsymbol{w} \in \mathbb{R}^{\mathsf{m}}$, $\boldsymbol{x} \in \mathcal{X}$, and $\phi : \mathcal{X} \to \mathbb{R}^{\mathsf{m}}$ is a feature map.

[1] We adopt here the same terminology used in [11].

Static prediction games have been introduced in [11] by modeling the learner and the data generator as the players of a noncooperative game that we identify as l-player and d-player, respectively. The strategies of l-player correspond to the parameterizations $\boldsymbol{w}$ of the prediction function f. The strategies of the data generator, instead, are assumed to live directly in the feature space, by regarding $\dot{\mathbf{X}} = (\dot{\boldsymbol{x}}_1^\top, \ldots, \dot{\boldsymbol{x}}_{\mathsf{n}}^\top)^\top \in \mathbb{R}^{\mathsf{mn}}$ as a data generator strategy, where $\dot{\boldsymbol{x}}_i = \phi(\boldsymbol{x}_i)$. By doing so, the decision function f becomes linear in either players' strategies. Each player is characterized also by a cost function that depends on the strategies played by either players. The cost function of d-player and l-player is denoted by c_d and c_l, respectively, and is given by

$$c_l(\boldsymbol{w}, \dot{\mathbf{X}}) = \rho_l \Omega_l(\boldsymbol{w}) + \sum_{i=1}^{\mathsf{n}} \ell_l(\boldsymbol{w}^\top \dot{\boldsymbol{x}}_i, y_i) \tag{1}$$

$$c_d(\boldsymbol{w}, \dot{\mathbf{X}}) = \rho_d \Omega_d(\dot{\mathbf{X}}) + \sum_{i=1}^{\mathsf{n}} \ell_d(\boldsymbol{w}^\top \dot{\boldsymbol{x}}_i, y_i) \tag{2}$$

where $\boldsymbol{w} \in \mathbb{R}^{\mathsf{m}}$ is the strategy of l-player, $\dot{\mathbf{X}} \in \mathbb{R}^{\mathsf{mn}}$ is the strategy of the d-player, and y_i denotes the label of $\dot{\boldsymbol{x}}_i$ as per $\hat{\mathcal{D}}$. Moreover, $\rho_{d/l} > 0$ is a tradeoff parameter, $\ell_{d/l}(\boldsymbol{w}^\top \dot{\boldsymbol{x}}_i, y)$ measures the loss incurred by the l/d-player when the decision function yields $\boldsymbol{w}^\top \dot{\boldsymbol{x}}_i$ for the ith training sample while the true label is y, and $\Omega_{d/l}$ can be regarded as a penalization for playing a specific strategy. For the d-player, this term quantifies the cost of perturbing $\hat{\mathcal{D}}$ in feature space.

The goal of this paper is to introduce a randomization component in the model of [11], particularly to what concerns the players' behavior. To this end, we take one abstraction step with respect to the aforementioned prediction game, where we let the learner and the data generator sample their playing strategy in the prediction game from a parameterized distribution, under the assumption that they are expected cost-minimizing (also known as expected utility maximizing). By doing so, we introduce a new noncooperative game that we call randomized prediction game between the l-player and the d-player with strategies being mapped to the possible parameterizations of the players' respective distributions, and cost functions being expected costs under the same distributions.

A. Definition of Randomized Prediction Game

Consider a prediction game as described earlier. We inject randomness in the game by assigning each player a parameterized probability distribution, i.e., $p_l(\boldsymbol{w}; \boldsymbol{\theta}_l)$ for the learner and $p_d(\dot{\mathbf{X}}; \boldsymbol{\theta}_d)$ for the data generator, that governs the players' strategy selection. Players are allowed to select the parameterization $\boldsymbol{\theta}_l$ and $\boldsymbol{\theta}_d$ for the respective distributions. For any choice of $\boldsymbol{\theta}_l$, the l-player plays a strategy $\boldsymbol{w}$ sampled from $p_l(\cdot; \boldsymbol{\theta}_l)$. Similarly, for any choice of $\boldsymbol{\theta}_d$, the d-player plays a strategy $\dot{\mathbf{X}}$ sampled from $p_d(\cdot; \boldsymbol{\theta}_d)$. If the players adhere to such rules, we obtain a randomized prediction game.

A randomized prediction game is a noncooperative game between a learner (l-player) and data generator (d-player) that has the following components.

1) An underlying prediction game with cost functions $c_{l/d}(\boldsymbol{w}, \dot{\mathbf{X}})$ as defined in (1) and (2).
2) two parametrized probability distributions $p_{l/d}(\cdot; \boldsymbol{\theta}_{l/d})$ with parameters in $\Theta_{l/d}$.

3) $\Theta_{l/d}$ are nonempty, compact, and convex subsets of a finite-dimensional metric space $\mathbb{R}^{\mathsf{s}_{l/d}}$.

The sets of parameters $\Theta_{l/d}$ are the pure strategy sets (also known as action spaces) for the l-player and d-player, respectively. The cost functions of the two players, which quantify the cost that each player incurs when a strategy profile $(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d$ is played, coincide with the expected costs, denoted by $\overline{c}_{l/d}(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)$, that the two players have in the underlying prediction game if strategies are sampled from $p_l(\cdot; \boldsymbol{\theta}_l)$ and $p_d(\cdot; \boldsymbol{\theta}_d)$, according to the expected cost-minimizing hypothesis

$$\overline{c}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) = \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \dot{\mathbf{X}}\sim p_d(\cdot;\boldsymbol{\theta}_d)}}[c_l(\boldsymbol{w}, \dot{\mathbf{X}})] \tag{3}$$

$$\overline{c}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) = \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \dot{\mathbf{X}}\sim p_d(\cdot;\boldsymbol{\theta}_d)}}[c_d(\boldsymbol{w}, \dot{\mathbf{X}})] \tag{4}$$

where $\mathbb{E}[\cdot]$ denotes the expectation operator. We assume $\overline{c}_{l/d}$ to be well-defined functions, i.e., the expectations to be finite for any $(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d$. To avoid confusion between $c_{l/d}$ and $\overline{c}_{l/d}$, in the following, we will refer them, respectively, as cost functions, and expected cost functions.

By adhering to a noncooperative setting, the two players involved in the prediction game are not allowed to communicate and they play their strategies simultaneously. Each player has complete information of the game setting by knowing the expected cost function and strategy set of either players. Under rationality assumption, each player's interest is to achieve the greatest personal advantage, i.e., to incur the lowest possible cost. Accordingly, the players are prone to play a Nash equilibrium, which in the context of our randomized prediction game is a strategy profile $(\boldsymbol{\theta}_l^\star, \boldsymbol{\theta}_d^\star) \in \Theta_l \times \Theta_d$, such that no player is interested in changing his/her own playing strategy. In formal terms, this yields

$$\boldsymbol{\theta}_l^\star \in \arg\min_{\boldsymbol{\theta}_l\in\Theta_l} \overline{c}_l\big(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d^\star\big), \quad \boldsymbol{\theta}_d^\star \in \arg\min_{\boldsymbol{\theta}_d\in\Theta_d} \overline{c}_d\big(\boldsymbol{\theta}_l^\star, \boldsymbol{\theta}_d\big). \tag{5}$$

B. Existence of a Nash Equilibrium

The existence of a Nash equilibrium of a randomized prediction game is not granted in general. A sufficient condition is given in the following.

Theorem 1 (Existence): A randomized prediction game admits at least one Nash equilibrium if it has the following.

1) $\overline{c}_{l/d}$ are continuous in $\Theta_l \times \Theta_d$.
2) $\overline{c}_l(\cdot, \boldsymbol{\theta}_d)$ is quasi-convex in Θ_l for any $\boldsymbol{\theta}_d \in \Theta_d$.
3) $\overline{c}_d(\boldsymbol{\theta}_l, \cdot)$ is quasi-convex in Θ_d for any $\boldsymbol{\theta}_l \in \Theta_l$.

Proof: The result follows directly from the Debreu–Glicksberg–Fan theorem [29]. ∎

C. Uniqueness of a Nash Equilibrium

In addition to the existence of a Nash equilibrium, it is of interest to investigate if the equilibrium is unique. However, determining tight conditions that guarantee the uniqueness of the Nash equilibrium for any randomized prediction game is challenging; in particular, due to the additional dependence on a probability distribution for the learner and the data generator.

We will make use of a classical result due to Rosen [30] to formulate sufficient conditions for the uniqueness of the Nash equilibrium of randomized prediction games in terms of the so-called pseudogradient of the game, defined as

$$\overline{g}_{\boldsymbol{r}} = \begin{bmatrix} r_l \nabla_{\boldsymbol{\theta}_l} \bar{c}_l \\ r_d \nabla_{\boldsymbol{\theta}_d} \bar{c}_d \end{bmatrix} \tag{6}$$

with any fixed vector $\boldsymbol{r} = [r_l, r_d]^\top \geq \mathbf{0}$. In particular, a randomized prediction game admits a unique Nash equilibrium if the following assumption is verified.

Assumption 1:

1) $\overline{c}_{l/d}$ are twice differentiable in $\Theta_l \times \Theta_d$.
2) $\overline{c}_l(\cdot, \boldsymbol{\theta}_d)$ is convex in Θ_l for any $\boldsymbol{\theta}_d \in \Theta_d$.
3) $\overline{c}_d(\boldsymbol{\theta}_l, \cdot)$ is convex in Θ_d for any $\boldsymbol{\theta}_l \in \Theta_l$

and $\overline{g}_{\boldsymbol{r}}$ is strictly monotone for some fixed $\boldsymbol{r} > \mathbf{0}$, that is,

$$\big[\overline{g}_{\boldsymbol{r}}(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) - \overline{g}_{\boldsymbol{r}}\big(\boldsymbol{\theta}_l', \boldsymbol{\theta}_d'\big)\big]^\top \begin{bmatrix} \boldsymbol{\theta}_l - \boldsymbol{\theta}_l' \\ \boldsymbol{\theta}_d - \boldsymbol{\theta}_d' \end{bmatrix} > 0$$

for any distinct strategy profiles $(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d), (\boldsymbol{\theta}_l', \boldsymbol{\theta}_d') \in \Theta_l \times \Theta_d$.[2]

Rosen [30] also provides a useful sufficient condition that guarantees a strictly monotone pseudogradient. This requires the Jacobian of the pseudogradient, also known as pseudo-Jacobian, given by

$$\overline{J}_{\boldsymbol{r}} = \begin{bmatrix} r_l \nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_l} \overline{c}_l & r_l \nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_d} \overline{c}_l \\ r_d \nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_l} \overline{c}_d & r_d \nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_d} \overline{c}_d \end{bmatrix} \tag{7}$$

to be positive definite.

Theorem 2: A randomized prediction game admits a unique Nash equilibrium if Assumption 1 holds, and the pseudo-Jacobian $\overline{J}_{\boldsymbol{r}}(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)$ is positive definite for all $(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d$ and some fixed $\boldsymbol{r} > \mathbf{0}$.

