# Adversarial ML: A Game Theoretic Survey

Murat Kantarcioglu

muratk@utdallas.edu

www.utdallas.edu/~muratk

# Adversarial ML: Motivating Examples

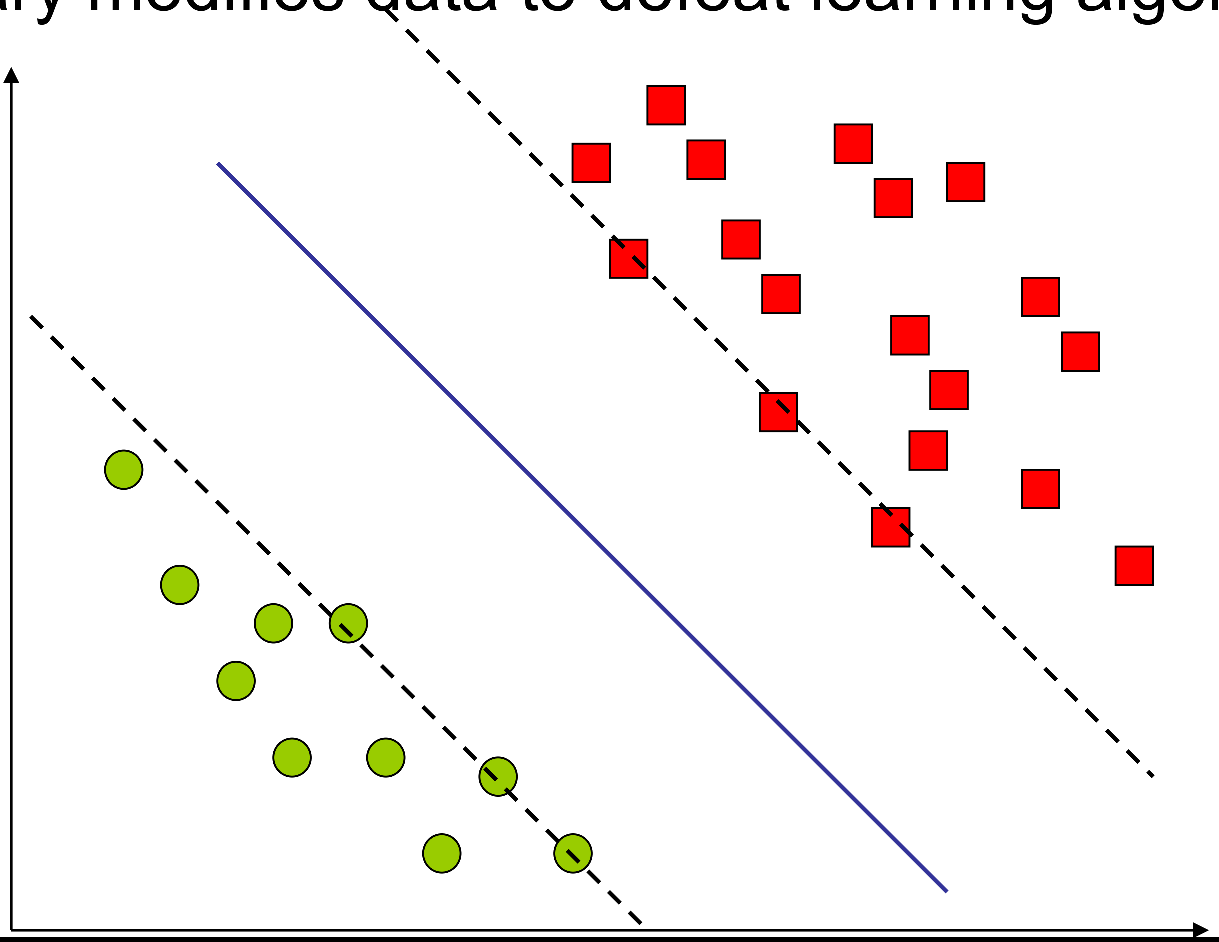
- Many adversarial learning problems in practice
  - Intrusion Detection
  - Fraud Detection
  - Spam Detection
  - Malware Detection

- Adversary adapts to avoid being detected.

- New solutions are explored to address this problem

# The Problem

- Violation of standard *i.i.d.* assumption
- Adversary modifies data to defeat learning algorithms



# Example: Spam Filtering

- Millions way to write Viagra

```
From: "Ezra Martens" <ezrabngktbbem...
To: "Eleftheria Marconi" <clifton@pu...
Subject: shunless Phaxrrmaceutical
Date: Fri, 30 Sep 2005 04:49:10 -0500

Hello,
Easy Fast =
Best Home Total
OrdeShipPrricDelivConf
ringpingeseryidentiality
VIAAmbCIALevVALXan
GRAienLISitraIUMax
$  $ $$
3.33 1.21 3.75
Get =additional informmation attempted to
```

# Understanding Adversarial Learning

- It is not concept drift
- It is not online learning
- Adversary adapts to avoid being detected
  - During training time (i.e., data poisoning)
  - During test time (i.e., modifying features when data mining is deployed)
- There is game between the ML model builder and the adversary

# Solution Ideas

- Constantly adapt your classifier to changing adversary behavior.

- Questions??
  - How to model this game?
  - Does this game ever end?
  - Is there an equilibrium point in the game?

# Agenda

- Summary of foundational results/models to reason about learning in the presence of an active adversary
  - No proofs/ Summary of the models
  - Not all the good work could be covered ☹
- Modified techniques resistant to adversarial behavior with some game theory inspiration
- Deep Learning

# Foundations

# Machine Learning Problems

| | Supervised Learning | Unsupervised Learning |
|---|---|---|
| Discrete | classification or categorization | clustering |
| Continuous | regression | dimensionality reduction |

# The machine learning framework

$$y = f(\mathbf{x})$$

Output:
Attack
No-Attack

prediction
function

Samples:
Malware
Network Packets

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function $f$ by minimizing some criteria on the training set
- **Testing:** apply $f$ to a *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

# Threat Models

- Training Time Attacks:
  - Poison/ modify the training data
  - Some attacks are tailored for specific f()
- Test time/ Deployment Time Attacks
  - Attacker modifies x to x'
    - E.g., modify packet length by adding dummy bytes
    - Add good word to spam e-mail
    - Add noise to an image
  - Could be specific to f()
  - Focus in this tutorial

# Adversarial Classification [1]

- Data is manipulated by an adversary to increase false negatives.
  - Spam detection
  - Intrusion detection
  - Fraud detection
- Classification is considered as a game between the classifier and the adversary.
  - Both are cost-sensitive

