The Many Faces of Adversarial Machine Learning

Yevgeniy Vorobeychik

Washington University in Saint Louis
yvorobeychik@wustl.edu

Abstract

Adversarial machine learning (AML) research is concerned with robustness of machine learning models and algorithms to malicious tampering. Originating at the intersection between machine learning and cybersecurity, AML has come to have broader research appeal, stretching traditional notions of security to include applications of computer vision, natural language processing, and network science. In addition, the problems of strategic classification, algorithmic recourse, and counterfactual explanations have essentially the same core mathematical structure as AML, despite distinct motivations. I give a simplified overview of the central problems in AML, and then discuss both the security-motivated AML domains, and the problems above unrelated to security. These together span a number of important AI subdisciplines, but can all broadly be viewed as concerned with *trustworthy AI*. My goal is to clarify both the technical connections among these, as well as the substantive differences, suggesting directions for future research.

1 Introduction

The deployment of machine learning in open-world safety critical settings has created a need to ensure that such approaches are robust to adversarial encounters. In response emerged a subfield of *adversarial machine learning (AML)*, which studies vulnerabilities in machine learning approaches. AML takes two common forms: 1) attacks on deployed (learned) models, commonly known as adversarial examples or perturbations, and 2) attacks on algorithms, commonly known as poisoning.

Historically, the field of AML was motivated by classical security issues, such as the use of machine learning for spam, intrusion, and malware detection (Biggio et al. 2013; Barreno et al. 2010; Dalvi et al. 2004; Laskov et al. 2014; Lowd and Meek 2005). Over time—particularly since mid-2010s—the scope expanded significantly to include domains such as computer vision (often motivated by autonomous driving), natural language processing (motivated, for example, by hate speech detection), and network science (motivated, for example, by identifying malicious actors on social media platforms), among others. Common to these domains are concerns about *malicious* actors who aim to subvert predictions to some anti-social ends. Moreover, recent years have seen the emergence of problems that have nothing to do with security considerations at all, and yet use essentially the same mathematical formalization as conventional AML. Most prominent among these are strategic classification, algorithmic recourse, and counterfactual explanations. Despite the close mathematical connection, however, cross-pollination of ideas between these three problems and AML has been surprisingly limited, and there appears to be limited appreciation in the literature of just how closely related they are, with commonly made distinctions often more superficial than substantive.

Below, I formally describe the canonical AML problem, then describe several common AML domains motivated by security, and finally discuss the formal connections between AML and the three non-security problems above that share a strong connection to AML despite distinct motivations.

2 Adversarial Machine Learning

Perhaps the most recognizable form of adversarial machine learning is the addition of small adversarial noise (*adversarial perturbation*) to inputs into machine learning models aimed at changing predictions. To formalize, let $f(x;\theta)$ be a model parameterized by θ (that are learned from data) which takes a feature vector x as an input and outputs a prediction. Let δ denote an adversarial perturbation. A common design goal for adversarial perturbations involves solving the following optimization problem:

$$\min_{\delta : c(\delta) \le \epsilon} \quad l(f(x+\delta;\theta), y_T), \tag{1}$$

where $c(\delta)$ measures the cost of implementing the perturbation δ if the original feature vector is x, y_T is the target prediction, and $l(y_{pred}, y_T)$ is the prediction error that the adversary aims to minimize. presumed to be exogenously specified. The $c(\delta) \le \epsilon$ constraint on the magnitude of the perturbation commonly leverages a highly stylized cost function that aims to capture whatever practical constraints are faced in generating such attacks (which are in actuality difficult to model formally). As we shall see below, it is precisely the nature of the choice of such a cost function that appears to vary most by particular domain in which adversarial machine learning has come to play a role.

Formulation (1) is commonly referred to as a *targeted* attack. An *untargeted* attack, on the other hand, aims to maximize error with respect to the true output y. Note, however, that we can fit such attacks into formulation (1) by simply redefining $y_T = y$ and redefining the objective to be the negative of the learner's loss.

Problem (1) admits many variations. For example, we can interchange the objective and the constraint, minimizing modification cost subject to a hard constraint that the target class y_T is predicted, or we can transform this into an unconstrained optimization of the weighted sum of the loss and cost terms. Further, it may be natural to restrict which features can be modified (for example, a malware designer may be unable to modify some features which are essential to the malicious payload (Laskov et al. 2014; Tong et al. 2019)). Moreover, one may impose further constraints that ensure that the attack is not easily detected, such as constraining it to be similar to examples in training data (Maiorca, Corona, and Giacinto 2013; Ghafouri, Vorobeychik, and Koutsoukos 2018). Finally, it is often natural to restrict only a subset of possible inputs to be adversarial; for example, in malware detection settings, it is only the malicious instances that would aim to evade detection by changing features (Vorobeychik and Kantarcioglu 2018).