Proof: Under Assumption 1, the positive definiteness of $\overline{J}_{\boldsymbol{r}}$ for all strategy profiles and some fixed vector $\boldsymbol{r} > \mathbf{0}$ implies the strict monotonicity of $\overline{g}_{\boldsymbol{r}}$, which in turn implies the uniqueness of the Nash equilibrium [30, Th. 6]. ∎

In the following, we provide sufficient conditions that ensure the positive definiteness of the pseudo-Jacobian and the uniqueness of the Nash equilibrium via Theorem. 2. To this end, we decompose $\bar{c}_{l/d}(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)$ as:

$$\begin{aligned} \bar{c}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) &= \rho_l \overline{\Omega}_l(\boldsymbol{\theta}_l) + \bar{L}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \\ \bar{c}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) &= \rho_d \overline{\Omega}_d(\boldsymbol{\theta}_d) + \bar{L}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \end{aligned} \tag{8}$$

where $\overline{\Omega}_{l/d}$ and $\bar{L}_{l/d}$ are the expected regularization and loss terms given by

$$\begin{aligned} \overline{\Omega}_l(\boldsymbol{\theta}_l) &= \mathbb{E}_{\boldsymbol{w}\sim p_l(\cdot,\boldsymbol{\theta}_l)}[\Omega_l(\boldsymbol{w})] \\ \overline{\Omega}_d(\boldsymbol{\theta}_d) &= \mathbb{E}_{\dot{\mathbf{X}}\sim p_d(\cdot,\boldsymbol{\theta}_d)}[\Omega_d(\dot{\mathbf{X}})] \\ \bar{L}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) &= \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \dot{\mathbf{X}}\sim p_d(\cdot;\boldsymbol{\theta}_d)}}\left[\sum_{i=1}^{\mathsf{n}} \ell_l(\boldsymbol{w}^\top \boldsymbol{x}_i, y_i)\right] \\ \bar{L}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) &= \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \dot{\mathbf{X}}\sim p_d(\cdot;\boldsymbol{\theta}_d)}}\left[\sum_{i=1}^{\mathsf{n}} \ell_d(\boldsymbol{w}^\top \boldsymbol{x}_i, y_i)\right]. \end{aligned}$$

Moreover, we require the following convexity and differentiability conditions on $\overline{\Omega}_{l/s}$ and $\bar{L}_{l/d}$.

[2] Assumption 1-1) could be relaxed to continuously differentiable.

Assumption 2:

1) $\overline{\Omega}_{l/d}$ is strongly convex and twice continuously differentiable in $\Theta_{l/d}$.
2) $\bar{L}_l(\cdot, \boldsymbol{\theta}_d)$ is convex and twice continuously differentiable in Θ_l for all $\boldsymbol{\theta}_d \in \Theta_d$.
3) $\bar{L}_d(\boldsymbol{\theta}_l, \cdot)$ is convex and twice continuously differentiable in Θ_d for all $\boldsymbol{\theta}_l \in \Theta_l$.

Finally, we introduce some quantities that are used in the subsequent lemma, which gives sufficient conditions for the positive definiteness of the pseudo-Jacobian

$$\begin{aligned}
\lambda_l^{\Omega} &= \inf_{\boldsymbol{\theta}_l \in \Theta_l} \lambda_{\min}\left[\nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_l} \overline{\Omega}_l(\boldsymbol{\theta}_l)\right] \\
\lambda_d^{\Omega} &= \inf_{\boldsymbol{\theta}_d \in \Theta_d} \lambda_{\min}\left[\nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_d} \overline{\Omega}_d(\boldsymbol{\theta}_d)\right] \\
\lambda_l^{L} &= \inf_{(\boldsymbol{\theta}_l,\boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d} \lambda_{\min}\left[\nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_l} \overline{L}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)\right] \\
\lambda_d^{L} &= \inf_{(\boldsymbol{\theta}_l,\boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d} \lambda_{\min}\left[\nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_d} \overline{L}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)\right] \\
\tau &= \sup_{(\boldsymbol{\theta}_l,\boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d} \lambda_{\max}[R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)^\top]
\end{aligned}$$

where $R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) = 1/2[\nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_d} \bar{L}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)^\top + \nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_l} \bar{L}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)]$ and $\lambda_{\max/\min}$ give the maximum/minimum eigenvalue of the matrix in input. The quantities listed above are finite and positive if Assumption 2 holds, given the compactness of $\Theta_{l/d}$.

Lemma 1: If Assumption 2 holds and

$$\left(\rho_l \lambda_l^{\Omega} + \lambda_l^{L}\right)\left(\rho_d \lambda_d^{\Omega} + \lambda_d^{L}\right) > \tau$$

then the pseudo-Jacobian $\overline{J}_r(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)$ is positive definite for all $(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d$ by taking $\boldsymbol{r} = (1, 1)^\top$.

Proof: The pseudo-Jacobian in (7) can be written as follows given the decomposition of $\bar{c}_{l/d}$ in (8):

$$\mathbf{J}_r = \begin{bmatrix} \rho_l \nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_l} \overline{\Omega}_l + \nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_l} \bar{L}_l & \nabla^2_{\boldsymbol{\theta}_l\boldsymbol{\theta}_d} \bar{L}_l \\ \nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_l} \bar{L}_d & \rho_d \nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_d} \overline{\Omega}_d + \nabla^2_{\boldsymbol{\theta}_d\boldsymbol{\theta}_d} \bar{L}_d \end{bmatrix}$$

where we omitted the arguments of $\overline{\Omega}_{l/d}$ and $\bar{L}_{l/d}$ for notational convenience. Let us denote by $\mathbf{J}_r^{ll}$, $\mathbf{J}_r^{ld}$, $\mathbf{J}_r^{dl}$, and $\mathbf{J}_r^{dd}$ the four matrices composing $\mathbf{J}_r$ (in top–down, left–right order).

Consider the following matrix:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}^{ll} & \mathbf{H}^{ld} \\ \mathbf{H}^{dl} & \mathbf{H}^{dd} \end{bmatrix} = \begin{bmatrix} \rho_l \lambda_l^{\Omega} + \lambda_l^{L} & R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)^\top \\ R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) & \rho_d \lambda_d^{\Omega} + \lambda_d^{L} \end{bmatrix}.$$

Then, we have for all $\boldsymbol{t} = (\boldsymbol{t}_l^\top, \boldsymbol{t}_d^\top) \neq \mathbf{0}$

$$\begin{aligned}
\boldsymbol{t}^\top \mathbf{J}_r \boldsymbol{t} &= \boldsymbol{t} \frac{\mathbf{J}_r + \mathbf{J}_r^\top}{2} \boldsymbol{t}^\top \\
&= \underbrace{\boldsymbol{t}_l \mathbf{J}_r^{ll} \boldsymbol{t}_l}_{\geq \boldsymbol{t}_l \mathbf{H}^{ll} \boldsymbol{t}_l} + \underbrace{\boldsymbol{t}_d \mathbf{J}_r^{dd} \boldsymbol{t}_d}_{\geq \boldsymbol{t}_d^\top \mathbf{H}^{dd} \boldsymbol{t}_d} + \boldsymbol{t}_l^\top \underbrace{\frac{\mathbf{J}_r^{ld} + \mathbf{J}_r^{dl\top}}{2}}_{\mathbf{H}^{ld} + \mathbf{H}^{dl\top}} \boldsymbol{t}_d \geq \boldsymbol{t}^\top \mathbf{H} \boldsymbol{t}
\end{aligned}$$

where the underbraced relations follow from the definitions of $\lambda_{l/d}^{\Omega}$, $\lambda_{l/d}^{L}$, and R. Accordingly, the positive definiteness of $\mathbf{J}_r$ can be derived from the positive definiteness of matrix $\mathbf{H}$. To prove the latter, we will show that all roots of the characteristic polynomial $\det(\mathbf{H} - \lambda \mathbf{I})$ of $\mathbf{H}$ are positive. By properties of the determinant,[3] we have

$$\begin{aligned}
\det(\mathbf{H} - \lambda \mathbf{I}) &= \det\left(\left(\rho_l \lambda_l^{\Omega} + \lambda_l^{L} - \lambda\right)\mathbf{I}\right) \\
&\quad \cdot \det\left(\left(\rho_d \lambda_d^{\Omega} + \lambda_d^{L} - \lambda\right)\mathbf{I} - \frac{\mathbf{S}}{\rho_l \lambda_l^{\Omega} + \lambda_l^{L} - \lambda}\right)
\end{aligned}$$

where $\mathbf{S}$ is a diagonal matrix with the eigenvalues of $R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)^\top$. The roots of the first determinant term are all equal to $\rho_l \lambda_l^{\Omega} + \lambda_l^{L}$, which is positive because $\rho_l > 0$ by construction and $\lambda_l^{\Omega} > 0$ follows from the strong convexity of $\overline{\Omega}_l$ in Assumption 2-1). As for the second determinant term, take the ith diagonal element $\mathbf{S}_{ii}$ of $\mathbf{S}$. Then, two roots are the solution of the following quadratic equation:

$$\lambda^2 - \lambda(a + b) + ab - \mathbf{S}_{ii} = 0$$

which are given by

$$\lambda_{1,2}^{(i)} = a + b \pm \sqrt{(a - b)^2 + 4\mathbf{S}_{ii}}$$

where $a = \rho_l \lambda_l^{\Omega} + \lambda_l^{L}$ and $b = \rho_d \lambda_d^{\Omega} + \lambda_d^{L}$. Among the two, $\lambda_2^{(i)}$ (the one with the minus) is the smallest one, which is strictly positive if

$$ab = \left(\rho_l \lambda_l^{\Omega} + \lambda_l^{L}\right)\left(\rho_d \lambda_d^{\Omega} + \lambda_d^{L}\right) > \mathbf{S}_{ii}.$$

Since the condition has to hold for any choice of the eigenvalue $\mathbf{S}_{ii}$ in the right-hand side of the inequality, we take the maximum one $\max_i \mathbf{S}_{ii}$, which coincides with $\lambda_{\max}(R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) R(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)^\top)$. We further maximize the latter quantity with respect to $(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d$, as we want the result to hold for any parameterization. Therefrom, we recover the variable τ and the condition $(\rho_l \lambda_l^{\Omega} + \lambda_l^{L})(\rho_d \lambda_d^{\Omega} + \lambda_d^{L}) > \tau$, which guarantees that all roots of the characteristic polynomial of $\mathbf{H}$ are strictly positive for any choice of $(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d$, and hence, $\overline{J}_r$ is positive definite over $\Theta_l \times \Theta_d$. ■

In addition to Lemma 1, we provide in the Supplementary Material alternative (stronger) sufficient conditions, which generalize the ones given in [11].

D. Finding a Nash Equilibrium

From the computational perspective, we can find a Nash equilibrium in our game by exploiting algorithms similar to the ones adopted for static prediction games [11]. In particular, we consider a modified extragradient descent algorithm [11], [31], [32] that finds a solution to the following variational inequality problem, provided that $\overline{g}_{\boldsymbol{r}}$ is continuous and monotone:

$$\overline{g}_{\boldsymbol{r}}\left(\boldsymbol{\theta}_l^\star, \boldsymbol{\theta}_d^\star\right)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^\star) \geq 0 \quad \forall (\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \in \Theta_l \times \Theta_d \tag{9}$$

where $\boldsymbol{\theta} = [\boldsymbol{\theta}_l^\top, \boldsymbol{\theta}_d^\top]^\top$ and similarly for $\boldsymbol{\theta}^\star$. Any solution $\boldsymbol{\theta}^\star$ to this problem can be shown to correspond bijectively to a Nash equilibrium of a game having $\overline{g}_{\boldsymbol{r}}$ as pseudogradient [11], [32].

If Theorem 1 holds, the pseudo-Jacobian $\bar{J}_{\boldsymbol{r}}$ can be shown to be positive semidefinite, and $\overline{g}_{\boldsymbol{r}}$ is thus continuous and

[3] $\det\begin{bmatrix} a\mathbf{I} & \mathbf{B}^\top \\ \mathbf{B} & d\mathbf{I} \end{bmatrix} = \det(a\mathbf{I}) \det(d\mathbf{I} - \frac{1}{a}\mathbf{B}\mathbf{B}^\top)$ and if $\mathbf{U}\mathbf{S}\mathbf{U}^\top$ is the eigendecomposition of $\mathbf{B}\mathbf{B}^\top$, then the latter determinant becomes $\det(\mathbf{U}(d\mathbf{I} - \frac{1}{a}\mathbf{S})\mathbf{U}^\top) = \det(d\mathbf{I} - \frac{1}{a}\mathbf{S})$

Algorithm 1 Extragradient Descent (Adapted From [11])

Input: Cost functions $\bar{c}_{l/d}$; parameter spaces Θ_l, Θ_d; a small positive constant ϵ.

Output: The optimal parameters $\boldsymbol{\theta}_l, \boldsymbol{\theta}_d$.

