Statistical Science
2024, Vol. 39, No. 3, 449–468
https://doi.org/10.1214/24-STS922

Protecting Classifiers from Attacks

Víctor Gallego, Roi Naveiro, Alberto Redondo, David Ríos Insua and Fabrizio Ruggeri

Abstract. In multiple domains such as malware detection, automated driving systems, or fraud detection, classification algorithms are susceptible to being attacked by malicious agents willing to perturb the value of instance covariates to pursue certain goals. Such problems pertain to the field of adversarial machine learning and have been mainly dealt with, perhaps implicitly, through game-theoretic ideas with strong underlying common knowledge assumptions. These are not realistic in numerous application domains in relation to security and business competition. We present an alternative Bayesian decision theoretic framework that accounts for the uncertainty about the attacker's behavior using adversarial risk analysis concepts. In doing so, we also present core ideas in adversarial machine learning to a statistical audience. A key ingredient in our framework is the ability to sample from the distribution of originating instances given the, possibly attacked, observed ones. We propose an initial procedure based on approximate Bayesian computation usable during operations; within it, we simulate the attacker's problem taking into account our uncertainty about his elements. Large-scale problems require an alternative scalable approach implementable during the training stage. Globally, we are able to robustify statistical classification algorithms against malicious attacks.

Key words and phrases: Classification, Bayesian methods, adversarial machine learning, adversarial risk analysis, deep models.

1. INTRODUCTION

Over this and the last decade, an increasing number of processes is being automated through classification algorithms (Bishop, 2006). It is thus essential that these are robust in order to trust key operations based on their output. As a fundamental hypothesis, statistical classification relies on the use of independent and identically distributed (i.i.d.) data for both the training and operation phases. State-of-the-art classifiers perform extraordinarily well on such data, but they have proved vulnerable to various types of attacks targeted at fooling the underlying algorithms (Comiter, 2019, Micro, 2020). Security aspects of classification, which form part of the emerging field of *adversarial machine learning* (AML) (Vorobeichyk and Kantarcioglu, 2019, Joseph et al., 2019), question the i.i.d. hypothesis due to the presence of adversaries ready to modify the data to obtain a benefit, making training and operation distributions different. The societal relevance of the problem is well reflected in the recently proposed EU Artificial Intelligence (AI) Act (European Commission, 2022), NIST AI Risk Management Framework (NIST, 2022) and NIST-MITRE AML terminology (Tabassi et al., 2020).

Work in AML has traditionally focused around three topics: (a) analyzing *attacks* against machine learning (ML) algorithms to uncover their vulnerabilities; (b) delivering *defenses* against such attacks; and, (c) consequently, developing *frameworks* encompassing both attacks and defenses. While an important methodological pillar in the area (see, e.g., Joseph et al., 2019) has been classical robust statistics (Hampel et al., 1986), the predominant paradigm used to frame the confrontation between classification systems and adversaries has been, sometimes implicitly, game theory (see reviews in Biggio and Roli, 2018 and Zhou, Kantarcioglu and Xi, 2018). As examples, (a) the most popular attacks, including the fast gradient sign method (FGSM) (Goodfellow, Shlens and Szegedy, 2014), may be viewed in game-theoretic terms as best responses (i.e., maximizing the adversary's utility) to the classification algorithms; similarly, (b), one of

Víctor Gallego is PhD, Komorebi AI Technologies, Madrid, Spain. Roi Naveiro is Assistant Professor, CUNEF Universidad, Madrid, Spain (e-mail: roi.naveiro@cunef.edu). Alberto Redondo is PhD, Institute of Mathematical Sciences (ICMAT-CSIC), Madrid, Spain. David Ríos Insua is Research Professor, Institute of Mathematical Sciences (ICMAT-CSIC), Madrid, Spain. Fabrizio Ruggeri is Senior Fellow, CNR-IMATI, Milano, Italy.

the most promising defense techniques, *adversarial training* (AT) (Madry et al., 2018), may be framed through best response defenses against a worst-case attack (i.e., those maximizing the defender's utility under the worst-case attack); and, (c), finally, the pioneering framework in adversarial classification (AC) (Dalvi et al., 2004) was introduced as a game between a classifier and an adversary. This perspective typically entails strong common knowledge (CK) assumptions (Hargreaves-Heap and Varoufakis, 2004) which, from a fundamental point of view, are not sustainable in important application domains like security, defense, law enforcement or competitive business, as adversaries attempt to hide and conceal information.

Recent work (Naveiro et al., 2019) presented ACRA, a novel approach to AC based on Adversarial Risk Analysis (ARA) (Rios Insua, Rios and Banks, 2009). ARA makes operational a Bayesian approach to games (as in Kadane and Larkey, 1982 and Raiffa, 1982) facilitating procedures to predict adversarial decisions used to enhance the robustness of classifiers. However, ACRA may be used only with generative classifiers (Goodfellow, Bengio and Courville, 2016), like the utility sensitive Naive Bayes classifier (Chai et al., 2004) or certain deep variational autoencoder classifiers (Kingma et al., 2014). Moreover, only binary classification problems were supported in ACRA.

Thus, besides reviewing core AML ideas relevant in AC, we present a general Bayesian decision theoretic framework that may be used with both discriminative and generative classifiers, deals with multiple class problems and provide efficient computational schemes in large scale settings. In this, we not only solve a current complex problem with a sophisticated solution but also aim to bring the attention of the statistical community to a very relevant and largely unexplored issue: performing statistical inference in presence of adversaries. This problem area is not only a matter of academic interest but also a serious concern for individual and societal security, as expressed in the recent *Executive Order on the Safe, Secure, and Trustworthy Development and Use of Artificial Intelligence* (The White House, 2023). The dramatic increase in data availability has made classification an even more urgent need, as well as that of protecting from misclassification, either caused by intentional agents or by nature. ML techniques are getting increasingly relevant and, thus, AML is gaining track within the computer science community (not so much among statisticians). Thus, our purpose is to stir leveraging statistical techniques to tackle core problems in AML.

With this in mind, a broad overview of the general AC problem structures this paper according to the three core topics mentioned above. First (topic (a)), Section 2 provides a setup of the general AC problem, showcases two examples of how the performance of classifiers degrades in presence of subtle attacks, and overviews key attacks. Then (topic (b)), Section 3 overviews state-of-the-art defenses and suggests a general Bayesian solution to robustify classifiers against adversarial data manipulations. This approach changes the way classification decisions are made *during operations* and we illustrate its performance in a sentiment analysis problem; computational issues are discussed and an efficient alternative for large-scale problems is proposed, affecting the *training stage*, and modifying the way inferences are performed to take into account the eventual presence of adversaries during operations. It is illustrated through a case in computer vision based on a deep neural network classifier. Finally (topic (c)), a computational pipeline encompassing the whole framework is presented in Section 4. Code to reproduce all the experiments in the paper and illustrate the proposed pipeline is available at https://github.com/datalab-icmat/aml_bayes_classification.

2. ATTACKS OVER CLASSIFIERS

2.1 Basic Setup

Consider a classifier C (she) which may receive instances from k different classes designated with a label $y \in \{1, \dots, k\}$. Instances have covariates/features $x \in \mathbb{R}^d$. Notationwise, for convenience, we distinguish between random variables and realizations using upper and lower cases, respectively; thus $X = x$ refers to the originating covariates. Uncertainty about the instances' class given its covariates is modeled through a distribution $p(y|x)$, usually parameterized with certain parameters β. Such distribution can come from a generative model, for example, a naive Bayes one, where distributions $p(x)$ and $p(x|y)$ are explicitly modeled and $p(y|x)$ is obtained through Bayes formula; or from a discriminative model, for example, a neural network, in which $p(y|x)$ is modeled directly (Bishop, 2006). The β parameters are estimated using training data $\mathcal{D} = \{(x_i, y_i)_{i=1}^N\}$. In classical approaches, data $\mathcal{D}$ is used to find an estimate $\hat{\beta}$ that maximizes a certain likelihood (or minimizes a loss function $L(\beta, x, y)$) and $p(y|\hat{\beta}, x)$ is employed for classifying new instances. In Bayesian approaches, a prior $p(\beta)$ is used to compute the posterior $p(\beta|\mathcal{D})$ and the predictive distribution $p(y|x, \mathcal{D})$ is used to classify new data. Herein, $p(y|x)$ refers to either the predictive distribution of y when evaluating covariates x if we are in a Bayesian setup or $p(y_i|\hat{\beta}, x)$ if we are in the classical one. Other nonprobabilistic classifiers like support vector machines (SVMs) adapt as well to the above notation using calibration methods such as Platt et al. (1999) scaling scheme.

Whatever the estimation method adopted, C aims at classifying x to pertain to the class defined through $\arg\max_{y_C} \sum_{y=1}^k u_C(y_C, y) p(y|x)$, where $u_C(y_C, y)$ is

TABLE 1
Accuracy comparison (with precision) of four classifiers on clean and attacked data

Classifier	Clean data	Attacked data Attacker A	Attacked data Attacker B
Logistic Regression	0.728 ± 0.005	0.322 ± 0.011	0.418 ± 0.010
Naive Bayes	0.722 ± 0.004	0.333 ± 0.009	0.405 ± 0.009
Neural Network	0.691 ± 0.019	0.338 ± 0.021	0.417 ± 0.015
Random Forest	0.720 ± 0.005	0.327 ± 0.011	0.397 ± 0.013

the utility that she perceives when an instance with label y is classified as of class y_C, thus using the maximum expected utility principle (French and Rios Insua, 2000). Quite often, in the classification domain a $0-1$ utility function is used, so that the classifier gets utility 1 for a correct classification, and 0 for an incorrect one. In this case, the decision rule is $\arg\max_{y_C} p(y = y_C|x)$ and we thus aim to find the class that maximizes the probability of a correct classification.

In the scenario of interest, another agent, called adversary A (he), is involved. He applies an attack a to the features x leading to the transformed covariates $x' = a(x)$ actually received by C. The adversary aims to fool the classifier by making her misclassify instances to attain some benefit as it happens, for example, in spam detection, where, by crafting his message, a spammer aims to fool a spam detection tool so as to make her classify a spam message as legitimate to increase his business opportunities. Upon observing x', C needs to determine the instance class. An adversary unaware classifier might be making gross mistakes as she classifies based on the received, possibly modified, features x', instead of the actual ones, which are not observed. We provide two societally relevant examples that will drive our proposals in Sections 3.2 and 3.3, respectively.

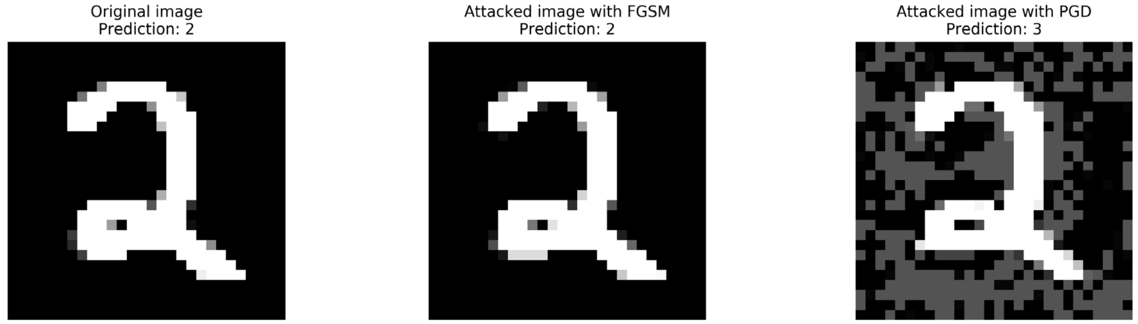
EXAMPLE 1. Consider a sentiment analysis problem. The goal is to assess whether a film review was positive or negative. We use a dataset containing 2400 IMDb reviews (1200 positive, 1200 negative) extracted from Kotzias et al. (2015). As covariates, we use 150 binary features indicating the presence or absence of the most common 150 words in the dataset after removing stopwords. A label indicates whether the review is positive ($y = 0$) or negative ($y = 1$).

We study the performance degradation of four standard statistical classification algorithms (*naive Bayes, logistic regression (LR), neural network (NN)* and *random forests*) under the actions of two types of attackers. For the first one, referred to as *attacker A*, the adversary aims to manipulate positive reviews in such a way that they are classified as negative, thus artificially decreasing the predicted quality of the film. The goal of the second adversary, denoted *attacker B*, is to manipulate negative reviews so as to make the classifier label them as positive ones, introducing bad reviews without being noticed. In both cases, we consider an attacker that modifies reviews by either adding or removing words. The perturbations have to be somehow restricted for the reviews to conceal their malicious intent. We do so by allowing at most two modifications per review.

