DATA POISONING ATTACKS AGAINST MRMR

Heng Liu and Gregory Ditzler

The University of Arizona
Dept. of Electrical & Computer Engineering
Tucson, AZ 85721
{hengl, ditzler}@email.arizona.edu

ABSTRACT

Many machine learning models lack the consideration that an adversary can alter data at the time of training or testing. Over the past decade, the machine learning models' vulnerability has been a concern and more secure algorithms are needed. Unfortunately, the security of feature selection (FS) remains an under-explored area. There are only a few works that address data poisoning algorithms that are targeted at embedded FS; however, data poisoning techniques targeted at information-theoretic FS do not exist. In this contribution, a novel data poisoning algorithm is proposed that targets failures in minimum Redundancy Maximum Relevance (mRMR). We demonstrate that mRMR can be easily poisoned to select features that would not normally have been selected.

Index Terms— Feature Selection, Information Theory, Adversarial Learning

1. INTRODUCTION

Machine learning has been shown to be tremendously successful in areas such as signal processing [1], spam filtering [2, 3], malware detection [4, 5], etc. Unfortunately, the security of machine learning models didn't draw attention until recently. In 2006, the security of the spam filtering model in an adversarial environment was questioned for the first time [6]. Therefore, more secure machine learning algorithms are in high demands. Note that one important aspect of machine learning is the preprocessing of the data and, unfortunately, techniques such as feature selection (FS) has been largely omitted from much of the adversarial learning literature. Furthermore, it is worth noting that FS plays a crucial role in many tasks related to machine learning and data science.

The vulnerability of machine learning along with countermeasures against an adversary has been extensively studied for over ten years with some of the earliest works being [7]. Biggio et al. proposed a taxonomy to model the adversary's behavior from three areas: *goal*, *knowledge* and *capability* [8]. Generally, an adversary seeks to violate the *security*, *availability* or *privacy* of machine learning models through *targeted* (white-box) or *indiscriminate* (black-box) attacks. The adversary's knowledge may vary from *Perfect Knowledge* (P-K) to *Limited Knowledge* (L-K) w.r.t. a model's components (e.g., training data, feature set, learning algorithm, etc.). Moreover, the adversary may have different levels of control over and access to the training and testing data.

There are typically two types of attacks: poisoning and evading. Poisoning attacks seek to contaminate the training data to mislead the learning algorithm at training time. Evasion attacks, on the other hand, occur during testing time and aim to evade detection for malicious data. There are many works that focus on algorithms for evading and poisoning a data sample against classifiers [8, 9]; however, far fewer works exist for generating such attacks against feature selection [10–12]. In this contribution, we assume the adversary has knowledge of the FS algorithm and has control over the training data (e.g., the capability to read and write training data samples).

Generally, FS methods are categorized into three categories: embedded, wrappers (classifier-dependent) and filters (classifier-independent). Xiao et al. shed lights on the vulnerability of Lasso [13] via training poisoning [11,14,15]. Zhang et al' proposed an adversary-aware wrapper algorithm against evasion attacks [12]. In this work, we examine the widely-used mRMR's [16], a filter FS algorithm, security against training poisoning. mRMR uses the Mutual Information [17] criteria to maximize the features' relevancy w.r.t. the class label and minimize the redundancy among selected features. Equation (1) shows the objective that mRMR optimizes. Let feature F_i be a feature that mRMR is evaluating to determine if it should be selected and F_j be the a feature that has already been selected.

$$J_{\text{mRMR}}(F_i) = \overbrace{I(F_i;Y)}^{\text{relevancy}} - \overbrace{\frac{1}{|\mathcal{S}|}\sum_{F_j\in\mathcal{S}} I(F_i;F_j)}^{\text{redundancy}} \qquad (1)$$

The objective in (1) has a trade-off between the relevancy of a feature F_i and its redundancy with all of the other features that have been selected. A feature is selected if it has the largest mRMR score and the feature is added into $\mathcal{S}$. This

process is repeated until the desired number of features have been selected.

