# Adversarial Learning: A Critical Review and Active Learning Study

David J. Miller[1]    Xinyi Hu [1]    Zhicong Qiu[1]    George Kesidis[1]

djm25@psu.edu    xzh106@psu.edu    zzq101@psu.edu    gik2@psu.edu

[1]School of Electrical Engineering and Computer Science, Pennsylvania State University, University Park, PA 16802 USA

**Abstract**

This papers consists of two parts. The first is a critical review of prior art on adversarial learning, identifying some significant limitations of previous works. The second part is an experimental study considering adversarial active learning and an investigation of the efficacy of a mixed sample selection strategy for combating an adversary who attempts to disrupt the classifier learning.

**Index Terms**

adversarial learning, active learning, reverse engineering a classifier, sample selection, mixed strategy

## I. INTRODUCTION

While there is still skepticism concerning the value of machine learning (ML) for network security [22], there has been growing interest in the "dual" problem of investigating the *security* of ML systems, as applied both to *security-sensitive* applications (network intrusion detection systems (NIDS), biometric authentication, email), as well as more generally (image, character, and speech recognition, document classification), e.g. [12], [9], [14]. Much of the focus is on attempting to degrade or foil supervised classifiers, as well as anomaly detectors (ADs). [9], [12] provide a useful taxonomy for various attacks on classifiers, including whether they affect training (what we will call *tampering*) or just testing/use (*foiling*). Moreover, if the attack is on training, one can distinguish attacks that involve *mislabeling* from those that add *correctly* labeled examples, but ones with contrived features chosen to bias learning.

[9] demonstrated naive Bayes spam filters can easily be degraded by labeled spam examples (from known spam sources) which use many tokens that commonly appear in normal (ham) email (an "indiscriminate dictionary attack"). This attack on training destroys discrimination power of most tokens in the "dictionary". Huge false positive rates ensued when as little as 5% of training data consisted of these contrived emails. [12] considered active learning (AL), a promising framework for security applications, as the classifier can adapt to track evolving threats and also because oracle labeling may discover *novel* classes [18], [19] that may be zero-day threats. [12] demonstrated, using SVMs, that if an adversary "salts" the unlabeled data batch in a *biased* fashion near the current decision boundary (where AL seeks to choose samples for labeling), one can induce classifier degradation – each adversarial sample was chosen such that, if labeled, it will decrease accuracy the most. We will discuss in the sequel that the approach in [12] appears to rely on (oracle) *mislabeling*, even though frequent mislabeling is not too realistic in practice. We will also demonstrate experimentally that such mislabeling is not necessary in order for the attacker to degrade classification accuracy.

[14] considers classifier testing/operation, constructing examples that will be classified *differently* by a classifier than by a human being. This is an attack, e.g. in the context of human and autonomous drivers, where one sees a STOP sign and the other does not, or in future man-machine interactions where robot servants may misunderstand their master's commands. [14] showed that one could make *relatively* small perturbations to images of digits (*presumably* below human visual detectability, although we discuss this further in the sequel) that alter a deep neural net's decision. Effectively, [14] minimally "pushes" patterns across the decision boundary. While not recognized in [14], their perturbation approach is related to boundary-finding algorithms (neural network inversion) [7], [4]. Foiling has also been performed in other domains, e.g. against detecting malware in PDF documents: synthetic "mimicry" attacks [21], [23] or natural documents with embedded malware [11] ("reverse" mimicry, as a trojan).

In the related study [3], the attacker generates "obfuscated" voice commands typically perceived as background noise by any human being who happens to be listening, but which are recognized as valid commands by a speech recognition system. Such commands could be used e.g. for financial fraud or to perpetrate a terrorist attack (controlling a crane, a train, etc.). To defend against this, they suggest, e.g.: a challenge (e.g. CAPTCHA, reactive two-factor authentication); supervised (speaker-dependent) training to recognize only authorized individuals; password[1], speaker/voice authentication, or some other kind of proactive two-factor authentication; training a classifier to discriminate between computer-generated obfuscated commands and nominal human commands; increasing command specificity. Also, disadvantages of these defenses are discussed, e.g. latency and additional human effort associated with e.g. a CAPTCHA for each utterance[2].

Regarding the latter defenses, the idea is to reduce the allowed variation of utterances classified to each word. Thus, effectively, an "unrecognized" (anomalous) class surrounds each word in feature (e.g. cepstral or wavelet coefficient) space. Note that such AD in speech recognition is not new, e.g. [1], [2]. Also, iPhone's Siri voice interface added "individualized speech" (voice/speaker) recognition in 2015 [17] (before [3] but, again, this is very old technology, and easily configured upon purchase of the phone).