# Cost and Utility for Classifier & Adversary

- Given a training set $S$ and a test set $T$,
  - CLASSIFIER
    - learn from $S$ a classification function $y_C = C(x)$
    - $V_i$: cost of measuring the $i^{th}$ feature $X_i$
    - $U_C(y_C, y)$: utility of classifying an instance as $y_C$ with true class $y$
      - $U_C(+, -) < 0,\ U_C(-, +) < 0,\ U_C(+,+) > 0,\ U_C(-,-) > 0$
  - ADVERSARY
    - modify a *positive* instance in $T$ from $x$ to $x' = A(x)$
    - $W_i(x_i, x'_i)$: cost of changing the $i^{th}$ feature from $x_i$ to $x'_i$
    - $U_A(y_C, y)$: ADVERSARY's utility when the classifier classifies an instance as $y_C$ with true class $y$
    - $U_A(-,+) > 0,\ U_A(+,+) < 0$ and $U_A(-,-) = U_A(+,-) = 0$

# A Single-step Two Players' Game

- For computational tractability, the adversarial classification game only considers one move by each of the players.
- It also assumes that all parameters of both players are known to each other.
- Classifier is Naïve Bayes:
  - an instance $x$ is classified positive if the expected utility of doing so exceeds that of classifying it as negative

$$\frac{P(+|x)}{P(-|x)} > \frac{U_C(-,-) - U_C(+,-)}{U_C(+,+) - U_C(-,+)}$$

# Adversary's Strategy

- Adversary's optimal strategy:
  - Two assumptions:
    - complete Information
    - CLASSIFIER is unaware of its presence.
  - Modify features such that
    - The transformation cost is less than the expected utility.
    - The new instances is classified as negative.
  - Solve an integer LP

# Classifier’s Strategy

- Classifier’s optimal strategy:
  - Three assumptions:
    - Adversary uses optimal strategy.
    - Training set is not tampered by Adversary.
    - The transformation cost $W_i(x_i, x'_i)$ is a semi-metric.
  - Make prediction $y_C$ that Maximizes conditional utility:

$$U(y_C|x) = \sum_{y \in \mathcal{Y}} P(y|x) U_C(y_C, y)$$

with a post-adversary conditional probability

$$P_A(x'|+) = \sum_{x \in \mathcal{X}} P(x|+) P_A(x'|x, +)$$

# Classifier Evaluation and Attribute Selection against Active Adversaries [2]

- Consider cases where the classifier is modified after observing adversaries action.
  - Spam filter rules.

- Stackelberg Games
  - Adversary chooses an action $a_1$
  - After observing $a_1$ , data miner chooses action $a_2$
  - Game ends with payoffs to each player

$$u_1(a_1, a_2), u_2(a_1, a_2)$$

# Adversarial Stackelberg Game Formulation

- Two class problem
  - Good class, Bad class
- Mixture model

$$x = (x_1, x_2, x_3, \ldots, x_n)$$

$$p_1 + p_2 = 1$$

$$f(x) = p_1 f_1(x) + p_2 f_2(x)$$

- Adversary applies a transformation T to modify bad class (i.e $f_2(x) \xrightarrow{T} f_2^T(x)$ )

# Adversarial Stackelberg Game Formulation Cont.

- After observing transformation, data miner chooses an updated classifier h
- We define the payoff function for the data miner

$$f(x) = p_1 f_1(x) + p_2 f_2^T(x)$$

$$c(T,h) = \int_{L_1^h} c_{11} p_1 f_1(x) + c_{12} p_2 f_2^T(x) dx + \int_{L_2^h} c_{21} p_1 f_1(x) + c_{22} p_2 f_2^T(x) dx$$

$$u_2(T,h) = -c(T,h)$$

- $C_{ij}$ is the cost for classifying x to class i to given that it is in class j
- Data miner tries to minimize c(T,h)

# Adversarial Stackelberg Game Formulation Cont.

- Transformation has a cost for the adversary
  - Reduced effectiveness for spam e-mails

- Let $g^T(x)$ be the gain of an element after transformation
- Adversary gains for the "bad" instances that are classified as "good"

$$u_1(T, h) = \int_{L_1^h} g^T(x) f_2^T(x) dx$$

# Adversarial Stackelberg Game Formulation Cont.

- Given the transformation T, we can find the best response classifier( R(T)) h that minimizes the c(T,h)

$$h_T(x) = \begin{cases} \pi_1, (c_{12} - c_{22}) p_2 f_2^T(x) \le (c_{21} - c_{11}) p_1 f_1(x) \\ \pi_2, \text{otherwise} \end{cases}$$

- For Adversarial Stackelberg game, subgame perfect equilibrium is:

$$T^* = \arg\max_{T \in S} \left( u_1 \left( T, R(T) \right) \right)$$

$$(T^*, R(T^*))$$

# Adversarial Stackelberg Game Formulation Cont.

$$\begin{aligned} g_e(T) &= u_1(T, R_2(T)) \\ &= \int_{L_1^{h_T}} (g^T(x) f_2^T(x)) dx \\ &= E_{f_2^T}(I_{\{L_1^{h_T}\}}(x) \times g^T(x)) \end{aligned}$$

$$T^* = \arg\max_{T \in S}(g_e(T))$$

- If the game is repeated finitely many times, after an equilibrium is reached, each party does not have incentive change their actions.

# Summary [2]: Attribute Selection for Adversarial Learning

- How to choose attributes for Adversarial Learning?
  - Choose the most predictive attribute
  - Choose the attribute that is hardest to change
- Example:

| Attribute | $\pi_1$ | $\pi_2$ | Penalty | Equilibrium Bayes Error |
|---|---|---|---|---|
| $X_1$ | N(1,1) | N(3,1) | a = 1 | 0.16 |
| $X_2$ | N(1,1) | N(3.5,1) | a = 0.45 | 0.13 |
| $X_3$ | N(1,1) | N(4,1) | a = 0 | 0.23 |

- Not so good ideas!!