While either the untargeted or targeted attack problem above is non-convex for complex non-linear models f, such as deep neural networks, there are several common heuristic approaches for obtaining high-quality solutions. If the feature vectors x are (approximately, or actually) continuous, a common option is a form of projected gradient descent algorithm, often referred to as PGD (which becomes ascent in the case of untargeted attacks) (Madry et al. 2018). If features are discrete, stochastic local search methods can be highly effective (Li and Vorobeychik 2018; Tong et al. 2019).

A conceptual dual of these adversarial problems is the problem of learning models that are *robust* to adversarial perturbations. Perhaps the most useful way to formalize the problem of robust machine learning in this context is as the following robust optimization problem:

$$\min_{\theta} \sum_i \max_{\delta : c(\delta) \le \epsilon} \quad l(f(x_i + \delta; \theta), y_i), \tag{2}$$

where i indexes input data points in the training data. Commonly, solutions to this problem are approximated by using *adversarial training (AT)*. In AT, one iterates between updating θ (e.g., using minibatch gradient updates) while fixing adversarial perturbations δ_i for the update, and updating δ_i (i.e., generating adversarial perturbations by solving Problem (1)) for inputs that will be used for training in the next iteration. An important feature of AT is that one need not make *any* specific assumptions on how adversarial perturbations δ_i are generated; they could even be generated using a model of the behavior of *human* adversaries (Ke, Li, and Vorobeychik 2016; Li and Vorobeychik 2018).

A crucial aspect of the adversarial models above that we have yet to instantiate is the cost $c(\delta)$ incurred by the attacker for the modification δ. Indeed, we will observe below that what is commonly viewed as a "natural" cost function varies by domain, although ultimately all of the commonly used cost function models are highly stylized.

Adversarial perturbation attacks that we described so far are just one class of threat models considered in the broader AML literature. Another important class of attacks presumes adversarial ability to tamper with *training data*, and are referred to as *poisoning attacks*. Many variations of poisoning attacks exist, but most share the following structure. The attacker modifies a training dataset D, transforming it into a new dataset D'. There is, again, a stylized cost function $c(D, D')$ that penalizes significant modifications, beyond whatever constraints the model itself admits, such as restricting modifications to flipping labels (Patrini et al. 2017), inserting data points (Biggio, Nelson, and Laskov 2012), etc. Additionally, the attacker aims to achieve some goal encapsulated by a utility function $U(\theta(D'))$, where $\theta(D')$ is a model parameter vector obtained if the learning algorithm trains on the transformed data D'. A generic way of modeling the poisoning problem is as a bi-level optimization problem (Vorobeychik and Kantarcioglu 2018):

$$\max_{D'} U(\theta(D')) - c(D, D') \tag{3a}$$

$$\text{s.t.}: \quad \theta(D') \in \arg\min_{\theta} \sum_{d \in D'} l(d, \theta), \tag{3b}$$

where $l(d, \theta)$ is the loss on data point d when model parameter vector is θ.

The bi-level nature of poisoning attack model as exemplified by Problem (3) means that in general these are extremely difficult optimization problems. Consequently, approaches tend to make strong assumptions, such as that the learning problem itself is convex (allowing one to leverage KKT conditions in the poisoning attack) (Mei and Zhu 2015). Interestingly, the problem of robust learning with poisoned data often admits simpler solutions, with common examples involving some form of data sanitization prior to, or as a part of, learning (Vorobeychik and Kantarcioglu 2018).

3 AML in Security

The most natural context for adversarial machine learning is security. And, indeed, this is a central concern. However, as we shall see presently, what constitutes "security" is extremely broad, covering areas within AI that range from machine learning applications within core security (such as malware detection) to network science. Nevertheless, our overview in this section concerns what are predominantly *security*-related considerations. Thereafter, I consider settings that involve no security implications whatsoever, and yet share the core mathematical structure with AML.

Core Security Security considerations served as the early motivation for adversarial machine learning research. The traditional problems in security in which this issue is most salient are *detection problems*, most notably, spam, malware, and intrusion detection. A canonical form of the detection problem involves a binary classifier $f(x; \theta)$ which predicts whether an input x is malicious (often, $+1$) or benign (often, either 0 or -1). In this context, the adversarial problem is only relevant for malicious inputs x, since benign inputs certainly wish to remain benign. In the context of detection, the most common (arguably, dominant) variant of the perturbation cost function $c()$ is $c(\delta) = \|\delta\|_p$, where $\|\cdot\|_p$ is an ℓ_p norm for $p \geq 0$, much as it is widely understood that this fails to capture actual constraints faced by attackers (such as ensuring that the malicious payload remains effective).