Still, the threats posed by the attacks like [3] need to be addressed[3]. Supervised learning to discriminate the attack from "valid" speech, which assumes either that *labeled* examples of the attack (and a sufficient *number* of these) have been captured *or* that the attack strategy is *known* (so that labeled examples can be synthesized) may not be realistic[4]. Alternatively, we will advocate for an AD defense, which requires *neither* detailed knowledge of the attack *nor* (up front) labeled examples of same.

All of the above research works purport to demonstrate "security holes" in ML techniques. These examples are provocative and they motivate further research. However, these studies also make convenient assumptions, e.g. about the attacker's knowledge, which may be *grossly* unrealistic. Moreover, these studies ignore *existing* ML techniques that are much more resilient to adversaries, even *without* considering *explicit* (and well known) defenses, which may detect (and thus foil) the attack. We next identify key limitations of some prior adversarial learning works.

**Unknown Classes and Authentication:** [9], [12], [14] assume each pattern *must* be classified to one of a *known* set of categories. In many systems, in fact, there is an augmented class space with an "unknown" (unrecognized) category. Patterns not confidently assigned to any known category may be assigned "unknown" – the attack examples in [14] could be assigned thus[5]. Moreover, in an AL setting, human oracles who cannot confidently classify selected samples may *reject* the sample (and thus the

---

[1]Recently, a child used Amazon Echo to place an order - the parents had not set up controls (a password) to prevent this [6].

[2]This is completely reasonable in some cases (e.g. accessing bank accounts or entering passwords), but may be considered very inconvenient in others (e.g. casual web surfing).

[3]As do other security concerns involving Siri, e.g. [5], though it is not clear that one could use Siri to input data into web pages via Safari on the iPhone.

[4]The attacker may have great degrees of freedom he can apply to create inobtrusive examples – it may be both difficult and impractical to try to create representative supervising examples "covering" all of these possibilities.

[5]For example, Siri responds "don't know" to speech it does not recognize as an english word, where every english word (or word-tense combination) here corresponds to a known category Siri has been trained to recognize.

attack). Likewise, emails that are part of an indiscriminate dictionary attack could easily be detected as anomalous even relative to the existing "spam" class, could thus be labeled as "unknown", rather than "spam", and then would not be used to help learn (i.e. corrupt) the "spam" model. Also, particularly in IDS settings with *unknown unknowns*, the full complement of classes is *a priori* unknown and one may *discover* new classes in an unlabeled or semisupervised data batch [13]. This is especially true in an AL context, starting from few labeled examples [18]. Finally, many security applications in fact involve *authentication*, *not* classification per se – [14], [3] assume the problem is classification. The difference between these problems is well-known – the latter *assumes* that a datum originates from one of the $N$ known classes whereas the former allows for the possibility that the datum is not associated with a known class – assignment to the "unknown unknown" class amounts to AD. As an obvious example, consider a perimeter authenticator (access controller) based on a challenge for a userid and password; here the number of correct responses ($N$ authorized individuals) is miniscule compared to the total number of possible inputs, the vast majority of which result in failed access. Moreover, authentication (at least to enter the perimeter) may require an *exact* (password) match for access. This is essentially an extreme example of giving emphasis to class specificity. Biometric based challenges (with well-known "replay attack" protections) can be used instead to explore usability/security trade-offs, or to further secure access based on passwords.

**Unrealistic Assumptions:** [9], [14], and [3] assume the classifier, both its structure *and its learned parameters*, are known to the attacker. [12] further assumes that the *true* joint (feature vector, class) distribution is *also* known to the attacker. Knowledge of this distribution is usually the "holy grail" in ML (i.e. it is *never* assumed known) – given it, one can form the Bayes-optimal classifier. In practice, *at best*, one can only imperfectly *estimate* this joint distribution, given a finite training set. So this latter assumption is wholly unrealistic even for an inside attacker. Even ignoring this latter assumption, the former one – full knowledge of the classifier – is really only reasonable under two cases: 1) a *non-security* setting, where public-domain classifiers *might* be used; 2) in a "security" setting, if the attacker is in fact an *insider*, e.g. if they work for the company that designed the classifier. Otherwise, in e.g. biometric authentication, where one seeks to gain access to sensitive or restricted resources, there is *zero* incentive to publicize many details of the authentication system. [12] *further* assumes that the classifier is precisely known *after every round of AL oracle labeling*. Note that such knowledge might only be obtainable by intensive probing of the system (to "relearn" the decision boundary) *in between* active labelings. Such probing may be very resource-intensive and must complete within the limited time window between labelings. Moreover, [12] considers just a 2-D example. The number of probes needed to accurately learn the decision boundary will grow with the feature dimensionality – consider, e.g. document or image domains, where one may easily work with tens to hundreds of thousands of features.