# Stackelberg Games for Adversarial Prediction Problems [3]

- Unlike the previous research, Bruckner & Scheffer consider Stackelberg games where the *classifier* is the leader and the *adversary* is the follower.
  - Data miner chooses an action $a_1$
  - After observing $a_1$ , the adversary chooses action $a_2$
  - Game ends with payoffs to each player

$$u_1(a_1, a_2), u_2(a_1, a_2)$$

# Cost Definition

- Two-players game between learner (-1) and adversary (+1).
- The costs of the two players are defined as follows:

$$\hat{\theta}_{-1}(\mathbf{w}, \dot{D}) = \sum_{i=1}^{n} c_{-1,i} \ell_{-1}(f_{\mathbf{w}}(\dot{x}_i), y_i) + \rho_{-1} \hat{\Omega}_{-1}(\mathbf{w}),$$

$$\hat{\theta}_{+1}(\mathbf{w}, \dot{D}) = \sum_{i=1}^{n} c_{+1,i} \ell_{+1}(f_{\mathbf{w}}(\dot{x}_i), y_i) + \rho_{+1} \hat{\Omega}_{+1}(D, \dot{D})$$

# Stackelberg Games

1. Learner decides on $w$.
2. Adversary observes $w$ and changes the data distribution.
3. Adversary minimizes its loss given $w$ by searching for a sample $D_w$ that leads to the global minimum of the loss

$$\mathcal{D}_{\mathbf{w}} = \left\{ \{(\dot{x}_i, y_i)\}_{i=1}^{n} : \{\dot{x}_i\}_{i=1}^{n} \in \underset{\dot{x}'_1,\ldots,\dot{x}'_n \in \mathcal{X}}{\operatorname{argmin}} \ \hat{\theta}_{+1}\left(\mathbf{w}, \{(\dot{x}'_i, y_i)\}_{i=1}^{n}\right) \right\}$$

# Stackelberg Equilibrium

- Assuming that the adversary will decide for any $D \in D_w$, the learner has to choose model parameters $w^*$ that minimize the learner's cost function $\theta_{-1}$ for any of the possible reactions $D \in D_w$ that are optimal for the adversary:

$$\mathbf{w}^* \in \underset{\mathbf{w}\in\mathbb{R}^m}{\operatorname{argmin}} \max_{\dot{D}\in\dot{\mathcal{D}}_{\mathbf{w}}} \hat{\theta}_{-1}(\mathbf{w}, \dot{D})$$

- An action $w^*$ that minimizes the learner's costs and a corresponding optimal action $D \in D_{w^*}$ of the adversary are called a Stackelberg equilibrium.

# Find Stackelberg Equilibrium [3]

## Finding Stackelberg Equilibrium

$$\min_{\mathbf{w}\in\mathbb{R}^m} \max_{\forall i:\, \dot{x}_i \in \mathcal{X}} \hat{\theta}_{-1}(\mathbf{w}, \{(\dot{x}_i, y_i)\}_{i=1}^n)$$

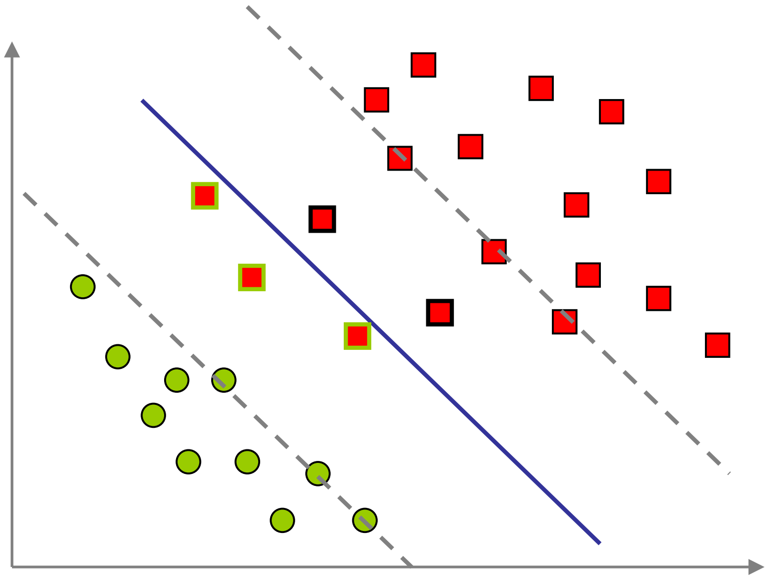
$$\text{s.t.} \qquad \{\dot{x}_i\}_{i=1}^n \in \operatorname*{argmin}_{\dot{x}'_1,\ldots,\dot{x}'_n \in \mathcal{X}} \hat{\theta}_{+1}(\mathbf{w}, \{(\dot{x}'_i, y_i)\}_{i=1}^n)$$

Stackelberg equilibrium is applicable when
(1.) the adversary is rational;
(2.) the predictive model is known to the adversary.

# TECHNIQUES

# Adversarial support vector machine learning [4]

- Support Vector machines try to find the hyperplane that has the highest possible separation margin.



# Adversarial Attack Model Example

- Free-range attack
  - Adversary can move malicious data anywhere in the domain

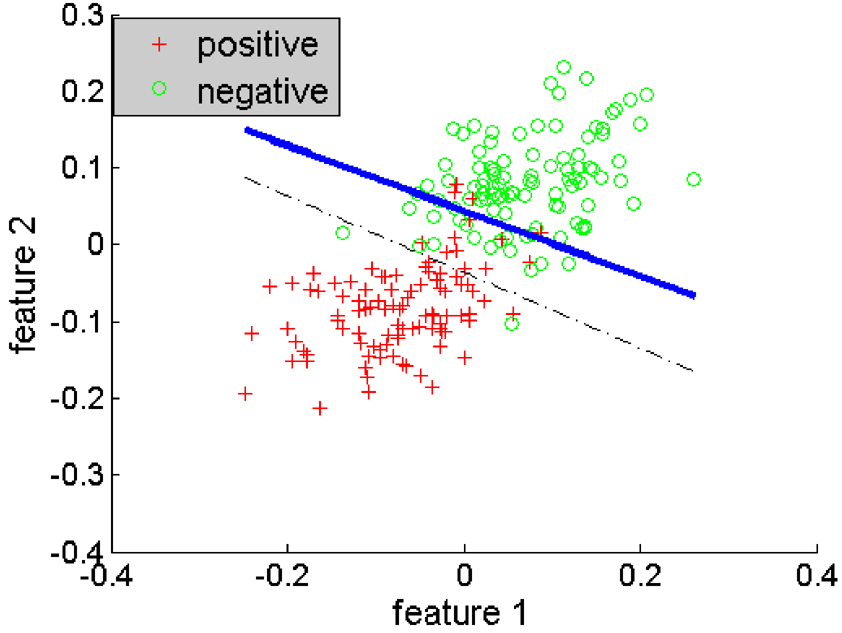
$$c_f(x_{.j}^{\min} - x_{ij}) \leq \delta_{ij} \leq C_f(x_{.j}^{\max} - x_{ij})$$