Randomly select $\boldsymbol{\theta}^{(0)} = (\boldsymbol{\theta}_l^{(0)}, \boldsymbol{\theta}_d^{(0)}) \in \Theta_l \times \Theta_d$.
Set iteration count $k = 0$, and select $\sigma, \beta \in (0, 1)$.
Set $\boldsymbol{r} = (r_l, r_d)^\top = (1, \rho_l/\rho_d)^\top$.
repeat
  Set $\boldsymbol{d}^{(k)} = \Pi_{\Theta_l \times \Theta_d}\left(\boldsymbol{\theta}^{(k)} - \overline{\boldsymbol{g}}_{\boldsymbol{r}}\left(\boldsymbol{\theta}_l^{(k)}, \boldsymbol{\theta}_d^{(k)}\right)\right) - \boldsymbol{\theta}^{(k)}$.
  Find maximum step size $t^{(k)} \in \{\beta^p | p \in \mathbb{N}\}$ s.t.

$$-\overline{\boldsymbol{g}}_{\boldsymbol{r}}\left(\bar{\boldsymbol{\theta}}_l^{(k)}, \bar{\boldsymbol{\theta}}_d^{(k)}\right)^\top \boldsymbol{d}^{(k)} \geq \sigma\left(\left\|\boldsymbol{d}^{(k)}\right\|_2^2\right),$$

  where $\bar{\boldsymbol{\theta}}^{(k)} = \boldsymbol{\theta}^{(k)} + t^{(k)}\boldsymbol{d}^{(k)}$.
  Set $\eta^{(k)} = -\frac{t^{(k)}}{\left\|\overline{\boldsymbol{g}}_{\boldsymbol{r}}\left(\bar{\boldsymbol{\theta}}_l^{(k)}, \bar{\boldsymbol{\theta}}_d^{(k)}\right)\right\|_2^2}\overline{\boldsymbol{g}}_{\boldsymbol{r}}\left(\bar{\boldsymbol{\theta}}_l^{(k)}, \bar{\boldsymbol{\theta}}_d^{(k)}\right)^\top \boldsymbol{d}^{(k)}$.
  Set $\boldsymbol{\theta}^{(k+1)} = \Pi_{\Theta_l \times \Theta_d}\left(\boldsymbol{\theta}^{(k)} - \eta^{(k)}\overline{\boldsymbol{g}}_{\boldsymbol{r}}\left(\bar{\boldsymbol{\theta}}_l^{(k)}, \bar{\boldsymbol{\theta}}_d^{(k)}\right)\right)$.
  Set $k = k + 1$.
until $\left\|\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)}\right\|_2^2 \leq \epsilon$
return $\boldsymbol{\theta}_l = \boldsymbol{\theta}_l^{(k)}$, $\boldsymbol{\theta}_d = \boldsymbol{\theta}_d^{(k)}$

monotone. Hence, the variational inequality can be solved by the modified extragradient descent algorithm given as Algorithm 1, which is guaranteed to converge to a Nash equilibrium point [31], [33]. The algorithm generates a sequence of feasible points whose distance from the equilibrium solution is monotonically decreased. It exploits a projection operator $\Pi_{\Theta_l \times \Theta_d}(\boldsymbol{\theta})$ to map the input vector $\boldsymbol{\theta}$ onto the closest admissible point in $\Theta_l \times \Theta_d$, and a simple line-search algorithm to find the maximum step t on the descent direction $\boldsymbol{d}$.[4]

In Section III, we apply our randomized prediction game to the case of linear SVM learners, and compute the corresponding pseudogradient, as required in Algorithm 1.

III. Randomized Prediction Games for Support Vector Machines

In this section, we consider a randomized prediction game involving a linear SVM learner [34] and Gaussian distributions as the underlying probabilities $p_{l/d}$.

Learner: The decision function of the learner is of the type $f(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^\top \phi(\boldsymbol{x})$ where the feature map is given by $\phi(\boldsymbol{x}) = [\boldsymbol{x}^\top \ 1^\top]$. For convenience, we consider a decomposition of $\boldsymbol{w}$ into $[\tilde{\boldsymbol{w}}^\top \ b]^\top$, where $\tilde{\boldsymbol{w}} \in \mathbb{R}^{\mathsf{m}-1}$ and $b \in \mathbb{R}$. Hence, the decision function can also be written as $f(\boldsymbol{x}; \boldsymbol{w}) = \tilde{\boldsymbol{w}}^\top \boldsymbol{x} + b$. Accordingly, the input space $\mathcal{X}$ is a $(\mathsf{m}-1)$-dimensional vector space, i.e., $\mathcal{X} \subseteq \mathbb{R}^{\mathsf{m}-1}$. The distribution p_l for the learner is assumed to be Gaussian. In order to guarantee the theoretical existence of the Nash equilibrium through Theorem 1, we assume the parameters of the Gaussian distribution to be bounded. For the sake of clarity, we use in this section axis-aligned Gaussians (i.e., with diagonal covariance matrices) for our analysis, even though general covariances could be adopted as well. Under these assumptions, we define the strategy set for the learner as $\Theta_l = \{(\boldsymbol{\mu}_{\boldsymbol{w}}, \boldsymbol{\sigma}_{\boldsymbol{w}}) \in \mathbb{R}^{\mathsf{m}} \times \mathbb{R}_+^{\mathsf{m}}\} \cap \mathcal{B}_l$, where $\mathcal{B}_l \subset \mathbb{R}^{\mathsf{m}} \times \mathbb{R}_+^{\mathsf{m}}$ is an application-dependent nonempty, convex, bounded set, restricting the set of feasible parameters. The parameter vectors $\boldsymbol{\mu}_{\boldsymbol{w}}$ and $\boldsymbol{\sigma}_{\boldsymbol{w}}$ encode the mean and standard deviation of the axis-aligned Gaussian distributions. The loss function ℓ_l of the learner corresponds to the hinge loss of the SVM, i.e., $\ell_l(z, y) = [1 - zy]_+$ with $[z]_+ = \max(0, z)$, while the strategy penalization term $\Omega_l(\boldsymbol{w})$ is the squared Euclidean norm of $\tilde{\boldsymbol{w}}$. As a result, the cost function c_l corresponds to the C-SVM objective function, and it is convex in $\boldsymbol{w}$

$$c_l(\boldsymbol{w}, \mathbf{X}) = \frac{\rho_l}{2}\|\tilde{\boldsymbol{w}}\|^2 + \sum_{i=1}^{\mathsf{n}}[1 - y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)]_+. \quad (10)$$

Data Generator: For convenience, we consider $\mathbf{X}$ rather than $\dot{\mathbf{X}}$ as the quantity undergoing the randomization. This comes without loss of generality, because there is a one-to-one correspondence between $\dot{\boldsymbol{x}}_i$ and $\boldsymbol{x}_i$ if we consider the linear feature map $\dot{\boldsymbol{x}}_i = \phi(\boldsymbol{x}_i) = [\boldsymbol{x}_i^\top, 1]^\top$. Moreover, we assume that samples $\boldsymbol{x}_i$ can be perturbed independently. Accordingly, the distribution p_d for the data generator factorizes as $p_d(\mathbf{X}; \boldsymbol{\theta}_d) = \prod_{i=1}^{\mathsf{n}} p_d(\boldsymbol{x}_i; \boldsymbol{\theta}_d^{(i)})$, where $\boldsymbol{\theta}_d = (\boldsymbol{\theta}_d^{(1)}, \ldots, \boldsymbol{\theta}_d^{(\mathsf{n})})$. We consider $p_d(\boldsymbol{x}_i; \boldsymbol{\theta}_d^{(i)})$ to be a k-variate axis-aligned Gaussian distribution with bounded mean and standard deviation given by $\boldsymbol{\theta}_d^{(i)} = (\boldsymbol{\mu}_{\boldsymbol{x}_i}, \boldsymbol{\sigma}_{\boldsymbol{x}_i})$. In summary, the strategy set adopted for the data generator is given by $\Theta_d = \prod_{i=1}^{\mathsf{n}} \Theta_d^{(i)}$, where $\Theta_d^{(i)} = \{(\boldsymbol{\mu}_{\boldsymbol{x}_i}, \boldsymbol{\sigma}_{\boldsymbol{x}_i}) \in \mathbb{R}^{\mathsf{k}} \times \mathbb{R}_+^{\mathsf{k}}\} \cap \mathcal{B}_d$. Here, $\mathcal{B}_d \subset \mathbb{R}^{\mathsf{k}} \times \mathbb{R}_+^{\mathsf{k}}$ is a nonempty, convex, bounded set. The loss function ℓ_d of the data generator is the hinge loss under wrong labeling, i.e., $\ell_d(z, y) = [1 + zy]_+$. In this way, the data generator is penalized if the learner correctly classifies a sample point. Finally, the strategy penalization function Ω_d is the squared Euclidean distance of the perturbed samples in $\mathbf{X}$ from the ones in the original training set $\hat{\mathcal{D}}$, i.e., $\Omega_d(\mathbf{X}) = \sum_{i=1}^{\mathsf{n}} \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|^2$. The resulting cost function c_d is convex in $\mathbf{X}$

$$c_d(\boldsymbol{w}, \mathbf{X}) = \frac{\rho_d}{2}\sum_{i=1}^{\mathsf{n}}\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|^2 + \sum_{i=1}^{\mathsf{n}}[1 + y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)]_+. \quad (11)$$

Existence of a Nash Equilibrium: The proposed randomized prediction game for the SVM learner admits at least one Nash equilibrium. This can be proven by means of Theorem 1. Indeed, the required continuity of $\bar{c}_{l/d}$ holds and, as for the quasi-convexity conditions, we can rewrite (3) as follows by exploiting the fact that p_l is a Gaussian distribution with mean $\boldsymbol{\mu}_{\boldsymbol{w}}$ and standard deviation $\boldsymbol{\sigma}_{\boldsymbol{w}}$:

$$\bar{c}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) = \mathbb{E}_{\substack{z \sim \mathcal{N}(\mathbf{0}, I) \\ \mathbf{X} \sim p_d(\cdot; \boldsymbol{\theta}_d)}}\left[c_l(\boldsymbol{\mu}_{\boldsymbol{w}} + D(\boldsymbol{\sigma}_{\boldsymbol{w}})z, \mathbf{X})\right] \quad (12)$$

where $\mathcal{N}(\mathbf{0}, I)$ is a m-dimensional standard normal distribution and $D(\boldsymbol{\sigma}_{\boldsymbol{w}})$ is a diagonal matrix having $\boldsymbol{\sigma}_{\boldsymbol{w}}$ on the diagonal. Since c_l is convex in its first argument and convexity is preserved under addition of convex functions, positive rescaling, and composition with linear functions, we have that

[4]We refer the reader to [11], [31], [32] (and references therein) for proofs that derive conditions for which d is effectively a descent direction.

$\overline{c}_l$ is convex (and thus quasi-convex) in $\boldsymbol{\theta}_l = (\boldsymbol{\mu}_{\boldsymbol{w}}, \boldsymbol{\sigma}_{\boldsymbol{w}})$. As for the quasi-convexity condition of the data generator's cost, we can exploit the separability of c_d to rewrite (4) as follows:

$$\overline{c}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) = \sum_{i=1}^{\mathsf{n}} \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ z\sim\mathcal{N}(\mathbf{0},I)}} \left[c_d^{(i)}(\boldsymbol{w}, \boldsymbol{\mu}_{\boldsymbol{x}_i} + D(\boldsymbol{\sigma}_{\boldsymbol{x}_i})z)\right]$$

where

$$c_d^{(i)}(\boldsymbol{w}, \boldsymbol{x}) = \frac{\rho_d}{2}\|\boldsymbol{x} - \hat{\boldsymbol{x}}_i\|^2 + [1 + y_i(\boldsymbol{w}^\top \tilde{\boldsymbol{x}}_i + b)]_+.$$

Since $c_d^{(i)}$ is convex in its second argument, by following the same reasoning used to show the quasi-convexity of the learner's expected cost, we have that each expectation in $\overline{c}_d$ is convex in $\boldsymbol{\theta}_d^{(i)} = (\boldsymbol{\mu}_{\boldsymbol{x}_i}, \boldsymbol{\sigma}_{\boldsymbol{x}_i})$, $1 \le i \le \mathsf{n}$. As a consequence, $\overline{c}_d$ is convex, and hence, quasi-convex in $\boldsymbol{\theta}_d$ is the sum of convex functions.