Natural Language Processing One traditional security application that was a particularly common motivation for early work on adversarial machine learning is spam (or phishing email) detection. The simplest variation embeds words in a document in a binary bag-of-words feature representation. In this case, one can naturally adopt a ℓ_1-norm cost function, and use local search to approximately solve the adversarial problem, where the aim of the attacker is to cause a predicted spam email to be misclassified as benign.

The preoccupation with adversarial text clasification problems has recently received a broader interest in the natural language processing (NLP) community for prediction tasks that go beyond traditional security (such as sentiment prediction). An important consideration both in the earlier, and more recent work on AML in NLP is preservation of semantic similarity. For example, arbitrary changes of features (words) in the superficial bag-of-word representation would typically fail to preserve semantic similarity, even if we only change a small number of words. One natural variation of the problem that helps address this is to use semantic word embedding to represent words in text. By working directly in the embedding space, one achieves two advantages over the older approaches: 1) we can now optimize Problem (1) over the real-valued embedded features directly, and 2) ℓ_p-norm cost is more semantically relevant. Nevertheless, such approaches still fail to achieve adequate semantics at a sentence level, and more complex mathematical models of modification cost functions have been proposed to better capture semantics (Zhang et al. 2020).

Computer Vision An explosion of recent interest in AML has been much stimulated by the work on adversarial examples in computer vision that specifically target deep neural networks in image classification problems (Goodfellow, Shlens, and Szegedy 2015). One of the most intriguing observations was that adversarial noise that is *imperceptible to a human* can change the prediction for an otherwise high-performing image classifier. Notable in this work is that it makes use of the cost function $c(\delta) = \|\delta\|_p$ just as in the security problems discussed above. However, there is an interesting particular distinction: in the earlier AML approaches in security, common cost functions were based on either ℓ_1 or ℓ_2 norms, whereas in the domain of vision, the choices tend to be either ℓ_2 or ℓ_∞. These distinctions seem to have arisen largely out of different research conventions in these areas of focus, rather through any principled threat analysis. In any case, the original motivation for the choice of ℓ_p cost in adversarial computer vision was simply that it was a natural and easy way to obtain imperceptible noise by setting ϵ to be sufficiently small. Not surprisingly, much work has subsequently identified alternative ways to obtain imperceptible adversarial examples (Hosseini and Poovendran 2018; Xiao et al. 2018), but it seems fair to say that the dominant focus remains on ℓ_p cost functions.

Network Science One of the major aims in network science is to develop technical tools for quantitative analysis and inference on data represented by a graph (often referred to as network analysis). Such a graph may be directed or undirected, with edges possibly weighted, and may change in time. There are too many particular network analysis problems to enumerate here; I will therefore limit discussion to link prediction and node classification on a fixed graph.

In link prediction, the observed (sub)graph is leveraged to predict existence or non-existence of particular edges of interest. Node classification, in turn, takes as input a graph in which all nodes are also associated with feature vectors, but only a subset of nodes have observed labels, and the task is to predict labels for the remaining nodes. Both link prediction and node classification can be viewed as tasks that apply some function $f(x;\theta)$ to predict a label for the target of interest (a node or a potential link), where the input x combines both the network identity of the prediction target (e.g., node index) and any relevant observed features thereof, and θ are any relevant model parameters. However, note that this representation is myopic, as it fails to account for longer-range interactions in networks which can be informative in inference. A key generalization, therefore, can include as inputs a combination of the full feature matrix X (with each row corresponding to a feature vector of the associated graph node) and the graph adjacency matrix A. Thus, the prediction model takes the form $f_i(X, A; \theta)$ where i can refer to either the identity of a node or an edge.

Clearly, in the context of network analysis of the form above, the adversary has considerable latitude for modification: they can modify X (that is, the full feature matrix of relevance) as well as the adjacency matrix A (for example, by removing or adding edges). If we take θ as fixed, however, the nature of the adversarial problem is qualitatively similar to what we had previously, if we view the combination of X and A as a "long" feature vector. In particular, one would typically define a cost function $c(\delta)$ in a manner quite similar to other adversarial machine learning settings. For example, a common adversarial model in the context of network analysis focuses solely on removing and/or adding edges (Wang et al. 2021; Zhou et al. 2019; Zhou, Michalak, and Vorobeychik 2019). In that context, a natural cost function simply bounds the number of edges that can be modified in the graph. Furthermore, if edges are weighted, the associated feature space is real-valued, and standard gradient-based approaches can be applied.

