# DS 4400

## Machine Learning and Data Mining I

Alina Oprea
Associate Professor, CCIS
Northeastern University

April 9 2019

# Logistics

- Final exams have been graded!
- Final project presentations
  - Thursday, April 11
  - Tuesday, April 16 in ISEC 655
  - 8 minute slot – 5 min presentation and 3 min questions
- Final report due on Tuesday, April 23
  - Template in Piazza
  - Schedule on Piazza

# What we covered

Adversarial ML

Ensembles
- Bagging
- Random forests
- Boosting
- AdaBoost

Deep learning
- Feed-forward Neural Nets
- Convolutional Neural Nets
- Recurrent Neural Nets
- Back-propagation

Unsupervised
- PCA
- Clustering

Linear classification
- Perceptron
- Logistic regression
- LDA
- Linear SVM

Non-linear classification
- kNN
- Decision trees
- Kernel SVM
- Naïve Bayes

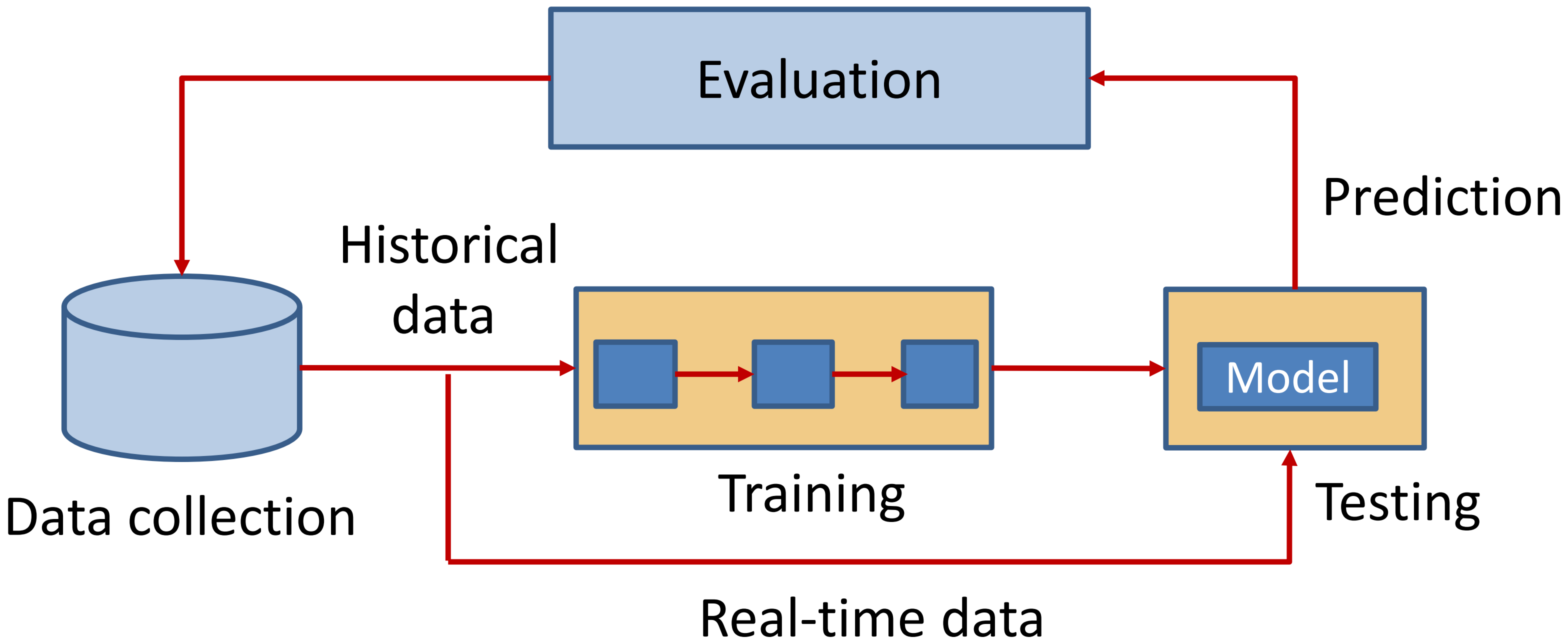
- Metrics
- Cross-validation
- Regularization
- Feature selection
- Gradient Descent
- Density Estimation

Linear Regression

Linear algebra

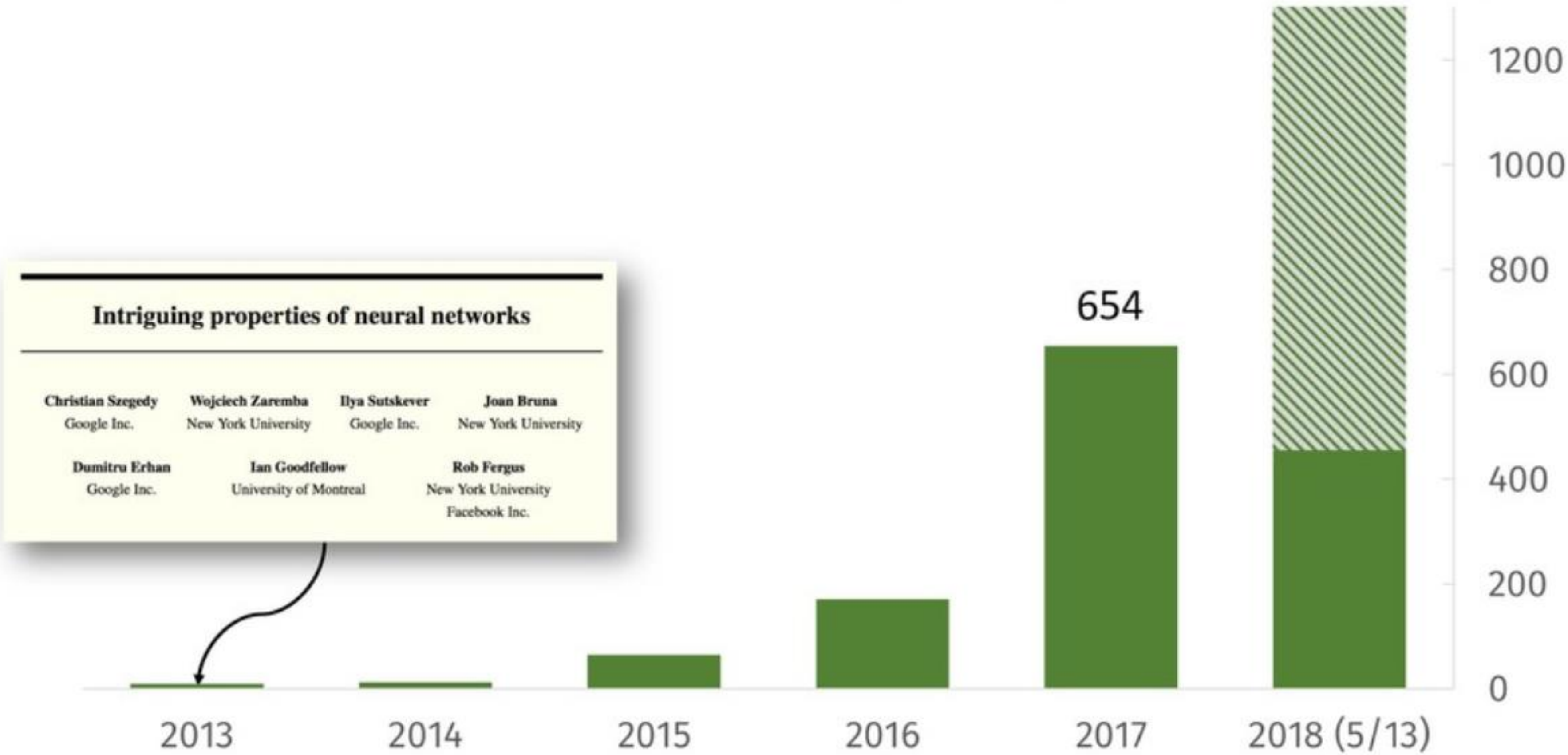
Probability and statistics

# Adversarial Machine Learning



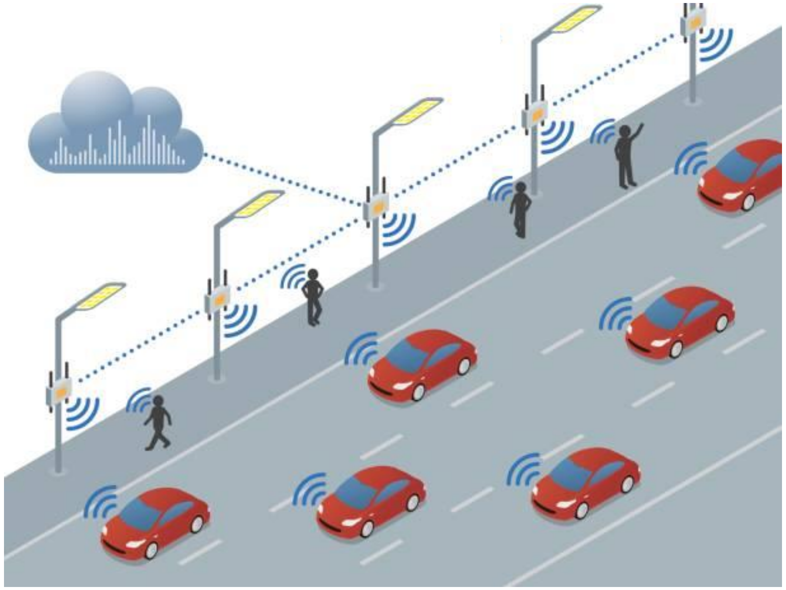

- Studies attacks against machine learning systems
- Designs robust machine learning algorithms that resist sophisticated attacks
- Many challenging open problems!