**Asymmetry: "Straw Man" vs. A Robust System:** While [9], [12] discuss possible defenses, the example attacks in [9], [12], [14] are on *completely defenseless* systems, as well as *inherently* vulnerable ones. Even without explicitly building defenses, there are ML techniques that are widely used and which *also* (as a side benefit) make the system robust to exploits. Consider the naive Bayes (NB) spam filter attacks [9], including the "indiscriminate dictionary" attack and the *red herring* attack, applicable both to email and to NIDS. Here, spurious tokens are introduced into samples that come from a known spam source (e.g. an email address recognized as a source for spam). Once the NB classifier "takes the bait" and adapts its classifier to focus on these (apparently) discriminating tokens, subsequent spam emails (from various addresses) *omit* these tokens and thus avoid detection. The reason such attacks are successful is because NB is a *weak* classifier, building only a *single* model to represent a spam class that may (in a time-varying fashion) exhibit great diversity – i.e. NB effectively puts all its eggs in one basket. Suppose, rather than a single NB model, that one uses a *mixture* of NB models, with new mixture components introduced in a time-varying fashion, as needed, to well-model patterns that are not well-fit by the existing model. Such a mixture can capture and *isolate* a red herring attack within a single (new) NB component. Thus, the main (legitimate) NB components representing the spam class will not be corrupted by the attack. Moreover, once the attack is over, the probability *mass* of this component will dwindle and this

component can eventually be removed. Thus, we suggest a dynamic mixture, responsive to attacks (as well as time-varying classes), which should be highly robust to red herring and indiscriminate dictionary attacks. Other ML techniques that may help achieve attack robustness include e.g. ensemble classification – here, *uncompromised* classifiers may compensate for compromised ones, helping to achieve robust decisions via ensemble decision fusion.

The attacks in [12], [14] could likely be defeated by relatively simple AD defenses. For example, the AL attack in [12] "tricks" the classifier to select for labeling *biased*, attacker-generated samples near the current decision boundary. To maximize the success of the attack, these synthetic examples should be chosen for active labeling with *high prevalence*. Moreover, as elaborated in the next section, this attack appears to rely on the assumption that the oracle may *mislabel* samples near the true (optimal) decision boundary. As demonstrated in the next section: 1) a successful attack does not require oracle mislabeling; 2) irrespective of whether there is such mislabeling, attacks can be defeated by using a *mixed* strategy for AL sample selection (not always selecting the sample nearest the current decision boundary), and with no significant loss in the efficiency and accuracy of classifier learning.

Likewise, [14] effectively assumes that if the *number* of altered features (image pixels, for character recognition) is below a fixed threshold, the attack will not be detectable by a human being. First, this is questionable, as the resulting salt and pepper noise is *quite* visible (see Fig. 1 in [14]) and is *not* typical of the original (clean) images in the database[6]. Second, whether or *not* the attack is perceptible to a human, it may be *easily* detected by an AD defense. Peculiarly, [14] limits the number of features (pixels) one is allowed to alter. In so doing, to induce a classification error, the magnitudes of the perturbations of the chosen features must be substantial[7]. Accordingly, an AD may easily detect salt and pepper noise pixels as anomalous, relative to intensity values of pixels in a surrounding local spatial neighborhood. Finally, we note that if a human and machine *do* disagree on an example – but if they can *share* their decisions – then the introduction of these "attack" examples becomes an *opportunity* – to actively learn – so as to rectify the machine's decision on such examples[8].

**Tampering Power:** In [9], while the authors *limit* the power of the attacker to corrupt training data (5% of email training data), it is unclear that this level of tampering power is plausible/corresponds to a realistic scenario. In principle, if the attacker is an insider, she may have unlimited capability to corrupt training data, in which case there may be no adequate defense. Otherwise, the chosen power of the attack may intricately depend on knowledge of the particular defense system in play, i.e. on the attacker's *tradeoff* between achieving high attack potency and minimizing the probability of the attack's detection and thwarting. Even if only a few training samples are altered, this tradeoff may exist. For example, in [25], tampering with a single support vector is show to dramatically degrade classification accuracy. However, to achieve such effect, the tampered sample may become an extreme outlier of its class, and thus may either be ignored (via use of margin slackness) or may be detectable as a suspicious sample.