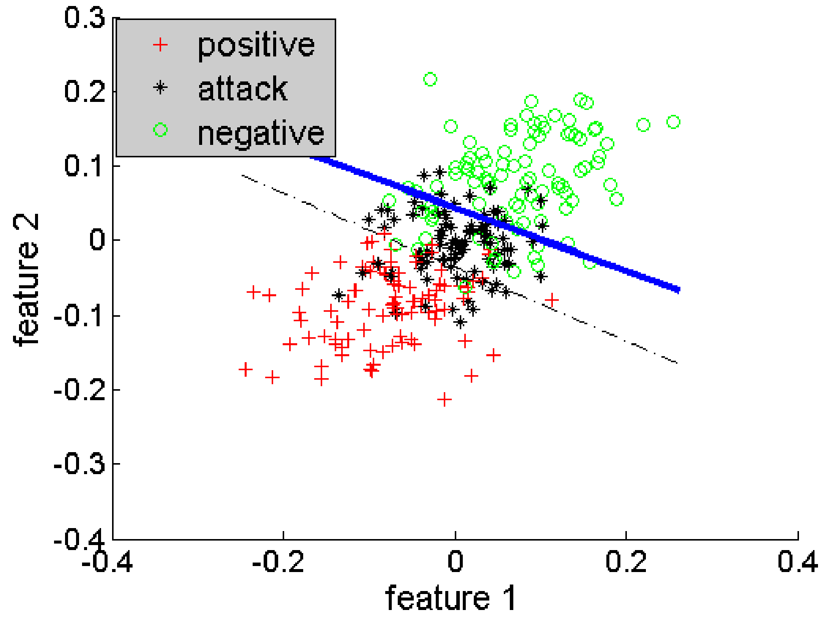
# Adversarial SVM Risk Minimization Model

## SVM risk minimization model: free-range attack

$$
\begin{aligned}
\underset{w,b,\xi_i,t_i,u_i,v_i}{\operatorname{argmin}} \quad & \tfrac{1}{2}||w||^2 + C\textstyle\sum_i \xi_i \\
s.t. \quad & \xi_i \geq 0 \\
& \xi_i \geq 1 - y_i \cdot (w \cdot x_i + b) + t_i \\
& t_i \geq \textstyle\sum_j C_f \left( v_{ij}(x_j^{max} - x_{ij}) - u_{ij}(x_j^{min} - x_{ij}) \right) \\
& u_i - v_i = \tfrac{1}{2}(1 + y_i)w \\
& u_i \succeq 0 \\
& v_i \succeq 0
\end{aligned}
$$

# AD-SVM Example:



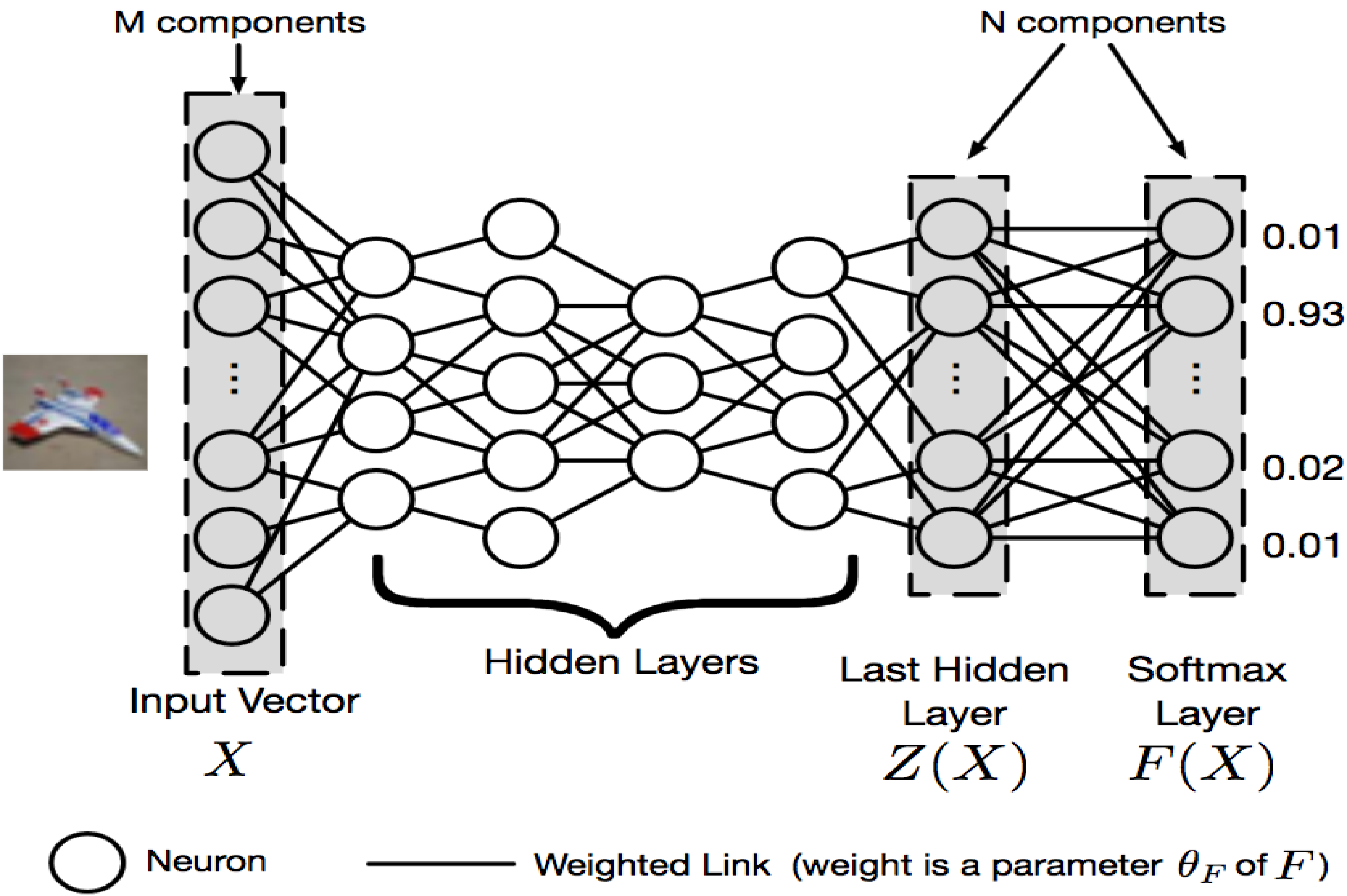

**black dashed** line is the standard SVM classification boundary, and the blue line is the Adversarial SVM (ADV-SVM) classification boundary