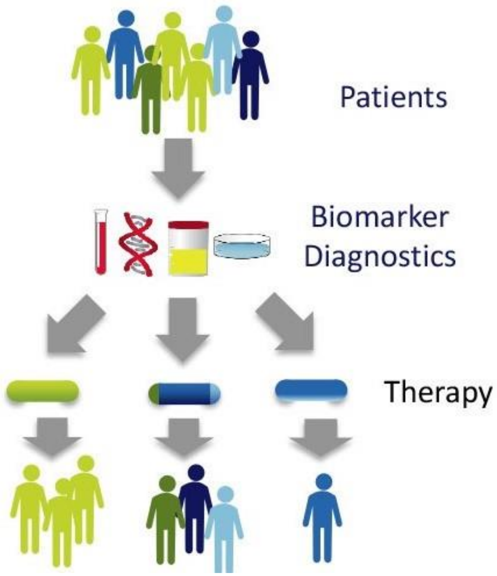
# Papers on “Adversarial Examples” (Google Scholar)




Source: David Evans, University of Virginia

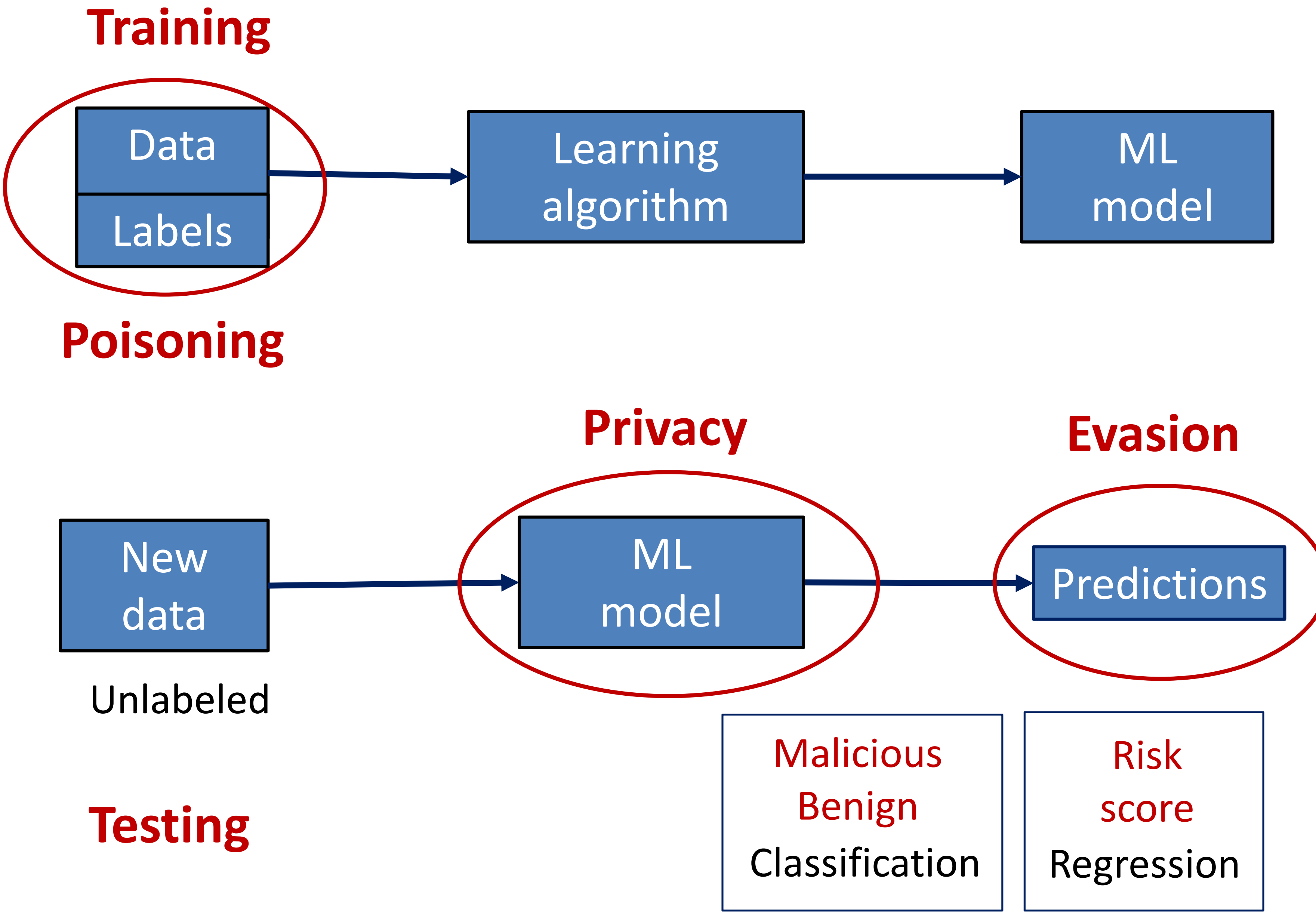
# Why is it important?






Many critical applications where ML/AI will be deployed

# Attacks against supervised learning



# Taxonomy

Attacker's Objective

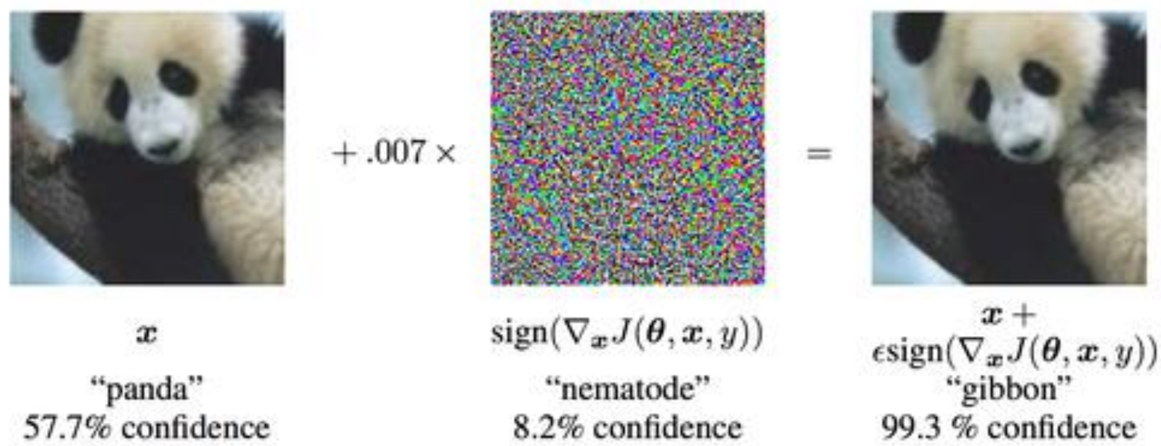
Learning Stage

| | **Targeted** Modify predictions on targeted set of points | **Availability** Corrupt entire ML model | **Privacy** Learn information about model and data |
|---|---|---|---|
| **Training** | Targeted poisoning<br>Backdoor<br>Trojan attacks | Poisoning availability | - |
| **Testing** | Evasion attacks<br>Adversarial examples | - | Model extraction<br>Model inversion |

# Outline

- Evasion (testing-time) attacks
  - Adversarial examples
  - Optimization formulation
  - Applications to connected cars
  - Applications to cyber security
- Poisoning (training-time) attacks
  - Availability attacks for linear regression
  - Applications to health care
  - Defenses

# Evasion attacks




Adversarial example

- [Szegedy et al. 13] Intriguing properties of neural networks
- [Biggio et al. 13] Evasion Attacks against Machine Learning at Test Time
- [Goodfellow et al. 14] Explaining and Harnessing Adversarial Examples
- [Carlini, Wagner 17] Towards Evaluating the Robustness of Neural Networks
- [Madry et al. 17**]** Towards Deep Learning Models Resistant to Adversarial Attacks
- [Kannan et al. 18] Adversarial Logit Pairing

# Adversarial example definition

- Given ML model $f$ and point $x$ with class $c$
  - $f(x) = c$
- Try to modify it minimally to get target class $t$
- Point $x'$ is an *adversarial example* if
  - $f(x') = t$ (prediction is targeted class)
  - $\mathrm{Dist}(x, x') \leq \delta$ (distance from original image is small)
- State-of-the-art attack based on Gradient Descent optimization to find closest adversarial example
  - [Carlini and Wagner 2017]

