Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning

Battista Biggio, Fabio Rolia

Pattern Recognition 84(2018)

Youngjoon Kim[1]

[1]Korea University
acorn421@korea.ac.kr

2020.10.10

Table of Contents

Table of Contents

Battista Biggio, Fabio Rolia

Youngjoon Kim

Introduction

Battista Biggio, Fabio Rolia

Youngjoon Kim

Adversarial Machine Learning

- Machine learning have reported impressive performance
- It can be fooled by *adversarial examples*
- Research papers have started proposing countermeasures to mitigate the threat associated to these *wild patterns*

Misconception

- Start date of the field of *adversarial machine learning*
- adversarial examples against linear classifiers(2004) $\rightarrow$ adversarial examples against deep networks(2014)

Goal of Paper

- Provide an overview of *adversarial machine learning*
- Connect between the security of non-deep learning and deep learning
- Highlight common *misconceptions* of security evaluation of learning algorithms

Table of Contents

Battista Biggio,
Fabio Rolia

Youngjoon Kim

Arms race

Security is an amrs race

- Security is an *arms race*
- Security of machine learning is not an exception

Example in spam filtering

- Rule-based filters & text classifiers → Obfuscate the content of spam emails(mispelling bad words, adding good words)
- Embed the spam message within an attached image → Detect spam using signatures of known spam hash & OCR tools → Obfuscate images with random noise
- Learning-based spam detection → Generate adversarial example

Reactive and proactive security

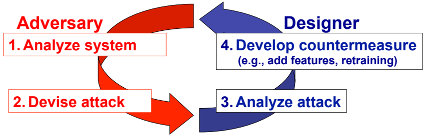


Figure: Reactive security

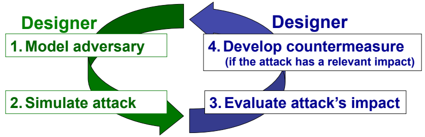


Figure: Proactive security

Security designer should follow proactive approach to prevent never-before-seen attacks

Table of Contents

Know your adversary

Battista Biggio, Fabio Rolia

Youngjoon Kim

Know your adversary

"If you know the enemy and know yourself, you need not fear the result of a hundred battles." (Sun Tzu, The Art of War, 500 BC)

Modeling components

- Attacker's Goal
- Attacker's Knowledge
- Attacker's Capability
- Attack Strategy

Attacker's Goal

Security Violation

- **Integrity** violation : evade detection without compromising normal system operation
- **Availability** violation : compromise the normal system functionalities available to legitimate users
- **Privacy** violation : obtain private information about the system

Attack Specificity

- **Targeted** : attack *specific set of samples*
- **Indiscriminate** : attack *any sample*

Error Specificity

- **Specific** : misclassified as a *specific class*
- **Generic** : misclassified as *any of other classes*

Attacker's Knowledge I

Knowledges of the target systems

- Training data D
- Feature set X
- Learning algorithm f
- Trained parameters/hyper-parameters w.
- Knowledges of systems $\theta = (D, X, f, w)$

Perfect-Knowledge (PK) White-Box Attacks

- X, f, D, w
- $\theta_{\text{PK}} = (D, X, f, w)$

Attacker's Knowledge II

Limited-Knowledge (LK) Gray-Box Attacks

1. LK-SD(Surrogate Data)
   - X, f, D, w
   - a surrogate data set $\hat{D}$, estimated parameters $\hat{w}$
   - $\theta_{\mathrm{LK-SD}} = (\hat{D}, X, f, \hat{w})$
2. LK-SL(Surrogate Learners)
   - X, f, D, w
   - $\theta_{\mathrm{LK-SL}} = (\hat{D}, X, \hat{f}, \hat{w})$.