Table 1 presents the accuracy of the four classifiers over clean and attacked test data; LR is applied with L2 regularisation (equivalent to performing maximum a posteriori (MAP) estimation in an LR model with a normal prior); the NN has two hidden layers. All models are trained under the same conditions: we randomly split the dataset into train and test subsets, respectively, with sizes 90%, 10%. Accuracy means and standard deviations are estimated via hold-out validation over 10 repetitions (Kim, 2009). Clearly, all four classifiers experience a considerable performance degradation that highlights their lack of robustness against adversarial attacks.

EXAMPLE 2. The second experiment is in the computer vision domain. It refers to a handwritten digit recognition problem based on a deep neural network classifier with $k = 10$ classes (1 per digit). The best-known attacks to classification algorithms in this domain, like the above mentioned FGSM, purposefully modify images so that alterations become imperceptible to the human eye, yet drive a model to misclassify the perturbed ones greatly degrading performance. These perturbed images are denominated *adversarial examples* (Szegedy et al., 2014).

For instance, with a relatively simple deep convolutional neural network (CNN) model (Krizhevsky, Sutskever and Hinton, 2012), we achieve 99% out of the sample accuracy when predicting the handwritten digits in the MNIST data set (LeCun, Cortes and Burges, 1998) with 28×28 pixels (thus, $d = 784$ features are used). However, if we perturb such a set with the FGSM attack, the accuracy gets drastically reduced to 62%. Moreover, if we use a more powerful attack such as the Projected Gradient Descent (PGD) method, accuracy rapidly decays to practically 0%. Figure 1 provides an example of (a) an original image, (b) a perturbed one with FGSM, and (c) a perturbed one with PGD. To our eyes, the three images look like a 2. However, with high probability our CNN



(a) Original image (b) FGSM-attacked image (c) PGD-attacked image

FIG. 1. *An original MNIST image identified as* 2. *The same image, perturbed with FGSM. Some pixels have changed, but it is still correctly classified. The same image, perturbed with PGD, is now incorrectly classified as a* 3.

classifier correctly identifies a 2 in the first two cases (Figures 1a and 1b), yet classifies as a 3 that under a PGD attack (Figure 1c).

A direct analog of this example in the automated driving system setup relates to the confounding of a yield sign with a stop sign (Caballero, Rios Insua and Naveiro, 2023), with potentially catastrophic consequences.

2.2 Overview on Attacks

The FGSM attack in Example 2 is defined through

$$x' = x + \epsilon \cdot \operatorname{sign}(\nabla_x L(\beta, x, y)),$$

where x' designates the attacked covariates; x are the original ones; $\nabla_x L(\beta, x, y)$ is the gradient of the loss function with respect to x; and ϵ is a small scalar reflecting attack intensity. Thus, the FGSM attack can be interpreted as a best response when the attacker can perturb each covariate by ϵ. Importantly, observe that this attack assumes that the adversary has precise knowledge of the underlying model and parameters of the involved classifier, what is designated a *white box* attack (Tabassi et al., 2020).

Another well-known white box attack is the PGD method (Madry et al., 2018) which iterates through the expression until the loss function plateaus

$$x_{t+1} = \operatorname{Clip}_{x,\epsilon}\{x_t + \alpha \cdot \operatorname{sign}(\nabla_x L(\beta, x_t, y_t))\}, \tag{1}$$

where α is the gradient step size and the Clip function forces the distance between the new instance covariates and their original values to be less than ϵ. Finally, Carlini and Wagner (2017) white box attack has gained popularity recently. It looks for perturbed instances $x' = x + \epsilon$, where the perturbation is obtained through

$$\min_{\epsilon} \|\epsilon\|_p + c \cdot f_y(x + \epsilon),$$

with $\|\cdot\|_p$ being the L_p norm and $f_y(x + \epsilon)$ is a function dependent on the loss $L(\beta, x, y)$ such that $f_y(x + \epsilon) \leq 0$ if and only if the output class for $x + \epsilon$ is the target class y. The positive constant c trades-off between minimizing the norm of the perturbation and maximizing its adversarial effect.

Notice that minimizing the most common loss functions in ML is equivalent to maximizing a certain posterior distribution since such loss functions can be typically written as

$$\sum_{i=1}^{N} L(\beta, x_i, y_i) = -\sum_{i=1}^{N} \log p(y_i | x_i, \beta) - \log p(\beta), \tag{2}$$

where $L(\beta, x_i, y_i)$ is the loss function evaluated at training instance (x_i, y_i). For instance, in a logistic regression setting with normal priors over the coefficients, finding their MAP estimate is equivalent to minimizing the cross-entropy loss with L_2 regularization. Thus, all the reviewed attacks can be seen as approximations of the optimization problem

$$x' = \arg \min_{z \in B(x)} \log p(y|z, \beta),$$

where $B(x)$ is some neighborhood of x over which the attacker has an influence. The exact solution to this problem is intractable in high-dimensional data and thus we resort to the above type of approximations based on gradient information. However, they all assume that the attacker has full knowledge of the target model, which is unrealistic in most security scenarios, except for the case of insiders (Joshi, Aliaga and Insua, 2021).

The type of attacks in Example 1 demands much less information from the defender model, but still some as it requires to know, for example, the features used by the classifier. Thus, it corresponds to a *gray box* attack (Tabassi et al., 2020) in which the classification system is partially known by the attacker. However, the entailed assumptions might be still debatable in most security domains.

Finally, there are *black box* attacks (Tabassi et al., 2020) which make minimal assumptions about the classifier. One example is the good word insertions attack against spam detectors in Naveiro et al. (2019) Another instance are the one-pixel attacks in Su, Vargas and Kouichi

(2017). The entailed assumptions are reasonable, yet their applicability is limited in numerous setups.

The above colored attack terminology is based on the knowledge that the attacker might have about the classifier. There are other taxonomies; for example, in Ríos Insua et al. (2018) adversaries are distinguished according to their capabilities or the type of problem they solve as *data fiddler*, *structural* and *parallel* attackers. Huang et al. (2011), Barreno et al. (2006) and Tabassi et al. (2020) provide further attack taxonomies based on attackers' goals, knowledge and capabilities and attack timing.

In this paper, we focus on exploratory attacks (defined to have influence just over operational data, leaving training data untainted). Moreover, we shall consider only integrity violations (the adversary just modifies malicious instances trying to make them be classified as legitimate ones). This is the most common case in numerous application domains. For instance, in fraud detection, fraudsters might modify the way their operations are done, trying to avoid them being classified as fraudulent by a classifier trained with clean data.

As a major driver of this paper, we shall emphasize modeling inherent uncertainty about attacks: in realistic settings, the classifier would not have precise information about how the attacker modifies a given instance, as, in general, his preference and probability assessments are not fully available. It seems therefore crucial to account explicitly for such uncertainty.

3. PROTECTING CLASSIFIERS FROM ATTACKS

3.1 Incorporating Defenses

Examples 1 and 2 showcased that an adversary unaware classifier may be purposefully fooled into issuing wrong classifications potentially incurring in severe expected utility underachievement. Assume for a moment that the classifier knows the attack a that she has suffered and that such attack is invertible, in the sense that she may recover the original x, designated $a^{-1}(x')$ when convenient. Then, rather than classifying based on $\arg\max_{y_C} \sum_{y=1}^{k} u(y_C, y)p(y|x')$, as an adversary unaware classifier would do, she should classify using

$$(3) \qquad \arg\max_{y_C} \sum_{y=1}^{k} u(y_C, y)p\big(y|x = a^{-1}(x')\big).$$

However, the classifier does not know the attack a, nor, more generally, the originating instance. The way around this issue entails constructing models of the attacks likely to be undertaken by the adversaries.

As the Introduction discussed, most of the previous work along these lines rely on game theoretic arguments and classical robust statistical concepts. Standard adversarial robustness definitions (see, e.g., Katz et al., 2017), aim at the condition

$$(4) \qquad \arg\max_{y} p(y|x, \beta) \approx \arg\max_{y} p\big(y|x', \beta\big),$$

entailing that for any input x and adversarial perturbation $x' \sim p(\cdot|x, y)$, the predicted class should not practically change under an adversarial attack. Optimization problems (4) can be stated by maximizing the log-likelihoods of the model under clear and attacked data, that is $\max \log p(y|x, \beta)$ and $\max \log p(y|x', \beta)$. In general, methods to increase model robustness against adversarial data manipulations fall into two main categories: those that affect operations, modifying the rule used to classify new instances, and those that alter the inferences made in the training phase to take into account the eventual presence of adversaries during operations.

The pioneering work of Dalvi et al. (2004) is an example of the first class of methods. This work assumes that the classifier has full knowledge of the adversary's problem. Next, the attacked covariates produced when the attacker receives a given instance can be computed exactly. Finally, during operations, provided that we observe x', the possible originating instances can be computed as those that, under the attack, lead to such observation.

The aim of the second class of robustifying approaches is to train using artificial data that somehow mimic actual, potentially attacked, operational data, through several heuristic approaches. Most of these methods model how the attacker would modify the instances in the training set. AT (Madry et al., 2018) is a mainstream proposal; instead of training a classifier to minimize the empirical risk based on training data, the authors minimize empirical risk under a worst-case attacker who chooses, for each instance, the worst modification within a constrained region. Having trained the classifier in this manner, $p(y|x')$ could be directly evaluated at the operational stage as this probability has been inferred taking into account the presence of an attacker. AT can be viewed as a zero-sum game where complete knowledge about the adversary is assumed, since the attacker has total knowledge of the model's loss function and parameters, overlooking existing uncertainty. Although AT strives for worst-case guarantees, it may be less effective if the real adversary behaves differently than the assumed scenario. Recall that a main theme within this paper is that, by introducing the uncertainties faced by the adversary and the defender, we can expand upon AT to arrive at more robust and principled defenses.

Similarly, Adversarial Logit Pairing (ALP) defenses (Kannan, Kurakin and Goodfellow, 2018) try to impose the stronger condition that the logits for clean and attacked instances are close,

$$(5) \qquad p(y|x, \beta) \approx p\big(y|x', \beta\big),$$

with improved robustness. This is done by including in the loss function minimized during training an extra term proportional to the absolute difference of the logits. Then problem to be solved is then $\min\{-\log p(y|x, \beta) +$

$\log p(y|x', \beta) + |f_\beta(x) - f_\beta(x')|\}$, with $f_\beta(x)$ being the logits of the model under input x. As with AT, these defenses assume models for how the attacker would modify training instances that do not take into account the existing uncertainty, as for each training instance x, x' is assumed to be available.

3.2 Robustifying Classifiers During Operation

This section analyses in detail the robustification of classifiers by modifying their operational stage. When observing instance x', we model our uncertainty about the latent originating instance x through a distribution $p(x|x')$ and perform inference about x proposing a formal way to sample from such distribution, thus accounting for the lack of knowledge about the attack. However, sampling from this distribution is harder, especially in large scale settings, and Section 3.3 proposes an approach that bypasses this harder step, modifying instead the training stage.

Suppose for the moment that we are actually able to model our uncertainty about the originating covariates x given the observed x' through a distribution $p(x|x')$ with support $\mathcal{X}_{x'}$, the set of reasonable instances x leading to x' if attacked. Based on (3), the expected utility that the classifier would get for a classification decision y_C would be

$$\psi(y_C) = \int_{\mathcal{X}_{x'}} \left(\sum_{y=1}^{k} u(y_C, y) p(y|x = a^{-1}(x')) \right) p(x|x')\, dx$$
$$= \sum_{y=1}^{k} u(y_C, y) \left[\int_{\mathcal{X}_{x'}} p(y|x = a^{-1}(x')) p(x|x')\, dx \right],$$

having to solve

$$\arg\max_{y_C} \psi(y_C). \tag{6}$$

Typically, we approximate expected utilities by Monte Carlo (MC) using a sample $\{x_n\}_{n=1}^N$ from $p(x|x')$ so that

$$\widehat{\psi}(y_C) = \frac{1}{N} \sum_{y=1}^{k} u(y_C, y) \left[\sum_{n=1}^{N} p(y|x_n) \right].$$

As a consequence, Algorithm 1 summarizes a general procedure for adversarial classification that we later specify.