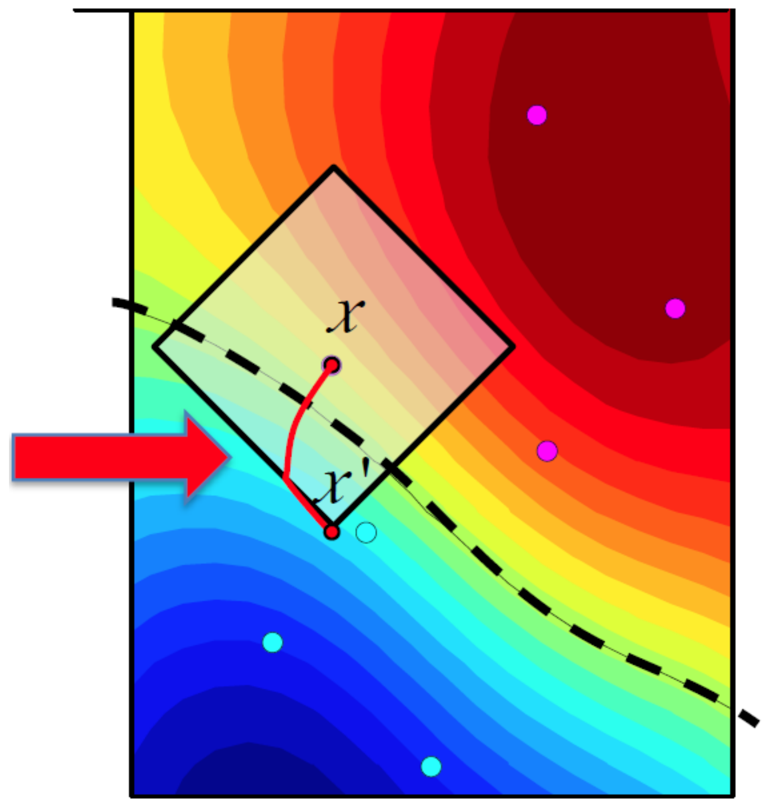
# Optimization Formulation

Given original example $x$, $f(x) = c$
Find adversarial example $x'$

$$\min \|x - x'\|_2^2$$

Such that $f(x') = t$

[Szegedy et al. 13] Intriguing properties of neural networks



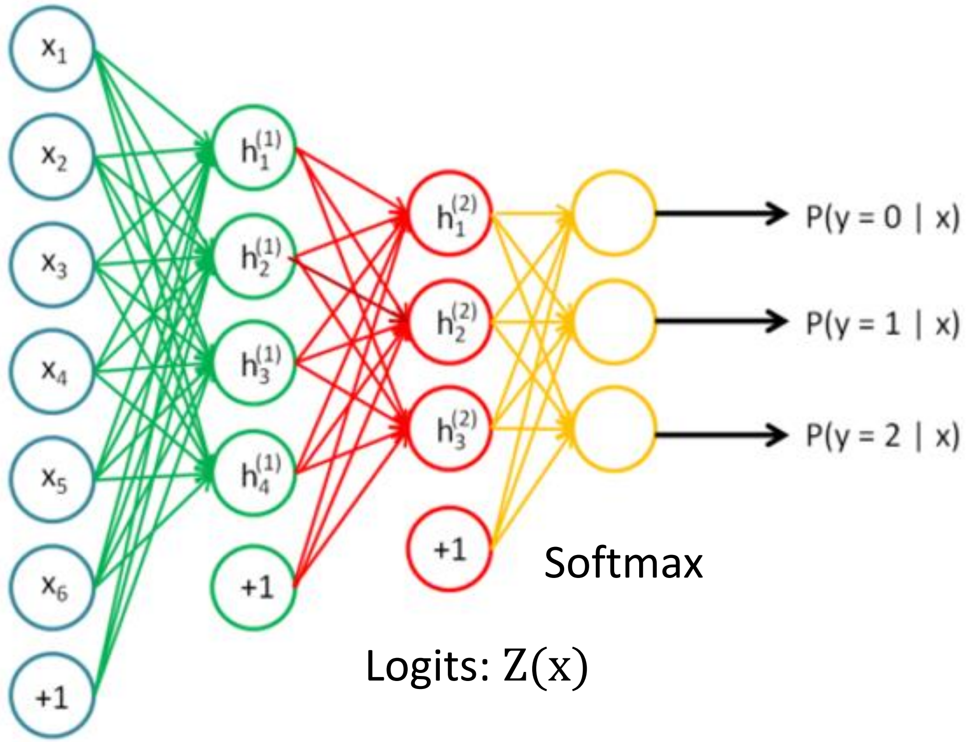

Equivalent formulation

$$\min c\|x - x'\|_2^2 + \ell_t(x')$$

$\ell_t(x')$ is loss function on $x'$

# Evasion attacks in logit layer

Input: Images represented as feature vectors




[Carlini and Wagner 2017]
Penalty method

$$\min c||\delta||_2^2 + Z_c(x') - Z_t(x')$$
$$x' = x + \delta$$

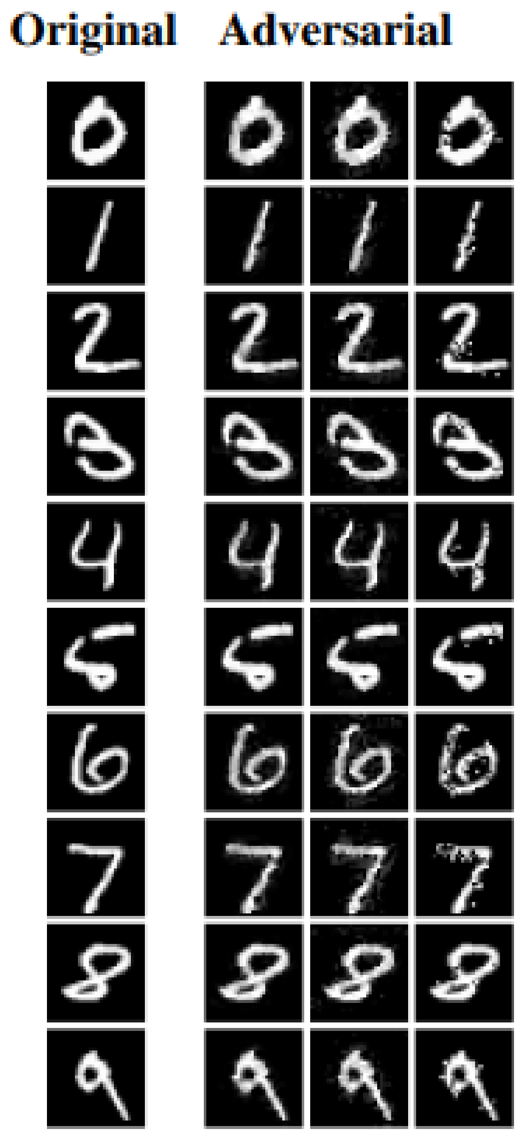
Solve iteratively using Gradient Descent by $\delta$

# Attacks on MNIST data

[Carlini and Wagner 2017]
Penalty method

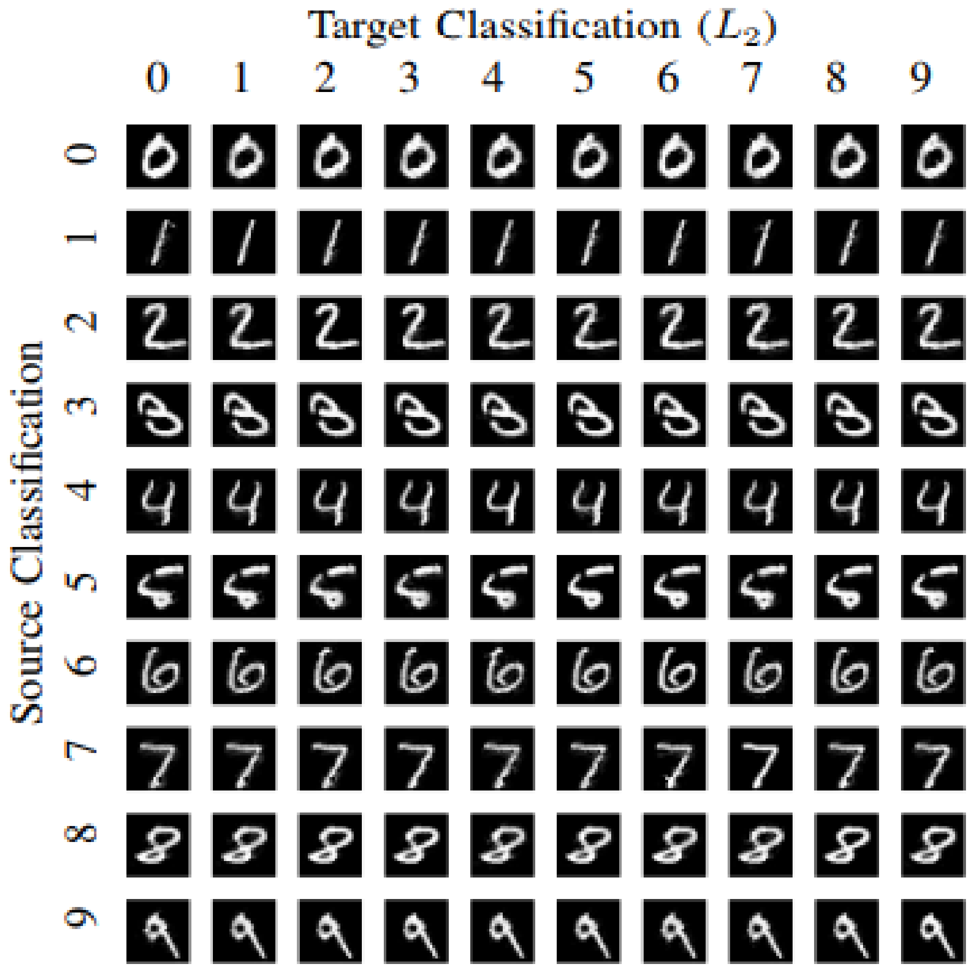
Uses 3 distance metrics

- $L_0$: number of pixels changed
- $L_2$: Euclidean distance
- $L_\infty$: max perturbation of each pixel



# Attacks on Euclidean distance

[Carlini and Wagner 2017]
Penalty method



# Adversarial Glasses

- Sharif et al. (ACM CCS 2016) attacked deep neural networks for face recognition with carefully-fabricated eyeglass frames
- When worn by a **41-year-old white male** (left image), the glasses mislead the deep network into believing that the face belongs to the famous actress **Milla Jovovich**

- Physically realizable attacks
- [Sharif et al. 2016] Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition

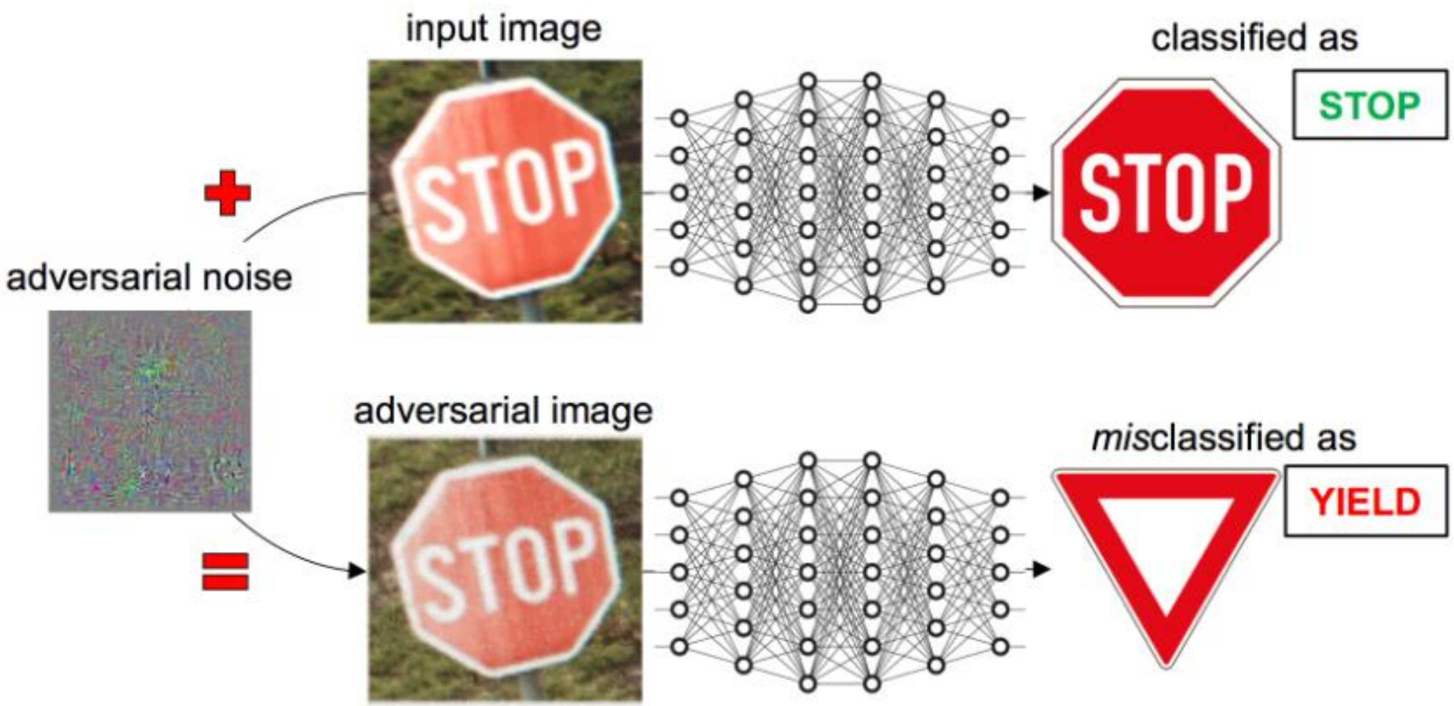
# Adversarial Road Signs

## Robust Physical-World Attacks on Machine Learning Models

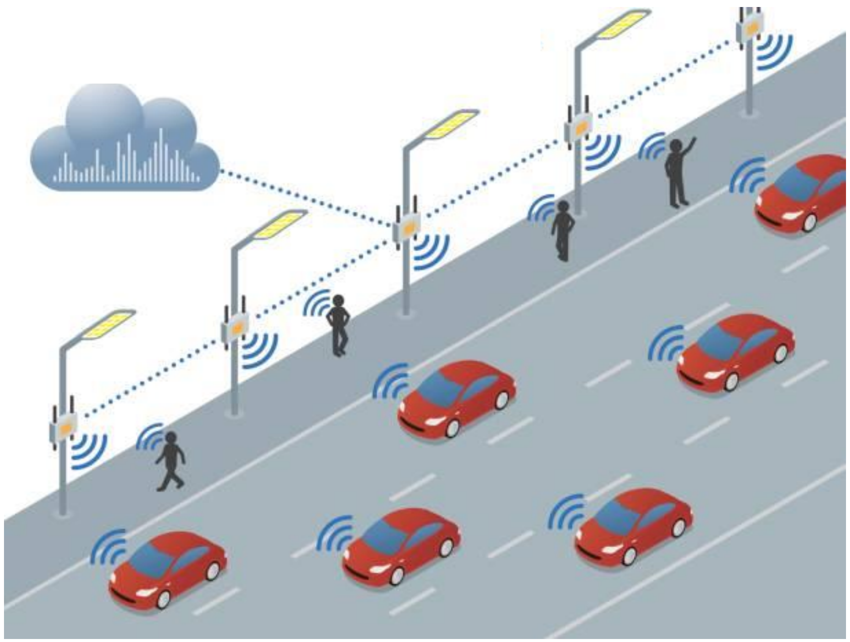
Ivan Evtimov[1], Kevin Eykholt[2], Earlence Fernandes[1], Tadayoshi Kohno[1],
Bo Li[4], Atul Prakash[2], Amir Rahmati[3], and Dawn Song*[4]

[1]University of Washington
[2]University of Michigan Ann Arbor
[3]Stony Brook University
[4]University of California, Berkeley

# Road Sign Misclassification



# Why Relevant in Self-Driving Cars?

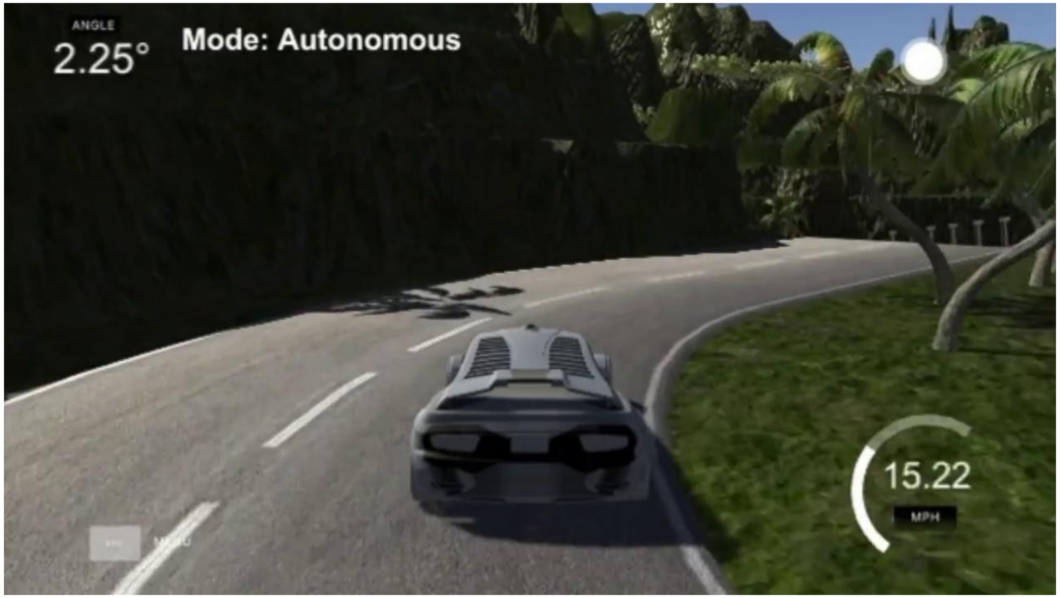


**Machine learning has tremendous potential:**
- Assist drivers by processing sensor data from ECUs
- Predict road conditions by interacting with other cars
- Recognize risky conditions and warn drivers

But safety is of paramount importance!

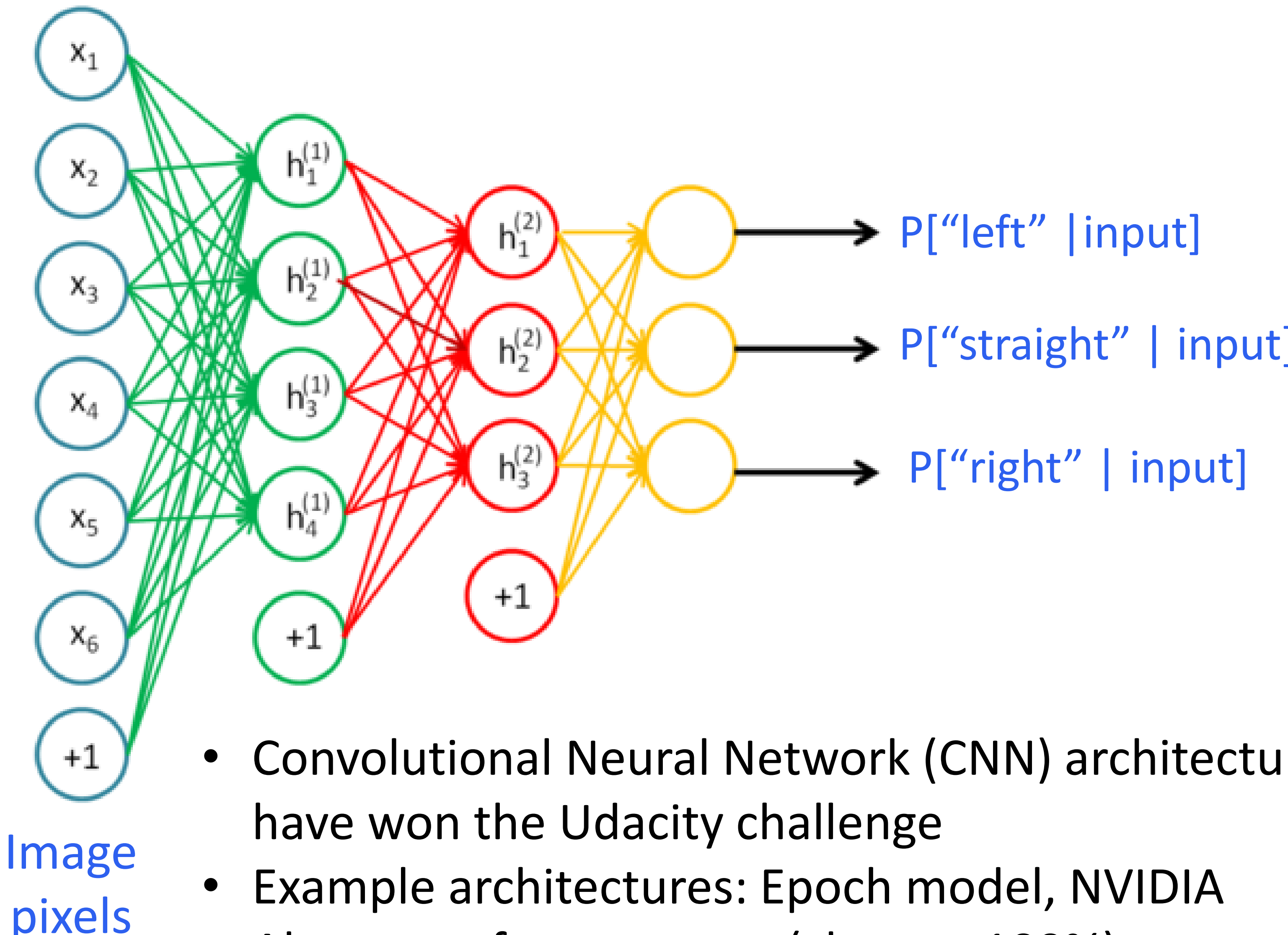
# Example Application

- Steering angle prediction by processing camera image
- Udacity challenge: public competition and dataset available




[A. Chernikova, M. Jagielski, A. Oprea, C. Nita-Rotaru, and B. Kim. Are Self-Driving Cars Secure? Evasion Attacks against Deep Neural Networks for Self-Driving Cars. In IEEE SafeThings, 2019]

# Deep Neural Networks



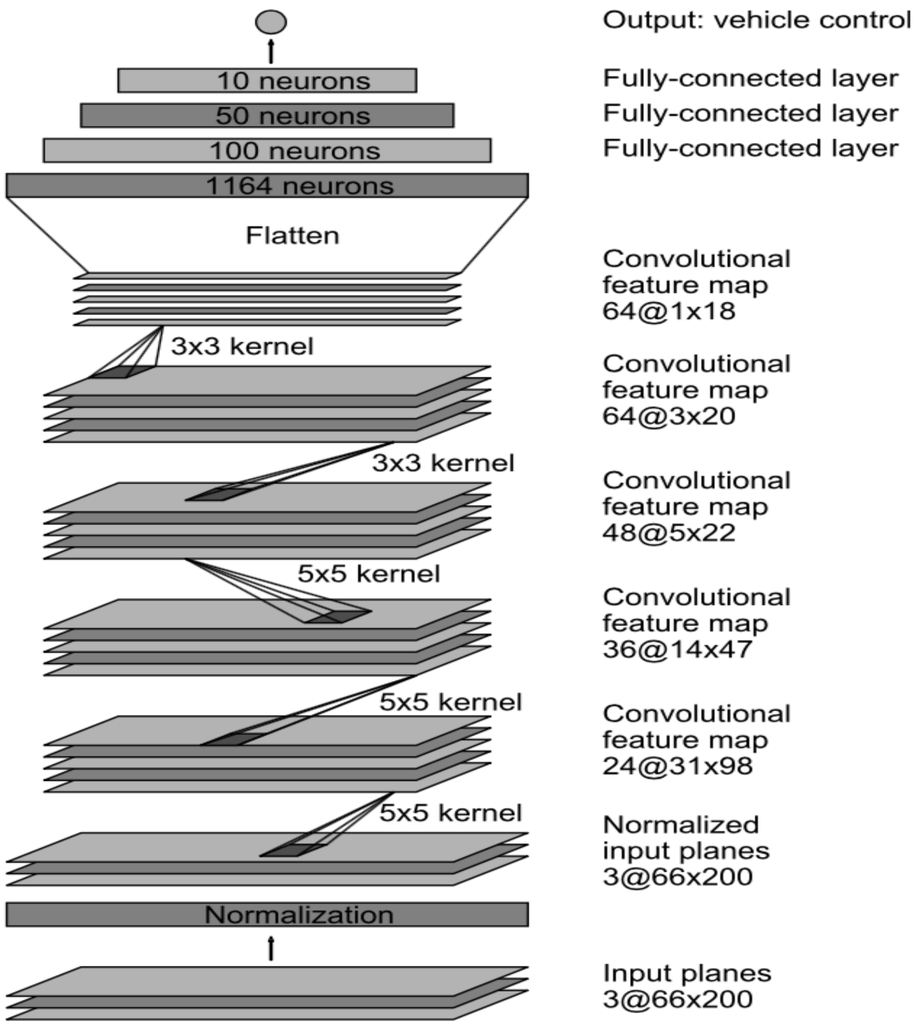

- Convolutional Neural Network (CNN) architectures have won the Udacity challenge
- Example architectures: Epoch model, NVIDIA
- Almost perfect accuracy (close to 100%)