Attacker's Knowledge III

Battista Biggio, Fabio Rolia

Youngjoon Kim

Zero-Knowledge (ZK) Black-Box Attacks

- X, f, D, w
- Attacker can query the system in a black-box manner and get feedback(labels or confidence scores)
- Purpose of classifier(e.g. object detection), kind of features(e.g. static feature or dynamic feature in malware classification), kind of data
- $\theta_{\text{ZK}} = (\hat{D}, \hat{X}, \hat{f}, \hat{w})$

Attacker's Capability

Battista Biggio, Fabio Rolia

Youngjoon Kim

Attack Influence

- **Poisoning** Attacks : can manipulate both training and test data
- **Evasion** Attacks : can only manipulate test data

Data Manipulation Constraints

- Presence of application specific constraints on data manipulation
- E.g. malicious code has to be modified without compromising its intrusive functionality
- Initial attack samples D_c can only be modified according to a space of possible modifications $\Phi(D_c)$

Attack Strategy

Battista Biggio, Fabio Rolia

Youngjoon Kim

Optimal Attack Strategy

- Given attacker's knowledge $\theta \in \Theta$ attack samples $D'_c \in \Phi(D_c)$
- Attacker's goal can be defined in terms of an objective function $A(D'_c, \theta) \in \mathbb{R}$

$$D^\star_c \in \underset{D'_c \in \Phi(D_c)}{\arg\max}\, A(D'_c, \theta) \tag{1}$$

Security Evaluation Curves

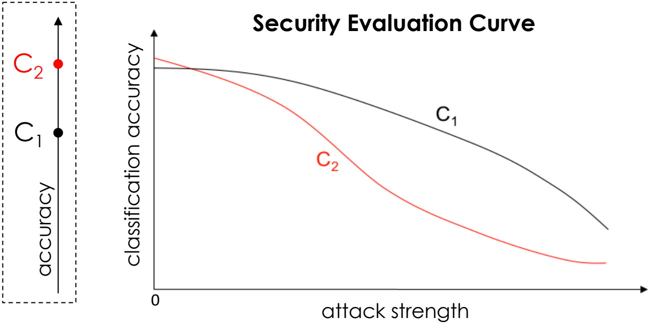


Figure: Security Evaluation Curve; Attack strength can be amount of perturbation or number of poisoning attack points

Summary

Attacker's Goal

	Misclassifications that do not compromise normal system operation	Misclassifications that compromise normal system operation	Querying strategies that reveal confidential information on the learning model or its users
Attacker's Capability	**Integrity**	**Availability**	**Privacy / Confidentiality**
Test data	Evasion (a.k.a. adversarial examples)	-	Model extraction / stealing and model inversion (a.k.a. hill-climbing attacks)
Training data	Poisoning (to allow subsequent intrusions) – e.g., backdoors or neural network trojans	Poisoning (to maximize classification error)	-

Figure: Categorization of attacks. Evasion, Poisoning, Model extraction, Model inversion, Backdoor

Table of Contents

Battista Biggio,
Fabio Rolia

Youngjoon Kim

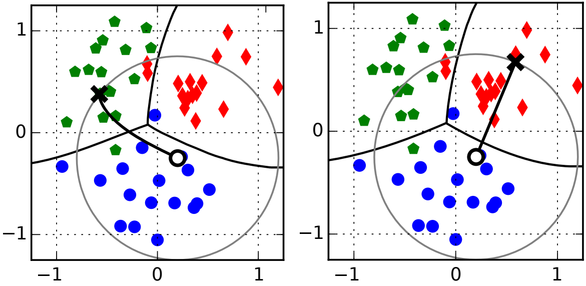
Evasion Attacks I

Battista Biggio, Fabio Rolia

Youngjoon Kim

Evasion Attacks

- **Evasion** attacks consist of manipulating input data to evade a trained classifier at test time
- *Error-generic*, *Error-specific*



Evasion Attacks II

Examples of Evasion Attacks

- Manipulation of malware code to have the corresponding sample misclassified as legitimate
- Manipulation of images to mislead object recognition

Notaion

$f_i(x)$: confidence score of the classifier on the sample x for class i

Error-generic Evasion Attacks

Definition

- Mislead classification to any other class

Problem Formulation

$$\max_{x'} \quad A(x', \theta) = \Omega(x') = \max_{l \neq k} f_l(x) - f_k(x) \,, \tag{2}$$

$$\text{s.t.} \quad d(x, x') \leq d_{\max} \,, \;\; x_{\text{lb}} \preceq x' \preceq x_{\text{ub}} \,, \tag{3}$$