**Reverse Engineering:** [14] (strongly) assumed that the classifier structure and its parameter values are known to the attacker. Recent works [24], [15] have proposed techniques to reverse-engineer a (black box) classifier without necessarily even knowing its structure. In [24], the authors consider black box machine learning *services*, offered by companies such as Google, where, for a given (presumably big data, big model) domain, a user pays for class decisions on individual samples (queries) submitted to the ML service. [24] demonstrates that, with a *relatively* modest number of queries (perhaps as many as ten thousand or more), one can *learn* a classifier on the given domain that closely mimics the black box ML service decisions. Once the black box has been reverse-engineered, the attacker need no longer

---

[6]Human subject testing *was* used in [14]. However, the authors did *not* ask respondents whether they thought images had been tampered with –they only asked them to classify the images.

[7]Even though the authors impose the *minimum* norm perturbations needed to induce classification errors (note again that the introduced salt and pepper noise is quite visible).

[8]This assumes that the human is more accurate than the machine. In some application domains, this is certainly the case. In domains where this is not the case, such disagreement may by the same token (appropriately) cause the human to reconsider their judgement.

subscribe to the ML service. One weakness of [24] is that it neither considers very large (feature space) classification domains nor very large networks (deep neural networks (DNNs)) – *orders* of magnitude more queries may be needed to reverse-engineer a DNN on a large-scale domain. However, a much more critical weakness of of [24] stems from one of its (purported) greatest *advantages* – the authors tout that their reverse-engineering does not require *any* labeled training samples from the domain[9]. In fact, in [24], the attacker's queries to the black box are *randomly* drawn, e.g. uniformly, over the given feature space. While such random querying is demonstrated to achieve reverse-engineering, what was not recognized in [24] is that this random queryinig makes the attack *easily detectable* by the ML service – randomly selected query patterns will typically look nothing like legitimate examples from any of the classes – they are very likely to be extreme outliers, of all the classes. Each such query is thus *individually* highly suspicious by itself – thus, even tens, let alone thousands of such queries will be trivially detected as jointly improbable under a null distribution (estimable from the training set defined over all the classes from the domain). Even if the attacker employed bots, each of which makes a small number of queries (even as few as ten), each bot's random queries should be easily detected as anomalous, likely associated with a reverse-engineering attack.

We next perform an experimental study involving adversarial active learning.

## II. ACTIVE LEARNING EXPERIMENTAL SETUP

In [12], as noted earlier, the attacker can perfectly estimate the current decision rule after every AL round (may require lots of probing between rounds) and knows the true joint density on the feature vector and class label $p(Y, X)$ (wholly unrealistic). It was furthered assumed that the attacker knows the AL sample selection strategy (uncertainty sampling) and injects one new (high decision uncertainty) sample into the unlabeled batch at each AL round, crafted so that it will be chosen by the oracle for labeling. Moreover, from the description given in [12], the authors do *not* assume that the oracle labels the attacker's sample consistent *either* with the Bayes-optimal decison rule *or* randomly, according to the true class posteriors. That is, in [12], although it is not very clearly stated, the oracle may *mislabel* the samples crafted by the attacker. This is crucial to the success of the attack in [12].

In this paper, we propose a more realistic framework, under which the attacker still possesses the ability to degrade classification accuracy even without the unrealistic oracle mislabeling assumption. However, we also demonstrate that attacks on AL can be defeated by a *mixed* sample selection strategy, through which the attacker's injected samples are not *so* frequently chosen for oracle labeling. Moreover, this mixed strategy does not make a significant sacrifice either in classifier accuracy or learning convergence (number of queries needed to achieve good accuracy). In fact, this strategy is also suitable for defeating the attack even in the presence of oracle mislabeling (though we focus on more realistic oracle labeling in the sequel).

### A. Preliminaries

As in [12], we assume a two-class problem and use a two-class linear support vector machine. We first select a (large) (unlabeled) training pool ($T_r$), then randomly select a relatively small number of samples from both classes in $T_r$ and assign (ground-truth) class labels to them[10]. One half of this labeled subset is used as a labeled training set ($T_l$) to train the initial SVM classifier. The other half of this labeled subset is used as a "validation" set ($V$), to estimate an approximate class posterior for the SVM, using the approach described in [16]. The (large) remainder in $T_r$ is taken as the pool of unlabeled samples available to the active learner ($T_u$), from which the active learner selects for Oracle labeling (and into

---

[9]For certain sensitive domains, or ones where obtaining real examples is expensive, the user may in fact have no realistic means of obtaining a significant number of real data examples from the domain. This is one main reason why the ML service is needed in the first place – the company or its client for this domain are the (exclusive) owners of this (labeled, precious) data resource.