# CNN Architecture Epoch

```
x = Convolution2D(32, 3, 3, activation='relu', border_mode='same')(img_input)
x = MaxPooling2D((2, 2), strides=(2, 2))(x)
x = Dropout(0.25)(x)

x = Convolution2D(64, 3, 3, activation='relu', border_mode='same')(x)
x = MaxPooling2D((2, 2), strides=(2, 2))(x)
x = Dropout(0.25)(x)

x = Convolution2D(128, 3, 3, activation='relu', border_mode='same')(x)
x = MaxPooling2D((2, 2), strides=(2, 2))(x)
x = Dropout(0.5)(x)

y = Flatten()(x)
y = Dense(1024, activation='relu')(y)
y = Dropout(.5)(y)
y = Dense(1)(y)

model = Model(input=img_input, output=y)
model.compile(optimizer=Adam(lr=1e-4), loss = 'mse')
```

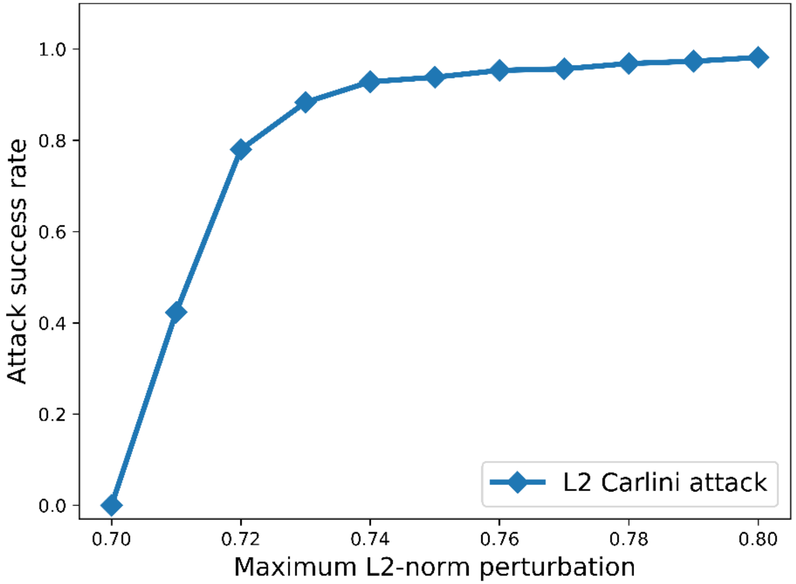
25 million parameters

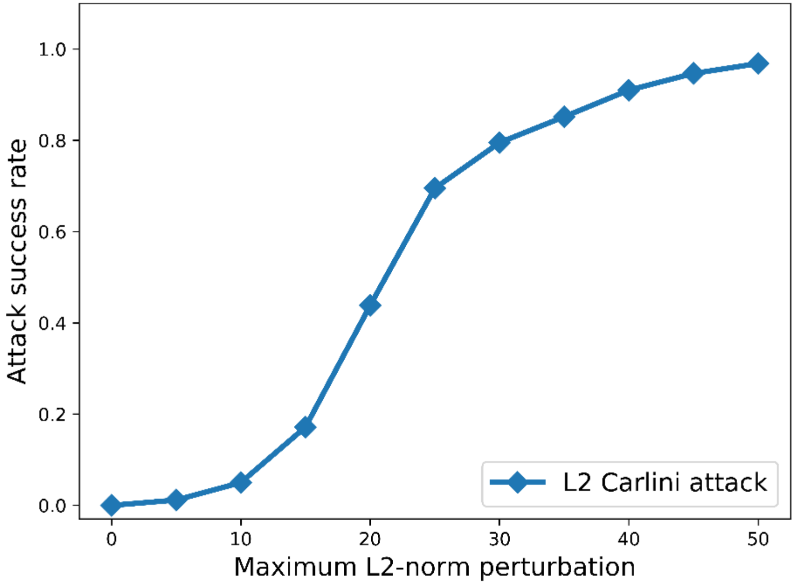
# CNN Architecture NVIDIA




467 million parameters

# How successful is the attack?

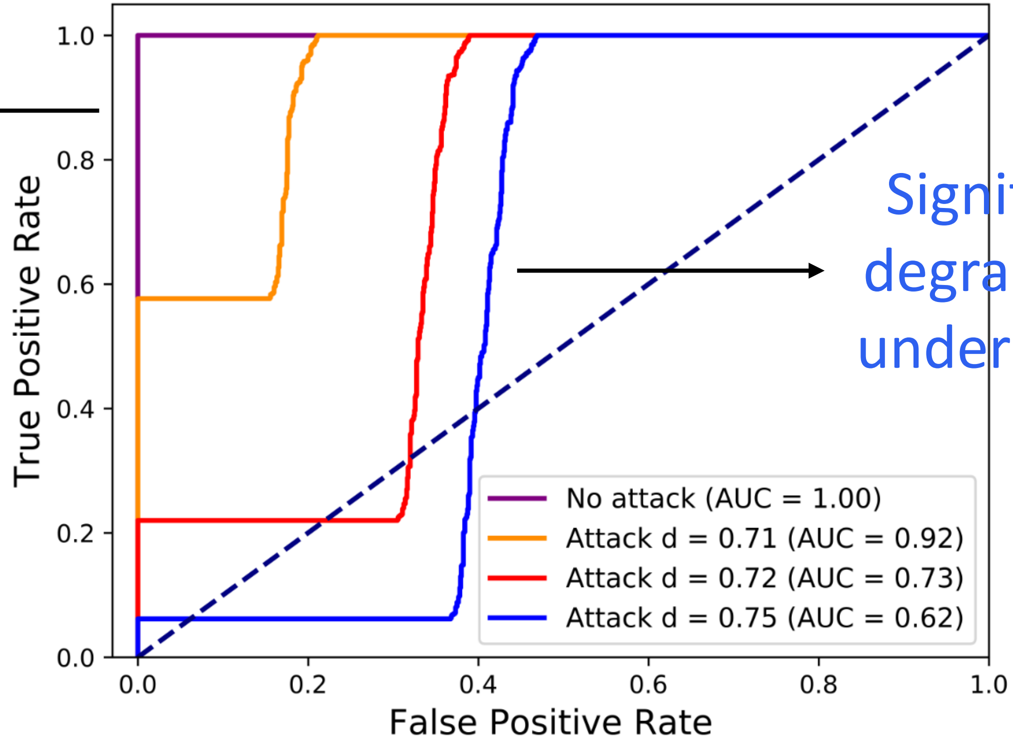



- Both models: small modification to the image results in 100% attack success
- NVIDIA model is more resilient!

# How much is the attack impacting the classification?

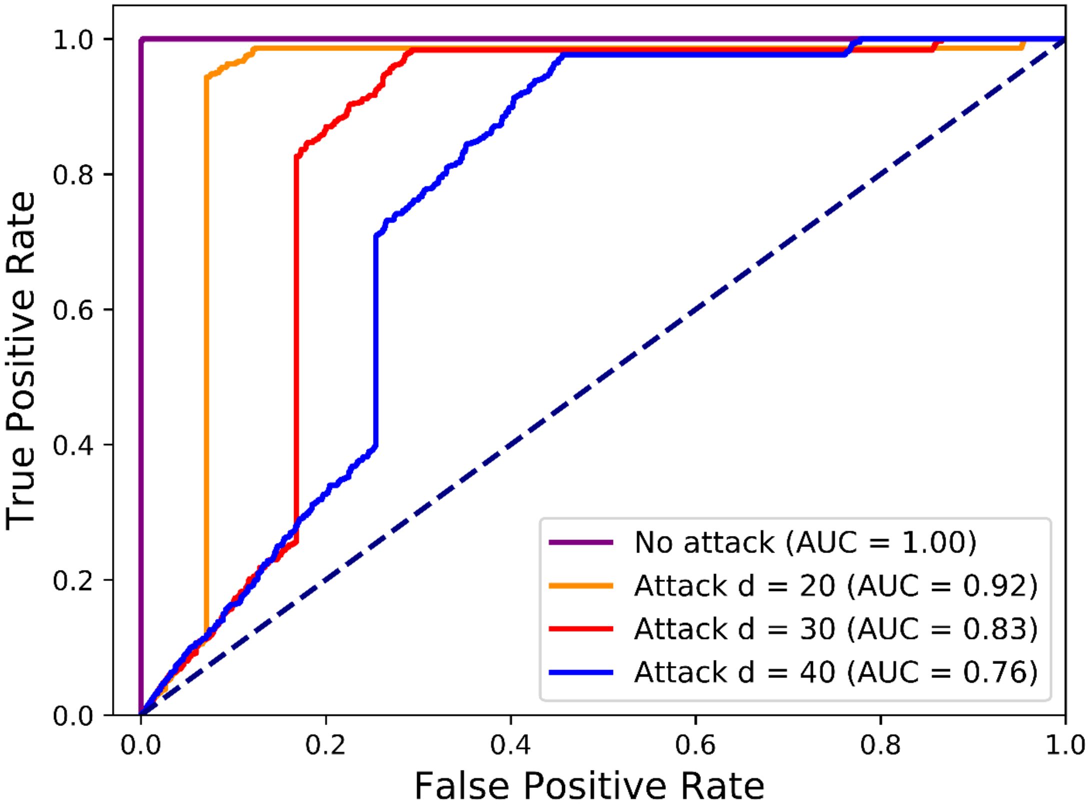
Perfect accuracy (no attack)