To implement this approach, inference about the latent originating instance x given the observed x' must be undertaken. This entails estimating $p(x|x')$ or, at least, being able to sample from it. To do so, one must define an *attack model* $p(x'|x)$, that is, a model of our beliefs about how the attacker modified instance x, and sample from it. From a modeling perspective, this is involved as it requires strategic thinking about the adversary. Later, we suggest a formal Bayesian decision-theoretic argument to produce such a sample. For the moment, assume that such a procedure is available. Note that, if we could evaluate $p(x'|x)$ and $p(x)$ analytically, then sampling from $p(x|x')$ could be done using standard MCMC methods (French and Rios Insua, 2000). However, in general, estimating an *attacking model* is much harder than simulating from it.

Algorithm 1 ARA procedure for Adversarial Classification during operations

Input: N, training data $\mathcal{D}$.
Output: A classification decision $y_C^*(x')$.
Training
  Based on $\mathcal{D}$ estimate a model for $p(y|x)$.
End Training
Operation
  Read instance x'
  Draw sample $\{x_n\}_{n=1}^N$ from $p(x|x')$.

  Find $y_C^*(x') = \arg\max_{y_C} \frac{1}{N} \sum_{y=1}^{k} (u(y_C, y) \times [\sum_{n=1}^{N} p(y|x_n)])$

End Operation
Return $y_C^*(x')$

3.2.1 *A specification*: *AB-ACRA*. We propose AB-ACRA, an approach to sample from $p(x|x')$ making use of samples from $p(x'|x)$ by leveraging the information available about the attacker using concepts from approximate Bayesian computation (ABC) (Csilléry et al., 2010) and ARA (Banks, Rios and Ríos Insua, 2016). As basic ingredients, the approach requires sampling from $x \sim p(x)$ and $x' \sim p(x'|x)$.

Assume initially that x and, thus, x' are discrete. In this case, we could easily generate samples $p(X|X' = x')$ using MCMC including a rejection step. This would entail proposing a candidate $\tilde{x}$ according to certain transition distribution $q(x \to \tilde{x})$, sampling $\tilde{x}' \sim p(X'|X = \tilde{x})$ and, if the generated $\tilde{x}'$ is equal to the instance x' actually observed by the classifier, accept $\tilde{x}$ with probability $\alpha = \min\{1, \frac{p(\tilde{x})q(\tilde{x}\to x_i)}{p(x_i)q(x_i\to\tilde{x})}\}$. Using standard reversibility arguments in Metropolis–Hastings algorithms, it is straightforward to prove that samples generated iterating through these steps are approximately distributed according to $p(X|X' = x')$.

Note a few things. First, this approach requires us to evaluate $p(X)$. If this is not possible, but we can generate samples from such distribution, we could choose $p(X)$ to be the proposal generating density q. It can be easily seen that, in this case, the acceptance probability is just $\mathbb{I}[\tilde{x}' = x']$, thus avoiding the evaluation of $p(X)$. Indeed, one iteration of the previous scheme will produce a sample from the desired distribution, being an instance of a rejection sampler (Casella, Robert and Wells, 2004). Note that we would be generating $\tilde{x} \sim p(X)$, $\tilde{x}' \sim p(X'|X = \tilde{x})$, and accepting $\tilde{x}$ only if $\tilde{x}'$ coincides with the actually observed instance x'. It is straightforward to prove that $\tilde{x} \sim p(X|X' = x')$: think of this procedure as generating instances x and indicators I, where $I = 0(1)$ if we reject (accept) the sample; then, accepted instances are distributed according to the required distribution since

$$\begin{aligned} p(X = \tilde{x}|I = 1) &\propto p(I = 1|X = \tilde{x})p(X = \tilde{x}) \\ &\propto p(X' = x'|X = \tilde{x})p(X = \tilde{x}) \\ &\propto p(X = \tilde{x}|X' = x'). \end{aligned}$$

In general, however the convergence of this MCMC scheme will be slow, as just samples for which $\tilde{x}' = x'$ are accepted. Speed is clearly affected by the choice of q: densities that produce instances $\tilde{x}$ such that $p(X'|X = \tilde{x})$ placing a lot of mass around the observed x' will have better mixing. However, when x' is high dimensional the acceptance rate would be very low, as $p(X' = x'|X)$ will be generally very small. Moreover, in the continuous case, it will be $p(X' = x'|X) = 0$, canceling the acceptance rate.

We thus leverage ABC techniques (Martin, Frazier and Robert, 2024). For this, let us relax the condition $\tilde{x}' = x'$ in the rejection step of the previous MCMC scheme, and allow samples which are sufficiently close in the sense that $\phi(\tilde{x}', x') < \epsilon$ for a given distance function ϕ and tolerance ϵ. Observe though that the probability of generating samples for which $\phi(\tilde{x}', x') < \epsilon$ decreases as the dimension of x' increases. A common ABC solution replaces the acceptance criterion with the condition $\mathbb{I}[\phi(s(\tilde{x}'), s(x')) < \epsilon]$, where $s(x)$ is a set of summary statistics that capture the relevant information in x, the particular choice of s being problem specific (see examples for ϕ and s in the case study in 3.3.2). Obviously, if we use $\mathbb{I}[s(\tilde{x}') = s(x')]$ the approach is exact, provided that $s(x)$ is a sufficient statistic for X in $p(X'|X)$. Following standard MCMC convergence arguments, the induced Markov chain converges to the stationary distribution $p(X|\phi(s(\tilde{X}'), s(x')) < \epsilon)$. In general, choosing smaller values of ϵ will improve the approximation of our actual target $p(X|X' = x')$. Algorithm 2 illustrates the whole procedure. If the approximation level required entails dealing with a very small ϵ, the acceptance rate would drop, resulting again in poor mixing. A possibility to improve this consists of building a Markov chain in an augmented state-space (x, ϵ) (Bortot, Coles and Sisson, 2007). Values of x simulated using large values of ϵ are less reliable but the transition to such values can improve mixing.

To complete the specification, we still need to be able to sample from or evaluate $p(X)$ and sample from $p(X'|X)$.

Algorithm 2 MCMC-ABC sampler for $p(X|X' = x')$

Input: Instance x', distribution $p(X'|X)$, prior $p(X)$, transition density $q(x \to \tilde{x})$ tolerance ϵ, set of statistics s and distance ϕ.
Output: Samples approximately distributed according to $p(X|X' = x')$.
Initialize x_0, $i = 0$.
Repeat until convergence:
Propose $\tilde{x}$ according to transition distribution $q(x_i \to \tilde{x})$.
Sample $\tilde{x}' \sim p(X'|X = \tilde{x})$.
Compute

$$\alpha = \min\left\{1, \frac{p(\tilde{x})q(\tilde{x} \to x_i)}{p(x_i)q(x_i \to \tilde{x})}\mathbb{I}[\phi(s(\tilde{x}'), s(x')) < \epsilon]\right\}$$

With probability α set $x_{i+1} = \tilde{x}$, otherwise $x_{i+1} = x_i$.
Set $i = i + 1$.
End Repeat

Sampling from $p(X)$ and $p(X'|X)$. Estimating $p(X)$ is standard using training data, which is untainted by assumption. We just need a density estimation technique. For example, approximately sampling from this distribution could be done via bootstrapping from the training data. In addition, other techniques such as generative adversarial networks (Goodfellow et al., 2014), energy-based models (Grathwohl et al., 2019) or mixture models (Wiper, Rios Insua and Ruggeri, 2001) could be used. For high-dimensional X this sampling might be complex, partly motivating our approach in Section 3.3.

On the other hand, sampling from $p(X'|X)$ entails strategic thinking as we need to model how the adversary would modify the originating instance X. Any attacking model in the literature, including the ones sketched in Section 2.2, could be used, making our framework applicability very general. However, we propose a formal Bayesian decision-theoretic argument to produce samples from $p(X'|X)$, employing the ARA methodology to account explicitly for the uncertainty about the adversary's behavior. In particular, we identify two sources of uncertainty that are relevant for adversarial purposes: (a) adversarial attacks might not be deterministic as, for example, an attacker might choose to randomize between multiple different data transformations and, thus, we should consider *aleatory* uncertainty when modeling attacks; (b) another source of uncertainty (*epistemic*) stems from our lack of knowledge about the adversary. Herein, we show how the ARA methodology can be used to model both types of uncertainties.

With no loss of generality, assume that, out of the relevant k classes, the attacker deems as interesting the first l (call them *bad*), the other ones being irrelevant to him (*good*): he is interested in modifying data associated with instances belonging to the bad classes to make C believe that they belong to the good ones. As an example, consider a fraudster who may commit l types of fraud. He crafts the corresponding x to x' to make the classifier think that she has received a legitimate instance from class $y > l$. As we only consider integrity violations (that is, the adversary has just control over bad instances) we use the decomposition

$$p(x'|x) = \sum_{y=1}^{k} p(x'|x, y)p(y|x) = \sum_{y=1}^{l} p(x'|x, y)p(y|x) + \sum_{y=l+1}^{k} \mathbb{I}(x' = x)p(y|x).$$

Sampling from $p(y|x)$ is simple, as those probabilities can be estimated from training data. Therefore, we can obtain samples from $p(x'|x)$ by first generating from $y \sim p(y|x)$ and, then, if $y > l$ return x or sample $x' \sim p(x'|x, y)$, otherwise.

We still need a procedure to sample from $p(x'|x, y)$. Again, any attack model could be used here, but we employ a Bayesian decision theoretic framework to model the adversary's decision problem when he has available an instance x with label y, employing the ARA methodology. Assume the attacker is an expected utility maximizer trying to fool C. His utility function has the form $u_A(y_C, y)$, when the classifier says y_C and the actual label is y. The attacker should choose the feature modification maximizing his expected utility by making C classify instances as most beneficial as possible to him. With no loss of generality, assume the utility that A derives from the classifier's decision has the structure

$$u_A(y_C, y) = \begin{cases} 0 & \text{if } y \le l \text{and } y_C \le l, \\ u_A^{y_C,y} > 0 & \text{if } y \le l \text{ and } y_C > l, \\ 0, & \text{if } y > l \end{cases}$$

reflecting the fact that the attacker just obtains benefit when he makes the defender classify a bad instance as good. By transforming instance x with label $y \in \{1, \dots, l\}$ into x', the attacker would get an expected utility

$$\sum_{y_C=1}^{k} u_A(y_C, y)p_A(y_C|x') = \sum_{y_C=l+1}^{k} u_A^{y_C,y} p_A(y_C|x'), \tag{7}$$

with $p_A(y_C|x')$ describing the probability that C classifies the observed instance x' as y_C, from A's perspective. Thus, the adversary should craft instance x with label y into the attacked instance $x'(x, y)$ with

$$x'(x, y) = \arg\max_z \sum_{y_C=l+1}^{k} u_A^{y_C,y} p_A(y_C|z), \tag{8}$$

where the optimization is performed over the set of all possible modifications of instance x.

However, since typically we shall not have access to the adversary to completely elicit his preferences and beliefs, we model our epistemic uncertainty about $u_A^{y_C,y}$ and $p_A(y_C|z)$ in a Bayesian way with, respectively, random utilities $U_A^{y_C,y}$ and random probabilities $P_A(y_C|z)$ defined, with no loss of generality, over an appropriate common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ with atomic elements $\omega \in \Omega$ (Chung, 2001). This induces a distribution over the attacker's expected utility, where the random expected utility for him would be $\sum_{y_C=l+1}^{k} U_A^{y_C,y,\omega} P_A^{y_C,\omega}(z)$. In turn, the random optimal attack is defined through

$$X'_\omega(x, y) = \arg\max_z \sum_{y_C=l+1}^{k} U_A^{y_C,y,\omega} P_A^{\omega}(y_C|z),$$

and we can make $p(x'|x, y) = \mathcal{P}(X'_\omega(x, y) = x')$. Defined this way, our model for $p(x'|x, y)$ properly accounts for the existing uncertainty about the adversary. Note that, by construction, if we sample $u_A \sim U_A$ and $p_A(y_C|z) \sim P_A(y_C|z)$ and solve (8), $x'(x, y)$ would be distributed according to $p(x'|x, y)$. As mentioned, this is the last ingredient needed to implement our approach and, thus, completes our *adversarial modeling* framework to adversarial classification.

Observe that we have assumed that the attacker is an expected utility maximizer. Thus, if the data manipulation maximizing expected utility is unique, the generated attack (given the attacker's utility and probability) will be deterministic, in the sense that instance (x, y) will always lead to the same manipulated x'. Under the proposed framework, we could easily model a nondeterministic attacker that randomizes possible attacks. One possibility would be to model the attacker as an agent that randomly selects an attack in such a way that those with higher expected utility are more likely to be chosen. This would just require introducing an extra sampling step when simulating from $p(x'|x, y)$. Thus, it is possible to not only integrate epistemic uncertainty into our attacker modeling strategy but also to account for aleatoric uncertainty.