However, many approaches in network analysis treat problems such as node classification as semi-supervised learning, where the parameters θ of f are learned from observed data about node labels, and subsequently the learned f is used for predicting unobserved labels (Zügner et al. 2020). In this context, the most meaningful threat model would involve modifying the graph (and/or the features) *prior to learning*—that is, this becomes a poisoning attack.

Our discussion of adversarial network analysis has presumed a security motivation. One example setting motivating security concerns would be network security (for example, if the goal is to predict the presence of compromised devices on the network). Another example would be in law enforcement, where the goal is to identify members of a crime organization, and their connections with one another, as well as with others who may facilitate (but not be directly involved in) criminal activities.

4 Strategic Classification

My overview thus far has considered adversarial machine learning settings motivated primarily by security considerations (hence, adversarial). The problem of *strategic classification*, in contrast, is motivated by the economic consideration of *incentives*, instead of any actual adversarial behavior (Hardt et al. 2016). Nevertheless, nearly all common variations of strategic classification are mathematically identical to adversarial machine learning, and differ primarily in the convention about what cost function $c(\delta)$ is used in the analysis. Moreover, the only reason for the difference in conventions appears to be a greater emphasis on mathematical convenience in the strategic classification literature.

A canonical motivation for strategic classification is a problem like college admissions. Suppose we use an algorithm (represented by $f(x;\theta)$, and perhaps using machine learning to learn f from data) for admissions decisions, with each applicant represented by a feature vector x (e.g., SAT scores, GPA, number of extracurricular activities, etc). For the moment, let us say that admissions are decided independently for each student based on their acceptability (the college is not particularly selective). Clearly, if individuals wish to be accepted but have $f(x;\theta)=0$, they have an incentive to change their features x. There are two ways this change can be implemented: if features are self-reported, the applicants could outright misreport them to make themselves look stronger; if features are collected from a neutral third party (such as SAT scores), students can expend resources, such as money, to change the associated features (for example, pay for expensive test preparation). Considering the latter situation, it is natural to consider a total budget on the amount of resources one can expend modifying application characteristics, so we obtain two constraints, $c(\delta)\leq\epsilon$, and $f(x+\delta;\theta)=1$, where δ is the change in the individual's features. This essentially mirrors the canonical adversarial machine learning problem (1), with the caveat that only individuals who would not otherwise be accepted do this (just as in the malware detection domain, where only malware authors would modify their code to evade detection).

In general, individuals in the strategic classification setting are faced with a selection algorithm, and their goal is to maximize their chances of obtaining a social or economic benefit which comes with being selected. Nevertheless, there is no substantive mathematical difference between this problem and the problem which is motivated by adversarial agents. A common difference in this literature, however, is that in place of the ℓ_p cost function $c(\delta)=\|\delta\|_p$ conventional in security settings, the strategic classification literature typically assumes separable costs, $c(\delta)=\max\{0,v^T\delta\}$, or simply linear costs $c(\delta)=v^T\delta$ for some known v. This assumption has proved to be very convenient for obtaining robust classifiers in closed form (Hardt et al. 2016), as well as other mathematical analyses (Hu, Immorlica, and Vaughan 2019; Milli et al. 2019). However, recent work considers the issue of training classifiers which are robust to strategic manipulation for more general cost functions (Levanon and Rosenfeld 2021). Interestingly, this work illuminates a surprising gap in cross-pollination of ideas between the AML and strategic classification literatures. For example, adversarial training—a major workhorse in AML—is not mentioned by Levanon and Rosenfeld (2021). On the other hand, the contributions to robust learning by the strategic classification literature (e.g., Hardt et al. (2016); Levanon and Rosenfeld (2021)) seem to have also had minimal impact on the adversarial machine learning literature.

One could argue that in strategic classification, the game is not strictly zero-sum, in that strategic agents are not necessarily maximizing error. However, this is unconvincing for two reasons. First, maximizing error is in any case a proxy in adversarial machine learning, but has proved an effective means of making progress. Second, one can note that adversarial training as described in Section 2 actually needs to make no assumptions on the way adversarial perturbations are generated; and, indeed, prior work has shown that this general scheme is effective even in non-zero-sum settings, and even if strategic behavior is generated using some alternative "mechanistic" adversarial model (e.g., a model that learns to mimic adversarial behavior from data) (Li and Vorobeychik 2018).