Uniqueness of a Nash Equilibrium: We have shown that $\bar{c}_l(\cdot, \boldsymbol{\theta}_d)$ and $\bar{c}_d(\boldsymbol{\theta}_l, \cdot)$ are convex as required by Assumption 1-2) and 1-3). In particular, we have that the single expected regularization terms $\overline{\Omega}_{l/d}(\cdot)$ and loss terms $\bar{L}_l(\cdot, \boldsymbol{\theta}_d)$, $\bar{L}_d(\boldsymbol{\theta}_l, \cdot)$ are convex as well. Moreover, they are twice continuously differentiable by having Gaussian distributions for $p_{l/d}$. It is then sufficient to have $\overline{\Omega}_{l/d}$, which are strongly convex to prove the uniqueness of the Nash equilibrium via Lemma 1. While it is easy to see that $\overline{\Omega}_d$ is strongly convex, we have that $\overline{\Omega}_l$ is not strongly convex with respect to b due to the presence of an unregularized bias term b in the learner. The problem derives from the fact that the SVM itself may not have a unique solution when the bias term is present and nonregularized (see [35], [36] for a characterization of the degenerate cases). As a result, the proposed game is not guaranteed to have a unique Nash equilibrium in its actual form. On the other hand, a unique Nash equilibrium may be obtained by either considering an unbiased SVM, i.e., by setting $b = 0$ as in [11], or a regularized bias term, e.g., by adding $\frac{\varepsilon}{2}b^2$ to the learner's objective function with $\varepsilon > 0$. In both cases, all conditions that ensure the uniqueness of the Nash equilibrium via Theorem 2 and Lemma 1 would be satisfied, under proper choices of $\rho_{l/d}$.

It is worth noting however that the necessary and sufficient conditions under which a biased (nonregularized) SVM has no unique solution are quite restricted [35], [36]. For this reason, we believe that uniqueness of the Nash equilibrium could also be proven for the biased SVM under mild assumptions. However, this requires considerable effort in trying to relax the sufficiency conditions of Rosen [30], which is beyond the scope of this paper. We thus leave this challenge to future investigations. Moreover, we believe that enforcing a unique Nash equilibrium in our game by making the original SVM formulation strictly convex may lead to worse results, similar to exploiting convex approximations to solve originally nonconvex problems in machine learning [37], [38]. For the above reasons, in this paper, we choose to retain the original SVM formulation for the learner, by sacrificing the uniqueness of the Nash Equilibrium. We nevertheless provide in Section V a discussion of why having a unique Nash Equilibrium is not so important in practice for our game, and we empirically show in Section VI that our approach can anyway achieve competitive performances with respect to other state-of-the-art approaches.

The rest of this section is devoted to showing how to compute the pseudogradient (6) by providing explicit formulas for $\nabla_{\boldsymbol{\theta}_l}\bar{c}_l$ and $\nabla_{\boldsymbol{\theta}_d}\bar{c}_d$.

A. Gradient of the Learner's Cost

In this section, we focus on computing the gradient $\nabla_{\boldsymbol{\theta}_l}\bar{c}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d)$, where $\bar{c}_l$ is defined as in (10). By properties of expectation and since $\boldsymbol{w}$ follows an axis-aligned Gaussian distribution with mean $\boldsymbol{\mu}_{\boldsymbol{w}}$ and standard deviation $\boldsymbol{\sigma}_{\boldsymbol{w}}$, we can reduce the cost of the learner to:

$$\bar{c}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) = \frac{\rho_l}{2}(\|\boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}\|^2 + \|\boldsymbol{\sigma}_{\tilde{\boldsymbol{w}}}\|^2) + \sum_{i=1}^{\mathsf{n}} \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \boldsymbol{x}_i\sim p_d\left(\cdot;\boldsymbol{\theta}_d^{(i)}\right)}} [[1 - y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)]_+] \quad (13)$$

where we are assuming the following decompositions for the mean $\boldsymbol{\mu}_{\boldsymbol{w}} = [\boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}^\top \ \mu_b]^\top$ and standard deviation $\boldsymbol{\sigma}_{\boldsymbol{w}} = [\boldsymbol{\sigma}_{\tilde{\boldsymbol{w}}}^\top \ \sigma_b]^\top$. The hard part for the minimization is the term in the expectation, which cannot be expressed to our knowledge in a closed-form function of the Gaussian's parameters. We thus resort to a central-limit-theorem-like approximation, by regarding $s_i = 1 - y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)$ as a Gaussian-distributed variable with mean μ_{s_i} and standard deviation σ_{s_i}, i.e., $s_i \sim \mathcal{N}(\mu_{s_i}, \sigma_{s_i})$. In general, s_i does not follow a Gaussian distribution, since the product of two normal deviates is not normally distributed. However, if the number of features k is large, the approximation becomes reasonable. Under this assumption, we can rewrite the expectation as follows:

$$\mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \boldsymbol{x}_i\sim p_d\left(\cdot;\boldsymbol{\theta}_d^{(i)}\right)}} [[1 - y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)]_+] = \mathbb{E}_{s_i\sim\mathcal{N}(\mu_{s_i},\sigma_{s_i})}[[s_i]_+]. \quad (14)$$

The mean and the variance of the Gaussian distribution in the right-hand side of (14) are, respectively, given by

$$\mu_{s_i} = \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \boldsymbol{x}_i\sim p_d\left(\cdot;\boldsymbol{\theta}_d^{(i)}\right)}} [1 - y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)] = 1 - y_i\left(\boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}^\top \boldsymbol{\mu}_{\boldsymbol{x}_i} + \mu_b\right) \quad (15)$$

$$\sigma_{s_i}^2 = \mathbb{V}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \boldsymbol{x}_i\sim p_d\left(\cdot;\boldsymbol{\theta}_d^{(i)}\right)}} [1 - y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)] = \boldsymbol{\sigma}_{\tilde{\boldsymbol{w}}}^{2\top}\left(\boldsymbol{\sigma}_{\boldsymbol{x}_i}^2 + \boldsymbol{\mu}_{\boldsymbol{x}_i}^2\right) + \boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}^{2\top}\boldsymbol{\sigma}_{\boldsymbol{x}_i}^2 + \sigma_b^2 \quad (16)$$

where $\mathbb{V}$ is the variance operator, and we assume that squaring a vector corresponds to squaring each single component.

The expectation in (14) can be transformed after simple manipulations into the following function involving the Gauss error function (integral function of the standard normal distribution) denoted by erf():

$$h(\mu_{s_i}, \sigma_{s_i}) = \frac{\sigma_{s_i}}{\sqrt{2\pi}} \exp\left(-\frac{\mu_{s_i}^2}{2\sigma_{s_i}^2}\right) + \frac{\mu_{s_i}}{2}\left[1 - \operatorname{erf}\left(-\frac{\mu_{s_i}}{\sqrt{2}\sigma_{s_i}}\right)\right]. \quad (17)$$

The learner's cost in (13) can thus be approximated as

$$\bar{c}_l(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) \approx L_l(\boldsymbol{\mu}_{\boldsymbol{w}}, \boldsymbol{\sigma}_{\boldsymbol{w}}) = \frac{\rho_l}{2}(\|\boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}\|^2 + \|\boldsymbol{\sigma}_{\tilde{\boldsymbol{w}}}\|^2) + \sum_{i=1}^{\mathsf{n}} h(\mu_{s_i}(\theta_l), \sigma_{s_i}(\theta_l)). \quad (18)$$

We can now approximate the gradient $\nabla_{\boldsymbol{\theta}_l}\overline{c}_l$ in terms of $\nabla_{\boldsymbol{\theta}_l} L_l$. In the following, we denote the Hadamard (also known as entrywise) product between any two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ as $\boldsymbol{a} \circ \boldsymbol{b}$, and we assume any scalar-by-vector derivative to be a column vector. The gradients of interest are given as

$$\frac{\partial L_l}{\partial \boldsymbol{\mu}_{\boldsymbol{w}}} = \rho_l \begin{bmatrix} \boldsymbol{\mu}_{\tilde{\boldsymbol{w}}} \\ 0 \end{bmatrix} + \sum_{i=1}^{\mathsf{n}} \left(\frac{\partial h}{\partial \mu_{s_i}} \frac{\partial \mu_{s_i}}{\partial \boldsymbol{\mu}_{\boldsymbol{w}}} + \frac{\partial h}{\partial \sigma_{s_i}^2} \frac{\partial \sigma_{s_i}^2}{\partial \boldsymbol{\mu}_{\boldsymbol{w}}} \right) \quad (19)$$

$$\frac{\partial L_l}{\partial \boldsymbol{\sigma}_{\boldsymbol{w}}} = \rho_l \begin{bmatrix} \boldsymbol{\sigma}_{\tilde{\boldsymbol{w}}} \\ 0 \end{bmatrix} + \sum_{i=1}^{\mathsf{n}} \left(\frac{\partial h}{\partial \mu_{s_i}} \frac{\partial \mu_{s_i}}{\partial \boldsymbol{\sigma}_{\boldsymbol{w}}} + \frac{\partial h}{\partial \sigma_{s_i}^2} \frac{\partial \sigma_{s_i}^2}{\partial \boldsymbol{\sigma}_{\boldsymbol{w}}} \right) \quad (20)$$

where it is not difficult to show that

$$\frac{\partial h}{\partial \mu_{s_i}} = \frac{1}{2}\left[1 - \operatorname{erf}\left(-\frac{1}{\sqrt{2}}\frac{\mu_{s_i}}{\sigma_{s_i}}\right)\right] \quad (21)$$

$$\frac{\partial h}{\partial \sigma_{s_i}^2} = \frac{1}{2}\frac{1}{\sqrt{2\pi}\sigma_{s_i}} \exp\left(-\frac{1}{2}\frac{\mu_{s_i}^2}{\sigma_{s_i}^2}\right) \quad (22)$$

and that

$$\frac{\partial \mu_{s_i}}{\partial \boldsymbol{\mu}_{\boldsymbol{w}}} = -y_i \begin{bmatrix} \boldsymbol{\mu}_{\boldsymbol{x}_i} \\ 1 \end{bmatrix}, \quad \frac{\partial \mu_{s_i}}{\partial \boldsymbol{\sigma}_{\boldsymbol{w}}} = \mathbf{0} \quad (23)$$

$$\frac{\partial \sigma_{s_i}^2}{\partial \boldsymbol{\mu}_{\boldsymbol{w}}} = \begin{bmatrix} 2\boldsymbol{\sigma}_{\boldsymbol{x}_i}^2 \circ \boldsymbol{\mu}_{\tilde{\boldsymbol{w}}} \\ 0 \end{bmatrix}, \quad \frac{\partial \sigma_{s_i}^2}{\partial \boldsymbol{\sigma}_{\boldsymbol{w}}} = 2\boldsymbol{\sigma}_{\boldsymbol{w}} \circ \begin{bmatrix} \boldsymbol{\sigma}_{\boldsymbol{x}_i}^2 + \boldsymbol{\mu}_{\boldsymbol{x}_i}^2 \\ 1 \end{bmatrix}. \quad (24)$$

B. Gradient of the Data Generator's Cost

In this section, we turn to the data generator and we focus on approximating $\nabla_{\boldsymbol{\theta}_d}\overline{c}_d$, where $\overline{c}_d$ is defined as in (11). We can separate $\overline{c}_d$ into the sum of n functions acting on each data sample independently, i.e., $\overline{c}_d(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d) = \sum_{i=1}^{\mathsf{n}} \overline{c}_d^{(i)}(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d^{(i)})$, where for each $i \in \{1, \ldots, \mathsf{n}\}$

$$\overline{c}_d^{(i)}\left(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d^{(i)}\right) = \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \boldsymbol{x}_i \sim p_d\left(\cdot;\boldsymbol{\theta}_d^{(i)}\right)}} \left[\frac{\rho_d}{2}\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|^2 + [1 + y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)]_+ \right]. \quad (25)$$