Significant degradation under attack

Epoch model

# How much is the attack impacting the classification?

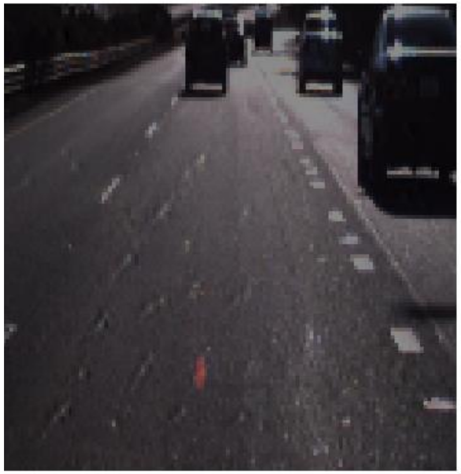



NVIDIA model

# Example Adversarial Images

Original Image
Class “Straight”

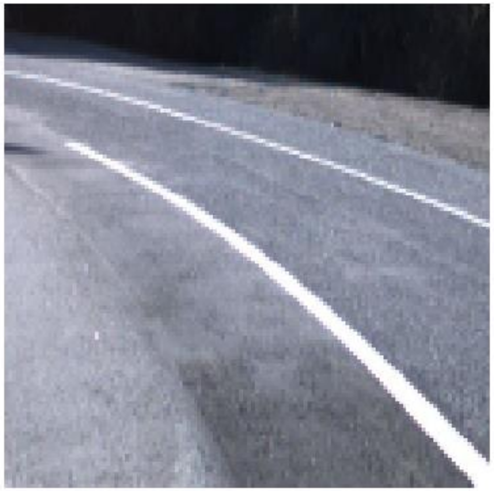
Adversarial Image
Class “Right”

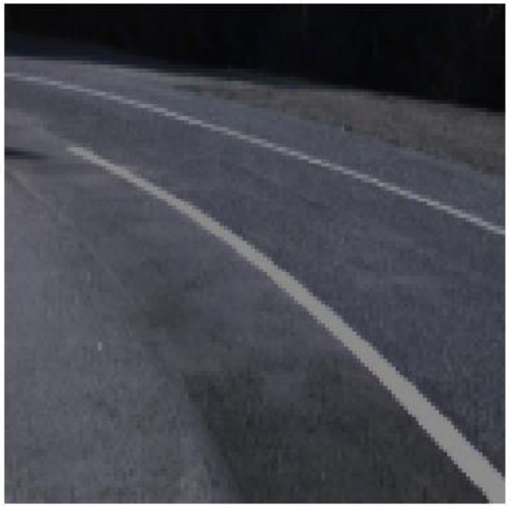


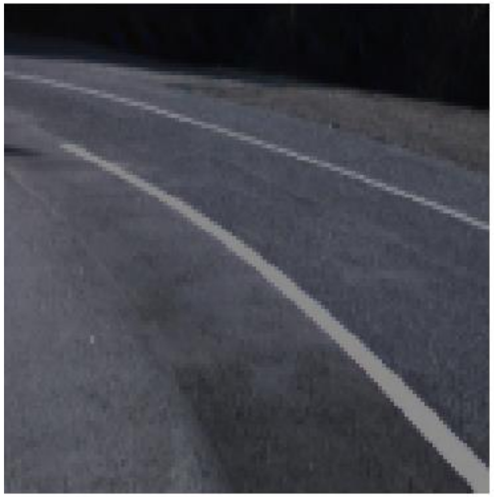
Adversarial Image
Class “Left”

Epoch model

# Example Adversarial Images







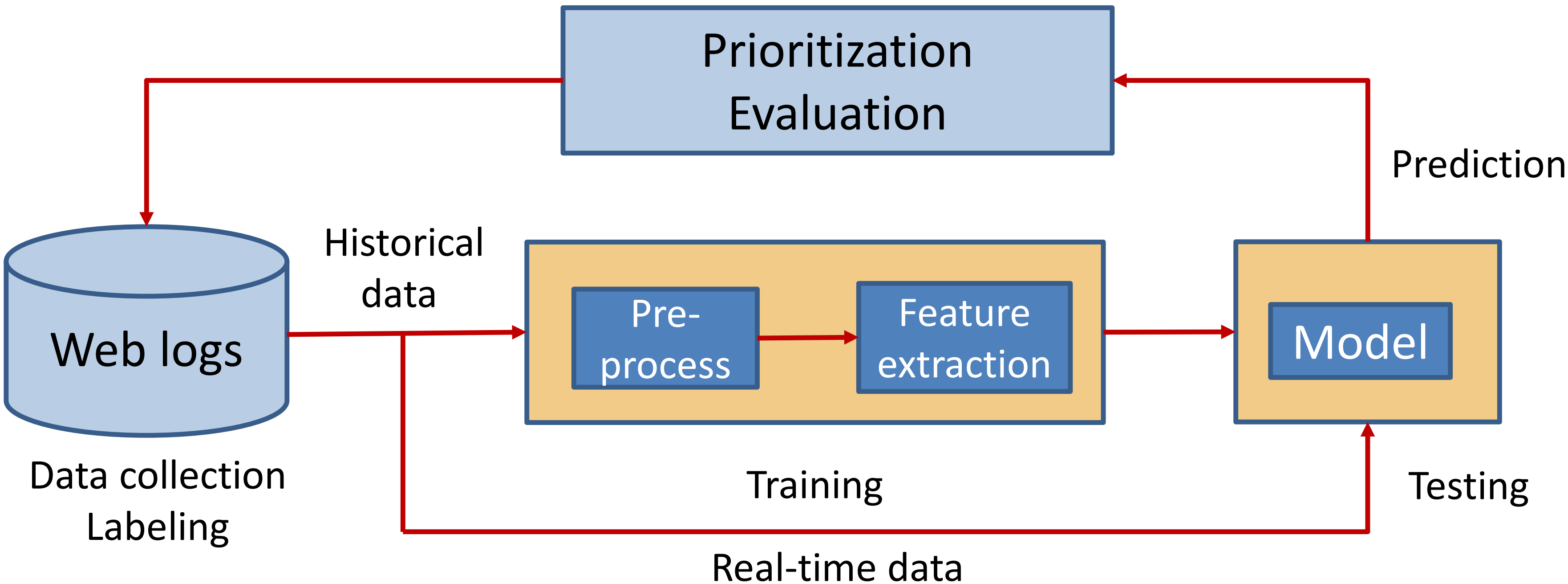
Original Image
Class "Left"

Adversarial Image
Class "Straight"

Adversarial Image
Class "Right"

NVIDIA model

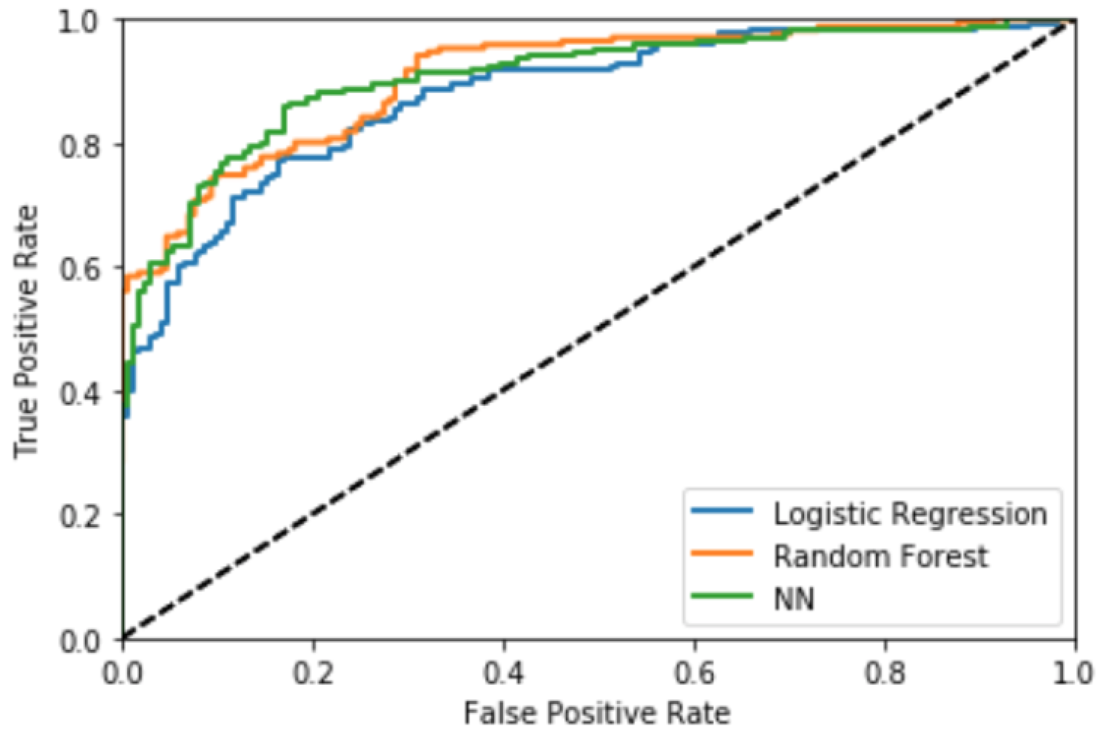
# Malware Detection




- Extract 89 features of malicious activities from web logs
  - Leverage security domain expertise
- Supervised learning models
  - Logistic regression, SVM, Decision trees, Random Forest
- Evaluation of higher risk alerts involves manual investigation
  - Prioritize most suspicious connections
- [A. Chernikova and A. Oprea. Adversarial Examples for Deep-Learning Cyber Security Analytics. In progress, 2019]