2. POISONING MRMR

In this section, we set up the problem of feature selection with an adversary then we propose an algorithm to manipulate mutual information and the FS poisoning algorithm.

2.1. Problem Settings and Notations

Given two random variables (R.V.'s): F and Y, the outcome spaces are $f \in \mathcal{F}$ and $y \in \mathcal{Y}$, respectively (Conventionally, probability theory uses "sample space", we use "outcome space" to avoid confusion with terms such as "data sample"). Similarly, we denote the joint outcome space for R.V. pair (F, Y) as $\Psi = \{(f, y) : (f, y) \in \mathcal{F} \otimes \mathcal{Y}, \mathbb{P}(f, y) \neq 0\}$. Note that Ψ excludes the zero-probability joint outcomes (possibly illegitimate) in $\mathcal{F} \otimes \mathcal{Y}$. Using the above notations, the entropy-based mutual information (MI) of (F, Y) is: $I(F; Y) = H(F) + H(Y) - H(F, Y)$ where entropy is calculated based on probabilities: $H(F) = -\sum_{f \in \mathcal{F}} \mathbb{P}(f) \log \mathbb{P}(f)$ [18].

First, we seek to manipulate (drag down or drive up) the MI of (F, Y) by injecting malicious data samples $D_i \in \widehat{\mathcal{D}}, i > N$ to training data $D_i \in \mathcal{D}$, $1 \leq i \leq N$, where N is the number of training data samples. The injection satisfies the following constraints: (a) we require the marginal outcomes for F and Y among $D_i \in \widehat{\mathcal{D}}$ are from $\mathcal{F}$ and $\mathcal{Y}$, respectively, (b) we require the joint outcome of (F, Y) among the $D_i \in \widehat{\mathcal{D}}$ is from Ψ. The reasoning is that the feature selector can deem an unrecognized outcome/joint outcome as illegitimate based on prior knowledge. We formulate the above requirements mathematically:

- The outcome spaces in $\widehat{\mathcal{D}}$ for F and Y are denoted as $\mathcal{F}_2$ ($\mathcal{F}_2 \subset \mathcal{F}$) and $\mathcal{Y}_2$ ($\mathcal{Y}_2 \subset \mathcal{Y}$), respectively. The complementary subsets are: $\mathcal{F}_1 = \mathcal{F} \backslash \mathcal{F}_2$, $\mathcal{Y}_1 = \mathcal{Y} \backslash \mathcal{Y}_2$.
- (F, Y)'s joint outcome space in $\widehat{\mathcal{D}}$ is denoted as Ψ_2 ($\Psi_2 \subset \Psi$). The complimentary subset is $\Psi_1 = \Psi \backslash \Psi_2$.

2.2. Manipulating Mutual Information

We now analyze how the distributions and MI are influenced by injecting malicious data samples $\widehat{\mathcal{D}}$, then we shed lights on determining $\widehat{\mathcal{D}}$ such that the MI is manipulated as desired. Here we use $\mathbb{P}(\cdot)$/$\mathbb{P}'(\cdot)$ to denote the probabilities before/after injection, respectively. Then we propose the Mutual Information Poisoning (MIP) algorithm.

Consider that the injection of data happens gradually (one by one), we use ρ to denote the incrementally injected data samples number ($\rho = 0, 1, 2, \cdots$). Let $\lambda(\rho) = \frac{N}{N+\rho}$, then we have $\lambda(\rho) \leq 1$ because $\rho \geq 0$. Furthermore, we assume that $\lambda(\rho)$ is a continuous function, which it is not, however, it simplifies the mathematics. We only consider $\lambda(\rho)$ meaningful for ρ taking discrete values. To simplify notations, we use λ directly to imply it is a function. We first write $\mathbb{P}'()$'s for $f \in \mathcal{F}_1$, $y \in \mathcal{Y}_1$ and $(f, y) \in \Psi_1$ in terms of $\mathbb{P}()$'s after ρ data samples injected:

- $\forall f \in \mathcal{F}_1, \mathbb{P}'(f) = \frac{N\mathbb{P}(f)}{N+\rho} = \lambda \mathbb{P}(f)$
- $\forall y \in \mathcal{Y}_1, \mathbb{P}'(y) = \frac{N\mathbb{P}(y)}{N+\rho} = \lambda \mathbb{P}(y)$
- $\forall (f, y) \in \Psi_1, \mathbb{P}'(f, y) = \frac{N\mathbb{P}(f,y)}{N+\rho} = \lambda \mathbb{P}(f, y)$

We still need the partial distributions for $f \in \mathcal{F}_2$, $y \in \mathcal{Y}_2$ and $(f, y) \in \Psi_2$ to calculate $I'(F; Y)$ after injection. $I'(F; Y)$ represents the mutual information after the dataset has been poisoned and $I(F; Y)$ is the mutual information before the dataset has been poisoned. Here we discuss a simple scenario to provide some insights: $|\mathcal{F}_2| = |\mathcal{Y}_2| = |\Psi_2| = 1$ (note that this assumption is not necessarily the final solution). Then the unknown partial distributions after injection can be expressed as follows:

- $\mathcal{F}_2 = \{f^*\}, \mathbb{P}'(f^*) = \lambda \mathbb{P}(f^*) + 1 - \lambda$
- $\mathcal{Y}_2 = \{y^*\}, \mathbb{P}'(y^*) = \lambda \mathbb{P}(y^*) + 1 - \lambda$
- $\Psi_2 = \{(f^*, y^*)\}, \mathbb{P}'(f^*, y^*) = \lambda \mathbb{P}(f^*, y^*) + 1 - \lambda$

We now calculate the individual and joint entropies:

$$
\begin{aligned}
H'(F) &= -\sum_{f \in \mathcal{F}} \mathbb{P}'(f) \log \mathbb{P}'(f) \\
&= -\sum_{f \in \mathcal{F}_1} \lambda \mathbb{P}(f) \log \left(\lambda \mathbb{P}(f)\right) - \sum_{f \in \mathcal{F}_2} \mathbb{P}'(f) \log \mathbb{P}'(f) \\
&= -\sum_{f \in \mathcal{F}_1} \lambda \mathbb{P}(f) (\log \lambda + \log \mathbb{P}(f)) \\
&- (\lambda \mathbb{P}(f^*) + 1 - \lambda) \log \left(\lambda \mathbb{P}(f^*) + 1 - \lambda\right) \\
&= -\lambda \log \lambda \sum_{f \in \mathcal{F}_1} \mathbb{P}(f) - \lambda \sum_{f \in \mathcal{F}_1} \mathbb{P}(f) \log \mathbb{P}(f) \\
&- (\lambda \mathbb{P}(f^*) + 1 - \lambda) \log \left(\lambda \mathbb{P}(f^*) + 1 - \lambda\right) \qquad (2)
\end{aligned}
$$

Similar for $H'(F)$ and $H'(F, Y)$:

$$
\begin{aligned}
H'(Y) &= -\sum_{y \in \mathcal{Y}} \mathbb{P}'(y) \log \mathbb{P}'(y) \\
&= -\lambda \log \lambda \sum_{y \in \mathcal{Y}_1} \mathbb{P}(y) - \lambda \sum_{y \in \mathcal{Y}_1} \mathbb{P}(y) \log \mathbb{P}(y) \\
&- (\lambda \mathbb{P}(y^*) + 1 - \lambda) \log \left(\lambda \mathbb{P}(y^*) + 1 - \lambda\right) \qquad (3)
\end{aligned}
$$

$$
\begin{aligned}
H'(F, Y) &= -\sum_{(f,y) \in \Psi} \mathbb{P}'(f, y) \log \mathbb{P}'(f, y) \\
&= -\lambda \log \lambda \sum_{(f,y) \in \Psi_1} \mathbb{P}(f, y) - \lambda \sum_{(f,y) \in \Psi_1} \mathbb{P}(f, y) \log \mathbb{P}(f, y) \\
&- (\lambda \mathbb{P}(f^*, y^*) + 1 - \lambda) \log \left(\lambda \mathbb{P}(f^*, y^*) + 1 - \lambda\right) \qquad (4)
\end{aligned}
$$