Models for random utilities and random probabilities. We describe now general guidelines that facilitate the specification of the remaining elements, that is models for random utilities U_A and random probabilities $P_A(y_C|z)$. Methodologically, these are prior assessment problems (Hanea et al., 2021); as a general statement, instead of a blind choice of hyperparameters, we try to use all the structural information that is available. Importantly, we can adjust the variance of the proposed prior assessments to reflect the reliability of the existing knowledge. Finally, we should submit the entire study to a sensitivity analysis, in particular with respect to the proposed priors (Ekin et al., 2023, Ríos Insua and Ruggeri, 2000).

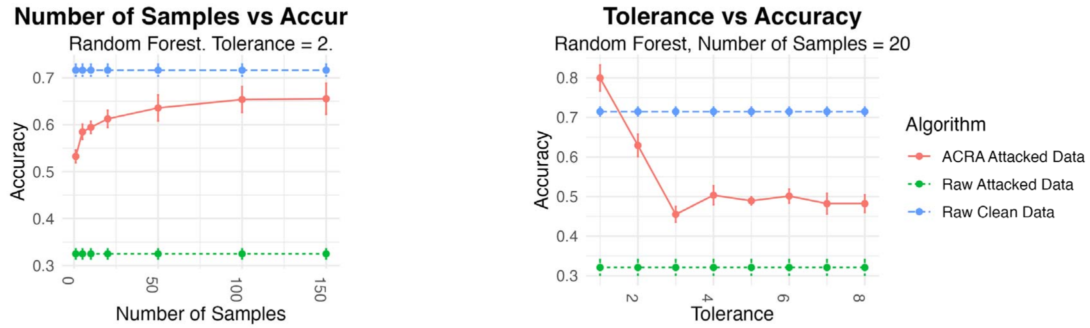
Consider first the random utilities U_A. Recall that, without loss of generality, we may scale utilities in our setup so that they have support $[0, 1]$ by just adopting an appropriate positive affine transformation based on the axiomatic properties of utility functions (French and Rios Insua, 2000). Among distributions with such support, a convenient choice due to their flexibility has been using $U_{y_C,y} \sim \text{Beta}(\alpha_{y_C,y}, \beta_{y_C,y})$ distributions. In particular, if no further additional information is available, we could use, for example, a uniform distribution (Yang and Berger, 1997). Rankings about perceived utilities are relatively easy to obtain and we just need to introduce them as constraints and sample from the random utilities by rejection. If eventually, further information about the likely values of the utilities is available, we may assess them through appropriate choices of $\alpha_{y_C,y}$ and $\beta_{y_C,y}$, using standard expert judgment elicitation approaches as in Morris, Oakley and Crowe (2014). This information can be incorporated by appropriately choosing the mean of the Beta distribution. In addition, we can regulate its variance while keeping the mean constant in order to reflect different levels of knowledge.

Modeling $P_A(y_C|x')$ is more delicate. It entails strategic thinking as C needs to model her opponent's beliefs about what classification she will adopt upon observing x'. Potentially, this leads to a hierarchy of recursive decision-making problems as the classifier needs *to think about what the attacker thinks about*... akin to level-k thinking in game theory (Stahl and Wilson, 1995), although we stop the entailed recursion at a level in which no more information is available when we introduce noninformative distributions over probabilities. We then solve recursively going down in the hierarchy by maximizing expected utility. This procedure is illustrated in Rios and Rios Insua (2012) in a simpler context. Here, we describe a level-2 modeling of $P_A(y_C|x')$ as a distribution based on $p(y_C|x')$ with some uncertainty around it. For this, consider the set $\mathcal{X}_{x'}$ of reasonable origins given the received x'. Since changing instances typically entails some cost for the adversary that increases with the number of features crafted, a reasonable choice is to define $\mathcal{X}_{x'}$ as the set of features x such that $\lambda(x, x') < \rho$ for a certain metric λ and threshold ρ. Next we consider an auxiliary distribution $p^*(x|x')$ over $\mathcal{X}_{x'}$; we adopt either a uniform distribution over $\mathcal{X}_{x'}$, or make $p^*(x|x') \propto [\lambda(x, x')]^{-1}$. Then, we define $\mu_{y_C} = \sum_x p(y_C|x)p^*(x|x')$, where $p(y_C|x)$ would come from estimates based on untainted training data. A Dirichlet distribution is used to model the behavior of the random vector encompassing all k possible classifications y_C given x'. The parameters will be chosen so that the mean for each y_C will coincide with μ_{y_C}. As a consequence of the properties of the Dirichlet distribution, the marginal distribution for each individual y_C will be $P_A(y_C|x') \sim \text{Beta}(\alpha^{y_C}, \beta^{y_C})$ having mean μ_{y_C} and a variance var_{y_C} properly chosen. To reduce the computational cost, we could approximate μ_{y_C} through $\frac{1}{M}\sum_{n=1}^{M} p(y_C|x_n)$, for a sample $\{x_n\}_{n=1}^{M}$ from $p^*(x|x')$.

3.2.2 *Case. Robustifying classification algorithms in sentiment analysis.* We illustrate the proposed approach with the sentiment analysis problem from Example 1. We test the performance of AB-ACRA as a defense mechanism against adversarial attacks, based on a 0–1 utility for the defender. Our discussion focuses on the random forest classifier there used, but we provide assessments for the other three classifiers in the example. As benchmark, recall that its accuracy over clean test data was 0.720 ± 0.005 (Table 1).

To compare AB-ACRA with raw RF on tampered data, let us simulate attacks over the instances in the test set using *attacker A*. For this, we solve problem (8) for each test review, removing the uncertainty that is not present from the adversary's point of view, restricting to attacks that involve changing at most the value of two of the words for each review. The utility that the attacker perceives when he makes the defender misclassify a review is 0.7. Finally, the adversary would have uncertainty about $p_A^{y_C}(x')$, as this quantity depends on the defender's decision. We test AB-ACRA against a worst case adversary who knows the actual value of $p(y_C|x_n)$ and estimates $p_A^c(x')$ through $\frac{1}{M}\sum_{n=1}^{M} p(y_C|x_n)$ for a sample $\{x_n\}_{n=1}^{M}$ from $p^*(x|x')$, with $M = 40$. For $p^*(x|x')$ we use a uniform distribution on the set of all instances at distance 1 from the observed x', using $\lambda(x, x') = \sum_{i=1}^{150} |x_i - x_i'|$ as distance. As we are in a binary classification setting the uncertainty about $p_A^{y_C}(x')$ and the attacker's utility function from the defender's perspective is modeled through beta distributions centered at the attacker's utility and probability values, with variances chosen to guarantee that the distribution is concave in its support. Otherwise, we would be believing that the attacker's utility and probability are peaked around 0 and 1 and low in between, which makes little sense in our context. For this, variances must be bounded from above by $\min\{[\mu^2(1-\mu)]/(1+\mu), [\mu(1-\mu)^2]/(2-\mu)\}$, were μ is the corresponding mean. The size of the variance will inform about the degree of knowledge the defender is assumed to have about the attacker. Reflecting a moderate lack of knowledge, we set the variance to be 10% of this upper bound. Thus, we are assuming a certain degree of knowledge about the adversary, as the expected values of the random utilities and probabilities coincide with the actual values used by the attacker. We later study how deviations from the assumed attacker behavior affect performance.

As summary statistic s for AB-ACRA, we use the 11 features (out of 150) having the highest permutation feature importance (Breiman, 2001). Given the discrete nature of the covariates, the distance used in the ABC scheme is Hamming's. Figure 2a compares the accuracy



(a) Experiment for a different number of samples. (b) Experiment for different tolerance values.

FIG. 2. *Accuracy comparison RF vs AB-ACRA.*

of AB-ACRA and RF for different sample sizes N in Algorithm 1, and tolerance $\epsilon = 2$. As we can see, AB-ACRA beats RF in tainted data with $N = 5$. The accuracy saturates quickly as we increase the number of samples: good performance is achieved with a relatively small sample size. Figure 2b plots the accuracy of AB-ACRA against RF for different tolerance ϵ values: as this parameter decreases, accuracy increases, albeit at a higher computational cost.

Tables 2 and 3 show average accuracies of the four classifiers from Example 1 robustified during operations against tainted data using attackers A and B, respectively. As can be seen, our approach allows us to mitigate performance degradation by showcasing the benefits of explicitly modeling the attacker's behavior in adversarial environments. Interestingly, in most cases, the classifiers even perform better on attacked data than the raw algorithm on clean data. This regularizing effect was mentioned in Goodfellow, Shlens and Szegedy (2014) for other algorithms and application areas.

The previous experiments quantified the defender's uncertainty about the attacker's utility and probability through beta distributions centered around the values actually employed by the adversary. In realistic settings, these values are often unknown and estimates are instead used. Therefore, it is natural to explore how performance is impacted when using estimates that may be potentially far from the truth. Our second batch of experiments tests the approach against an attacker whose utilities and probabilities differ from the baselines elicited by the defender. In particular, the attacker will deviate uniformly around the assumed probability and utility for each attack. The size of the deviation is constrained to be less than 25% the assumed value: if we center our beta distribution for, for example, the attacker's probability at value μ, the attacker will deviate from the assumed behavior in the range $(0.75 \cdot \mu, 1.25 \cdot \mu)$, with the upper bound truncated to 1 if this value is exceeded. Hence, in this experiment, our beta distributions will be centered around *wrong values*. We set the variance of the beta priors to be relatively high, 50% of the upper bound, and compare our approach with the CK one, in which the elements of the attacker are assumed to be known, and thus are point masses (on wrong values).

Tables 4 and 5 show average accuracies of the four classifiers on attacked data without defense (col. 2), the standard CK defense (col. 3) and, finally, our AB-ACRA defense (col. 4). Note first that the overall performance drops with respect to the results in Tables 2 (col. 4) and 3 (col 4.): when the attacker deviates from the assumed behavior, the performance recovery of both AB-ACRA and CK defenses is worse. Importantly, these results suggest as well that when the adversary deviates from the common knowledge assumption, AB-ACRA is as accurate as the common knowledge approach for certain classifiers and more accurate for others. This reflects that accounting for the uncertainty over the attacker elements is in-

TABLE 2
Accuracy comparison (with precision) of four classifiers with and without protection on clean and attacked data. Attacker A

Classifier	Clean data (raw)	Attacked data (raw) Attacker A	Attacked data (AB-ACRA) Attacker A
Logistic Regression	0.728 ± 0.005	0.322 ± 0.011	0.589 ± 0.023
Naive Bayes	0.722 ± 0.004	0.333 ± 0.009	0.968 ± 0.008
Neural Network	0.691 ± 0.019	0.338 ± 0.021	0.761 ± 0.030
Random Forest	0.720 ± 0.005	0.327 ± 0.011	0.837 ± 0.014

TABLE 3
Accuracy comparison (with precision) of four classifiers with and without protection on clean and attacked data. Attacker B

Classifier	Clean data (raw)	Attacked data (raw) Attacker B	Attacked data (AB-ACRA) Attacker B
Logistic Regression	0.728 ± 0.005	0.418 ± 0.010	0.840 ± 0.010
Naive Bayes	0.722 ± 0.004	0.405 ± 0.009	0.880 ± 0.027
Neural Network	0.691 ± 0.019	0.417 ± 0.015	0.700 ± 0.064
Random Forest	0.720 ± 0.005	0.397 ± 0.013	0.826 ± 0.012

deed beneficial when the attacker deviates from the assumed behavior. This experiment showcases the increase in robustness due to modeling uncertainty in scenarios in which CK assumptions are not realistic.

3.3 Robustifying Classifiers During Training

Case 3.2.2 allowed us to illustrate the framework during operations in a relatively complex setup with a moderate number (150) of binary features and constraints on the maximum (2) allowed number of changes. However, the scheme presented may entail heavy computational costs in large-scale scenarios as sampling from $p(x|x')$ gets costly, even becoming infeasible computationally in high-dimensional domains. As an example, images are typically represented as matrices of size *width* × *height* if in grayscale, or *width* × *height* × *channels* if multiple channels of color are used. Even for baselines such as MNIST in Example 2 with 28 × 28 pixels, the feature number grows fast with the dimensions.