# Summary of [4]

- AD-SVM solves a convex optimization problem where the constraints are tied to adversarial attack models
- AD-SVM is more resilient to modest attacks than other SVM learning algorithms

# DEEP LEARNING

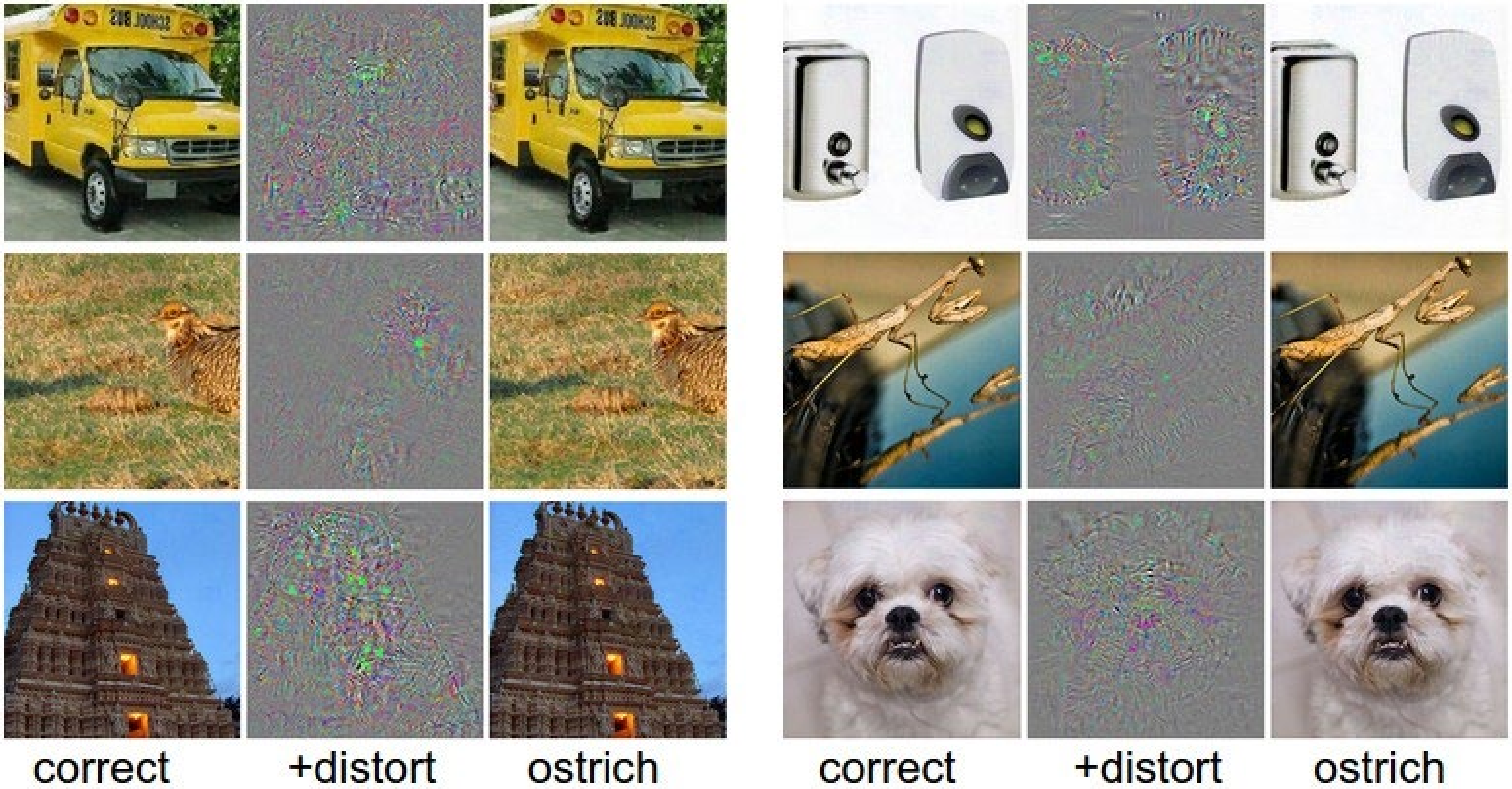
# DNN: Attacks and Defenses [5]



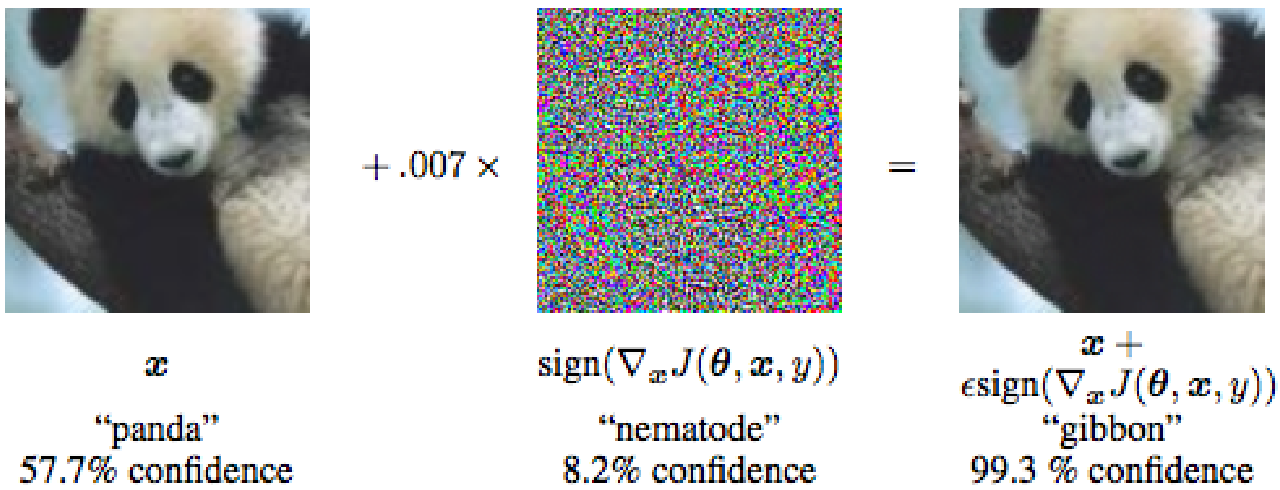
# Attacks against Deep Neural Networks

- Recent work in the machine learning and security communities have shown that adversaries can force DNNs to produce adversary-selected outputs using carefully crafted input.
- $For\ given\ x, find\ min\|\delta\|$ s.t. $f(x+\delta) \neq y$
- Targeted attack find $min\|\delta\|$ s.t. $f(x+\delta) = y_t$