- $f_k(x)$: the discriminant function associated to the true class k of the source sample x
- $\max_{l \neq k} f_l(x)$: the closest competing class
- manipulation constraints $\Phi(D_c)$:
  - a distance constraint $d(x, x') \leq d_{\max}$, which sets a bound on the maximum input perturbation between x
  - a box constraint $x_{\text{lb}} \preceq x' \preceq x_{\text{ub}}$, which bounds the values of the attack sample x'

Error-specific Evasion Attacks

Definition

- Mislead classification to specific class

Problem Formulation

$$\max_{x'} \quad A(x', \theta) = -\Omega(x') = f_k(x) - \max_{l \neq k} f_l(x) , \tag{4}$$

$$\text{s.t.} \quad d(x, x') \leq d_{\max} , \;\; x_{\text{lb}} \preceq x' \preceq x_{\text{ub}} , \tag{5}$$

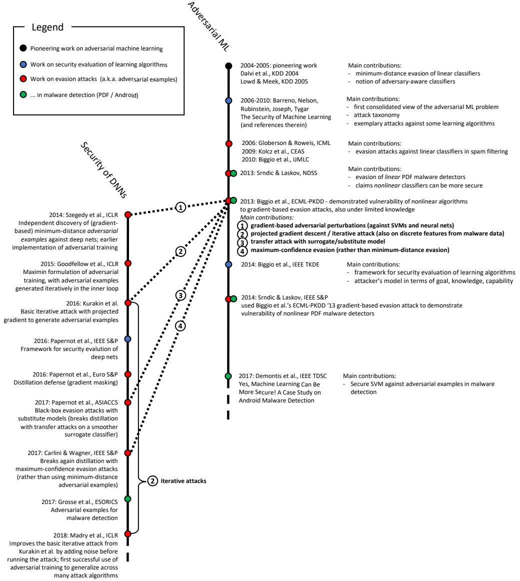
- $f_k(x)$: the discriminant function associated to the targeted class k
- $\max_{l \neq k} f_l(x)$: the closest competing class
- manipulation constraints $\Phi(D_c)$:
  - a distance constraint $d(x, x') \leq d_{\max}$, which sets a bound on the maximum input perturbation between x
  - a box constraint $x_{\text{lb}} \preceq x' \preceq x_{\text{ub}}$, which bounds the values of the attack sample x'

Attack Algorithm

Algorithm

- Differentiable learning algorithm : gradient-based attack
- Non-differentiable learning algorithm : more complex strategies[Kantchelian et al] or using same algorithm against a differentiable surrogate learner

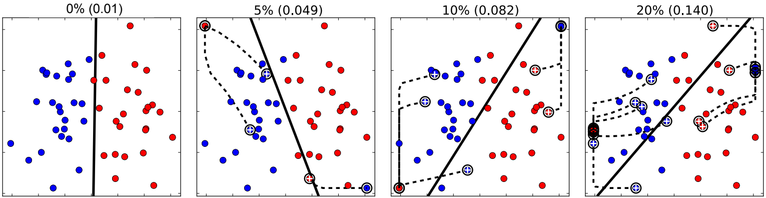
Timeline of Evasion Attacks



Poisoning Attacks

Poisoning Attacks

- **Poisoning** attacks aim to increase the number of misclassified samples at test time by injecting a small fraction of poisoning samples into the training data
- *Error-generic*, *Error-specific* in PK white-box setting



Error-generic Poisoning Attacks

Definition

- Aims to cause a *denial of service*, by inducing as many misclassifications as possible, regardless of the classes

Problem Formulation

$$D_c^\star \in \underset{D_c' \in \Phi(D_c)}{\arg\max} \quad A(D_c', \theta) = L(D_{\mathrm{val}}, w^\star), \tag{6}$$

$$\text{s.t.} \quad w^\star \in \underset{w' \in W}{\arg\min}\, L(D_{\mathrm{tr}} \cup D_c', w'), \tag{7}$$

- D_{tr} and D_{val} : two data sets available to the attacker
-

Error-specific Poisoning Attacks

Definition

- Aims to cause specific misclassifications.