To overcome this computational bottleneck, we shall argue that some of the steps from the approach in Section 3.2 may be skipped when dealing with differentiable classifiers. By this, we understand classifiers whose structural form $p(y|\beta, x)$ is differentiable with respect to the β parameters. A particularly relevant case is

$$p(y|\beta, x) = \text{softmax}\big(f_\beta(x)\big)[y], \qquad (9)$$
$$\text{where} \quad \text{softmax}(x)[j] = \frac{\exp x_j}{\sum_{i=1}^{k} \exp x_i},$$

which covers a large class of models. For example, if f_β is linear in inputs, we recover multinomial regression (MR) (McCullagh and Nelder, 1989); if we take f_β to be a sequence of linear transformations alternating nonlinear activation functions, such as Rectified Linear Units (ReLU), we obtain a feed-forward neural network (Gallego and Ríos Insua, 2022). These models are amenable to training through *stochastic gradient descent* (SGD) (Bottou, 2010). In particular, scalable optimization methods facilitate training deep neural networks (DNNs) with large amounts of high-dimensional data, like images, since they enable optimization using only mini-batches at each iteration.

Importantly, instead of dealing with the attacker in the operational phase, as in Section 3.2, we shift to modifying the training phase to account for future adversarial perturbations. To enable so, we require the ability to draw posterior samples from the posterior distribution $p(\beta|\mathcal{D})$.With this paradigm shift, we avoid the expensive step of sampling from $p(x|x')$, only requiring to do it from $p(x'|x)$ using gradient information from the defender model. Note that it is much easier to estimate $p(x'|x)$ from an adversary, just requiring an opponent model, than to estimate $p(x|x')$, which requires inverting such opponent model. Obviously, since there is no gradient notion in every model, we would need to resort to the approach in Section 3.2 in those cases.

For clarity, let us use $0-1$ utilities, although extensions to more general utilities follow a similar path. We thus focus on implementing the decision rule

$$\arg\max_{y_C} \iint p(y_C|x, \beta)\, p\big(x|x'\big)\, p(\beta|\mathcal{D})\, dx\, d\beta \qquad (10)$$

in a general scalable and robust manner.

TABLE 4
Accuracy comparison (with precision) of four classifiers on tainted data with no defense, CK defense, and AB-ACRA defense. Attacker A. Best result in boldface

Classifier	Attacked data (raw) Attacker A	Attacked data (CK) Attacker A	Attacked data (AB-ACRA) Attacker A
Logistic Regression	0.315 ± 0.007	0.499 ± 0.008	$\mathbf{0.513} \pm 0.008$
Naive Bayes	0.325 ± 0.007	0.645 ± 0.025	$\mathbf{0.665} \pm 0.024$
Neural Network	0.389 ± 0.024	0.592 ± 0.032	$\mathbf{0.638} \pm 0.030$
Random Forest	0.313 ± 0.009	$\mathbf{0.720} \pm 0.013$	0.710 ± 0.017

TABLE 5

Accuracy comparison (with precision) of four classifiers on tainted data with no defense, CK defense, and AB-ACRA defense. Attacker B. Best result in boldface

Classifier	Attacked data (raw) Attacker B	Attacked data (CK) Attacker B	Attacked data (AB-ACRA) Attacker B
Logistic Regression	0.412 ± 0.004	0.713 ± 0.008	$\mathbf{0.760} \pm 0.011$
Naive Bayes	0.406 ± 0.008	0.783 ± 0.039	$\mathbf{0.800} \pm 0.035$
Neural Network	0.437 ± 0.014	$\mathbf{0.727} \pm 0.050$	0.725 ± 0.052
Random Forest	0.402 ± 0.005	0.779 ± 0.011	$\mathbf{0.782} \pm 0.008$

3.3.1 *Protecting differentiable classifiers at training*. Beyond usual adversarial robustness conditions such as (4) and (5), our proposal will require the slightly stronger condition

$$p(y, x|\beta) \approx p(y, x'|\beta). \tag{11}$$

As we shall see, though this imposes some extra computational burden, it motivates a scalable training scheme that improves the robustness of the classifier.

To start with, to compute $p(y, x|\beta)$, we reinterpret its logits as in energy-based models (Grathwohl et al., 2019), leading to the expression $p(y, x|\beta) = \frac{\exp\{f_\beta(x)[y]\}}{Z(\beta)}$, where $Z(\beta)$ is the usually intractable normalizing constant. Now, by factoring the joint distribution $p(y, x|\beta)$ as $p(y|x, \beta)p(x|\beta)$, then for a sample $x \sim \mathcal{D}$ and the corresponding adversarial perturbation $x' \sim p(x'|x)$, consider maximizing the objective function $\mathcal{L}(\beta, x, y)$ with

$$\begin{aligned} \mathcal{L}(\beta, x, y) = \{&[\log p(y|x, \beta) + \log p(y|x', \beta)] \\ &- |f_\beta(x) - f_\beta(x')| \\ &- |\log p(x|\beta) - \log p(x'|\beta)|\}. \end{aligned} \tag{12}$$

The first two terms in (12) promote high predictive power for both $p(y|x, \beta)$ and $p(y|x', \beta)$. In turn, the third one encourages the logits of x and its corresponding perturbed sample x' to be similar, so that $p(y|x, \beta) \approx p(y|x', \beta)$. Finally, the last term acts as a new regularizer, encouraging $p(x|\beta) \approx p(x'|\beta)$ and, together with the previous term, leads to condition (11). Interestingly, note that since in this third term we assess the difference $\log p(x|\beta) - \log p(x'|\beta)$ of the originally intractable terms, the normalizing constant $Z(\beta)$ cancels out, rendering tractable the analysis in (12).

As a consequence, at the end of the training, by maximizing (12) we would expect that $p(x|\beta) \approx p(x'|\beta)$. Therefore, using Bayes formula, we would expect that $p(x|x') \approx p(x'|x)$. Then, considering that $p(y|x, \beta) \approx p(y|x', \beta)$, we swap the original decision rule (10) by

$$\arg\max_{y_C} \iint p(y_C|x', \beta)p(x'|x)p(\beta|\mathcal{D})\,dx'\,d\beta. \tag{13}$$

Note that since the observed input x' might be tainted, it is not necessary to attack it via $p(x'|x)$ anymore and just suffices to use the test time decision rule, $\arg\max_{y_C} p(y_C|x', \beta)$.

To sum up, by promoting $p(x|\beta) \approx p(x'|\beta)$ during training, we learn a robust model for which starting in either x or x', it does not matter whether we use $p(x'|x)$ or $p(x|x')$ as we arrive at the same distribution, being much simpler to sample from $p(x'|x)$ than from $p(x|x')$, as explained below. We emphasize that (12) is more than just a mash-up of AT, ALP, and our new regularizer, with certain computational advantages. In addition, (12) can be seen as an augmented model likelihood, in which we expand the original model likelihood $p(y|x, \beta)$ with extra regularizers to improve adversarial robustness.

The next paragraphs apply the ARA methodology to add a layer of uncertainty over the previous terms with two objectives: (i) enabling departure from the above mentioned standard CK assumptions typical in AC; and (ii) enhancing robustness and preventing overfitting. Indeed, based on ARA, we acknowledge the two sources of uncertainty that motivated our interest in AC and bring in further realism to the proposed analysis: the defender lacks full knowledge about the specific attack employed by her adversary and the latter usually does not have full knowledge of the model he desires to attack.

To address the first source, instead of performing an optimization to arrive at a single point as in for example, (1), we replace SGD with an SG-MCMC sampler such as stochastic gradient Langevin dynamics (SGLD) (Welling and Teh, 2011) to sample from regions with high adversarial loss, thus being proportional to $\exp\{-\log p(y|x, \beta)\}$. This leads to iterations

$$x_{t+1} = x_t - \epsilon_t \operatorname{sign} \nabla_x \log p(y|x_t, \beta) + \xi_t$$

with $\xi_t \sim \mathcal{N}(0, 2\epsilon_t)$ and $t = 1, \dots, T$, where ϵ_t are step sizes that decay to zero following the usual Robbins and Monro (1951) convergence conditions. We also consider uncertainty over the hyperparameters ϵ_t (from a Gamma distribution, or better a re-scaled Beta, since too high or too low learning rates are futile) and the number T of iterations (from a Poisson). In addition, we can consider mixtures of attacks, for instance, by sampling a Bernoulli random variable and, then, choosing the gradient corresponding to either FGSM or another attack from those

Algorithm 3 Large scale attack simulation

Input: Defender model $p(y|x,\beta)$, a set of K particles $\{\beta_i\}_{i=1}^K$ and attacker model $p(x'|x)$ (in this example, FGSM with multiple rounds).
Output: A set of adversarial examples $\{x_i\}_{i=1}^K$ from attacker model.
Sample $T \sim p(T)$
Sample $\epsilon \sim p(\epsilon)$
for each attack iteration t from 1 to T **do**
  $x_{i,t+1} = x_{i,t} - \epsilon_t \operatorname{sign}\nabla \log p(y|x_{i,t},\beta) + \mathcal{N}(0, 2\epsilon I)$
end for
Return $x_i = x_{i,T}$

Algorithm 4 Large scale ARA training

Input: Defender model $p(y|x,\beta)$ and attacker model $p(x'|x)$.
Output: A set of K particles $\{\beta_i\}_{i=1}^K$ approximating the defender posterior learned using ARA training.
for each training iteration t **do**
  sample $x_1,\dots,x_K \sim p(x'|x)$ using Alg. 3
  $\beta_{i,t+1} = \beta_{i,t} + \epsilon\nabla(\mathcal{L}(\beta_{i,t}, x_i, y) + \log p(\beta_{i,t})) + \mathcal{N}(0, 2\epsilon I)$ for each i (SGLD)
end for
return $\beta_i = \beta_{i,T}$

in Section 2.2. Algorithm 3 generates K adversarial examples that take into account the uncertainty over the attacker's model and are amenable to large-scale settings.

Concerning the second source of uncertainty, since the attacker may not know the actual $p(y|x,\beta)$, our model for his behavior takes into account the uncertainty over β. A first possibility considers an augmented model $p(y|x,\beta,\gamma)$ with $\gamma \sim \mathcal{B}er(p)$. Then, for example, if $\gamma = 0$, $p(y|x,\beta,0)$ may be given by MR, whereas if $\gamma = 1$, $p(y|x,\beta,1)$ is a neural network. This would reflect the lack of information that the attacker has about the architecture he is targeting. The case can be straightforwardly implemented as an ensemble model (Hastie, Tibshirani and Friedman, 2009), performing simulated attacks over it. Alternatively, β may have continuous support. In the case of a NN, this would reflect that the attacker has uncertainty over the parameter values (Müller and Insua, 1998). This can be implemented using scalable Bayesian approaches in deep models, such as SG-MCMC schema (Ma, Chen and Fox, 2015). To this end, we propose the defender model to be trained using SGLD, obtaining posterior samples via the iteration $\beta_{t+1} = \beta_t + \eta\nabla_\beta(\mathcal{L}(\beta_t, x, y) + \log p(\beta)) + \mathcal{N}(0, 2\eta I)$, with objective $\mathcal{L}(\beta, x, y)$ as in (12) and sampling x' using $p(x'|x)$ as in the previous paragraph. Note that $p(\beta)$ is the prior placed over the model's parameters, which in the case of differentiable classifiers such as deep neural networks is typically a zero-centered Gaussian with scale in the order of 0.01. Algorithm 4 employs the previous perturbations to robustly train the classifier using ARA principles.

Observe that to sample from the posterior $p(\beta|\mathcal{D}_{\text{att}})$, where $\mathcal{D}_{\text{att}}$ designates the attacked dataset, the Langevin SG-MCMC sampler can be written as

$$\beta_{t+1} = \beta_t + \epsilon\nabla \log p(\beta|\mathcal{D}_{\text{att}}) + \mathcal{N}(0, 2\epsilon I).$$

Since $\log p(\beta|\mathcal{D}_{\text{att}}) = \log p(\mathcal{D}_{\text{att}}|\beta) + \log p(\beta) - \log p(\mathcal{D}_{\text{att}})$ and, taking gradients wrt to β, we have $\nabla \log p(\beta|\mathcal{D}_{\text{att}}) = \nabla \log p(\mathcal{D}_{\text{att}}|\beta) + \nabla \log p(\beta)$. The previous sampler uses the log-likelihood under the attacked data. To further robustify it, we can replace that likelihood with the objective (12), that also includes the likelihood over the clean samples and the regularization term for improved adversarial robustness, in the sense of condition (11). Assume we work with a set of K samples $\{\beta_i\}_{i=1}^K$ from the defender posterior, then an unbiased estimator of the gradient is obtained by sampling a minibatch $\{x_i\}$ of attacked points using Algorithm 3, leading to the sampler

$$\beta_{i,t+1} = \beta_{i,t} + \epsilon\nabla\big(\mathcal{L}(\beta_{i,t}, x_i, y) + \log p(\beta_{i,t})\big) + \mathcal{N}(0, 2\epsilon I).$$

Finally, Algorithm 5, to be compared with Algorithm 1, integrates and aggregates the general procedure to robustify a classifier in a scalable manner.