By exploiting the properties of the expectation and since $p_d(\cdot; \boldsymbol{\theta}_d^{(i)})$ is an axis-aligned Gaussian distribution with mean $\boldsymbol{\mu}_{\boldsymbol{x}_i}$ and standard deviation $\boldsymbol{\sigma}_{\boldsymbol{x}_i}$, we can simplify (25) as

$$\overline{c}_d^{(i)}\left(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d^{(i)}\right) = \frac{\rho_d}{2}(\|\boldsymbol{\mu}_{\boldsymbol{x}_i} - \hat{\boldsymbol{x}}_i\|^2 + \|\boldsymbol{\sigma}_{\boldsymbol{x}_i}\|^2) + \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \boldsymbol{x}_i \sim p_d\left(\cdot;\boldsymbol{\theta}_d^{(i)}\right)}} [[1 + y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)]_+]. \quad (26)$$

As in the case of the learner, the expectation is a troublesome term having the same form of (14), except for an inverted sign. We adopt the same approximation used in Section III-A to obtain a closed-form function. Accordingly, $t_i = 1 + y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)$ is assumed to be normally distributed with mean μ_{t_i} and σ_{t_i}. Then, the expectation in (26) can be approximated as $h(\mu_{t_i}, \sigma_{t_i})$, where function h is defined as in (17). The variance $\sigma_{t_i}^2$ is equal to $\sigma_{s_i}^2$ (16), while μ_{t_i} is given by

$$\mu_{t_i} = \mathbb{E}_{\substack{\boldsymbol{w}\sim p_l(\cdot;\boldsymbol{\theta}_l) \\ \boldsymbol{x}_i \sim p_d\left(\cdot;\boldsymbol{\theta}_d^{(i)}\right)}} [1 + y_i(\tilde{\boldsymbol{w}}^\top \boldsymbol{x}_i + b)] = 1 + y_i\left(\boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}^\top \boldsymbol{\mu}_{\boldsymbol{x}_i} + \mu_b\right).$$

The samplewise cost of the data generator (26) can thus be approximated as

$$\overline{c}_d^{(i)}\left(\boldsymbol{\theta}_l, \boldsymbol{\theta}_d^{(i)}\right) \approx L_d(\boldsymbol{\mu}_{\boldsymbol{x}_i}, \boldsymbol{\sigma}_{\boldsymbol{x}_i}) = \frac{\rho_d}{2}(\|\boldsymbol{\mu}_{\boldsymbol{x}_i} - \hat{\boldsymbol{x}}_i\|^2 + \|\boldsymbol{\sigma}_{\boldsymbol{x}_i}\|^2) + h\left(\mu_{t_i}\left(\theta_d^{(i)}\right), \sigma_{t_i}\left(\theta_d^{(i)}\right)\right). \quad (27)$$

The corresponding gradient is given by

$$\frac{\partial L_d}{\partial \boldsymbol{\mu}_{\boldsymbol{x}_i}} = (\boldsymbol{\mu}_{\boldsymbol{x}_i} - \hat{\boldsymbol{x}}_i) + \rho_d \left(\frac{\partial h}{\partial \mu_{t_i}} \frac{\partial \mu_{t_i}}{\partial \boldsymbol{\mu}_{\boldsymbol{x}_i}} + \frac{\partial h}{\partial \sigma_{t_i}^2} \frac{\partial \sigma_{t_i}^2}{\partial \boldsymbol{\mu}_{\boldsymbol{x}_i}} \right) \quad (28)$$

$$\frac{\partial L_d}{\partial \boldsymbol{\sigma}_{\boldsymbol{x}_i}} = \boldsymbol{\sigma}_{\boldsymbol{x}_i} + \rho_d \left(\frac{\partial h}{\partial \mu_{t_i}} \frac{\partial \mu_{t_i}}{\partial \boldsymbol{\sigma}_{\boldsymbol{x}_i}} + \frac{\partial h}{\partial \sigma_{t_i}^2} \frac{\partial \sigma_{t_i}^2}{\partial \boldsymbol{\sigma}_{\boldsymbol{x}_i}} \right) \quad (29)$$

where $(\partial h)/(\partial \mu_{t_i})$ and $(\partial h)/(\partial \sigma_{t_i}^2)$ are given as in (21) and (22), and

$$\frac{\partial \mu_{t_i}}{\partial \boldsymbol{\mu}_{\boldsymbol{x}_i}} = y_i \boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}, \quad \frac{\partial \mu_{t_i}}{\partial \boldsymbol{\sigma}_{\boldsymbol{x}_i}} = \mathbf{0} \quad (30)$$

$$\frac{\partial \sigma_{t_i}^2}{\partial \boldsymbol{\mu}_{\boldsymbol{x}_i}} = 2\boldsymbol{\sigma}_{\tilde{\boldsymbol{w}}}^2 \circ \boldsymbol{\mu}_{\boldsymbol{x}_i} \quad \frac{\partial \sigma_{t_i}^2}{\partial \boldsymbol{\sigma}_{\boldsymbol{x}_i}} = 2\boldsymbol{\sigma}_{\boldsymbol{x}_i} \circ \left(\boldsymbol{\sigma}_{\tilde{\boldsymbol{w}}}^2 + \boldsymbol{\mu}_{\tilde{\boldsymbol{w}}}^2\right). \quad (31)$$

IV. KERNELIZATION

Our game, as in Brückner *et al.* [11], assumes explicit knowledge of the feature space ϕ, where the data generator is assumed to randomize the samples $\dot{\boldsymbol{x}} = \phi(\boldsymbol{x})$. However, in many applications, the feature mapping is only implicitly given in terms of a positive semidefinite kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that measures the similarity between samples as a scalar product in the corresponding kernel Hilbert space, i.e., there exists $\phi : \mathcal{X} \to \mathbb{R}$ such that $k(\boldsymbol{x}, \boldsymbol{x}') = \phi(\boldsymbol{x})^\top \phi(\boldsymbol{x}')$. Note that in this setting, the input space $\mathcal{X}$ is not restricted to a vector space like in Section III (e.g., it might contain graphs or other structured entities).

For the representer theorem to hold [39], we assume that the randomized weight vectors of the learner live in the same subspace of the reproducing kernel Hilbert space, i.e., $\boldsymbol{w} = \sum_j \alpha_j \phi(\hat{\boldsymbol{x}}_j)$, where $\boldsymbol{\alpha} \in \mathbb{R}^{\mathsf{n}}$. Analogously, we restrict the randomized samples obtained by the data generator to live in the span of the mapped training instances, i.e., $\dot{\boldsymbol{x}}_i = \sum_{j=1}^{\mathsf{n}} \xi_{ij} \phi(\hat{\boldsymbol{x}}_j)$, where $\boldsymbol{\xi}_i = (\xi_{i1}, \ldots, \xi_{i\mathsf{n}})^\top \in \mathbb{R}^{\mathsf{n}}$.

Now, instead of randomizing $\boldsymbol{w}$ and $\dot{\mathbf{X}}$, we let the data generator and the learner randomize $\Xi = (\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{\mathsf{n}})$ and $\boldsymbol{\alpha}$, respectively. We assume that the expected costs $\bar{c}_{l/d}$ can be rewritten in terms of $\boldsymbol{\alpha}$ and Ξ in a way that involves only inner products of $\phi(\boldsymbol{x})$, to take advantage of the kernel trick. This is possible for the term $\boldsymbol{w}^\top \dot{\boldsymbol{x}}_i = \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\xi}_i$ in (1) and (2), where $\mathbf{K}$ is the kernel matrix. Hence, the applicability of the kernel trick only depends on the choice of the regularizers. It is easy to see that due to the linearity of the variable shift, existence

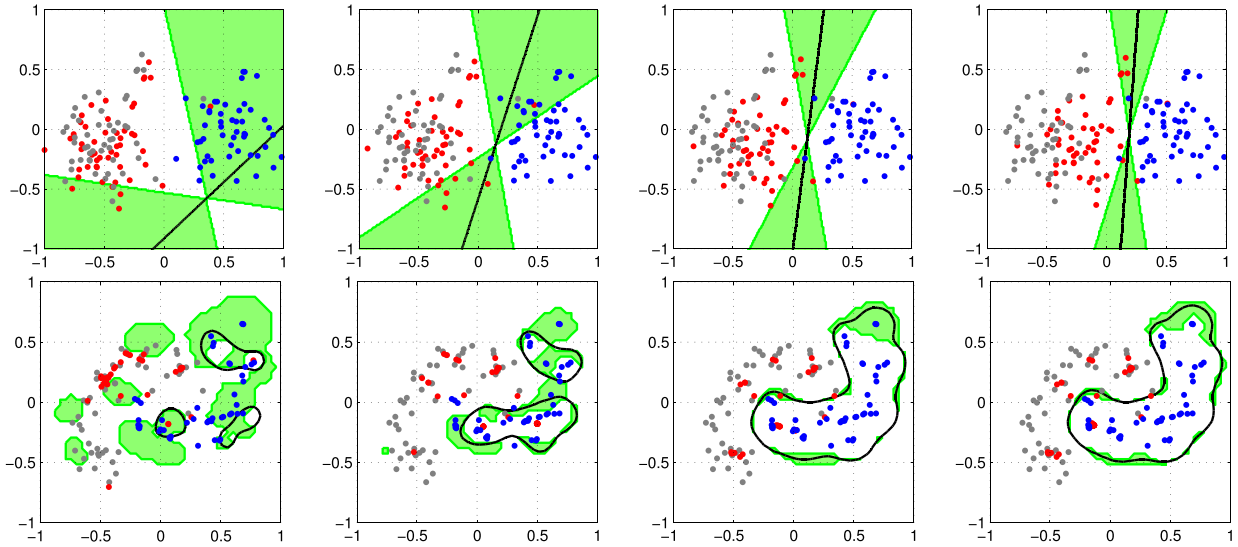

Fig. 1. 2-D examples of randomized prediction games, for SVMs with linear (top) and RBF kernels (bottom). Each row shows how the algorithm gradually converges to a Nash equilibrium. Blue (gray) points represent the legitimate (malicious) class. The mean of each manipulated attack sample is shown as a red point (for clarity, its variance is not shown). The black solid line represents the expected decision boundary, and the green shaded area highlights its variability within one standard deviation. Note how the linear SVM's decision boundary tends to shift toward the legitimate class, while the nonlinear boundary provides a better enclosing of the same class. This intuitively allows for a higher robustness to different kinds of attack, as it requires the adversary to make a higher number of modifications to the attack samples to evade detection, at the expense of a higher number of misclassified legitimate samples.

and uniqueness of a Nash equilibrium in our kernelized game hold under the same conditions given for the linear case.[5]

Although the data generator is virtually randomizing strategies in some subspace of the reproducing kernel Hilbert space, in reality manipulations should occur in the original input space. Hence, to construct the real attack samples $\{\boldsymbol{x}_i\}_{i=1}^n$ corresponding to the data generator's strategy at the Nash equilibrium, one should solve the so-called preimage problem, inverting the implicit feature mapping $\phi^{-1}(\mathbf{K}\boldsymbol{\xi}_i)$ for each sample. This problem is in general neither convex, nor it admits a unique solution. However, reliable solutions can be easily found using well-principled approximations [11], [39]. It is finally worth remarking that solving the preimage problem is not even required from the learner's perspective, i.e., to train the corresponding, secure classification function.

V. Discussion

In this section, we report a simple case study on a 2-D data set to visually demonstrate the effect of randomized prediction games on SVM-based learners. From a pragmatic perspective, this example also suggests that uniqueness of the Nash equilibrium should not be taken as a strict requirement in our game.