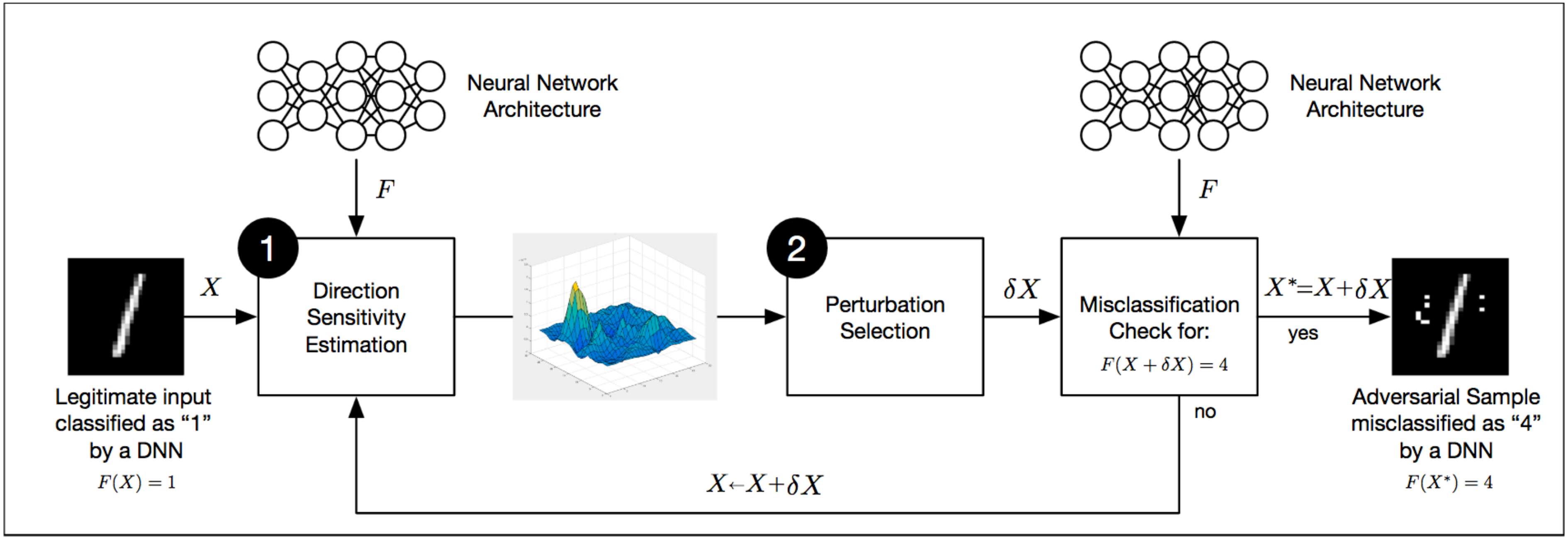
# Szagedy et al. [6]



# Goodfellow et al. Example [7]



# Adversarial Sample Crafting



# Adversarial Crafting

- Crafting consists of two steps: **direction sensitivity estimation** and **perturbation selection**.
- Step 1 evaluates the sensitivity of model *F* at the input point corresponding to sample *X*.
- Step 2 uses this knowledge to select a perturbation affecting sample X's classification.
  - If the resulting sample $X + \delta X$ is misclassified by model *F* in the adversarial target class, an adversarial sample $X^*$ has been found.
  - If not, the steps can be repeated on updated input $X + \delta X$ .

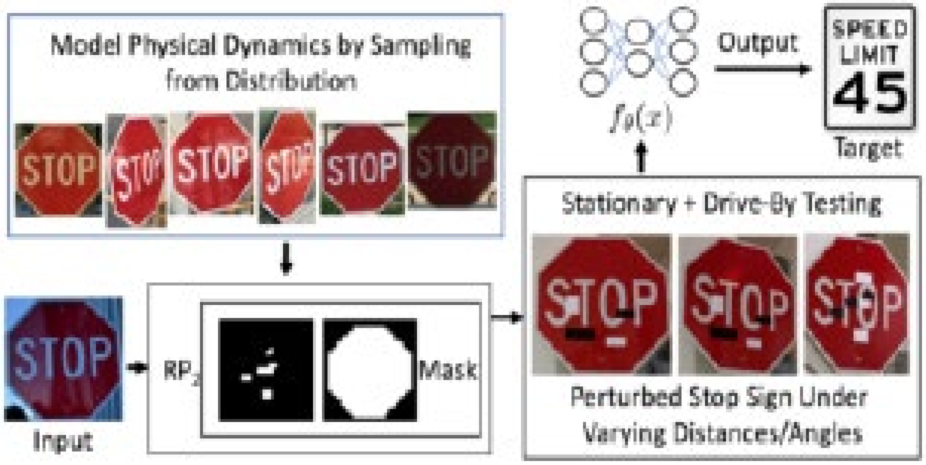
# Direction Sensitivity Estimation

- The goal is to find the dimensions of *X* that will produce the expected adversarial behavior with the smallest perturbation.
- Goodfellow et al. propose the fast sign gradient method
  - Computes the gradient of the cost function with respect to the input of the neural network
  - compute $\mathrm{x}' = \mathrm{x} + \epsilon . sgn(\nabla_x L(F(x), y)$
- Multi-step attack version
  - Compute $x_t = Proj_\epsilon[x_{t-1} + \epsilon_{t-1} . sgn(\nabla_x L(F(x_{t-1}), y) ]$
  - $Proj_\epsilon$ guarantees that $\|x - x_t\|_\infty < \epsilon$

# $l_0$ Attacks

- Papernot et al. [5] propose the forward derivative, which is the Jacobian of *F*
  - Directly compute the gradients of the output components with respect to each input component.
- Papernot et al. follow a more complex process involving saliency maps to only select a limited number of input dimensions to perturb.
  - Saliency maps assign values to combinations of input dimensions indicating whether they will contribute to the adversarial goal or not if perturbed.