3.3.2 *Case. Robustifying deep neural networks in computer vision.* We apply the proposed approach to two mainstream datasets in computer vision, MNIST and CIFAR-10, showcasing its benefits via experiments.

In the first experiment, MNIST, the defender aims to classify digits (from 0 to 9) in presence of adversarial attacks, recall Example 2. The underlying classifier is a two-layer feed-forward neural network with ReLU activations and a final softmax layer to get the predictions over the 10 classes (Gallego and Ríos Insua, 2022). The net is trained using SGD with momentum 0.5 for 5 epochs, learning rate of 0.01, and batch size of 32. The training set corresponds to 50,000 digit images and results are reported over a 10,000 digit test set. We use both of the uncertainties (attacker and defender) mentioned above, except that we do not adopt mixtures of different attacks or different models, focusing on a single-attacker setup.

Figure 3 shows the effect of an attack of increasing strength. Note that whereas the first FGSM attack fails to change the predicted label, the stronger FGSM attack successfully makes the network predict an incorrect label. Figure 4 plots the *security evaluation curves* (Biggio and Roli, 2018) for three different defenses (AT, ALP, and our ARA-based proposal) and an undefended model (NONE in the legends) under the MNIST dataset, using two attacks at test time: FGSM (left) and PGD (right). Such curves depict the accuracy of the defender model (y-axis),

Algorithm 5 ARA procedure for Adversarial Training of Differentiable Classifiers

Input: training data $\mathcal{D}$, prior $p(\beta)$.
Output: A classification decision $y_C^*(x')$.
Training
  Use Algorithm 4 to obtain an approximation of $p(\beta|\mathcal{D})$ (robustified posterior).
End Training
Operation
  Read instance x'
  Find $y_C^*(x') = \arg\max_{y_C} \sum_{y=1}^{k} u(y_C, y) \int p(y|x', \beta)\, p(\beta|\mathcal{D})\, d\beta$
End Operation
return $y_C^*(x')$

under different attack intensities ϵ (x-axis), with larger attack intensities implying more powerful attacks, as Figure 3 exemplifies.

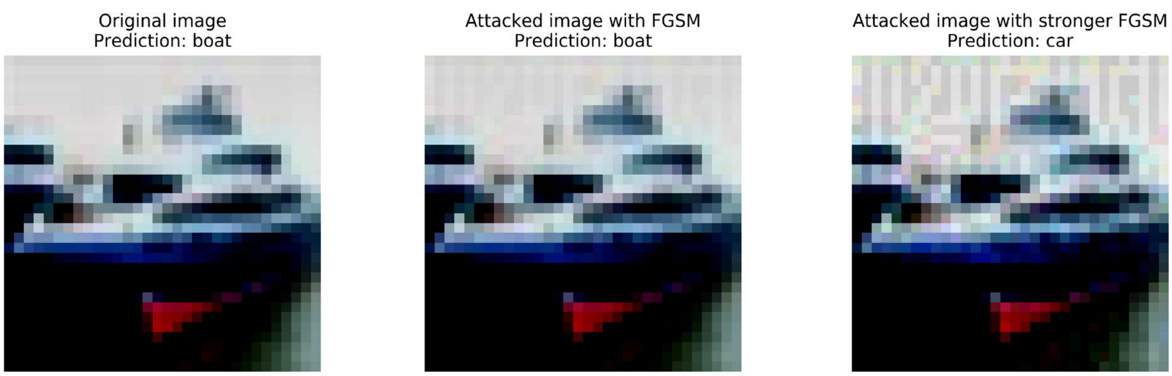
Inspection of Figure 4 immediately reveals noteworthy patterns. At low attack intensities, all four defenses perform comparably, although AML defenses appear to yield marginally lower accuracies. However, as attack intensity increases, the performance of the undefended technique rapidly deteriorates illustrating the relevance of AML defenses. The undefended model's accuracy on the untainted data is 98% and quickly degrades to 75% under FGSM attack at an intensity of 0.1. The three robustified approaches mitigate this degradation, with varying degrees of success. The AT and ALP defenses perform comparably under FGSM; however, the AT defense degrades quicker under the PGD attack. The ARA approach generally appears more robust than the AT and ALP defenses at much higher intensities for both attacks. For example, a 0.1 FGSM-attack intensity induces AT and ALP accuracies of 90% but an ARA accuracy of 92%; this difference increases further with FGSM-attack intensity. The disparities are even more pronounced under the PGD attack. Although such attacks degrade all defenses, higher attack intensities are required to affect the ARA defense than the AT and ALP defenses. Thus, Figure 4 suggests that the uncertainties provided by the ARA training method substantially improve the robustness of the neural network under two different attacks with that provided by state-of-the-art AT and ALP defenses. Note also that robustifying the model against attacks is essential as its performance rapidly deteriorates.

We also compute the *energy gap*

$$\Delta E := \mathbb{E}_{x\sim\mathcal{D}}[-\log p(x)] - \mathbb{E}_{x'\sim\mathcal{D}'}[-\log p(x')]$$

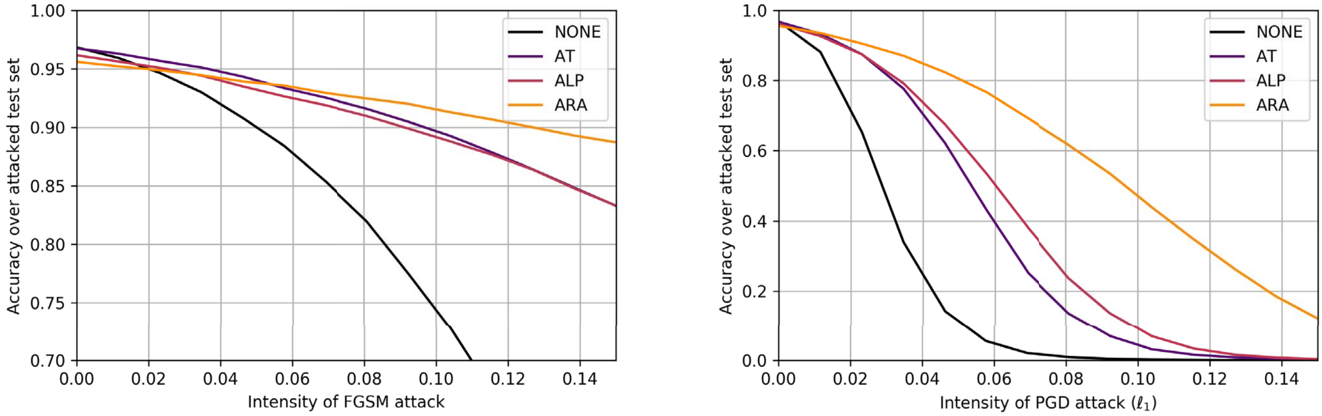
(using Eqs. (6) and (7) in Grathwohl et al. (2019)) for a given test set $\mathcal{D}$ and its attacked counterpart $\mathcal{D}'$ under the PGD attack. This serves as a proxy to measure the degree of fulfillment of our enabling assumption (11). We obtain that $\Delta E_{\text{None}} = 2.204$, $\Delta E_{\text{AT}} = 1.763$, and $\Delta E_{\text{ARA}} = 0.070$. The ARA version thus reduces the gap with respect to their counterparts, getting closer to the desired adversarial assumption that a robust model should fulfill ($p(x) \approx p(x')$), having a clear regularization effect which contributes to our enhanced ARA approach.

Figure 5 displays the same setting, analysing the protection from attacks over the classic CIFAR-10 dataset, which includes 60,000 32×32 color images (thus, of dimension 3072) in 10 classes. A much more complex



(a) Original image (no attack) (b) Perturbed image under FGSM intensity 0.01 (c) Perturbed image under FGSM intensity 0.04

FIG. 3. *A figure sample from CIFAR nonattacked (a) and under two increasingly intensive attacks (b and c).*



(a) FGSM attack. (b) PGD attack under ℓ_1 norm.

FIG. 4. *Robustness of deep network for MNIST under three defense mechanisms* (*ARA*, *AT*, *ALP*) *and two attacks* (*FGSM*, *PGD*).

architecture, a deep residual network consisting of 18 layers (He et al., 2016, Gallego and Ríos Insua, 2022), is required for classification purposes. A similar picture emerges justifying the need for robustifying classifiers against adversarial attacks and our ARA improving upon AT and ALP defenses. Observe though, Figure 5 b, that PGD attacks stress considerably the defenses in such a complex problem.

4. A PIPELINE FOR ADVERSARIAL CLASSIFICATION

We end up with the third major issue which refers to global frameworks for AML. Due to its historical significance, we start by sketching Dalvi et al. (2004) pioneering approach to enhance classification algorithms when an attacker is present, adapting it to our notation (although they only focus on the utility-sensitive naive Bayes classifier for binary problems). The authors view the problem as a game between C and A, using the following forward myopic approach.

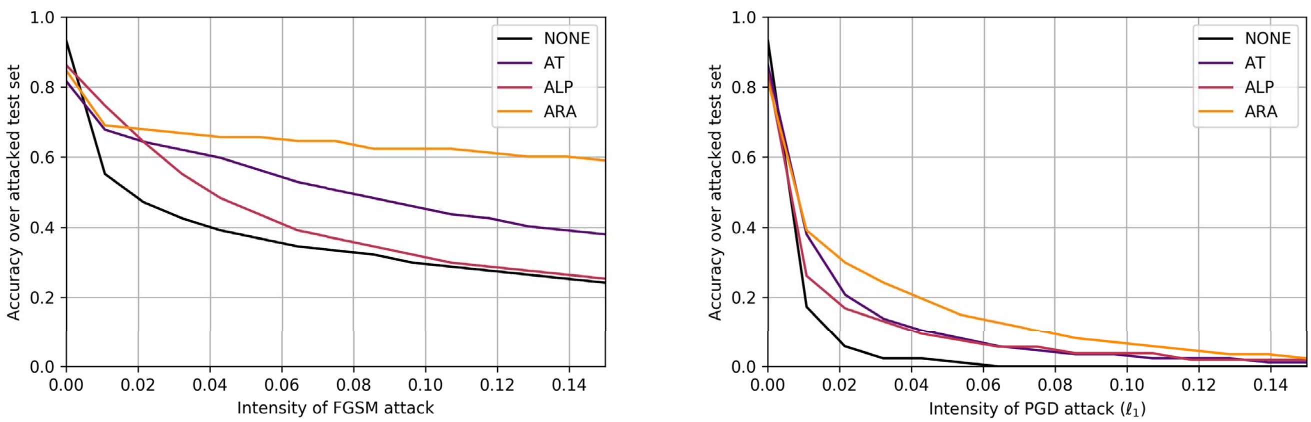
1. C first assumes that data is untainted and computes her optimal classifier through

$$\arg\max_{y_C} \sum_{i=1}^{k} u(y_c, y_i) p(y_i|x),$$

where $p(y_i|x)$ is inferred using training data, clean by assumption.

2. Then, assuming that A has complete information about the classifier's elements (a CK assumption) and that C is not aware of the attacker's presence, the authors compute A's optimal attack. To that end, they propose solving an integer programming problem, reflecting the fact that the adversary tries to minimize the cost of modifying an instance, provided that such modification induces the change/s in the classification decision that A is interested in.

3. Subsequently, the classifier, assuming that A implements the previous attack (again a CK assumption) and that the training data is untainted, deploys her optimal classifier against it, by choosing y_C maximizing



(a) FGSM attack. (b) PGD attack under ℓ_1 norm.

FIG. 5. *Robustness of a deep network for CIFAR*-10 *under three different defense mechanisms* (*ARA*, *AT*, *ALP*) *and two attacks* (*FGSM*, *PGD*).