Problem Formulation

$$D_c^\star \in \arg\max_{D_c' \in \Phi(D_c)} \quad A(D_c', \theta) = -L(D'_{\mathrm{val}}, w^\star)\,, \tag{8}$$

$$\text{s.t.} \quad w^\star \in \arg\min_{w' \in W} L(D_{\mathrm{tr}} \cup D_c', w')\,, \tag{9}$$

- D'_{val} contains the same samples as D_{val}, but their labels are chosen by the attacker according to the desired misclassifications.

Attack Algorithm

Algorithm

- Replace the inner optimization by its equilibrium conditions
- Deep Networks : *back-gradient poisoning*

Table of Contents

Reactive Defenses

Reactive Defenses

Aims to counter *past* attacks

- **Timely detection** of novel attacks
- Frequent classifier **retraining**
- **Verification** of consistency of classifier decisions against training data and ground-truth labels

Proactive Defenses

Battista Biggio, Fabio Rolia

Youngjoon Kim

Proactive Defenses

Aims to prevent *future* attacks

- Security by Design
- Security by Obscurity

Security-by-Design Defenses against White-box Attacks I

Countering Evasion Attacks

- **Iteratively retraining** the classifier which is similar with adversarial training
- Approaches based on **game theory**
- **Robust optimization**; formulates adversarial learning as a minimax problem
- **Detecting and rejecting** samples which are sufficiently far from the training data
- **Classifier ensembles**

Security-by-Design Defenses against White-box Attacks II

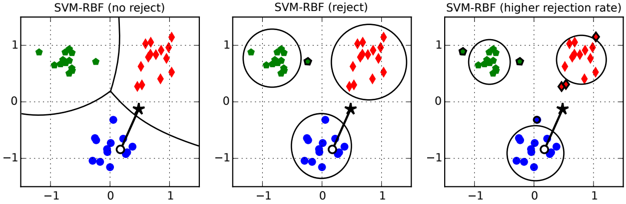


Figure: Effect of *class-enclosing* defenses against blind-spot adversarial examples on multiclass SVMs with RBF kernels

Effect on Decision Boundaries

- retraining and rejection can make decision functions may ten to *enclose* training classes more tightly

Security-by-Design Defenses against White-box Attacks III

Countering Poisoning Attacks

- Attack has to be exhibit different characteristics from the original training data
- **Data sanitization**; attack detection and removal
- **Robust learning**; learning algorithm based on robust statistics

Security-by-Obscurity Defenses against Black-box Attacks

Battista Biggio, Fabio Rolia

Youngjoon Kim

Security-by-Obscurity

- Disinformation technique; hide information to improve security
- Aim to counter gray-box and black-box attacks
- **Randomizing training data**
- Using **difficult to reverse-engineer classifiers**
- **Denying access** to the actual classifier or training data
- **Randomizing the classifier's output**
- **Gradient masking** has been proposed to hide the gradient direction, but it has been shown that it can be easily circumvented with surrogate learners

Table of Contents

Battista Biggio, Fabio Rolia

Youngjoon Kim

Conclusion

Discussion

- Machine learning can deal with *known unknowns*
- Adversarial machine learning often deals with *unknown unknowns*
- *Unknown unknowns* are the real threat in many security problems (e.g., zero-day attacks in computer security)
- Machine learning algorithms should be able to detect *unknown unknowns*

Future works

Future works

- Formal verification and certified defenses
- Robust artificial intelligence
- Interpretability of machine learning

Table of Contents

Battista Biggio,
Fabio Rolia

Youngjoon Kim

My Opinions and Questions

Attack Strength

Is it meaningful to an adversarial example that even people recognize as different classes?

Proactive Defense

Is perfect proactive defense possible in theoretically?

Trade-off

What is the trade-off between the model's performance and security?

Battista Biggio, Fabio Rolia

Youngjoon Kim

Thank you!