An instance of the proposed randomized prediction game for a linear SVM and for a nonlinear SVM with the RBF kernel is shown in Fig. 1. As one may note from the plots, the main effect of simulating the presence of an attacker that manipulates malicious data to evade detection is to cause the linear decision boundary to gradually shift toward the legitimate class and the nonlinear boundary to find a better enclosing of the legitimate samples. This should generally improve the learner's robustness to any kind of evasion attempt, as it requires the attacker to mimic more carefully the feature values of legitimate samples—a task typically harder in several adversarial settings than just obfuscating the content of a malicious sample to make it sufficiently different from the known malicious ones [7], [9].

Based on this observation, any attempt aiming to satisfy the sufficient conditions for uniqueness of the Nash equilibrium will result in an increase of the regularization strength in either the learner's or the attacker's cost function. Indeed, to satisfy the condition in Lemma 1, one could sufficiently increase ρ_l, ρ_d, or both. This amounts to increasing the regularization strength of either players, which in turn reduces in some sense their power. Hence, it should be clear that enforcing satisfaction of the sufficient conditions that guarantee the uniqueness of the Nash equilibrium might be counterproductive, by inducing the learner to weakly enclose the legitimate class, either due to a too strong regularization of the learners' parameters, or by limiting the ability of the attacker to manipulate the malicious samples, thus allowing the learner to keep a loose boundary. This will in general compromise the quality of the adversarial learning procedure. This argument shares similarities with the idea of addressing nonconvex machine learning problems directly, without resorting to convex approximations [37], [38].

Besides improving classifier robustness, finding a better enclosure of the legitimate class may, however, cause a higher number of legitimate samples to be misclassified as malicious. There is indeed a tradeoff between the desired level of robustness and the fraction of misclassified legitimate samples. The benefit of using randomization here is to make the attacker's strategy less pessimistic than in the case of static prediction games [10], [11], which should allow us to eventually find a better tradeoff between robustness and legitimate misclassifications. This aspect is investigated more systematically in the experiments reported in Section VI.

[5]Note that, on the contrary, manipulating samples directly in the input space would not even guarantee the existence of a Nash equilibrium, as the data generator's expected cost becomes nonquasi-convex in $\boldsymbol{x}$ for many (nonlinear) kernels, invalidating Theorem 1.

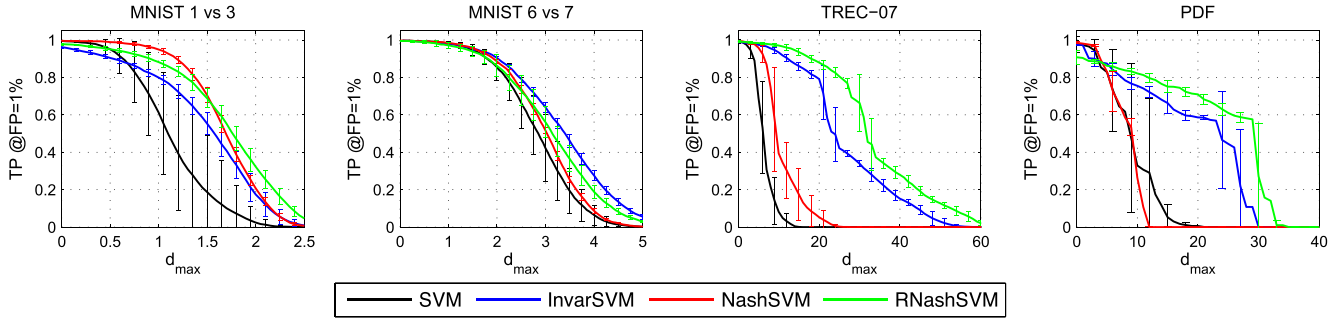

Fig. 2. Security evaluation curves, reporting the average TP at FP = 1% (along with its standard deviation, shown with error bars) against an increasing amount of manipulations to the attack samples (measured by $d_{\max}$), for handwritten digit (first and second plot), spam (third plot), and PDF (fourth plot) data.

VI. Experiments

In this section, we present a set of experiments on handwritten digit recognition, spam filtering, and PDF malware detection. Despite handwritten digit recognition is not a proper adversarial learning task as spam and malware detection, we consider it in our experiments to provide a visual interpretation of how secure learning algorithms are capable of improving robustness to evasion attacks.

We consider only linear classifiers, as they are a typical choice in these settings, and especially in spam filtering [2], [7], [14]. This also allows us to carry out a fair comparison with the state-of-the-art secure learning algorithms, as they yield linear classification functions. We compare our secure linear SVM learner (Section III) with the standard linear SVM implementation [40], and with the state-of-the-art robust classifiers invariant SVM (InvarSVM) [21], [22], and NashSVM [11] (see Section VII).

The goal of these experiments is to test whether these secure algorithms also work well under attack scenarios that differ from those hypothesized during design—a typical setting in security-related tasks, e.g., what happens if game-based classification techniques like that proposed in this paper and NashSVM are used against attackers that exploit a different attack strategy, i.e., attackers that may not act rationally according to the hypothesized objective function? What happens when the attacker does not play at the expected Nash equilibrium? These are important questions to address, as real-world attackers may not play according to the hypothesized objective function.

Security Evaluation: To address the above issues, we consider the security evaluation procedure proposed in [7]. It evaluates the performance of the considered classifiers under attack scenarios of increasing strength. We consider the true positive (TP) rate (i.e., the fraction of detected attacks) evaluated at 1% false positive (FP) rate (i.e., the fraction of misclassified legitimate samples) as performance measure. We evaluate the performance of each classifier in the absence of attack, as in standard performance evaluation techniques, and then start manipulating the malicious test samples to simulate attacks of different strengths. We assume a worst case adversary, i.e., an adversary that has perfect knowledge of the attacked classifier, since we are interested in understanding the worst case performance degradation. Note, however, that other choices are possible, depending on specific assumptions on the adversary's knowledge and capability [7], [13], [14]. In this setting, we assume that the optimal (worst case) sample manipulation $\mathbf{x}^*$ operated by the attacker is obtained by solving the following optimization problem:

$$\boldsymbol{x}^* \in \arg\min_{\boldsymbol{x}} \; yf(\boldsymbol{x};\boldsymbol{w}) \quad \text{s.t. } d(\boldsymbol{x},\hat{\boldsymbol{x}}_i) \leq d_{\max} \tag{32}$$

where y is the malicious class label and $d(\boldsymbol{x},\boldsymbol{x}_i)$ measures the distance between the perturbed sample $\boldsymbol{x}$ and the i^{th} malicious data sample $\hat{\boldsymbol{x}}_i$ (in this case, we use the ℓ_2 norm, as done by the considered classifiers). The maximum amount of modifications is bounded by $d_{\max}$, which is a parameter representing the attack strength. It is obvious that the more modifications the adversary is allowed to make on the attack samples, the higher the performance degradation incurred by the classifier is expected to be. Accordingly, the performance of more secure classifiers is expected to degrade more gracefully as the attack strength increases [7], [14].

The solution of the above problem is trivial when we consider linear classifiers, the Euclidean distance, and $\mathbf{x}$ is unconstrained: it amounts to setting $\boldsymbol{x}^* = \hat{\boldsymbol{x}}_i - y d_{\max}(\mathbf{w})/(||\mathbf{w}||)$. If $\mathbf{x}$ lies within some constrained domain, e.g., [0, 1], then one may consider a simple gradient descent with box constraints on $\mathbf{x}$ [9]. If $\mathbf{x}$ takes on binary values, e.g., {0, 1}, then the attack amounts to switching from 0 to 1 or vice versa the value of a maximum of $d_{\max}$ features which have been assigned the highest absolute weight values by the classifier. In particular, if $y\,w_k > 0$ ($y\,w_k < 0$) and the kth feature satisfies $\hat{x}_{ik} = 1$ ($\hat{x}_{ik} = 0$), then $x^*_k = 1$ ($x^*_k = 0$) [7], [14].

Parameter Selection: The considered methods require setting different parameters. From the learners' perspective, we have to tune the regularization parameter C for the standard linear SVM and InvarSVM, while we have ρ_{-1} and ρ_l for NashSVM and our method, respectively. In addition, the robust classifiers require setting the parameters of their attacker's objective. For InvarSVM, we have to set K, i.e., the number of modifiable features, while for NashSVM and our method, we have to set the value of the regularization parameter ρ_{+1} and ρ_d, respectively. Furthermore, to guarantee existence of a Nash Equilibrium point, we have to enforce some box constraints on the distribution's parameters. For the attacker, we restrict the mean of the attack points to lie in $[0, 1]$ (as the considered data sets are normalized in that interval), and their variance in $[10^{-3}, 0.5]$. For the learner, the variance of $\boldsymbol{w}$ is allowed to vary in $[10^{-6}, 10^{-3}]$, while its mean takes values on $[-W, W]$, where W is optimized together with the

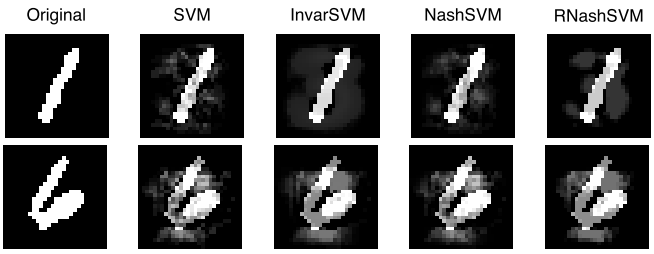

Fig. 3. Examples of obfuscated digits against each classifier when $d_{\max} = 2.5$ for 1 versus 3 (top row), and when $d_{\max} = 5$ for 6 versus 7 (bottom row).

other parameters. All the above-mentioned parameters are set by performing a grid search on the parameter space (C, ρ_{-1}, $\rho_d \in \{0.01, 0.1, 1, 10, 100\}$; $K \in \{8, 13, 25, 30, 47, 52, 63\}$; $\rho_{+1}, \rho_d \in \{0.01, 0.05, 0.1, 1, 10\}$; $W \in \{0.01, 0.05, 0.1, 1\}$), and retaining the parameter values that maximize the area under the security evaluation curve on a validation set. The reason is to find a parameter configuration for each method that attains the best average robustness over all attack intensities (values of $d_{\max}$), i.e., the best average TP rate at FP $= 1\%$.

A. Handwritten Digit Recognition

Similar to [21], we focus on two two-class subproblems of discriminating between two distinct digits from the MNIST data set [41], i.e., 1 versus 3, and 6 versus 7, where the second digit in each pair represents the attacking class ($y = +1$). The digits are originally represented as grayscale images of 28×28 pixels. They are simply mapped to feature vectors by ordering the pixels in raster scan order. The overall number of features is thus $d = 784$. We normalize each feature (pixel value) in $[0, 1]$, by dividing its value by 255. We build a training and a validation set of 1000 samples each by randomly sampling the original training data available for MNIST. As for the test set, we use the default set provided with this data, which consists of approximately 1000 samples for each digit (class). The results averaged on five repetitions are shown in the first and second plot of Fig. 2, respectively for 1 versus 3, and 6 versus 7. As one may notice, in the absence of attack (i.e., when $d_{\max} = 0$), all classifiers achieved comparable performance (the TP rate is almost 100% for all of them), due to a conservative choice of the operating point (FP $= 1\%$), that should also guarantee a higher robustness against the attack. In the presence of attack, our approach (RNashSVM) exhibits comparable performance with NashSVM on the problem of discriminating 1 versus 3, and to InvarSVM on 6 versus 7. NashSVM outperforms the standard SVM implementation in both cases, but exhibits lower security (robustness) than InvarSVM on 6 versus 7, despite the attacker's regularizer in InvarSVM is not even based on the ℓ_2 norm.

Finally, in Fig. 3, we report two attack samples (a digit from class 1 and one from class 6) and show how they are obfuscated by the attack strategy of (32) to evade detection against each classifier. Notice how the original attacking samples (1 and 6) tend to resemble more the corresponding attacked classes (3 and 7) when natively robust classifiers are used. This visual example confirms the analysis of Section V, i.e., that higher robustness is achieved when the adversary is required to mimic the feature values of samples of the legitimate class, instead of slightly modifying the attack samples to differentiate them from the rest of the malicious data.