$\sum_{i=1}^{k} u(y_C, y_i) p(y_i|x')$, her posterior expected utility given that she observes the possibly modified instance x'. This corresponds to optimizing

$$\sum_{i=1}^{k} u(y_C, y_i) p(x'|y_i) p(y_i).$$

Estimating all these elements is straightforward, except for $p_C(x'|y_i)$. Again, appealing to a CK assumption, the authors assume that C, who knows all of A's elements, can exactly solve the adversary problem from step two, and thus compute $x' = a(x, y_i)$, the attack deployed by the adversary when he receives instance x with label y_i. Thus,

$$p(x'|y_i) = \sum_{x \in \mathcal{X}'} p(x|y_i) p(x'|x, y_i),$$

where $\mathcal{X}'$ is the set of possible instances possibly leading to the observed one after an attack and $p(x'|x, y_i) = 1$ if $a(x, y_i) = x'$ and 0 otherwise.

The procedure could continue for more stages. However, the authors consider sufficient to use these three. As presented (and actually Dalvi et al stress in their paper), very strong common knowledge assumptions are made: all parameters of both players are known to each other. Although standard in game theory, such an assumption is unrealistic in the security scenarios typical of AC yet has pervaded most of the later literature in the field.

Throughout the paper, we have emphasized an alternative view of AC that can be condensed into a general pipeline that consists of three main activities: gathering intelligence, forecasting likely attacks, and protecting classification algorithms. A first attempt to organize AML research within such three stages framework is due to Biggio and Roli (2018). However, their framework relies on unrealistic CK assumptions. As such, the authors propose to forecast likely attacks as solutions to certain constrained optimization problems, thus entailing deterministic attacks.

In line with our discussion in Sections 2 and 3, we provide a probabilistic version that mitigates these hypotheses. As argued in previous sections, attacks are probabilistic in order to reflect the lack of knowledge about the attacker's problem. The steps are then:

1. *Gathering intelligence.* The goal of this stage is to model the attacker's problem. This requires assessing attacker *goals*, *knowledge*, and *capabilities*. As a result, a model for how the adversary manipulates an instance with covariates x and label y is constructed.

One possibility to construct such a model is to use a normative decision-theoretic perspective, where the adversary is assumed to behave as a rational agent choosing data manipulations to maximize expected utility.

$$\begin{aligned} & x'(x, y) \\ & = \arg\max_z \sum_{i=1}^{k} u_A(y_C, y_i) p_A(y_C|z = a(x)), \end{aligned} \tag{14}$$

where $p_A(y_C|z = a(x))$ models the adversary's belief about the defender's decision upon observing the manipulated instance $z = a(x)$.

2. *Forecasting likely attacks.* The goal of this stage is to produce an *attacking model* that incorporates not only the information gathered in step 1 but also the uncertainty that we have about the adversary's elements. Within our framework, an attacking model is assimilated with a probability distribution over attacked covariates $p(x'|x)$ given unattacked ones. Evaluating such a model is generally unfeasible but, as presented in Section 3, sampling from it is sufficient.

To sample from the attacking model, first, observe that $p(x'|x) = \sum_{i=1}^{k} p(x'|x, y_i) p(y_i|x)$. Sampling from $p(y_i|x)$ is standard. Sampling from $p(x'|x, y_i)$ is more complex, as we lack information about how the adversary will modify an instance with covariates x and true label y_i. In Dalvi's framework, as sketched above, CK was assumed, and as a consequence, $p(x'|x, y_i)$ was a point mass on the optimal adversarial modification of instance (x, y_i). Instead, we propose modeling our uncertainty about the adversary placing priors on the utilities and probabilities in (14). This induces a distribution over the Attacker's optimal attack defined through

$$X'_\omega(x, y) = \arg\max_z \sum_{i=1}^{k} U^\omega(y_C, y_i) P_A^\omega(y_C|z),$$

where U_A and $P_A^{y_D}$ are random utilities and probabilities defined over an appropriate common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ with atomic elements $\omega \in \Omega$. Then, by construction, $p(x'|x, y) = \mathcal{P}(X'_\omega(x, y) = x')$. Sampling from this distribution requires sampling from the random utilities and probabilities and computing the optimal attack conditioned on those samples.

3. *Protecting classifiers.* Once with a reasonable attacking model, the last step is to protect the classifier against it. As we have seen, this can be done either at operations, modifying the way decisions about the label of a new instance are made; or at training, changing the way it is done to anticipate the future presence of an adversary.

During operations, an adversary-aware classifier will determine the label of a possibly modified instance x' maximizing

$$\sum_{i=1}^{k} \int_{\mathcal{X}_{x'}} u(y_C, y_i) p(y_i|x) p(x|x') \, dx,$$

where x are the covariates of the unknown originating instance, which are marginalized out. In order to solve this problem, samples from $p(x|x')$ need to be generated using Steps 1 and 2 of the pipeline to sample from $p(x'|x)$ and see, for example, the AB-ACRA approach proposed to generate from $p(x|x')$.

Finally, protecting during training requires modifying how inference about the parameters of a given model is made, to guarantee robustness to adversarial manipulations, expressed through condition (11) incorporated into (12).

A natural question is how to deal with the case in which the attacker modifies its behavior as a response to an implemented defense. In our first approach, dealing with adaptive attackers is relatively easy as we are robustifying at operation time: if a change in the adversary's behavior is detected, it can be accounted for in the models used for the attacker's random utilities and probabilities, without retraining the algorithms. Indeed, if data about the adversary is available, models for random utilities and probabilities could be updated online in a Bayesian manner. In our second approach, we could retrain once we detect changes in attack patterns. However, note that in this approach learning (and robustification) is made over data minibatches. That is, the attacker perturbs the first data minibatch with respect to the original defender model. Then, the defender retrains using this attacked minibatch, leading to a slightly more robustified model. Next, the attacker perturbs the second minibatch of data, with respect to the updated model (not the original one), so this attacker is also slightly more powerful, and so on.

5. CONCLUSIONS

Adversarial classification is an increasingly important problem within the emerging field of AML with relevance in numerous security, cybersecurity, law enforcement, and competitive business applications. The pioneering work by Dalvi et al. (2004) has framed, perhaps implicitly, most approaches in AC within a standard game-theoretic context, in spite of the unrealistic common knowledge assumptions required (even questioned by those pioneers). On the other hand, in line with developments in robust Bayesian analysis (Ríos Insua and Ruggeri, 2000), there have been several attempts in the Bayesian community to develop robust models, such as Miller and Dunson (2019). However, none of these approaches model explicitly the presence of adversaries and consequently would not perform properly in adversarial setups.

We have proposed a general Bayesian framework for adversarial classification that models explicitly the presence of an adversary and our uncertainty about his decision-making process. It is general in the sense that application-specific assumptions are kept to a minimum. A key ingredient required by our framework is the ability to sample from the distribution of originating instances given the (possibly attacked) observed one. For this, we first introduced AB-ACRA, a sampling scheme that leverages ARA and ABC to explicitly model the adversary's knowledge and interests, adding our uncertainty about them, and mitigating strong common knowledge assumptions prevalent in the literature. In large-scale problems, this approach easily becomes computationally expensive and we have presented an alternative proposal for differentiable, probabilistic classifiers. In it, the computational load is moved to the training phase, simulating attacks from an adversary using the ARA approach, and then adapting the training framework to obtain a classifier robustified against such attacks.

The proposed methods have performed effectively in the experiments considered. Bayesian methods seem indeed of high relevance to the AML community, since the uncertainties predicted by the models can be used to assess if an instance has been attacked (as a drift in the data distribution would suggest). As an example, Lakshminarayanan, Pritzel and Blundell (2017) proposed a baseline to get predictive uncertainties in large neural models using deep ensembling and AT to smooth those predictive estimates. However, they do not evaluate their framework against adversarial attacks. Our scalable framework showcases the advantages of doing so. First, by estimating uncertainties (e.g., using SG-MCMC algorithms) much better principled than their deep ensembles counterpart, as they target the usual posterior distribution in the Bayesian paradigm. Second, by generalizing AT to further improve robustness, we duly take into account the uncertainties faced by the attacker, inspired by the ARA framework. Besides, our experiments have suggested a relevant regularizing effect.

Numerous lines for further research are worth pursuing in this arena. We highlight four of relevance to the statistical community. First, we have just considered integrity violation attacks. Extensions to availability violation attacks, whose goal is to increase the wrong classification rate, would be important.

Second, the AB-ACRA scheme could be improved in several ways. We have introduced a vanilla ABC version exclusively; integrating recent advancements in ABC methods into the suggested probabilistic framework for AML presents a compelling avenue for further research which could alleviate the AB-ACRA computational bottleneck. For instance, better sampling strategies proposed in the ABC literature could be adapted to our specific context, such as Bortot, Coles and Sisson (2007). Moreover, exploring how to build relevant summary statistics for our algorithm, for example, by following ideas in Fearnhead and Prangle (2012), could be fruitful. Finally, it is important to note our focus on Monte Carlo ABC. As demonstrated by Papamakarios and Murray (2016), there are situations where learning an accurate parametric representation of the entire true posterior distribution requires fewer model simulations than what Monte Carlo ABC methods demand for producing a single sample from an approximate posterior. Therefore, exploring the adaptation of parametric approaches for likelihood-free inference to our AML framework could be of interest.

Third, the approach for differentiable models could be improved as well. Since it requires an SG-MCMC method to simulate attacks, instead of the vanilla SGLD sampler, we could use more efficient samplers, such as those introduced in Gallego and Insua (2018). Finally, we have only touched upon classification problems but the ideas may be used in other areas like standard regression or autoregressive tasks such as time series analysis or natural language processing. Other relevant areas include considering nonmaximum expected utility adversaries, as with the prospect theory models presented in decision analytic contexts in Banks, Rios and Ríos Insua (2016), and integrating our proposals with standard robust Bayesian analysis tools mentioned above.

From a computational perspective, the previous sections illustrate the efficacy of the ARA approach in protecting classification algorithms. However, the aforementioned framework essentially simulates the attacker problem to forecast attacks and utilizes this information to optimize the defender's decision. This entails a nontrivial amount of computational resources and effort, in limited supply in many real-time environments. Therefore, future ARA research of an algorithmic nature is crucial. An approach based on augmented probability simulation (Ekin et al., 2023) is promising in that it combines Monte Carlo sampling and optimization routines. However, should this approach prove computationally infeasible as well, alternative means may be required. Such alternatives may include expedient heuristics for the attacker's problem or approximation techniques that regress the attacker's best response function in a metamodeling sense.

All in all, we would finally stress that AML research has remained largely unexplored by the statistical community, remaining mostly within the computer science domain. We hope that this paper will stimulate research oriented towards leveraging powerful statistical tools to tackle a problem of major relevance in modern societies.

ACKNOWLEDGMENTS

The first and the second authors contributed equally. Roi Naveiro is the corresponding author.

FUNDING

This work was supported by the Severo Ochoa Excellence Programme CEX2023-001347-S, the European Union's Horizon 2020 Research and Innovation Program under Grant Agreements 815003 (Trustonomy), 101021797 (STAR- LIGHT), the EOARD-AFOSR project RC2APD, the FBBVA project AMALFI as well as NSF Grant DMS-1638521 at SAMSI.

DRI is supported by the AXA-ICMAT Chair and the Spanish Ministry of Science program PID2021-124662OB-I00.

AR was supported by project RTC-2017-6593-7.

VG acknowledges support from grant FPU16-05034 and PTQ2021-011758.

Part of the work was performed during the visit of VG, RN, DRI and FR to SAMSI (Statistical and Applied Mathematical Sciences Institute), Durham, NC, USA, within the "Games and Decisions in Risk and Reliability" program.

REFERENCES

BANKS, D. L., RIOS, J. and RÍOS INSUA, D. (2016). *Adversarial Risk Analysis*. CRC Press, Boca Raton, FL. MR3409217

BARRENO, M., NELSON, B., SEARS, R., JOSEPH, A. D. and TYGAR, J. D. (2006). Can machine learning be secure? In *Proceedings of the* 2006 *ACM Symposium on Information*, *Computer and Communications Security* 16–25. ACM, New York.

BIGGIO, B. and ROLI, F. (2018). Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognit*. **84** 317–331.