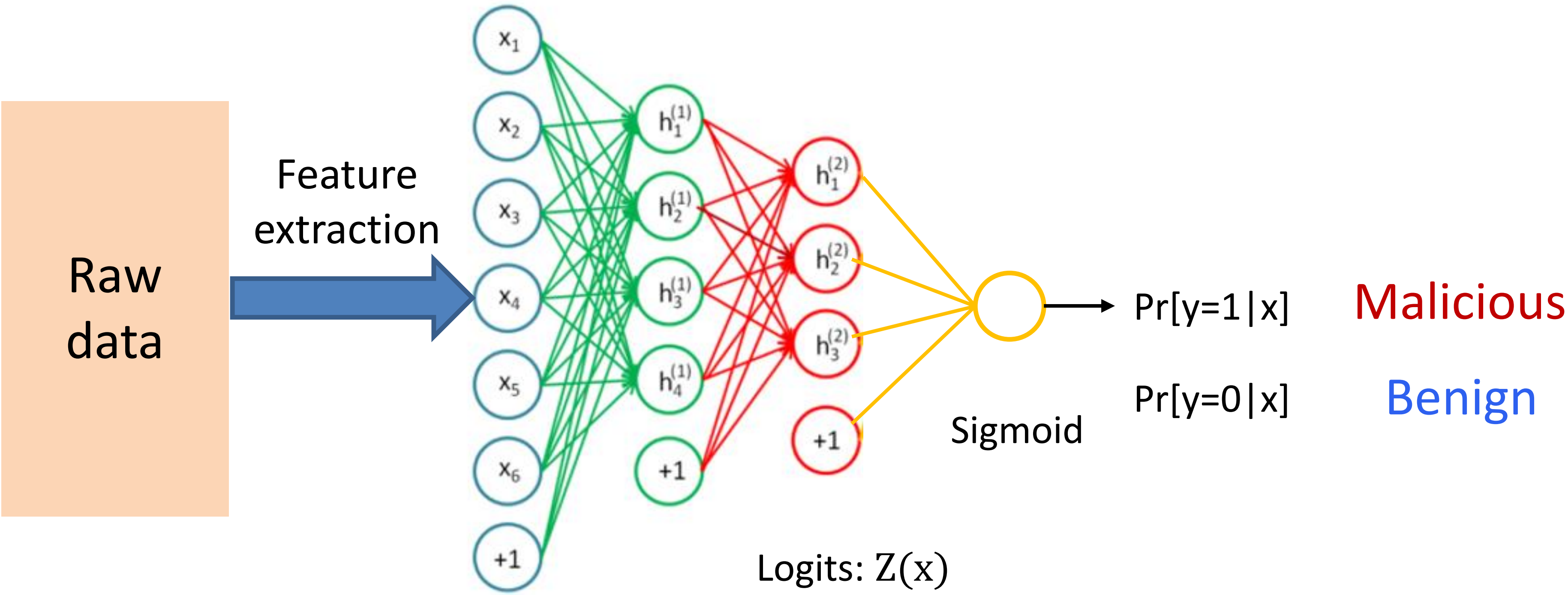
# Classification results

- Feed-Forward Neural Network (3 hidden layers)
- Highly imbalanced setting
  - 227k legitimate domains, 1730 malicious domains




How resilient are Feed-Forward Neural Networks to adversarial evasion attacks?

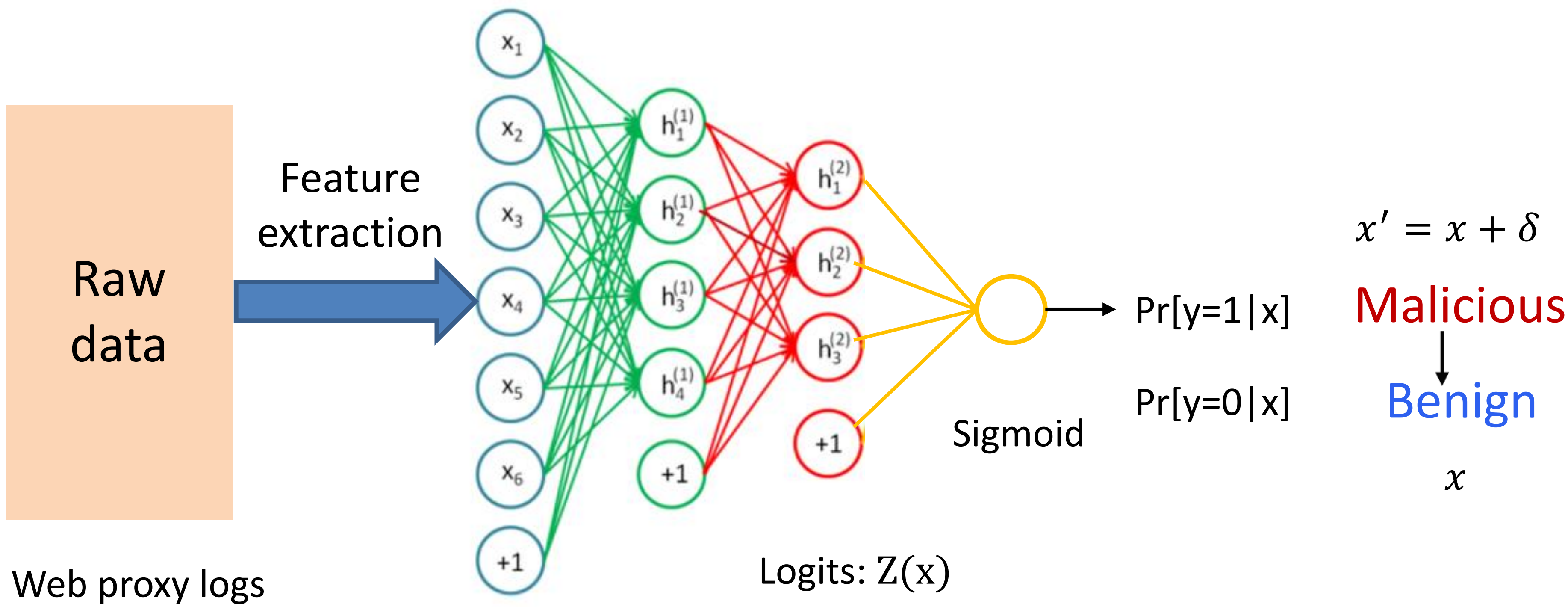
# Evasion attacks in security




Challenges

- In cyber security, classifiers are usually applied to pre-processed features, not raw data
- Features have constraints (e.g., min, max, and avg number of connections per host)

# Iterative evasion attack algorithm

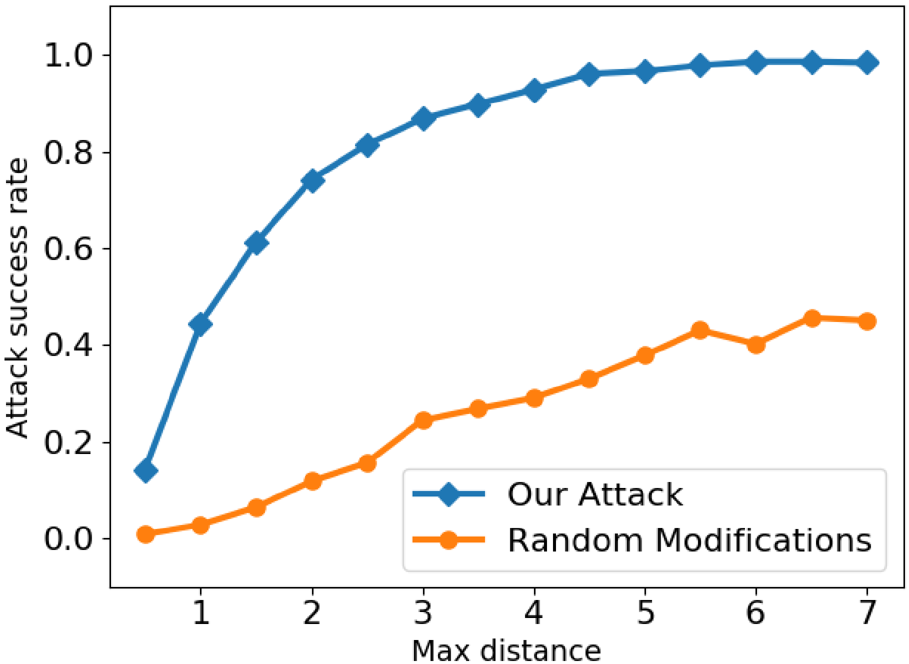


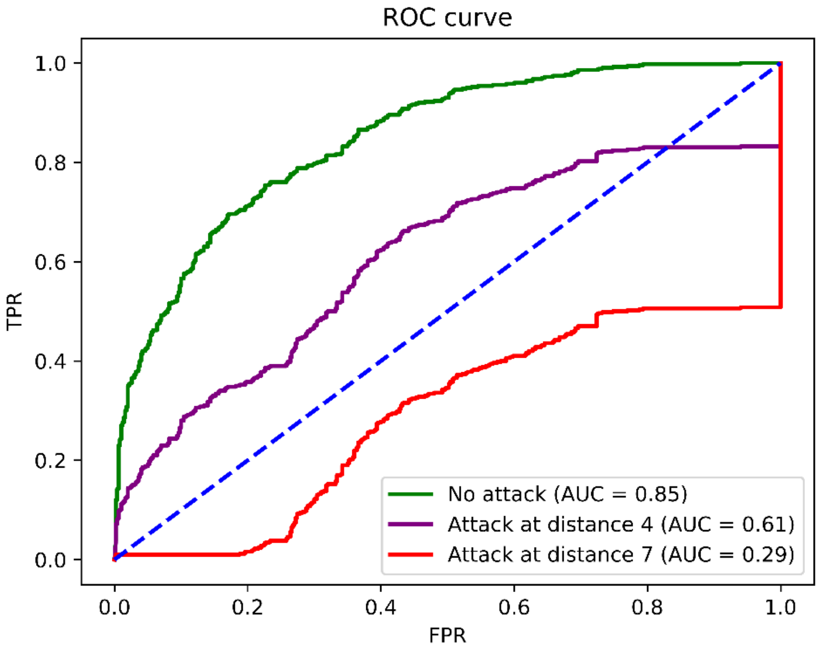

Attack Algorithm

Repeat
- Compute gradient on all features
- Select feature of max gradient
- Modify subset of related features while preserving constraints
- Project to feasible space

Until max distance reached or attack successful

# How Effective are Evasion Attacks in Security?




Feed-Forward Neural Network
83 features extracted from enterprise network traffic