Does all this mean that strategic classification is just another variation on the adversarial machine learning theme, with nothing of its own to contribute? I do not think so. The central motivating problem behind strategic classification is in fact very different, and gives rise to interesting new kinds of analysis and approaches that make little sense when the setting is purely adversarial. Some of these pertain to algorithmic recourse; we discuss these in Section 5. Two others are: scarcity of resources and incentive compatibility; I discuss these briefly next.

Strategic Classification with Scarce Resources Consider again the college admission setting. In our example, everyone gets in as long as they are "above the bar", so-to-speak. However, it seems unlikely that people would expend financial resources just to get into an "above-the-bar" type of college; this is a problem that's salient in highly competitive college admissions, and it is the highly competitive part that's critical: scarcity is significant.

Indeed, many settings of social interest where strategic classification is well-motivated involve scarce resource allocation: allocation of promotions and social services are among the many that come to mind. And scarcity breaks the key assumption above that the decisions $f(x;\theta)$ depend only on the features x of the individual: clearly, scarce resources will be allocated to individuals from a *population*, and allocation will depend on the features of *all* members of this population. For example, if college admissions are particularly competitive one year, what would have been sufficient to be accepted the previous year may no longer be enough.

Let us simplify the problem somewhat. Suppose we still have a scoring function $f(x;\theta)$ that yields a score for each individual with features x. But now, instead of a decision made independently for each x, we have a population of n individuals with a set of feature vectors $X = \{x_1, \ldots, x_n\}$. A natural way to allocate the scarce resource is to rank individuals by their scores, $f(x_i;\theta)$, from high to low, and allocate the resource in this order until nothing is left to allocate (assuming scarcity means that the number of available resources is $k < n$). In this setting, the efficacy of an individual i's feature manipulations δ_i in obtaining a resource now depends on the manipulations of all other individuals in the population X; thus, we have a game. For this game, Problem (1) is not a particularly good model, and more natural is a variation in which i's utility is defined by

$$u_i(\delta_i, X) = I(R_i(f(x_i + \delta_i, \theta), f(X_{-i};\theta)) \leq k) - c(\delta_i),$$

where $f(X_{-i};\theta)$ is a vector of scores for (observed) feature vectors in X other than i, R_i is a rank of i's score, and $I(\cdot)$ is an indicator function (we ignore the issue of tie-breaking here for simplicity). Note that since any collection of feature modifications δ_j for all players j other than i will induce some modified dataset X_{-i}, this utility definition is general.

As defined above, strategic classification is a game of complete information. We can also consider an incomplete information game, where individuals only know the distribution of feature vectors x_i, but not the realization, and perhaps are even uncertain about the population size n.

While strategic classification has been extensively studied when there are no resource constraints, there has been remarkably little work studying the setting with resource constraints. One of the efforts that considers this setting explicitly was our work on *auditing* in the context of strategic classification (Estornell, Das, and Vorobeychik 2021). The setting here is as follows. We wish to allocate k units of a scarce resource based on some ranking criterion, such as vulnerability or marginal benefit. To this end, we obtain objective features z, and also elicit important but self-reported features x, from all individuals, and use these as an input in a score function $f(x, z)$ just as above, allocating the resource to those with the k highest scores. This setup creates an incentive for individuals to misreport their self-reported features x, although note that z potentially constrains the credibility of these misreports. To address the concern about misreporting (and, as a result, misallocation), a decision-maker has a limited audit budget B which can be used to inspect self-reported features x, and penalize those who misreport (which can also involve preventing them from obtaining the resource). The central object of study in Estornell, Das, and Vorobeychik (2021) was the computational complexity of designing audit mechanisms that induce the smallest incentives for individuals to misreport their features, taking this to be a setting with incomplete information (i.e., individuals do not know one another's self-reported features, but only the distribution of these), as well as the complexity of verifying the incentives to misreport preferences under a given audit mechanism.

The question of auditing in the context of scarce resources is a natural segue into the second issue that seems particular to strategic classification settings: incentive compatibility.