Let Δ and β be the following:

$$\Delta = \sum_{(f,y)\in\Psi_1} \mathbb{P}(f,y) - \sum_{f\in\mathcal{F}_1} \mathbb{P}(f) - \sum_{y\in\mathcal{Y}_1} \mathbb{P}(y)$$

$$\beta = \sum_{(f,y)\in\Psi_1} \mathbb{P}(f,y)\log\mathbb{P}(f,y) - \sum_{f\in\mathcal{F}_1} \mathbb{P}(f)\log\mathbb{P}(f) - \sum_{y\in\mathcal{Y}_1} \mathbb{P}(y)\log\mathbb{P}(y)$$

Then $I'(F;Y)$ can be expressed as:

$$\begin{aligned} I'(F;Y) &= \lambda\log(\lambda)\Delta + \beta\lambda \\ &- (\lambda\mathbb{P}(f^*)+1-\lambda)\log(\lambda\mathbb{P}(f^*)+1-\lambda) \\ &- (\lambda\mathbb{P}(y^*)+1-\lambda)\log(\lambda\mathbb{P}(y^*)+1-\lambda) \\ &+ (\lambda\mathbb{P}(f^*,y^*)+1-\lambda)\log(\lambda\mathbb{P}(f^*,y^*)+1-\lambda) \end{aligned} \quad (5)$$

Recall that we expressed $I'(F;Y)$ as a function $\Phi(\lambda, f^*, y^*)$ of ρ through λ, and of (f^*, y^*) through its distributions. We take partial derivative for $\Phi(\lambda, f^*, y^*)$ w.r.t. λ (note that although $\Phi(\lambda, f^*, y^*)$ is differentiable w.r.t. λ, while it's only meaningful on discrete values of λ).

$$\begin{aligned} \frac{d\Phi}{d\lambda} &= \Delta\log\lambda + \beta + (1-\mathbb{P}(f^*))\log(\lambda\mathbb{P}(f^*)+1-\lambda) \\ &+ (1-\mathbb{P}(y^*))\log(\lambda\mathbb{P}(y^*)+1-\lambda) \\ &+ (\mathbb{P}(f^*,y^*)-1)\log(\lambda\mathbb{P}(f^*,y^*)+1-\lambda) \end{aligned} \quad (6)$$

$$\frac{d\Phi}{d\lambda}|_{\lambda=1} = I(F;Y) - \log\frac{\mathbb{P}(f^*,y^*)}{\mathbb{P}(f^*)\mathbb{P}(y^*)} \quad (7)$$

Equation (6) shows the rate of changing of $I'(F;Y)$ when a particular (f^*, y^*) is injected ρ times ((7) denotes the potential changing rate when $\rho = 0$ and it only depends on the choice of (f^*, y^*) and the original MI $I(F;Y)$). Therefore, the derivative (7) can be used as a guide to choose (f^*, y^*) to inject. We now present our strategy: Firstly, use (7) to find the (f^*, y^*) that contributes to the maximal desired manipulation of mutual information. Then MIP keeps injecting (f^*, y^*) until the changing rate drops below a limit by monitoring (6) (note that a negative derivative implies $I'(F;Y)$ increases as ρ augments because λ decreases as ρ increases, vice versa). Then we find the next (f^*, y^*) using (7) after updating the distribution. This process is repeated until a desired manipulation is achieved or the injected data sample amount exceeds a limit.

The MIP algorithm is given in Algorithm 1. In step 1, MIP takes the following inputs: R.V.'s F, Y of length N; desired MI operation sig ($+1/-1$ means increase/decrease); desired manipulation extent δ; injection limit θ; changing rate limit ϵ. In step 2, the distributions are calculated; the $baseline$ is set to the original MI; ρ and λ are set to 0 and 1; the length before distribution evaluation L is initialized as N. In step 3, we solve for (f^*, y^*) that contributes to the maximal desired manipulation. Step 4 enters a for loop: we continue to inject (f^*, y^*) to F, Y if the rate of changing of mutual information is above the limit ϵ. Then MIP updates the derivative $\frac{d\Phi}{d\lambda}$ w.r.t. updated λ (step 5-8). If the derivative drops below ϵ, MIP recalculates the distributions w.r.t. F, Y and reset $\lambda = 1, L = L + \rho$. The loop repeats the above procedures until a desired manipulation is achieved $|I(F;Y) - standard| < \delta$ or the injected sample amount exceeds a limit $x \le \theta$.