B. Spam Filtering

In these experiments, we use the benchmark, publicly available, TREC 2007 email corpus [42], which consists of 75 419 real emails (25 220 legitimate and 50 199 spam messages) collected between April and July 2007. We exploit the bag-of-words feature model, in which each binary feature denotes the absence (0) or presence (1) of the corresponding word in a given email [7], [11], [13], [14]. Features (words) are extracted from training emails using the tokenization method of the widely known antispam filter SpamAssassin,[6] and then $n = 1000$ distinct features are selected using a supervised feature selection approach based on the information gain criterion [43]. We build a training and a validation set of 1000 samples each by randomly sampling the first 5000 emails in chronological order of the original data set, while a test set of about 2000 samples is randomly sampled from the subsequent set of 5000 emails. The results averaged on five repetitions are shown in the third plot of Fig. 2. As in the previous case, in the absence of attack ($d_{\max} = 0$), all the classifiers exhibit a very high (and similar) performance. However, as the attack intensity ($d_{\max}$) increases, their performance degrades more or less gracefully, i.e., their robustness to the attack is different. Surprisingly, one may notice that only InvarSVM and RNashSVM exhibited an improved level of security. The reason is that these two classifiers are able to find a more uniform set of weights than SVM and NashSVM, and in this case, this essentially requires the adversary to manipulate a higher number of features to significantly decrease the value of the classifier's discriminant function. Note that a similar result has also been heuristically found in [13] and [14].

C. PDF Malware Detection

We consider here another relevant application example in computer security, i.e., the detection of malware in PDF files. The main reason behind the diffusion of malware in PDF files is that they exhibit a very flexible structure that allows embedding several kinds of resources, including `Flash`, `JavaScript`, and even executable code. Resources simply consists of keywords that denote their type, and of data streams that contain the actual object, e.g., an embedded resource in a PDF file may be encoded as follows:

```
13 0 obj ≪ /Kids [ 1 0 R 11 0 R ]
/Type /Page... ≫ end obj
```

where keywords are highlighted in bold face. Recent work has exploited machine learning techniques to discriminate between malicious and legitimate PDF files, based on the analysis of their structure and, in particular, of the embedded keywords [44]–[48]. We exploit here a similar feature representation to that proposed in [45], where each feature denotes the presence of a given keyword in the PDF file. We collected 5993 recent malware samples from the Contagio data set,[7] and 5951 benign samples from the Web. Following the procedure described in [45], we extracted 114 keywords

[6] http://spamassassin.apache.org

[7] http://contagiodump.blogspot.it

from the first 1000 samples (in chronological order) to build our feature set. Then, we build training, validation, and test sets as in the spam filtering case, and the average results over five repetitions. Attacks in this case are simulated by allowing the attacker only to increase the feature values of malicious samples, which corresponds to adding the constraint $\boldsymbol{x} \geq \hat{\boldsymbol{x}}_i$ (where the inequality holds for all features) to Problem 32. The reason is that removing objects (and keywords) from malicious PDFs may compromise the intrusive nature of the embedded exploitation code, whereas adding objects can be easily done through the PDF versioning mechanism [9], [46], [48].

The results are shown in the fourth plot of Fig. 2. The considered methods mostly exhibit the same behavior shown in the spam filtering case, besides the fact that, here, there is a clearer tradeoff between the performance in the absence of attack, and robustness under attack. In particular, InvarSVM and RNashSVM are significantly more robust under attack (i.e., when $d_{\max} > 0$) than SVM and NashSVM, at the expense of a slightly worsened detection rate in the absence of attack (i.e., when $d_{\max} = 0$).

To summarize, the reported experiments show that, even if the attacker does not play the expected attack strategy at the Nash equilibrium, most of the proposed state-of-the-art secure classifiers are still able to outperform classical techniques, and, in particular, that the proposed RNashSVM classifier may guarantee an even higher level of robustness. Understanding how this property relates to the use of probability distributions over the set of the classifier's and the attacker's strategies remains an interesting future question.

VII. Related Work

The problem of devising secure classifiers against different kinds of manipulation of samples at test time has been widely investigated in previous work [1], [6], [10], [11], [21]–[27]. Inspired by the seminal work by Dalvi *et al.* [1], several authors have proposed a variety of modifications to existing learning algorithms to improve their security against different kinds of attack. Globerson and Roweis [21] and Teo *et al.* [22] have formulated the so-called InvarSVM in terms of a minimax approach (i.e., a zero-sum game) to deal with worst case feature manipulations at test time, including feature addition, deletion, and rescaling. This paper has been further extended in [23] to allow features to have different *a priori* importance levels, instead of being manipulated equally likely. Notably, more recent research has also considered the development of secure learning algorithms based on zero-sum games for sensor networks, including distributed secure algorithms [27] and algorithms for detecting adversarially corrupted sensors [26].

The rationale behind shifting from zero-sum to nonzero-sum games for adversarial learning is that the classifier and the attacker may not necessarily aim at maximizing antagonistic objective functions. This in turn implies that modeling the problem as a zero-sum game may lead one to design overly-pessimistic classifiers, as pointed out in [11]. Even considering a nonzero-sum Stackelberg game may be too pessimistic, since the attacker (follower) is supposed to move after the classifier (leader), while having full knowledge of the chosen classification function (which again is not realistic in practical settings) [11], [24]. For these reasons, Brückner *et al.* [10], [11] have formalized adversarial learning as a nonzero-sum game, referred to as static prediction game. Assuming that the players act simultaneously (conversely to Stackelberg games [24]), they devised conditions under which a unique Nash equilibrium for this game exists and developed algorithms for learning the corresponding robust classifiers, including the so-called NashSVM. This paper essentially extends this approach by introducing randomization over the players' strategies.

For completeness, we also mention here that in [25] Bayesian games for adversarial regression tasks have been recently proposed. In such games, uncertainty on the objective function's parameters of either player is modeled by considering a probability distribution over their possible values. To the best of our knowledge, this is the first attempt toward modeling the uncertainty of the attacker and the classifier on the opponent's objective function.

VIII. Conclusion

In this paper, we have extended the work in [11] by introducing randomized prediction games. To operate this shift, we have considered parameterized, bounded families of probability distributions defined over the set of pure strategies of either players. The underlying idea, borrowed from [3], [6], and [28], consists of randomizing the classification function to make the attacker select a less effective attack strategy. Our experiments, conducted on an handwritten digit recognition task and on realistic application examples involving spam and malware detection, show that competitive, secure SVM classifiers can be learned using our approach, even when the conditions behind uniqueness of the Nash equilibrium may not hold, i.e., when the attacker may not play according to the objective function hypothesized for her by the classifier. This mainly depends on the particular kind of decision function learned by the learning algorithm under our game setting, which tends to find a better enclosing of the legitimate class. This generally requires the attacker to make more modifications to the malicious samples to evade detection, regardless of the attack strategy chosen. We can thus argue that the proposed methods exhibit robustness properties particularly suited to adversarial learning tasks. Moreover, the fact that the proposed methods may also perform well when the Nash equilibrium is not guaranteed to be unique suggests us that the conditions behind its uniqueness may hold under less restrictive assumptions (e.g., when the SVM admits a unique solution [35], [36]). We thus leave a deeper investigation of this aspect to future work.

Another interesting extension of this paper may be to apply randomized prediction games in the context of unsupervised learning, and, in particular, clustering algorithms. It has been recently shown that injecting a small percentage of well-crafted poisoning attack samples into the input data may significantly subvert the clustering process, compromising the subsequent data analysis [49], [50]. In this respect, we believe that randomized prediction games may help devising secure countermeasures to mitigate the impact of such attacks, e.g., by explicitly modeling the presence of poisoning samples (generated according to a probability distribution chosen by the attacker) during the clustering process.

It is worth finally mentioning that this paper is also slightly related to previous work on security games, in which the goal of the defender is to adopt randomized strategies to protect his or her assets from the attacker, by allocating a limited number of defensive resources, e.g., police officers for airport security and protection mechanisms for network security [51]–[53]. Although our game is not directly concerned to the protection of a given set of assets, we believe that investigating how to bridge the proposed approach within this well-grounded field of study may provide promising research directions for future work, e.g., in the context of network security [52], [53], or for suggesting better user attitudes toward security issues [54]. This may also suggest interesting theoretical advancements, e.g., to establish conditions for the equivalence of Nash and Stackelberg games [51] and to address issues related to the uncertainty on the players' strategies, or on their (sometimes bounded) rationality, e.g., through the use of Bayesian games [25], security strategies, and robust optimization [52], [53]. Another suggestion to overcome the aforementioned issues is to exploit higher level models of the interactions between attackers and defenders in complex, real-world problems, e.g., through the use of replicator equations to model adversarial dynamics in security-related tasks [55]. Exploiting conformal prediction may also be an interesting research direction toward improving current adversarial learning systems [56]. To conclude, we believe that these are all relevant research directions for future work.

REFERENCES

[1] N. Dalvi, P. Domingos, S. Sanghai, and D. Verma, "Adversarial classification," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, Seattle, WA, USA, 2004, pp. 99–108.

[2] D. Lowd and C. Meek, "Adversarial learning," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, Chicago, IL, USA, 2005, pp. 641–647.

[3] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, "Can machine learning be secure?" in *Proc. Symp. Inf., Comput. Commun. Secur. (ASIACCS)*, New York, NY, USA, 2006, pp. 16–25.

[4] A. A. Cárdenas and J. S. Baras, "Evaluation of classifiers: Practical considerations for security applications," in *Proc. AAAI Workshop Eval. Methods Mach. Learn.*, Boston, MA, USA, Jul. 2006, pp. 1–6.

[5] P. Laskov and M. Kloft, "A framework for quantitative security analysis of machine learning," in *Proc. 2nd ACM Workshop Secur. Artif. Intell. (AISec)*, New York, NY, USA, 2009, pp. 1–4.

[6] L. Huang, A. D. Joseph, B. Nelson, B. I. P. Rubinstein, and J. D. Tygar, "Adversarial machine learning," in *Proc. 4th ACM Workshop Artif. Intell. Secur. (AISec)*, Chicago, IL, USA, 2011, pp. 43–58.

[7] B. Biggio, G. Fumera, and F. Roli, "Security evaluation of pattern classifiers under attack," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 4, pp. 984–996, Apr. 2014.

[8] B. Biggio *et al.*, "Security evaluation of support vector machines in adversarial environments," in *Support Vector Machines Applications*, Y. Ma and G. Guo, Eds. Cham, Germany: Springer, 2014, pp. 105–153.

[9] B. Biggio *et al.*, "Evasion attacks against machine learning at test time," in *Proc. Eur. Conf. Mach. Learn. Principles Pract. Knowl. Discovery Databases (ECML PKDD)*, vol. 8190. 2013, pp. 387–402.

[10] M. Brückner and T. Scheffer, "Nash equilibria of static prediction games," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 22, Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, Eds. Cambridge, MA, USA: MIT Press, 2009, pp. 171–179.

[11] M. Brückner, C. Kanzow, and T. Scheffer, "Static prediction games for adversarial learning problems," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 2617–2654, Jan. 2012.

[12] G. L. Wittel and S. F. Wu, "On attacking statistical spam filters," in *Proc. 1st Conf. Email Anti-Spam (CEAS)*, Mountain View, CA, USA, 2004, pp. 1–16.

[13] A. Kołcz and C. H. Teo, "Feature weighting for improved classifier robustness," in *Proc. 6th Conf. Email Anti-Spam (CEAS)*, Mountain View, CA, USA, 2009, pp. 1–8.

[14] B. Biggio, G. Fumera, and F. Roli, "Multiple classifier systems for robust classifier design in adversarial environments," *Int. J. Mach. Learn. Cybern.*, vol. 1, no. 1, pp. 27–41, Dec. 2010.

[15] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, "Semantics-aware malware detection," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2005, pp. 32–46.