[10]For our synthetic data experiment, consistent with the ground-truth class distributions from [12], these labels are assigned according to the Bayes-optimal rule. For our real-world digits experiment, we use the labels provided with the given data set.

which the attacker inserts adversarial samples). In addition, there is a labeled test set to evaluate classifier accuracy.

## B. Sample Selection Criteria for Active Learning

The active learner selects samples from $T_u$, one by one, for labeling by the oracle, with the SVM classifier retrained after each oracle labeling. We investigate the following AL sample selection strategies:
**Uncertainty sampling:** Uncertainty sampling is the simplest and most commonly used sample selection strategy [20]. In this strategy, the AL chooses the nearest sample in $T_u$ to the current boundary (the most uncertain sample).
**Max-Expected-Utility (MEU):** Sampling to maximize expected utility (classifier gain) [8]. For data $x_i \in T_u$, $i^* = \mathrm{argmax}_i \tilde{U}_i(\theta)$, where
$\tilde{U}_i(\theta) =$

$$\sum_{y_i} p_\theta(y_i|x_i) \frac{1}{N} \left( \sum_{j \in L \cup i} p_{\theta_{+i}}(y_j|x_j) + \sum_{j \in U \setminus i} \sum_{y_j} p_\theta(y_j|x_j) p_{\theta_{+i}}(y_j|x_j) \right). \tag{1}$$

Here, $\theta$ is the current set of class posterior parameters, and $\theta_{+i}$ reflects the updated parameters after adding $x_i$ to $T_l$ with its *putative* label $y_i$. Using Platt probabilistic outputs for SVMs [16], the posterior probability $p_\theta(y_i|x_i)$ is based on a logistic regression approximation to the SVM (hard) decision function. Finally, $N = |T_l| + |T_u|$.
**Random sampling:** Select from $T_u$ according to a uniform distribution. This sample selection acts as a baseline.
**Mixed strategies:** Choose the sample by MEU (or random sampling) with probability $p$; otherwise by uncertainty sampling with probability $1 - p$. Note that a non-zero proportion for uncertainty sampling is warranted because uncertainty sampling is a very good mechanism for discovering unknown classes that may be latently present in $T_u$ [18]. At the same time, using uncertainty sampling plays into the hands of the attacker.

## C. The Attacker

We assume the attacker has the same knowledge as the active learner (i.e. the attacker knows $T_l$, $T_u$, and the current SVM classifier boundary). The attacker adds one sample to $T_u$ at each active sample selection round, chosen as follows: 1) he *projects* all training pool samples ($T_u$ and $T_l$) onto the current decision boundary (hence creating a (rich) candidate pool of high uncertainty samples), as shown in the figure below; 2) the attacker injects into $T_u$ the candidate from this pool with *minimum* expected utility, based on the expected utility objective given above.
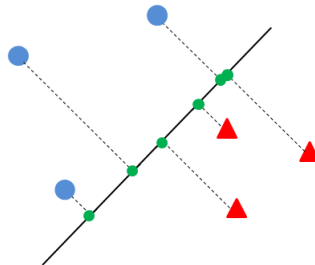


Fig. 1: Candidate adversarial samples (green dots on the boundary)

## III. Synthetic Data Experiments

### A. Dataset

We first consider the two-dimensional feature space problem from [12]. The data $X$ is generated such that the class 1 instances have a bivariate normal distribution centered at (2,0), with the class 2 instances bivariate normal centered at (-2,0); both classes use an isotropic (identity) covariance matrix, and the classes are equally likely. We generated 105 instances from each of the two classes as $T_r$ initially, from which we drew 5 samples at random from each class to form $T_l$, and another 5 samples from each class to form $V$. The remaining 190 instances were taken as $T_u$. We also generated 200 instances from each class as test set. This dataset is (obviously) not linearly separable. The optimal decision boundary is the $Y$ axis. Our oracle deterministically assigns labels consistent with this optimal decision boundary.

### B. Results

First, we conducted a single experimental trial, for which AL selects samples strictly using uncertainty sampling; thus, the attacker's adversarial samples are selected and labeled at each round[11]. Fig. 2 shows that the labeled adversarial samples induce a decision function that deviates from the optimal rule (the boundary becomes tilted from vertical). Hence, the attacker does have the ability to degrade AL classification accuracy (even without any oracle mislabeling).