Algorithm 1 MIP pseudo-code

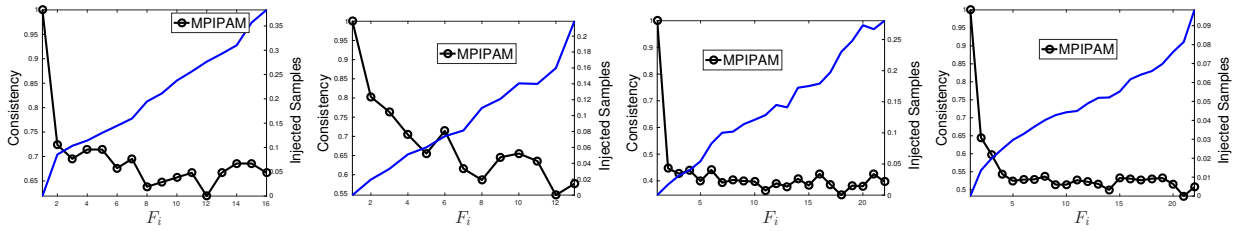
1: **Input**: R.V.'s F, Y of length N; Operation $sig = +1/-1$; Shift δ; Injection limit θ; Rate limit ϵ.
2: **Initialization**: Calculate $\mathbb{P}(f) : \forall f \in \mathcal{F}$; $\mathbb{P}(y) : \forall y \in \mathcal{Y}$; $\mathbb{P}(f,y) : \forall (f,y) \in \Psi$; $baseline = I(F;Y)$; $\rho = 0, \lambda = 1$; Evaluated length $L = N$.
3: $(f^*, y^*) = \arg\min_{(f,y)\in\Psi} sig \cdot \frac{d\Phi}{d\lambda}|_{\lambda=1}$
4: **for** $\rho \le \theta$ or $|I(F;Y) - baseline| < \delta$ **do**
5: **if** $|\frac{d\Phi}{d\lambda}| \ge \epsilon$ **then**
6: Insert (f^*, y^*) to F, Y; $\rho = \rho + 1, \lambda = \frac{L}{L+\rho}$
7: Update $\frac{d\Phi}{d\lambda}$ w.r.t. λ
8: **else if** $|\frac{d\Phi}{d\lambda}| < \epsilon$ **then**
9: Re-calculate $\mathbb{P}(f)$, $\mathbb{P}(y)$, $\mathbb{P}(f,y)$ w.r.t. F, Y
10: $\theta = \theta - \rho$, $L = L + \rho, \lambda = 1, \rho = 0$
11: $(f^*, y^*) = \arg\min_{(f,y)\in\Psi} sig \cdot \frac{d\Phi}{d\lambda}|_{\lambda=1}$
12: **end if**
13: **end for**

2.3. mRMR Poisoning Algorithm

Filter-based FS typically uses a greedy forward search algorithm to avoid the intractable exhaustive search, which has weaknesses that can be exploited by an adversary. For example in mRMR, mistakes at the beginning can lead to learning failures because mRMR evaluates candidates' redundancy w.r.t. the previously selected features. We seek to poison the first selected feature of mRMR in our algorithm: MIP poisoning against mRMR (MIPPAM).

Assume F is an arbitrary feature and F' is the feature having maximal MI with label Y: $F' = \arg\max\{I(F;Y)\}$. If $F \neq F'$, the MIPPAM algorithm's objective is to drive up $I(F;Y)$ and drag down $I(F';Y)$ simultaneously such that $I(F;Y) - I(F';Y) > 0$. After the objective is optimized, F will be the first feature selected that we refer to as "fake best".