Incentive Compatible Classification The problem of robustness in strategic classification quite naturally fits into the general *mechanism design* paradigm (Myerson 1989). In mechanism design, we have a set of n agents I, space of outcomes O, space of agent types T which index their utility functions $u_i(o, t)$ for outcomes $o \in O$ for each agent $i \in I$, and a set of actions A for the agents.[1] A mechanism f maps a collection of actions of participating agents, $\{a_1, \ldots, a_n\}$, $a_i \in A$, to an outcome $o \in O$. A *direct* mechanism is one in which actions are identical to types, that is, $A = T$. In a useful restriction of the general problem, the principal incurs a loss associated with each agent i, $l(o, g(t_i))$, where t_i is the actual type of agent i and $g(\cdot)$ some function, and aims to minimize the total loss, $\sum_i l(o, g(t_i))$. Thus, implementing a mechanism f yields the total loss

$$\sum_i l(f(t'_1, \ldots, t'_n), g(t_i)),$$

where t'_i is the *reported* (and not necessarily actual) type of each agent i. Now, suppose that T is the sample space, and $t_i = x_i$—that is, types are feature vectors. Reported types, in turn, are $t'_i = x'_i = x_i + \delta_i$, where δ_i is the perturbation implemented by agent i. Finally, let $g(\cdot) \equiv f(\cdot)$, and let $y_i = g(t_i) = g(x_i) \equiv f(x_i)$, where $f()$ is now a classifier. The designer's objective then becomes to identify a classifier f from some hypothesis class so as to minimize

$$\sum_i l(f(x_1 + \delta_1, \ldots, x_n + \delta_n), y_i).$$

If we treat each agent as independent, this transforms into empirical risk minimization with strategic agents.

Among the most important results in mechanism design is the *revelation principle*; informally, if there are no constraints on which type t'_i the agent can report in place of its true type t_i, then for any mechanism f there always exists another f' such that f' achieves the same outcome as f after accounting for agent strategic behavior, and f' is *incentive compatible*, that is, all agents maximize their respective utilities by reporting true types t_i. Remarkably, Zhang and Conitzer (2021) showed that the revelation principle does not generally obtain in the strategic classification setting. In fact, they show that the necessary and sufficient condition for it to hold is that misreporting is *transitive*: if an agent can misreport t_i to be t'_i and t'_i to be t''_i, it should also be able to misreport t_i as t''_i. This transitivity fails, for example, if we impose an ℓ_p-norm constraint on the magnitude of the manipulation cost δ, which is a common AML setting.

The discussion of strategic classification and incentive compatibility thus far has been the direct counterpart of *decision-time*, or adversarial perturbation, attacks in adversarial machine learning (Vorobeychik and Kantarcioglu 2018). I now turn to the problem of *strategic regression* (in most cases in the literature, linear regression), which in fact is the counterpart of *poisoning attacks* (Chen et al. 2018; Dekel, Fischer, and Procaccia 2010). The setting here is as follows. Consider a training dataset $D = \{(x_i, y_i)\}$, but now

[1] In this, we have assumed for simplicity that all agents have an identical set of types and actions.

suppose that instead of the true labels y_i, we obtain these from strategic agents who wish for the regression to be as accurate as possible on their own datapoint i. To this end, strategic agents may actually misreport the labels, and the resulting dataset becomes $\tilde{D} = \{(x_i, \tilde{y}_i)\}$, with $\tilde{y}_i$ the reported labels for the datapoints (which may or may not be accurate). A linear regression model is then fit to the obtained dataset $\tilde{D}$. Of central interest in this literature has also been an issue of incentive compatibility, but now it is the problem of devising a learning algorithm which incentivizes the agents to provide true labels y_i for their datapoints. Unlike the somewhat negative result in the case of incentive-compatible strategic classification above, here there are several strong positive results, the simplest of which is that l_1 loss is *group-strategyproof*, in that no subset of agents can gain by jointly misreporting their labels.

While the connection to robust learning in the context of data poisoning attacks is typically acknowledged in the strategic regression literature, the intersection of these two problems seems fruitful to explore further. For example, strategic regression models assume that *every* data point may have been strategically mislabeled, in contrast to robust learning, in which the set of problems explored is far more rich, ranging from very small to very high fraction of data that may have been poisoned (Karmalkar, Klivans, and Kothari 2019; Kearns and Li 1993; Liu et al. 2017). In the strategic setting especially, it seems quite reasonable to suppose that only a small fraction of agents is strategic. In this case, conventional techniques that identify and remove a subset of outliers (Klivans, Long, and Servedio 2009) would perhaps also yield approximate incentive compatibility for a richer class of loss functions.

5 Algorithmic Recourse

As techniques such as machine learning are increasingly used in high-impact settings, often involving historically marginalized populations, concerns about fairness of such approaches has come to the fore in recent years. An important way in which algorithmic fairness connects to adversarial machine learning is through its consideration of *algorithmic recourse* (Karimi, Schölkopf, and Valera 2021; Ustun, Spangher, and Liu 2019). In algorithmic recourse, one supposes that the classifier $f(x;\theta)$ is binary, where class 1 is associated with "being selected" (for some preferred outcome) and 0 with not being selected. Then, for an individual x who is not selected ($f(x;\theta) = 0$), we wish to additionally provide an actionable alternative $x' = x + \delta$ such that $c(\delta)$ is small and $f(x';\theta) = 1$. The interpretation is that we wish to provide individuals the simplest recourse to be selected (e.g., all you need to do is slightly improve your SAT score to be accepted to a particular college). Note, however, that this problem has essentially the identical mathematical form as the standard AML problem.