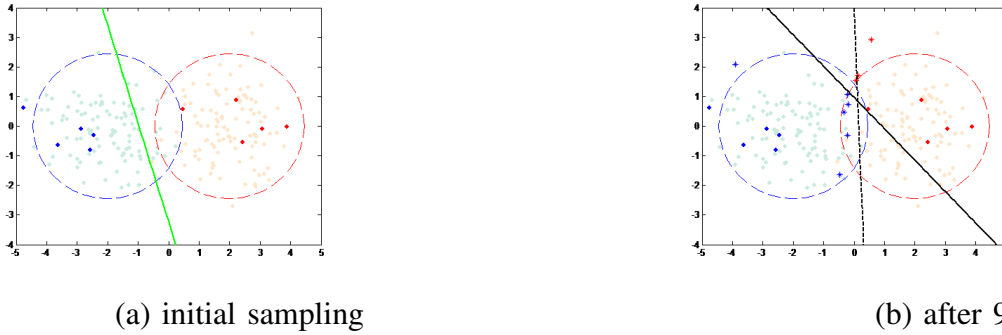(a) initial sampling                (b) after 9 queries

Fig. 2: Classifier decision boundary after initial training and after 9 queries without attack (dash lines) and with attack (solid lines).

In the following experiment, we performed 10 random trials. In each trial, we used the same $T_r$, but randomly chose the initial $T_l$ and $V$ from $T_r$. We computed average performance for different AL strategies, over these 10 trials.

Fig. 3 and Fig. 4 show performance both in the presence of and in the absence of the attack, respectively. Different sample selection strategies show different abilities to defeat the attack, as shown in Fig. 3. We have the following observations:

- When AL uses strict uncertainty sampling $(p = 0)$, adversarial samples degrade classification accuracy successfully for the first 15 queries, which is consistent with single-trial results. However, the fabricated samples the attacker inserts do not degrade performance in the long run. Thus, the attack effectively only delays convergence to a good decision boundary[12]. When the current boundary is very close to the optimal one, if the attacker still adds fabricated samples to the current boundary, these samples may even be helpful to refine the boundary. As shown in the sequel, however, the attack is more successful on a real-world, high-dimensional digit recognition domain.

---

[11]Note, though, that, unlike [12], the oracle does not perform any mislabeling.
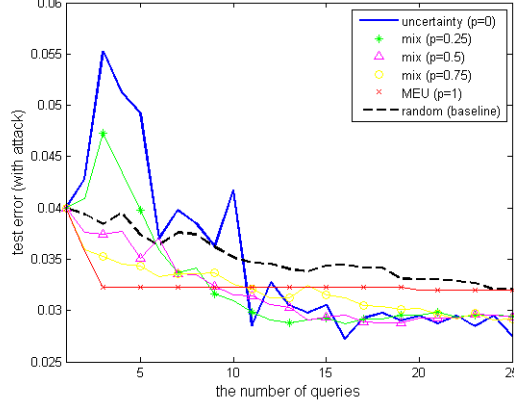[12]Mislabeling would likely allow *perpetuated* accuracy degradation.

Fig. 3: Average test error performance of different strategies with attack on the synthetic dataset.
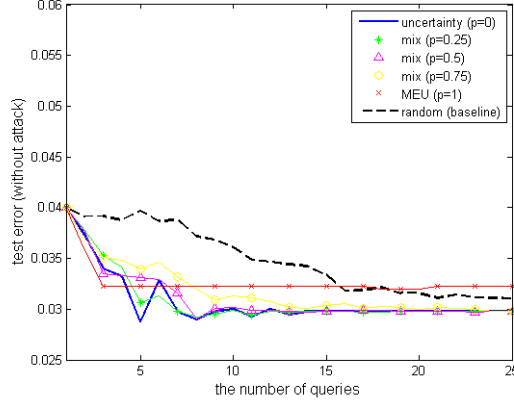


Fig. 4: Average test error performance of different strategies without attack on the synthetic dataset.

- Fig. 3 indicates MEU ($p = 1$) is not substantially affected by adversarial samples (as one would expect, since the MEU sample selection criterion is the antithesis of the adversary's sample generation strategy), and makes the test error decrease the most at the beginning. However, we also noticed that MEU selects samples in a class-biased fashion (many samples from one class), leading to a biased active learner after multiple queries. It is also clear from Fig.4 and Fig. 3 that MEU converges to a suboptimal decision boundary in the long run.
- With the attack present, the mixed strategies shows improved accuracy for increasing $p$, $p \in [0.25, 0.75]$. Moreover, in the absence of the attack, there is little accuracy difference for different choices of $p$.
- Note also that the random strategy is robust to the attack, but does not converge to a solution as accurate as the mixed strategies.