Assume f, f' and y are outcomes of F, F' and Y. We solve the above objective by adopting MIP algorithm. Firstly, MIPPAM takes three R.V.'s as inputs, sig is set to $+1$ (step 1). In step 2 we initialize three marginal and two joint distributions. Secondly, recall $\Phi(\lambda, f, y)$ denotes $I(F,Y)$ after ρ ($\rho = \frac{N(1-\lambda)}{\lambda}$) $(f,y)'s$ injected to F and Y. Thus, $\Phi_1(\lambda, f, y) - \Phi_2(\lambda, f', y)$ denotes $I(F;Y) - I(F';Y)$ after ρ $(f, f', y)'s$ injected to F, F' and Y. Thus, we use $\frac{d\Phi_1 - d\Phi_2}{d\lambda}$ and $\frac{d\Phi_1 - d\Phi_2}{d\lambda}|_{\lambda=1}$ to adopt equations (6) and (7) in steps 3, 5, 7, 8, and 11, respectively. Finally, the loop in step 4 repeats when $\rho \le \theta$ or $I(F;Y) > I(F';Y)$. The remaining feature values are determined by choosing the nearest neighbors of



(a) congress (b) heart (c) parkinsons (d) spect

Fig. 1. Consistency of 4 datasets where different "fake best" is used. X-axis are the "fake best" features sorted in descending order of original mutual information with Y (F_i means the feature has i^{th} MI with label), Y-axis are the corresponding Kuncheva consistency.

the above output values of F, F' and Y.

3. EXPERIMENTS

In this section, we present a comprehensive evaluation of the MIPPAM algorithm. In these experiments we run MIPPAM algorithm on ten datasets where malicious samples are generated by MIPPAM then run mRMR on the same datasets without adversarial data (i.e., benign and poisoned, respectively). The performance of the FS algorithm is measured by the stability of mRMR and classification accuracy. Information about the datasets are shown in Table 1 and all experiments are averaged by 30 bootstrap runs. Finally, mRMR selects ten percent of the features for each dataset.

Note that MIPPAM algorithm require a manually selected "fake best" to substitute the authentic best feature, we here experiment all choices of "fake best" (e.g., all candidates except the authentic best feature) and report the malicious selected feature subset's consistency with selection result on benign dataset, for each choice of "fake best". Moreover, we report the required injected malicious sample amount for each choice of "fake best". We here use the Kuncheva index (range in $-1 \sim 1$) to measure the degree of agreement [19] (high value implies high agreement), We report consistency for 4 datasets (due to limited space) in Figure 1. The first observation is, the MIPPAM algorithm can result in a general disagreement on two selection results, where the poisoning extent relies on the choice of "fake best". Moreover, as the "fake best" feature's mutual information decreases, the required malicious sample amount increases.

In many machine learning tasks, FS is not the final objective and classification accuracy is the desired statistic. A KNN and decision tree are trained on features selected on the benign and malicious datasets. Note that we obtain different FS results for various "fake best" features on each dataset; therefore, we choose the "fake best" feature that leads to the maximal decrease in the FS stability. Table 1 shows the testing error for the benign and malicious datasets. (for selection on the malicious dataset, we report the injected sample amount in parenthesis), we mark the failed poisoning cases in boldface. The results show that MIPPAM can lead to a common decrease in the classification accuracy for both classifiers. We also observe that some datasets require a significant amount of data to impact the error; however, the change in error between the benign and poisoned datasets are quite different. For example, splice and krvskp both have 775+ adversarial samples, but the error for splice, krvskp increase 5% and 30%, respectively.

Table 1. The testing error on ten datasets, the failed poisoning scenarios are marked in boldface. For each dataset, we also report the injected sample amount in parenthesis for corresponding "fake best", which achieves the most consistency decrement with FS result on the benign dataset.