A key concept in algorithmic recourse is that it be *actionable*, and alternative models propose to formalize recourse in terms of feasible actions that have causal impact on features (Karimi, Schölkopf, and Valera 2021). One may argue that the constraint that recourse be actionable is a crucial distinction between this problem and adversarial machine learning; however, this does not seem convincing, since in adversarial machine learning also, the stylized problem formulation is a proxy for realizable attacks—that is, attacks that can be implemented in practice (Tong et al. 2019).

Algorithmic recourse can be viewed as adding a form of transparency to algorithmic systems. Indeed, it is closely connected to *counterfactual explanations* discussed next.

6 Counterfactual Explanations

Ability to explain algorithmic decisions is widely recognized to be a crucial building block of trust. A variety of concepts and approaches for explaining algorithms and decisions they make have been proposed; of particular interest to us are *counterfactual explanations*, the goal of which is to explain a particular algorithmic outcome by presenting the closest alternative feature vector input for which the outcome changes (e.g., to be more favorable to the individual) (Verma, Dickerson, and Hines 2020). To this end, Problem (1) and its natural variations (particularly, where we minimize distance to change the prediction) are clearly well-suited, and have indeed been proposed as a way to generate such explanations. A number of additional variations exist that aim to ensure that explanations are "natural", such as adding regularization to the optimization problem that penalizes explanations ($x' = x + \delta$) which are too dissimilar from the distribution of observed feature vectors in the training dataset. In their survey, Verma, Dickerson, and Hines (2020) note the close connection to adversarial learning, but highlight that explanations have the additional desiderata of parsimony or sparsity, as well as actionability. However, these are also common desiderata in adversarial ML: sparsity is a feature of l_1 attacks, studied extensively in the adversarial ML literature (Vorobeychik and Kantarcioglu 2018), and actionability is a central consideration in realizable adversarial perturbations (Tong et al. 2019; Wu, Tong, and Vorobeychik 2020).

7 Conclusion

In this paper, I described several variations of the adversarial machine learning modeling paradigm. Some of these, such as malware detection, are typically framed as security problems. Several others, however, have entirely different motivations, but nevertheless share much of the core mathematical structure. These include strategic classification, algorithmic recourse, and counterfactual explanations. In addition to describing the formal connections, I attempted to illustrate some substantive differences, particularly in strategic classification. My overarching goal is to facilitate cross-pollination of ideas across these research areas, and further clarify both the similarities and substantive differences among them to help stimulate new research directions.

Acknowledgments

This work was supported in part by the National Science Foundation (IIS-1905558, IIS-1939677, IIS-1903207, ECCS-2020289), Army Research Office (W911NF-19-1-0241), Amazon, and NVIDIA.

References

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

Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Šrndić, N.; Laskov, P.; Giacinto, G.; and Roli, F. 2013. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, 387–402.

Biggio, B.; Nelson, B.; and Laskov, P. 2012. Poisoning attacks against support vector machines. In *International Conference on Machine Learning*.

Chen, Y.; Podimata, C.; Procaccia, A. D.; and Shah, N. 2018. Strategyproof linear regression in high dimensions. In *ACM Conference on Economics and Computation*, 9–26.

Dalvi, N.; Domingos, P.; Sanghai, S.; Verma, D.; et al. 2004. Adversarial classification. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 99–108.

Dekel, O.; Fischer, F.; and Procaccia, A. D. 2010. Incentive compatible regression learning. *Journal of Computer and System Sciences*, 76(8): 759–777.

Estornell, A.; Das, S.; and Vorobeychik, Y. 2021. Incentivizing Truthfulness Through Audits in Strategic Classification. In *AAAI Conference on Artificial Intelligence*, 5347–5354.

Ghafouri, A.; Vorobeychik, Y.; and Koutsoukos, X. 2018. Adversarial regression for detecting attacks in cyber-physical systems. In *International Joint Conference on Artificial Intelligence*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

Hardt, M.; Megiddo, N.; Papadimitriou, C.; and Wootters, M. 2016. Strategic classification. In *ACM Conference on Innovations in Theoretical Computer Science*, 111–122.

Hosseini, H.; and Poovendran, R. 2018. Semantic adversarial examples. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 1614–1619.