## IV. HANDWRITTEN DIGIT EXPERIMENTS

### A. Dataset

We used the MNIST dataset, consisting of $28 \times 28$ pixel grayscale images, learning a linear SVM to discriminate between the digits "5" and "6", that is, we have 784 (pixel) features and a two-class problem. We initially chose 105 "5" digits and 105 "6" digits as $T_r$ and again labeled 5 samples from each class to form $T_l$, and 5 samples from each class to form $V$. The remainder of $T_r$ was taken as $T_u$. Also, we randomly chose another (distinct set of) 456 "5" and 462 "6" samples to form a test set. Our oracle is

8

assumed to be an SVM trained on the entire data set. Since the entire data set is linearly separable, this is a plausible choice for the oracle. Note, also, that because the data set is linearly separable, this oracle assigns *ground-truth* labels to all original data samples that are chosen for oracle labeling – the oracle only manufactures labels for the adversarial samples that are selected for labeling (and in this case it does so objectively, consistent with maximum margin linear separation of the entire data set).

### B. Results

As described in II-C, candidate adversarial samples are the projections of all samples in $T_l$ and $T_u$ onto the current boundary. Some candidates shown in Fig. 5 involve superposition of the two digits – labeling such samples and subsequent classifier retraining is expected to have a (negative) impact on the classifier decision boundary.



Fig. 5: Digit projections on the boundary, Oracle labeling (from top to bottom, from left to right): 6, 6, 5, 6

Again, we performed 10 random trials to get the average performance with/without attack in Fig. 6 and Fig. 7, respectively.
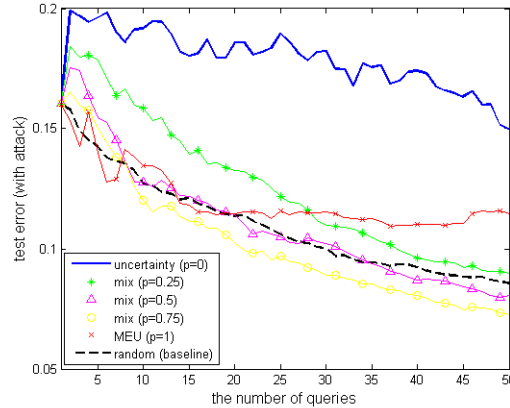


Fig. 6: Average test error performance of different strategies with attack on Digit Dataset.

To summarize the results:
- Using uncertainty sampling $(p = 0)$, AL always selects the attacker's injected samples, and is the best strategy without attack, but worst with attack. Note also that the attack much more substantially delays learning progress for this high-dimensional domain, compared with the 2-D example.
- MEU $(p = 1)$ makes the test error decrease the most at the beginning, but it converges to a suboptimal decision boundary, both with and without the attack.
- The mixed strategies have a good defensive capability against the attack (by sometimes using MEU), and they also alleviate the unbalanced selection problem of MEU by sometimes using uncertainty
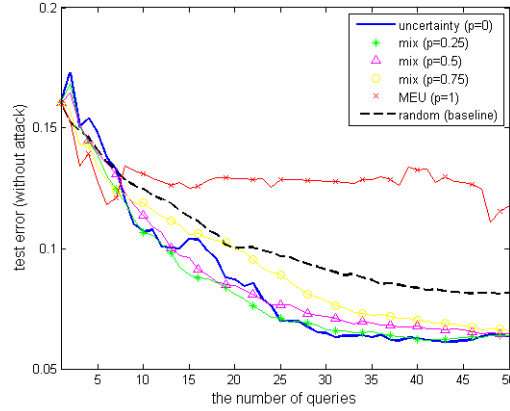
9

Fig. 7: Average test error performance of different strategies without attack on Digit Dataset.

sampling. With the attack, there is monotonically improving performance with $p$, for $p \in [0.25, 0.75]$ – using the mixed strategy with $p = 0.75$, Fig. 6 shows the test error has a fast and steady decrease. All the mixed strategies perform similarly without the attack.

- The random strategy fares well with the attack, but not when the attack is absent.

## V. FUTURE WORK

In future work, we will investigate some of the other adversarial learning defenses suggested in our review section. We may also investigate alternative AL sample selection strategies – e.g., a modification of the MEU strategy that does not suffer from the biased sampling we observed in our experiments (This may be achieved by estimating class proportions and modifying the MEU strategy to maximize expected utility, but while also sampling consistently with these class prior estimates.). Further, we will continue to study mixed strategies to discover the unknown classes.

## REFERENCES

[1] L. Atlas, D. Cohn, and R. Ladner. Training connectionist networks with queriers and selective sampling. In *Advances in Neural Information Systems Processing*, 1990.