# Robust Physical World Attacks [9]


- Attacks against DNNs trained for
  - Text classification
  - Speech Classification
  - Lidar based Object Recognition
- Many more attacks
  - See Carlini-Wagner Attacks [8]

# Defense Requirements

- Low impact on the architecture
- Maintain accuracy
- Maintain speed of network
- Defenses should work for adversarial samples relatively close to points in the training dataset

# Adversarial Training [10]

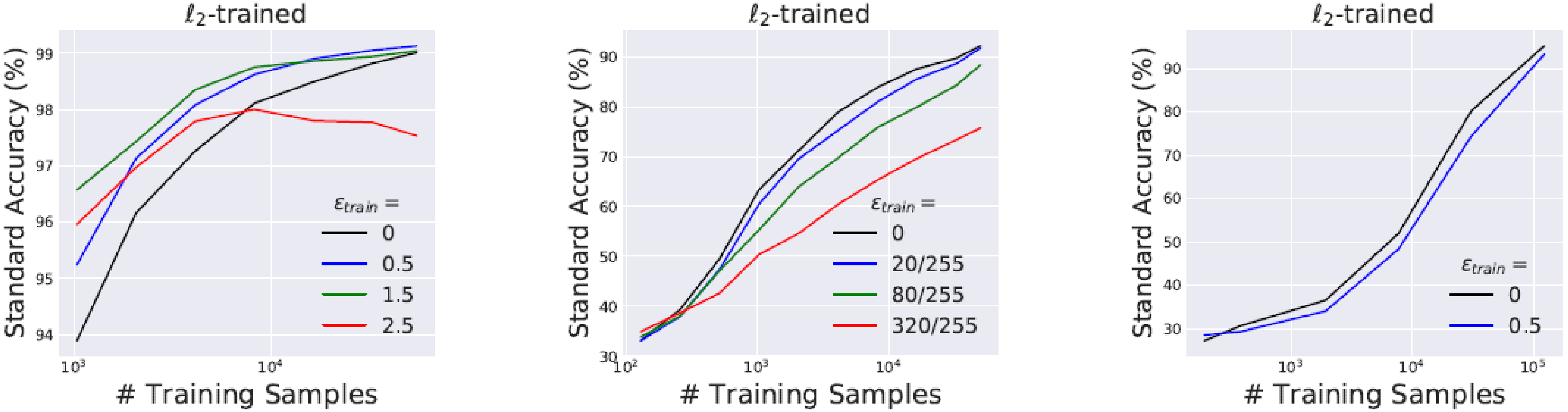
$$\theta^* = \arg\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\in\mathcal{X}} \left[ \max_{\delta\in[-\epsilon,\epsilon]^N} \ell(x+\delta; y; F_\theta) \right].$$

- Solve above optimization by running PGSM to find an attack and do SGD using $x + \delta$
  - Robust optimization based learning takes more time
- Given more data and more time, can we always learn a good model with high robustness ??
- Hard to train for Imagenet
  - Slower
- Robust for the norm it is trained
- Robustness to adversarial noise could be seen as an invariance property.

# Adversarial Training as Data Augmentation ?

- Idea: Adding adversarially modified samples is a form of data augmentation
  - Data Augmentation (e.g., adding rotated, cropped images ) usually helps in practice
  - Exps. Show that this may be the case when we do not have enough data.
  - The results does not seem to change if you use natural data combined with adversarially modified data.

# Adv ML models may not perform well when there is no attack. [11]

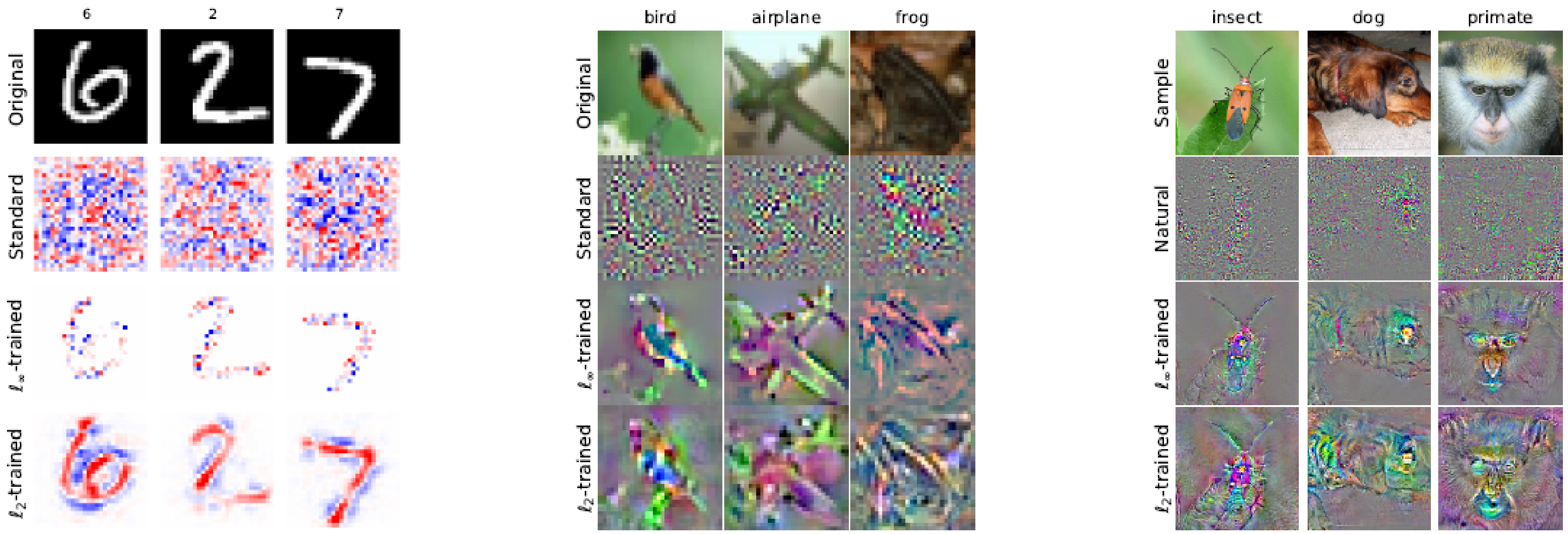



(a) MNIST (b) CIFAR-10 (c) Restricted ImageNet

# Empirical Observations on MNIST

- "it is possible to obtain a robust classifier by directly training a standard model using only features that are relatively well-correlated with the label (without adversarial training)".
- "As expected, as more features are incorporated into the training, the standard accuracy is improved at the cost of robustness"

# Robust Training Learning Better Features??




(a) MNIST (b) CIFAR-10 (c) Restricted ImageNet

Figure 2: Visualization of the loss gradient with respect to input pixels. Recall that these gradients highlight the input features which affect the loss most strongly, and thus the classifier's prediction. We observe that the gradients are significantly more human-aligned for adversarially trained networks – they align well with perceptually relevant features. In contrast, for standard networks they appear very noisy. (For MNIST, blue and red pixels denote positive and negative gradient regions respectively. For CIFAR-10 and ImageNet, we clip gradients to within $\pm 3$ standard deviations of their mean and rescale them to lie in the $[0,1]$ range.)