Hu, L.; Immorlica, N.; and Vaughan, J. W. 2019. The disparate effects of strategic manipulation. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 259–268.

Karimi, A.-H.; Schölkopf, B.; and Valera, I. 2021. Algorithmic recourse: from counterfactual explanations to interventions. In *ACM Conference on Fairness, Accountability, and Transparency*, 353–362.

Karmalkar, S.; Klivans, A.; and Kothari, P. 2019. List-decodable linear regression. *Neural Information Processing Systems*, 32.

Ke, L.; Li, B.; and Vorobeychik, Y. 2016. Behavioral experiments in email filter evasion. In *AAAI Conference on Artificial Intelligence*.

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

Klivans, A. R.; Long, P. M.; and Servedio, R. A. 2009. Learning Halfspaces with Malicious Noise. *Journal of Machine Learning Research*, 10(12).

Laskov, P.; et al. 2014. Practical evasion of a learning-based classifier: A case study. In *IEEE Symposium on Security and Privacy*, 197–211.

Levanon, S.; and Rosenfeld, N. 2021. Strategic classification made practical. In *International Conference on Machine Learning*, 6243–6253.

Li, B.; and Vorobeychik, Y. 2018. Evasion-robust classification on binary domains. *ACM Transactions on Knowledge Discovery from Data*, 12(4): 1–32.

Liu, C.; Li, B.; Vorobeychik, Y.; and Oprea, A. 2017. Robust linear regression against training data poisoning. In *ACM Workshop on Artificial Intelligence and Security*, 91–102.

Lowd, D.; and Meek, C. 2005. Adversarial learning. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 641–647.

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*.

Maiorca, D.; Corona, I.; and Giacinto, G. 2013. Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious pdf files detection. In *ACM SIGSAC Symposium on Information, Computer and Communications Security*, 119–130.

Mei, S.; and Zhu, X. 2015. Using machine teaching to identify optimal training-set attacks on machine learners. In *AAAI Conference on Artificial Intelligence*.

Milli, S.; Miller, J.; Dragan, A. D.; and Hardt, M. 2019. The social cost of strategic classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 230–239.

Myerson, R. B. 1989. Mechanism design. In *Allocation, Information and Markets*, 191–206. Springer.

Patrini, G.; Rozza, A.; Krishna Menon, A.; Nock, R.; and Qu, L. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1944–1952.

Tong, L.; Li, B.; Hajaj, C.; Xiao, C.; Zhang, N.; and Vorobeychik, Y. 2019. Improving robustness of ML classifiers against realizable evasion attacks using conserved features. In *USENIX Security Symposium*, 285–302.

Ustun, B.; Spangher, A.; and Liu, Y. 2019. Actionable recourse in linear classification. In *ACM Conference on Fairness, Accountability, and Transparency*, 10–19.

Verma, S.; Dickerson, J.; and Hines, K. 2020. Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*.

Vorobeychik, Y.; and Kantarcioglu, M. 2018. *Adversarial Machine Learning*. Morgan & Claypool Publishers.

Wang, B.; Jia, J.; Cao, X.; and Gong, N. Z. 2021. Certified robustness of graph neural networks against adversarial structural perturbation. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1645–1653.

Wu, T.; Tong, L.; and Vorobeychik, Y. 2020. Defending against physically realizable attacks on image classification. In *International Conference on Learning Representations*.

Xiao, C.; Zhu, J.-Y.; Li, B.; He, W.; Liu, M.; and Song, D. 2018. Spatially transformed adversarial examples. In *International Conference on Learning Representations*.

Zhang, H.; and Conitzer, V. 2021. Incentive-aware PAC learning. In *AAAI Conference on Artificial Intelligence*, 5797–5804.

Zhang, W. E.; Sheng, Q. Z.; Alhazmi, A.; and Li, C. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology*, 11(3): 1–41.

Zhou, K.; Michalak, T. P.; Rahwan, T.; Waniek, M.; and Vorobeychik, Y. 2019. Attacking Similarity-Based Link Prediction in Social Networks. In *International Conference on Autonomous Agents and Multiagent Systems*.

Zhou, K.; Michalak, T. P.; and Vorobeychik, Y. 2019. Adversarial robustness of similarity-based link prediction. In *IEEE International Conference on Data Mining*, 926–935.

Zügner, D.; Borchert, O.; Akbarnejad, A.; and Günnemann, S. 2020. Adversarial attacks on graph neural networks: Perturbations and their patterns. *ACM Transactions on Knowledge Discovery from Data*, 14(5): 1–31.