[2] N. Borges and G.G.L. Meyer. Unsupervised distributional anomaly detection for a self-diagnostic speech activity detector. In *Proc. Conference on information Sciences and Systems*, 2008.

[3] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou. Hidden voice commands. In *Proc. 25th USENIX Security Symposium*, Austin, TX, August 2016.

[4] D.T. Davis and J.-N. Hwang. Solving Inverse Problems by Bayesian Neural Network Iterative Inversion with Ground Truth Incorporation. *IEEE Trans. Sig. Proc.*, 45(11), Nov. 1997.

[5] J. Edwards. There's A Huge Password Security Quirk In iOS 7 That Lets Siri Control Your iPhone. http://www.businessinsider.com/password-security-flaw-in-ios-7-lets-siri-control-your-iphone-2013-9, Sep. 24, 2013.

[6] R. Hackett. Amazon Echo's Alexa Went Dollhouse Crazy. http://fortune.com/2017/01/09/amazon-echo-alexa-dollhouse/, Jan 09, 2017.

[7] D.A. Hoskins, J.-N. Hwang, and J. Vagners. Iterative inversion of neural networks and its application to adaptive control. *IEEE Trans. Neural Networks*, 3, Mar. 1992.

[8] T.M. Hospedales, S. Gong, and T. Xiang. A unifying theory of active discovery and learning. In *European Conference on Computer Vision*, LNCS 7576, pp. 453-466, 2012

[9] L. Huang, A.D. Joseph, B. Nelson, B.I.P. Rubinstein, and J.D. Tygar. Adversarial machine learning. In *Proc. 4th ACM Workshop on Artificial Intelligence and Security (AISec)*, 2011.

[10] Y. LeCun, C. Cortes, and C.J.C. Burges. The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.

[11] D. Maiorca, I. Corona, and G. Giacinto. Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious PDF files detection. In *Proc. ACM SIGSAC*, Hangzhou, China, May 2013.

[12] B. Miller, A. Kantchelian, S. Afroz, R. Bachwani, E. Dauber, L. Huang, M.C. Tschantz, A.D. Joseph, and J.D. Tygar. Adversarial active learning. In *Proc. Workshop on Artificial Intelligence and Security (AISec)*, 2014.

[13] D.J. Miller and J. Browning. A mixture model and EM-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. *IEEE Trans. on Pattern Anal. and Machine Intell.*, pages 1468–1483, 2003.

[14] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *Proc. 1st IEEE European Symp. on Security and Privacy*, 2016.

[15] N. Papernot, P. D. McDaniel, I.J. Goodfellow, S. Jha, Z.B. Celik, and A. Swami. Practical Black-Box Attacks against Deep Learning Systems using Adversarial Examples. *CoRR*, abs/1602.02697, Nov. 2016.

[16] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, MIT Press, 1999.

[17] D. Price. Complete guide to Siri: How to set up Siri in iOS 9 & train it to recognise your voice. http://www.macworld.co.uk/feature/iosapps/complete-guide-siri-ios-features-commands-questions-voices-ios-9-apple-music-proactive-3495151, Oct. 22, 2015.

[18] Z. Qiu, D.J. Miller, and G. Kesidis. A maximum entropy framework for semisupervised and active learning with unknown and label-scarce classes. *IEEE Trans. on Neural Networks and Learning Systems*, 28(4), 917-933.

[19] Z. Qiu, D.J. Miller, and G. Kesidis. Flow based botnet detection through semi-supervised active learning. In *Proc. IEEE ICASSP, New Orleans*, March 2017.

[20] B. Settles B. Active learning literature survey. University of Wisconsin, Madison, Computer Sciences Technical Report 1648, Available at http://burrsettles.com/pub/settles.activelearning.pdf, Jan. 26, 2010.

[21] C. Smutz and A. Stavrou. Malicious PDF Detection using Metadata and Structural Features. In *Proc. ACSAC*, Orlando, FL, Dec. 2012.

[22] R. Sommer and V. Paxson. Outside the Closed World: On Using Machine Learning For Network Intrusion Detection. In *Proc. IEEE Symposium on Security and Privacy*, 2010.

[23] N. Srndic and P. Laskov. Detection of malicious PDF files based on hierarchical document structure. In *Proc. NDSS*, 2013.

[24] F. Tamer, F. Zhang, A. Juels, M.K. Reiter, and T. Ristenpart. Stealing Machine Learning Models via Prediction APIs. In *USENIX Security Symposium*, Austin, TX, Aug. 2016.

[25] H. Xiao, B. Biggio, B. Nelson, H. Xiao C. Eckert, and F. Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160(C):53–62, July 2015.