# Robust Trained

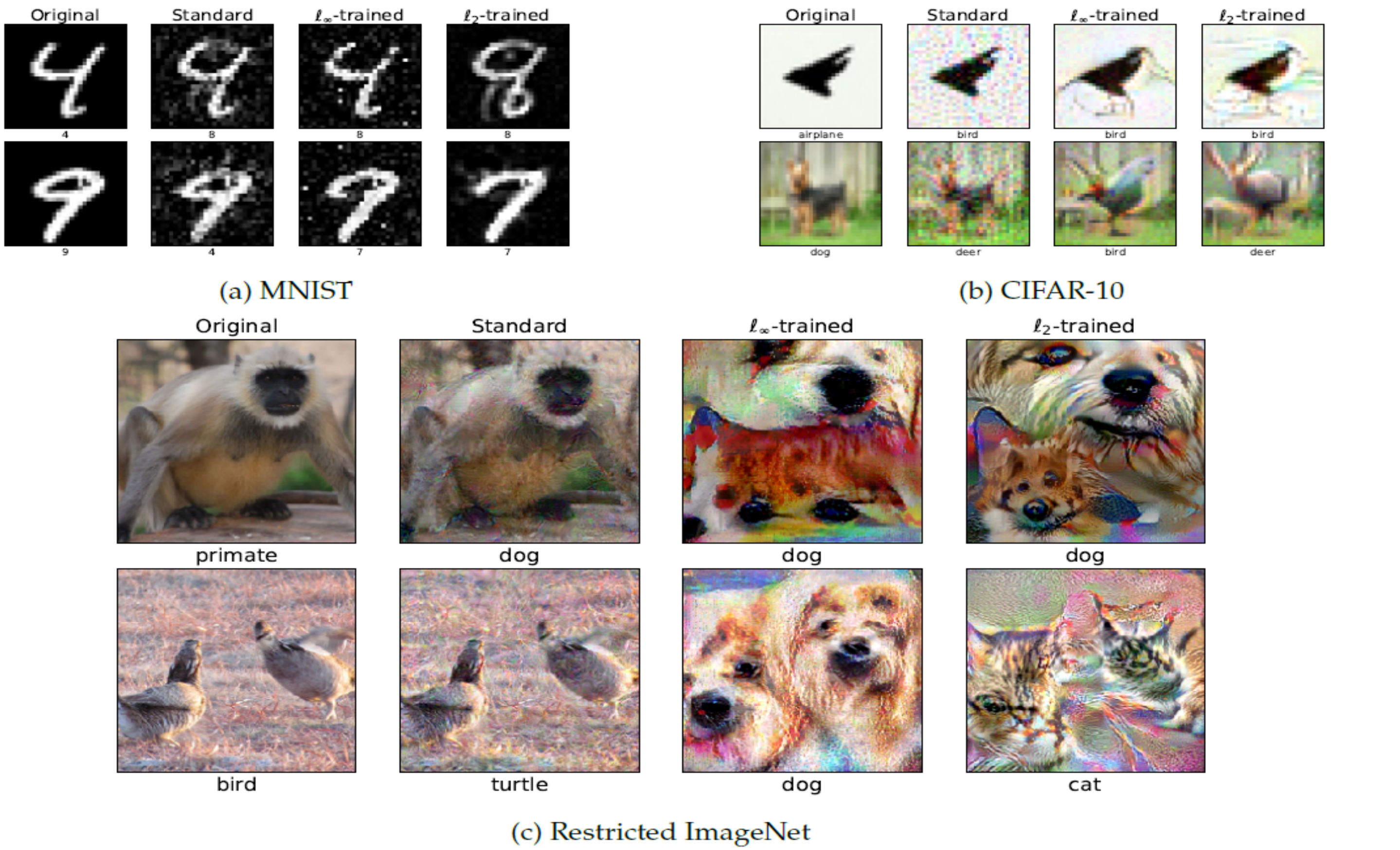



Figure 3: Visualizing large-$\varepsilon$ adversarial examples for standard and robust ($\ell_2/\ell_\infty$-adversarial training) models. We construct these examples by iteratively following the (negative) loss gradient while staying with $\ell_2$-distance of $\varepsilon$ from the original image. We observe that the images produced for robust models effectively capture salient data characteristics and appear similar to examples of a different class. (The value of $\varepsilon$ is

# Conclusions

- Adv ML especially Adv. DNNs became an important research direction
- New game theoretic ideas may help.
- May need to get ready for building systems based on unreliable ML components.
- **Please see our survey**:
  - Yan Zhou, Murat Kantarcioglu, Bowei Xi: A survey of game theoretic approach for adversarial machine learning. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 9(3) (2019)
- **Please see our book**:
  - Yevgeniy Vorobeychik, Murat Kantarcioglu: Adversarial Machine Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers 2018

# References


1. N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma, Adversarial classification, KDD '04.
2. M. Kantarcioglu, B. Xi, and C. Clifton, Classifier evaluation and attribute selection against active adversaries, Data Min. Knowl. Discov., vol. 22, pp. 291-335, January 2011.
3. M. Bruckner and T. Scheffer. Stackelberg games for adversarial prediction problems, SIGKDD, 2011.
4. Y. Zhou, M. Kantarcioglu, B. Thuraisingham, and B. Xi, Adversarial support vector machine learning, SIGKDD '12.
5. Distillation as a defense to adversarial perturbations against deep neural networks, Papernot et al., 2016
6. Szegedy et al. “Intriguing properties of neural networks”, ICLR 2013
7. Goodfellow et al. “Explaining and harnessing adversarial examples”, ICLR 2015
8. Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In ICML 2018
9. Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. Robust physical-world attacks on deep learning visual classification. In IEEE CVPR June 2018
10. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In ICLR 2018, Conference Track Proceedings
11. Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In ICLR 2019.-647