[16] P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee, "Polymorphic blending attacks," in *Proc. 15th Conf. USENIX Secur. Symp. (USENIX-SS)*, Berkeley, CA, USA, 2006, pp. 241–256.

[17] B. Nelson *et al.*, "Exploiting machine learning to subvert your spam filter," in *Proc. 1st USENIX Workshop Large-Scale Exploits Emergent Threats (LEET)*, Berkeley, CA, USA, 2008, pp. 1–9.

[18] B. I. P. Rubinstein *et al.*, "ANTIDOTE: Understanding and defending against poisoning of anomaly detectors," in *Proc. 9th ACM Internet Meas. Conf. (IMC)*, New York, NY, USA, 2009, pp. 1–14.

[19] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 1807–1814.

[20] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning?" in *Proc. 32nd Int. Conf. Mach. Learn. (JMLR W&CP)*, vol. 37. 2015, pp. 1689–1698.

[21] A. Globerson and S. Roweis, "Nightmare at test time: Robust learning by feature deletion," in *Proc. 23rd Int. Conf. Mach. Learn.*, vol. 148. 2006, pp. 353–360.

[22] C. H. Teo, A. Globerson, S. Roweis, and A. J. Smola, "Convex learning with invariances," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 20, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds. Cambridge, MA, USA: MIT Press, 2008, pp. 1489–1496.

[23] O. Dekel, O. Shamir, and L. Xiao, "Learning to classify with missing and corrupted features," *Mach. Learn.*, vol. 81, no. 2, pp. 149–178, Nov. 2010.

[24] M. Brückner and T. Scheffer, "Stackelberg games for adversarial prediction problems," in *Proc. 17th ACM Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2011, pp. 547–555.

[25] M. Großhans, C. Sawade, M. Brückner, and T. Scheffer, "Bayesian games for adversarial regression problems," in *Proc. 30th Int. Conf. Mach. Learn. (JMLR W&CP)*, 2013, vol. 28. no. 3, pp. 55–63.

[26] K. G. Vamvoudakis, J. P. Hespanha, B. Sinopoli, and Y. Mo, "Detection in adversarial environments," *IEEE Trans. Autom. Control*, vol. 59, no. 12, pp. 3209–3223, Dec. 2014.

[27] R. Zhang and Q. Zhu, "Secure and resilient distributed machine learning under adversarial environments," in *Proc. 18th Int. Conf. Inf. Fusion*, Jul. 2015, pp. 644–651.

[28] B. Biggio, G. Fumera, and F. Roli, "Adversarial pattern classification using multiple classifiers and randomisation," in *Proc. 12th Joint IAPR Int. Workshop Struct. Syntactic Statist. Pattern Recognit.*, vol. 5342. 2008, pp. 500–509.

[29] I. L. Glicksberg, "A further generalization of the Kakutani fixed point theorem, with application to Nash equilibrium points," *Proc. Amer. Math. Soc.*, vol. 3, no. 1, pp. 170–174, Feb. 1952.

[30] J. B. Rosen, "Existence and uniqueness of equilibrium points for concave N-person games," *Econometrica*, vol. 33, no. 3, pp. 520–534, Jul. 1965.

[31] D. L. Zhu and P. Marcotte, "Modified descent methods for solving the monotone variational inequality problem," *Oper. Res. Lett.*, vol. 14, no. 2, pp. 111–120, Sep. 1993.

[32] P. T. Harker and J.-S. Pang, "Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications," *Math. Program.*, vol. 48, no. 1, pp. 161–220, Mar. 1990.

[33] C. Geiger and C. Kanzow, *Theorie und Numerik Restringierter Optimierungsaufgaben*. New York, NY, USA: Springer, 1999.

[34] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[35] S. Abe, "Analysis of support vector machines," in *Proc. 12th IEEE Workshop Neural Netw. Signal Process.*, 2002, pp. 89–98.

[36] C. J. C. Burges and D. J. Crisp, "Uniqueness of the SVM solution," in *Advances in Neural Information Processing Systems (NIPS)*, S. A. Solla, T. K. Leen, and K.-R. Müller, Eds. Cambridge, MA, USA: MIT Press, 1999, pp. 223–229.

[37] R. Collobert, F. Sinz, J. Weston, and L. Bottou, "Trading convexity for scalability," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2006, pp. 201–208.

[38] Y. Bengio and Y. LeCun, "Scaling learning algorithms towards AI," in *Large-Scale Kernel Machines*, L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, Eds. Cambridge, MA, USA: MIT Press, 2007.

[39] B. Schölkopf *et al.*, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1000–1017, Sep. 1999.
[40] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, 2011.
[41] Y. LeCun *et al.*, "Comparison of learning algorithms for handwritten digit recognition," in *Proc. Int. Conf. Artif. Neural Netw.*, 1995, pp. 53–60.
[42] G. V. Cormack. "Trec 2007 spam track overview," in *TREC*, E. M. Voorhees and L. P. Buckland, Eds. National Institute of Standards and Technology, 2007.
[43] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 27–66, Jan. 2012.
[44] C. Smutz and A. Stavrou, "Malicious PDF detection using metadata and structural features," in *Proc. 28th Annu. Comput. Secur. Appl. Conf. (ACSAC)*, New York, NY, USA, 2012, pp. 239–248.
[45] D. Maiorca, G. Giacinto, and I. Corona, "A pattern recognition system for malicious PDF files detection," in *Machine Learning and Data Mining in Pattern Recognition* (Lecture Notes in Computer Science), vol. 7376, P. Perner, Ed. Heidelberg, Germany: Springer, 2012, pp. 510–524.
[46] D. Maiorca, I. Corona, and G. Giacinto, "Looking at the bag is not enough to find the bomb: An evasion of structural methods for malicious PDF files detection," in *Proc. 8th ACM Symp. Inf., Comput. Commun. Secur. (ASIACCS)*, New York, NY, USA, 2013, pp. 119–130.
[47] N. Šrndić and P. Laskov, "Detection of malicious PDF files based on hierarchical document structure," in *Proc. 20th Annu. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, 2013, pp. 1–16.
[48] N. Šrndić and P. Laskov, "Practical evasion of a learning-based classifier: A case study," in *Proc. IEEE Symp. Secur. Privacy (SP)*, Washington, DC, USA, May 2014, pp. 197–211.
[49] B. Biggio, I. Pillai, S. R. Bulò, D. Ariu, M. Pelillo, and F. Roli, "Is data clustering in adversarial settings secure?" in *Proc. ACM Workshop Artif. Intell. Secur. (AISec)*, New York, NY, USA, 2013, pp. 87–98.
[50] B. Biggio *et al.*, "Poisoning complete-linkage hierarchical clustering," in *Structural, Syntactic, and Statistical Pattern Recognition* (Lecture Notes in Computer Science), vol. 8621, P. Fränti, G. Brown, M. Loog, F. Escolano, and M. Pelillo, Eds. Joensuu, Finland: Springer, 2014, pp. 42–52.
[51] D. Korzhyk, Z. Yin, C. Kiekintveld, V. Conitzer, and M. Tambe, "Stackelberg vs. Nash in security games: An extended investigation of interchangeability, equivalence, and uniqueness," *J. Artif. Intell. Res.*, vol. 41, no. 2, pp. 297–327, May 2011.
[52] T. Alpcan and T. Başar, *Network Security: A Decision and Game-Theoretic Approach*. Cambridge, U.K.: Cambridge Univ. Press, 2010.
[53] M. Tambe, *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge, U.K.: Cambridge Univ. Press, 2011.
[54] J. Grossklags, N. Christin, and J. Chuang, "Secure or insure?: A game-theoretic analysis of information security games," in *Proc. 17th Int. Conf. World Wide Web (WWW)*, New York, NY, USA, 2008, pp. 209–218.
[55] G. Cybenko and C. E. Landwehr, "Security analytics and measurements," *IEEE Security Privacy*, vol. 10, no. 3, pp. 5–8, May 2012.
[56] H. Wechsler, "Cyberspace security using adversarial learning and conformal prediction," *Intell. Inf. Manage.*, vol. 7, no. 4, pp. 195–222, Jul. 2015.

Samuel Rota Bulò (M'12) received the Ph.D. degree in computer science from the Ca' Foscari University of Venice, Venice, Italy, in 2009.

He has been a Post-Doctoral Researcher with the Ca' Foscari University of Venice since 2013. He is currently a Researcher with the Technologies of Vision Laboratory, Fondazione Bruno Kessler, Trento, Italy. He regularly publishes his research in well-recognized conferences and top-level journals mainly in the areas of computer vision and pattern recognition. He held a research visiting positions with Instituto Superior Técnico–Technical University of Lisbon, Lisbon, Portugal, the University of Vienna, Vienna, Austria, the Graz University of Technology, Graz, Austria, the University of York, York, U.K., Microsoft Research Ltd., Cambridge, U.K., and the University of Florence, Florence, Italy. His current research interests include areas of computer vision and pattern recognition with particular emphasis on discrete and continuous optimization methods, graph theory, game theory, and field of stochastic modeling.

Battista Biggio (M'07) received the M.Sc. degree (Hons.) in electronics engineering and the Ph.D. degree in electronics engineering and computer science from the University of Cagliari, Cagliari, Italy, in 2006 and 2010, respectively.

He has been with the Department of Electrical and Electronic Engineering, University of Cagliari, since 2007, where he is currently a Post-Doctoral Researcher. In 2011, he visited the University of Tübingen, Tübingen, Germany, where he was involved in the security of machine learning to training data poisoning. His current research interests include secure machine learning, multiple classifier systems, kernel methods, biometrics, and computer security.

Dr. Biggio is a member of the International Association of Pattern Recognition. He serves as a Reviewer for several international conferences and journals.

Ignazio Pillai received the M.Sc. degree (Hons.) in electronics engineering and the Ph.D. degree in electronics engineering and computer science from the University of Cagliari, Cagliari, Italy, in 2002 and 2007, respectively.

He has been with the Department of Electrical and Electronic Engineering, University of Cagliari, since 2003, where he holds a post-doctoral position with the Research Laboratory on Pattern Recognition and Applications. He has authored over 20 publications in international journals and conferences, and a Reviewer for several international conferences and journals. His current research interests include multilabel classification, multimedia document categorization, and classification with a reject option.

Marcello Pelillo (M'90–SM'04–F'13) is a Professor of Computer Science at Ca' Foscari University of Venice, Italy, where he directs the European Centre for Living Technology (ECLT). He held visiting research positions at Yale University, McGill University, the University of Vienna, York University (U.K.), the University College London, and the National ICT Australia (NICTA). He has published more than 200 technical papers in refereed journals, handbooks, and conference proceedings in the areas of machine learning, computer vision, and pattern recognition.

Prof. Pelillo is a fellow of the IAPR. He serves (has served) on the Editorial Boards of the journals, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *IET Computer Vision*, *Pattern Recognition*, and *Birain Informatics*, and he serves on the Advisory Board of the *International Journal of Machine Learning and Cybernetics*. He has served (serves) as a Guest Editor for various special issues of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *Pattern Recognition*, and *Pattern Recognition Letters*. He has initiated several conferences series as Program Chair (EMMCVPR, IWCV, and SIMBAD) and will serve as a General Chair for ICCV 2017. He has recently been appointed as IEEE SMC Distinguished Lecturer.

Fabio Roli (F'12) received the Ph.D. degree in electronics engineering from the University of Genoa, Genoa, Italy.

He was a Research Group Member of the University of Genoa from 1988 to 1994. He was an Adjunct Professor with the University of Trento, Trento, Italy, from 1993 to 1994. In 1995, he joined the Department of Electrical and Electronic Engineering, University of Cagliari, Cagliari, Italy, where he is currently a Professor of Computer Engineering and Head of the Research Laboratory on Pattern Recognition and Applications. He was a Very Active Organizer of the international conferences and workshops, and established the popular workshop series on multiple classifier systems. His current research interests include design of pattern recognition systems and their applications.

Dr. Roli is a fellow of the International Association of Pattern Recognition.