BISHOP, C. M. (2006). *Pattern Recognition and Machine Learning*. *Information Science and Statistics*. Springer, New York. MR2247587 https://doi.org/10.1007/978-0-387-45528-0

BORTOT, P., COLES, S. G. and SISSON, S. A. (2007). Inference for stereological extremes. *J. Amer. Statist. Assoc.* **102** 84–92. MR2345549 https://doi.org/10.1198/016214506000000988

BOTTOU, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'*2010 177–186. Physica-Verlag, Heidelberg. MR3362066

BREIMAN, L. (2001). Random forests. *Mach. Learn.* **45** 5–32. MR3874153

CABALLERO, W. N., RIOS INSUA, D. and NAVEIRO, R. (2023). Some statistical challenges in automated driving systems. *Appl. Stoch. Models Bus. Ind.* **39** 629–652. MR4662205

CARLINI, N. and WAGNER, D. (2017). Towards evaluating the robustness of neural networks. In 2017 *IEEE Symposium on Security and Privacy* (*SP*) 39–57. IEEE Press, New York.

CASELLA, G., ROBERT, C. P. and WELLS, M. T. (2004). Generalized accept–reject sampling schemes. In *A Festschrift for Herman Rubin*. *Institute of Mathematical Statistics Lecture Notes—Monograph Series* **45** 342–347. IMS, Beachwood, OH. MR2126910 https://doi.org/10.1214/lnms/1196285403

CHAI, X., DENG, L., YANG, Q. and LING, C. X. (2004). Test-cost sensitive naive Bayes classification. In *Fourth IEEE International Conference on Data Mining* (*ICDM'*04) 51–58. IEEE Press, New York.

CHUNG, K. L. (2001). *A Course in Probability Theory*, 3rd ed. Academic Press, San Diego, CA. MR1796326

COMITER, M. (2019). Attacking artificial intelligence. Belfer Center paper.

EUROPEAN COMMISSION (2022). Laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain Union Legislative acts. European Commission.

CSILLÉRY, K., BLUM, M. G. B., GAGGIOTTI, O. E. and FRANÇOIS, O. (2010). Approximate Bayesian Computation (ABC) in practice. *Trends Ecol. Evol.* **25** 410–418. https://doi.org/10.1016/j.tree.2010.04.001

DALVI, N., DOMINGOS, P., MAUSAM, S. S. and VERMA, D. (2004). Adversarial classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. *KDD '04* 99–108.

EKIN, T., NAVEIRO, R., RÍOS INSUA, D. and TORRES-BARRÁN, A. (2023). Augmented probability simulation methods for sequential games. *European J. Oper. Res.* **306** 418–430. MR4522449 https://doi.org/10.1016/j.ejor.2022.06.042

FEARNHEAD, P. and PRANGLE, D. (2012). Constructing summary statistics for approximate Bayesian computation: Semi-automatic approximate Bayesian computation. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **74** 419–474. MR2925370 https://doi.org/10.1111/j.1467-9868.2011.01010.x

FRENCH, S. and RIOS INSUA, D. (2000). *Statistical Decision Theory*. Wiley, New York.

GALLEGO, V. and INSUA, D. R. (2018). Stochastic gradient MCMC with repulsive forces. Bayesian Deep Learning Workshop, Neural Information and Processing Systems (NIPS). arXiv preprint, available at arXiv:1812.00071.

GALLEGO, V. and RÍOS INSUA, D. (2022). Current advances in neural networks. *Annu. Rev. Stat. Appl.* **9** 197–222. MR4394906 https://doi.org/10.1146/annurev-statistics-040220-112019

GOODFELLOW, I., BENGIO, Y. and COURVILLE, A. (2016). *Deep Learning*. *Adaptive Computation and Machine Learning*. MIT Press, Cambridge, MA. MR3617773

GOODFELLOW, I., POUGET-ABADIE, J., MIRZA, M., XU, B., WARDE-FARLEY, D., OZAIR, S., COURVILLE, A. and BENGIO, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems* 27 (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence and K. Q. Weinberger, eds.) 2672–2680. Curran Associates, Red Hook.

GOODFELLOW, I., SHLENS, J. and SZEGEDY, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint, available at arXiv:1412.6572.

GRATHWOHL, W., WANG, K.-C., JACOBSEN, J.-H., DUVENAUD, D., NOROUZI, M. and SWERSKY, K. (2019). Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*.

HAMPEL, F. R., RONCHETTI, E. M., ROUSSEEUW, P. J. and STAHEL, W. A. (1986). *Robust Statistics*: *The Approach Based on Influence Functions*. *Wiley Series in Probability and Mathematical Statistics*: *Probability and Mathematical Statistics*. Wiley, New York. MR0829458

HANEA, A. M., NANE, G. F., BEDFORD, T. and FRENCH, S., eds. (2021). *Expert Judgement in Risk and Decision Analysis*. *International Series in Operations Research & Management Science* **293**. Springer, Cham. MR4238538 https://doi.org/10.1007/978-3-030-46474-5

HARGREAVES-HEAP, S. and VAROUFAKIS, Y. (2004). *Game Theory*: *A Critical Introduction*. Routledge, London.

HASTIE, T., TIBSHIRANI, R. and FRIEDMAN, J. (2009). *The Elements of Statistical Learning*: *Data Mining*, *Inference*, *and Prediction*, 2nd ed. *Springer Series in Statistics*. Springer, New York. MR2722294 https://doi.org/10.1007/978-0-387-84858-7

HE, K., ZHANG, X., REN, S. and SUN, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 770–778.

HUANG, L., JOSEPH, A. D., NELSON, B., RUBINSTEIN, B. I. P. and TYGAR, J. D. (2011). Adversarial machine learning. In *Proceedings of the* 4*th ACM Workshop on Security and Artificial Intelligence*. *AISec '11* 43–58.

JOSEPH, A., MELSON, B., RUBISNTEIN, B. and TYGAR, J. D. (2019). *Adversarial Machine Learning*. Cambridge Univ. Press, Cambridge.

JOSHI, C., ALIAGA, J. R. and INSUA, D. R. (2021). Insider threat modeling: An adversarial risk analysis approach. *IEEE Trans. Inform. Forensics Secur.* **16** 1131–1142. https://doi.org/10.1109/TIFS.2020.3029898

KADANE, J. B. and LARKEY, P. D. (1982). Subjective probability and the theory of games. *Manage. Sci.* **28** 113–125. MR0652016 https://doi.org/10.1287/mnsc.28.2.113

KANNAN, H., KURAKIN, A. and GOODFELLOW, I. (2018). Adversarial logit pairing. arXiv preprint, available at arXiv:1803.06373.

KATZ, G., BARRETT, C., DILL, D. L., JULIAN, K. and KOCHENDERFER, M. J. (2017). Towards proving the adversarial robustness of deep neural networks. arXiv preprint, available at arXiv:1709.02802.

KIM, J.-H. (2009). Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. *Comput. Statist. Data Anal.* **53** 3735–3745. MR2749918 https://doi.org/10.1016/j.csda.2009.04.009

KINGMA, D. P., MOHAMED, S., REZENDE, D. J. and WELLING, M. (2014). Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems* 3581–3589.

KOTZIAS, D., DENIL, M., DE FREITAS, N. and SMYTH, P. (2015). From group to individual labels using deep features. In *Proceedings of the* 21*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 597–606.

KRIZHEVSKY, A., SUTSKEVER, I. and HINTON, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* 1097–1105.

LAKSHMINARAYANAN, B., PRITZEL, A. and BLUNDELL, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Adv. Neural Inf. Process. Syst.* **30** 6402–6413.

LECUN, Y., CORTES, C. and BURGES, C. (1998). THE MNIST DATABASE of handwritten digits. Available at http://yann.lecun.com/exdb/mnist/.

MA, Y.-A., CHEN, T. and FOX, E. (2015). A complete recipe for stochastic gradient MCMC. *Adv. Neural Inf. Process. Syst.* 2917–2925.

MADRY, A., MAKELOV, A., SCHMIDT, L., TSIPRAS, D. and VLADU, A. (2018). Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

MARTIN, G. M., FRAZIER, D. T. and ROBERT, C. P. (2024). Approximating Bayes in the 21st century. *Statist. Sci.* **39** 20–45. MR4718525 https://doi.org/10.1214/22-sts875

MCCULLAGH, P. and NELDER, J. A. (1989). *Generalized Linear Models*. *Monographs on Statistics and Applied Probability*. CRC Press, London. Second edition of [MR0727836]. MR3223057 https://doi.org/10.1007/978-1-4899-3242-6

MICRO, T. (2020). Exploiting AI: How cybercriminals misuse and abuse AI and ML. Trend micro.

MILLER, J. W. and DUNSON, D. B. (2019). Robust Bayesian inference via coarsening. *J. Amer. Statist. Assoc.* **114** 1113–1125. MR4011766 https://doi.org/10.1080/01621459.2018.1469995

MORRIS, D., OAKLEY, J. and CROWE, J. (2014). A web-based tool for eliciting probability distributions from experts. *Environ. Model. Softw.* **52** 1–4.

MÜLLER, P. and INSUA, D. R. (1998). Issues in Bayesian analysis of neural network models. *Neural Comput.* **10** 749–770. https://doi.org/10.1162/089976698300017737

NAVEIRO, R., REDONDO, A., RÍOS INSUA, D. and RUGGERI, F. (2019). Adversarial classification: An adversarial risk analysis approach. *Internat. J. Approx. Reason.* **113** 133–148. MR3983223 https://doi.org/10.1016/j.ijar.2019.07.003

NIST (2022). AI risk management framework: Second draft. NIST.

PAPAMAKARIOS, G. and MURRAY, I. (2016). Fast ε-free inference of simulation models with Bayesian conditional density estimation. *Adv. Neural Inf. Process. Syst.* **29**.

PLATT, J. et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Adv. Larg. Margin Classif.* **10** 61–74.

RAIFFA, H. (1982). *The Art and Science of Negotiation*. Harvard Univ. Press, Cambridge.

RÍOS INSUA, D., GONZÁLEZ-ORTEGA, J., BANKS, D. and RÍOS, J. (2018). Concept uncertainty in adversarial statistical decision theory. In *The Mathematics of the Uncertain*. *Stud. Syst. Decis. Control* **142** 527–542. Springer, Cham. MR3793229

RÍOS INSUA, D. and RUGGERI, F., eds. (2000). *Robust Bayesian Analysis*. *Lecture Notes in Statistics* **152**. Springer, New York. MR1795206 https://doi.org/10.1007/978-1-4612-1306-2

RIOS INSUA, I., RIOS, J. and BANKS, D. (2009). Adversarial risk analysis. *J. Amer. Statist. Assoc.* **104** 841–854. MR2541597 https://doi.org/10.1198/jasa.2009.0155

RIOS, J. and RIOS INSUA, D. (2012). Adversarial risk analysis for counterterrorism modeling. *Risk Anal.* **32** 894–915. MR2946741 https://doi.org/10.1287/deca.1110.0225

ROBBINS, H. and MONRO, S. (1951). A stochastic approximation method. *Ann. Math. Stat.* **22** 400–407. MR0042668 https://doi.org/10.1214/aoms/1177729586

STAHL, D. O. and WILSON, P. W. (1995). On players' models of other players: Theory and experimental evidence. *Games Econom. Behav.* **10** 218–254. MR1347938 https://doi.org/10.1006/game.1995.1031

SU, J., VARGAS, V. and KOUICHI, S. (2017). One pixel attack for fooling deep neural networks. Available at arXiv:1710.08864.

SZEGEDY, C., ZAREMBA, W., SUTSKEVER, I., BRUNA, J., ERHAN, D., GOODFELLOW, I. and FERGUS, R. (2014). Intriguing properties of neural networks. arXiv preprint, available at arXiv:1312.6199.

TABASSI, E., BURNS, K., HADJIMICHAEL, M., MOLINA-MARKHAM, A. and SEXTON, J. (2020). A taxonomy and terminology of adversarial machine learning. NIST-MITRE.

THE WHITE HOUSE (2023). Executive order on the safe, secure, and trustworthy development and use of artificial intelligence. Briefing Room, Presidential Actions.

VOROBEICHYK, Y. and KANTARCIOGLU, M. (2019). *Adversarial Machine Learning*. Morgan & Claypool, CA.

WELLING, M. and TEH, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the* 28*th International Conference on Machine Learning* (*ICML*-11) 681–688.

WIPER, M., RIOS INSUA, D. and RUGGERI, F. (2001). Mixtures of gamma distributions with applications. *J. Comput. Graph. Statist.* **10** 440–454. MR1939034 https://doi.org/10.1198/106186001317115054

YANG, R. and BERGER, J. (1997). A catalog of noninformative priors. ISDS Discussion paper 97-42.

ZHOU, Y., KANTARCIOGLU, M. and XI, B. (2018). A survey of game theoretic approach for adversarial machine learning. Wiley interdisciplinary reviews: Data mining and knowledge discovery e1259.