Dataset	#features	#samples	k-NN (k = 5)		Decision Tree	
			Benign	Poisoned	Benign	Poisoned
congress	16	327	0.0503	0.1062 (114)	0.0506	0.1123 (114)
heart	13	203	0.2343	0.2532 (38)	0.2139	0.2413 (38)
ionosphere	34	264	0.0969	0.1475 (42)	0.0843	0.1249 (42)
krvskp	36	2397	0.0738	0.3668 (780)	0.0640	0.3122 (780)
parkinsons	22	147	0.1556	0.1597 (37)	0.1368	0.166 (37)
pengcolon	2000	47	0.1711	0.2022 (21)	0.1667	0.1844 (21)
penglung	325	55	0.1722	0.1926 (23)	0.3481	**0.3167 (23)**
semeion	256	1195	0.2750	**0.2663 (276)**	0.2708	0.2839 (276)
spect	22	201	0.2071	0.2404 (17)	0.2035	0.2217 (17)
splice	60	2382	0.0980	0.1534 (775)	0.0606	0.1189 (775)

4. CONCLUSIONS

The vulnerability of machine learning has been investigated for over ten years, while the FS's behavior in a malicious environment is an under-explored topic. In the meanwhile, the ever-increasing scale of machine learning tasks heavily relying on FS to reduce the dimensionality of the data, which becomes risky in a malicious environment. In this contribution, we revisited the information-theoretic FS algorithm mRMR and propose MIP algorithm that allows manipulating the mutual information of two random variables by injecting malicious samples, based on which we introduced the MIPPAM algorithm which led to a decrease in the consistency and as well as classification accuracy. The future work includes devising more general poisoning algorithm for information-theoretic FS and proposing more robust FS objectives.

5. REFERENCES

[1] K. G. Hartmann, R. T. Schirrmeister, and T. Ball, "EEG-GAN: Generative adversarial networks for electroencephalograhic (eeg) brain signals," *arXiv:1806.01875*, 2018.

[2] D. Lowd and C. Meek, "Good word attacks on statistical spam filters," in *Conference on Email Anti-Spam*, 2005.

[3] A. Kolcz and C. H. Teo, "Feature weighting for improved classifier robustness," in *Conference on Email Anti-Spam*, 2009.

[4] J. Gardiner and S. Nagaraja, "On the security of machine learning in malware c&c detection: A survey," *ACM Computing Surveys*, vol. 49, no. 3, 2016.

[5] W. Xu, Y. Qi, and D. Evans, "Automatically evading classifiers: A case study on PDF malware classifiers," in *Network and Distributed System Security Symposium*, 2016.

[6] M. Barreno, B. Nelson, R. Sears, A. Joseph, and J. D. Tygar, "Can machine learning be secure?," in *ACM Symposium on InformAtion, Computer and Communications Security*, 2006.

[7] D. Lowd and C. Meek, "Adversarial maching learning," in *Proceedings of Knowledge and Data Discovery*, pp. 641–647, 2005.

[8] B. Biggio, G. Fumera, and F. Roli, "Security evaluation of pattern classifiers under attack," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 4, pp. 984–996, 2013.

[9] F. Tramer, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," *arXiv:1705.07204*, 2017.

[10] K. K. Budhraja and T. Oates, "Adversarial feature selection," in *IEEE International Conference on Data Mining Workshop*, pp. 288–294, 2015.

[11] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning?," in *International Conference on Machine Learning*, 2015.

[12] F. Zhang, P. P. K. Chan, B. Biggio, D. Yeung, and F. Roli, "Adversarial feature selection against evasion attacks," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 766–777, 2016.

[13] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of Royal Statistics Society*, vol. 58, no. 1, pp. 267–288, 1996.

[14] C. Frederickson, M. Moore, G. Dawson, and R. Polikar, "Attack strength vs. detectability dilemma in adversarial machine learning," in *IEEE/INNS International Joint Conference on Neural Networks*, 2018.

[15] G. Ditzler and A. Prater, "Fine tuning lasso in an adversarial environment against gradient attacks," in *IEEE Symposium on Computational Intelligence and Data Mining*, 2017.

[16] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max–dependency, max–relevance, and min–redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[17] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[18] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley-Interscience, 2006.

[19] L. I. Kuncheva, "A stability index for feature selection," in *International Conference on Artifical Intelligence and Application*, pp. 390–395, 2007.