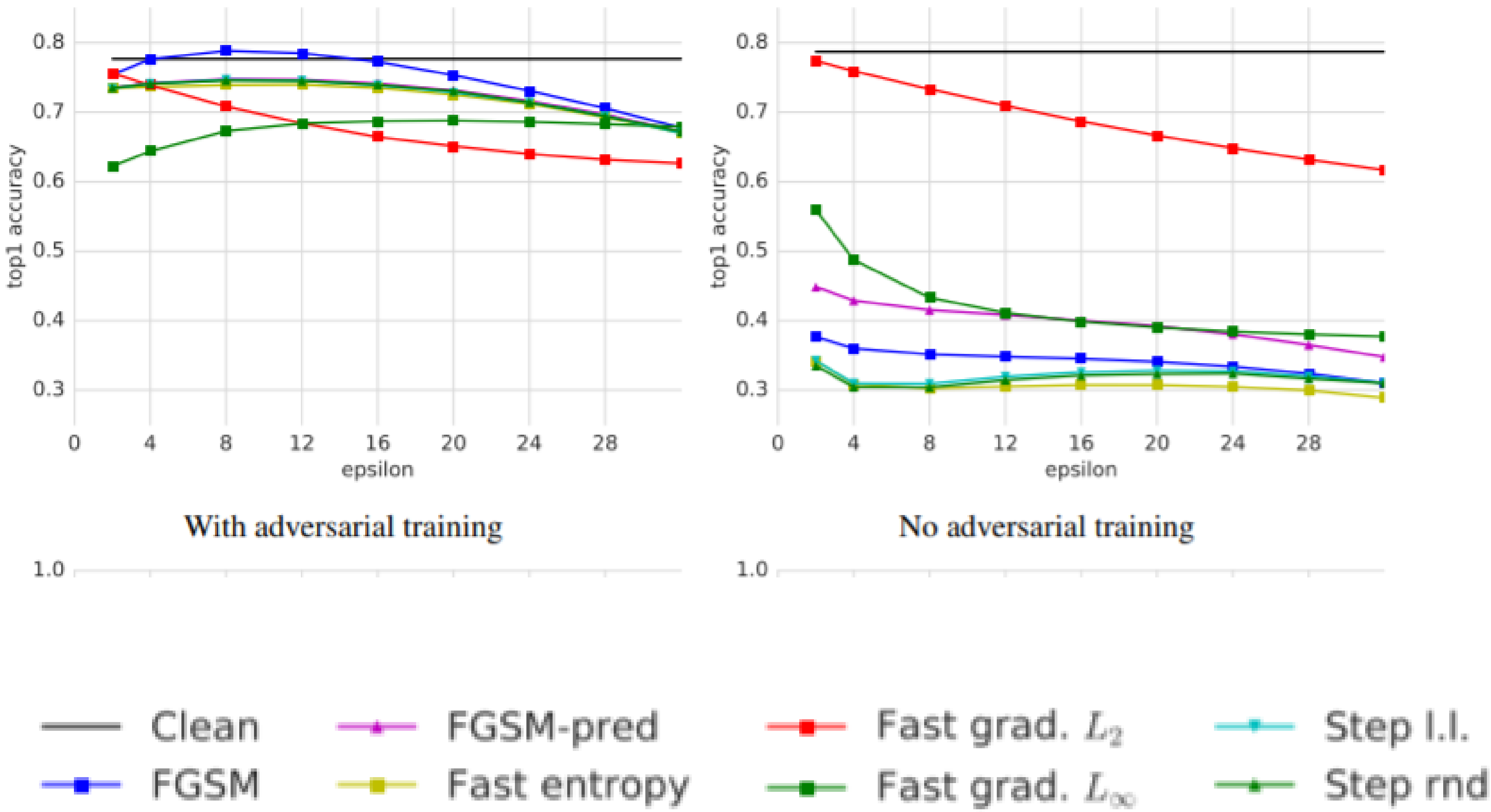
# Adversarial Training

**Algorithm 1** Adversarial training of network $N$.
Size of the training minibatch is $m$. Number of adversarial images in the minibatch is $k$.

1: Randomly initialize network $N$
2: **repeat**
3: Read minibatch $B = \{X^1, \ldots, X^m\}$ from training set
4: Generate $k$ adversarial examples $\{X^1_{adv}, \ldots, X^k_{adv}\}$ from corresponding clean examples $\{X^1, \ldots, X^k\}$ using current state of the network $N$
5: Make new minibatch $B' = \{X^1_{adv}, \ldots, X^k_{adv}, X^{k+1}, \ldots, X^m\}$
6: Do one training step of network $N$ using minibatch $B'$
7: **until** training converged

- I. Goodfellow et al. Explaining and harnessing adversarial examples, ICLR 2015.
- A. Kurakin et al. Adversarial Machine Learning at Scale, ICLR 2017.
- Many other defenses have been broken
  - [Athalye et al. ICML 2018]: Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples

# Is Adv Training Effective?



# Outline

- Evasion (testing-time) attacks
  - Adversarial examples
  - Optimization formulation
  - Applications to connected cars
  - Applications to cyber security
- Poisoning (training-time) attacks
  - Availability attacks for linear regression
  - Applications to health care
  - Defenses

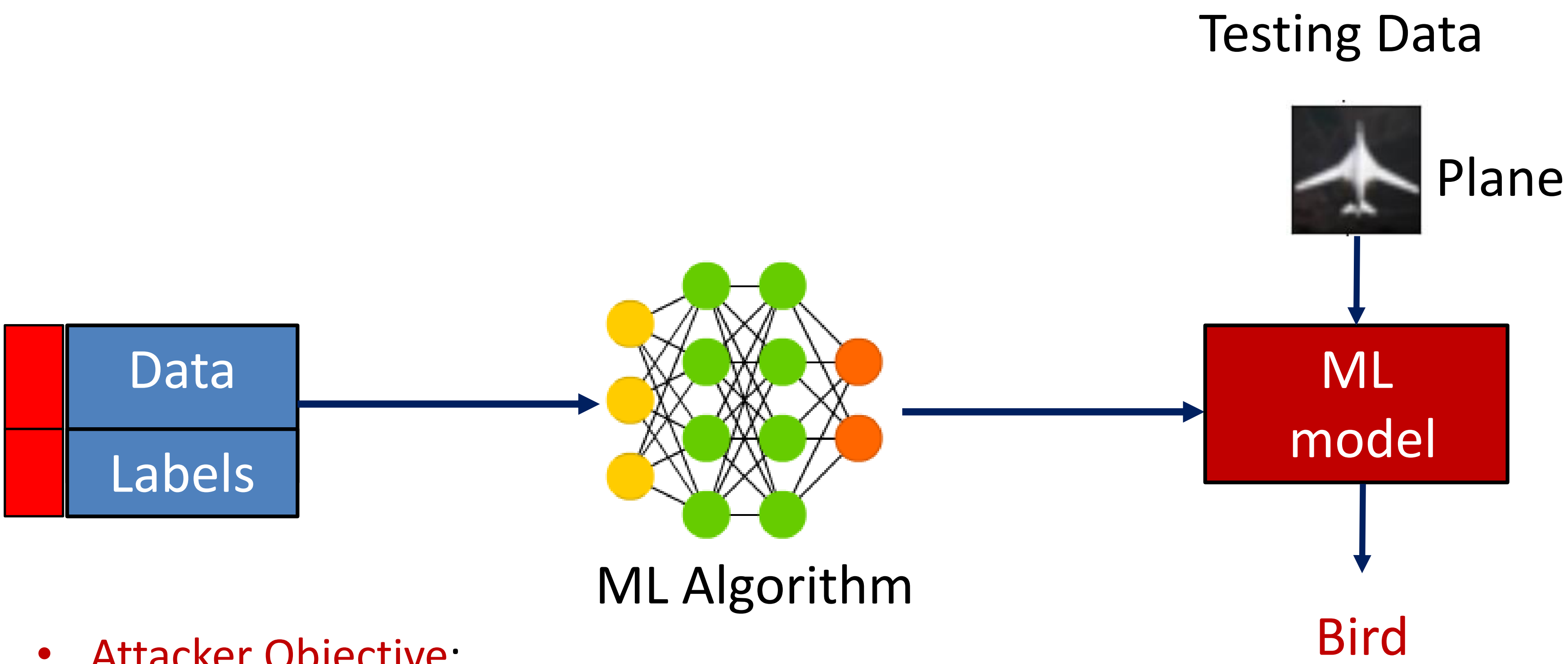
# Training-Time Attacks

- ML is trained by crowdsourcing data in many applications

- Social networks
- News articles
- Tweets


- Cannot fully trust training data!

# Poisoning Availability Attacks




- Attacker Objective:
  - Corrupt the predictions by the ML model significantly
  - Predictions on *most points* are impacted in testing
- Attacker Capability:
  - Insert fraction of poisoning points in training
- [M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li. Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. In IEEE S&P 2018]

# Optimization Formulation

Given a training set $D$ find a set of poisoning data points $D_p$

that maximizes the adversary objective $A$ on validation set $D_{val}$

where corrupted model $\boldsymbol{\theta}_p$ is learned by minimizing the loss function $L$ on $D \cup D_p$

$$\operatorname*{argmax}_{D_p} A(D_{val}, \boldsymbol{\theta}_p)\ s.t.$$
$$\boldsymbol{\theta}_p \in \operatorname*{argmin}_{\boldsymbol{\theta}} L(D \cup D_p, \boldsymbol{\theta})$$

Implicit dependence

Optimization formulation in white-box setting

– Attacker knows training data D, ML model

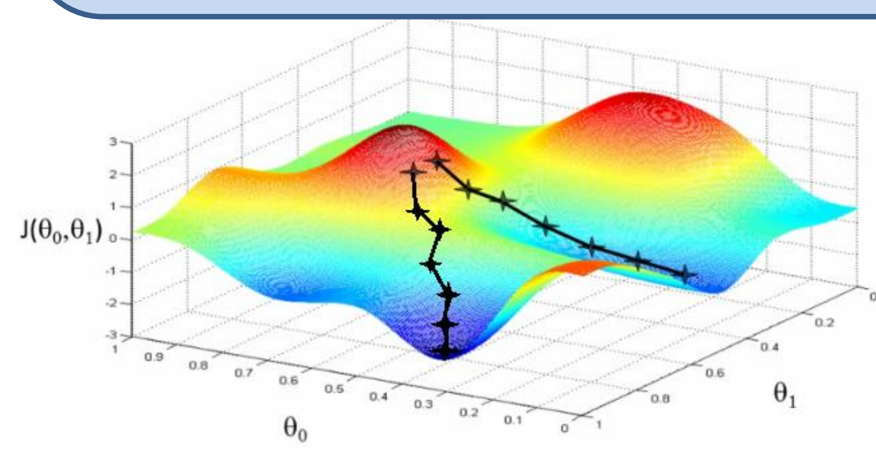
*Bilevel optimization problem is* NP hard in the general case

# Poisoning attack for Linear Regression

- Gradient ascent for classification [Biggio et al. 12, Xiao et al. 15]
- First white-box attack for regression [Jagielski et al. 18]
  - Determine optimal poisoning point ($\boldsymbol{x}_c$,$y_c$)
  - Objective is MSE; optimize by both $\boldsymbol{x}_c$ and $y_c$

$$\frac{\partial A}{\partial \boldsymbol{x}_c} = \sum_{i=1}^{n} 2(f(\boldsymbol{x}_i) - y_i)\left(\boldsymbol{x}_i^T \frac{\partial \boldsymbol{w}}{\partial \boldsymbol{x}_c} + \frac{\partial b}{\partial \boldsymbol{x}_c}\right) + \frac{\partial \Omega}{\partial \boldsymbol{w}} \frac{\partial \boldsymbol{w}}{\partial \boldsymbol{x}_c}$$

$$\frac{\partial A}{\partial y_c} = \sum_{i=1}^{n} 2(f(\boldsymbol{x}_i) - y_i)\left(\boldsymbol{x}_i^T \frac{\partial \boldsymbol{w}}{\partial y_c} + \frac{\partial b}{\partial y_c}\right) + \frac{\partial \Omega}{\partial \boldsymbol{w}} \frac{\partial \boldsymbol{w}}{\partial y_c}$$




- Different initializations and objectives
- Can be extended to multiple poisoning points

# Gradient Ascent Algorithm

- **Input**: poisoned point $x_0$, label $y_0$
  - Adversarial objective $A$
- **Output**: poisoned point $x$, label $y$

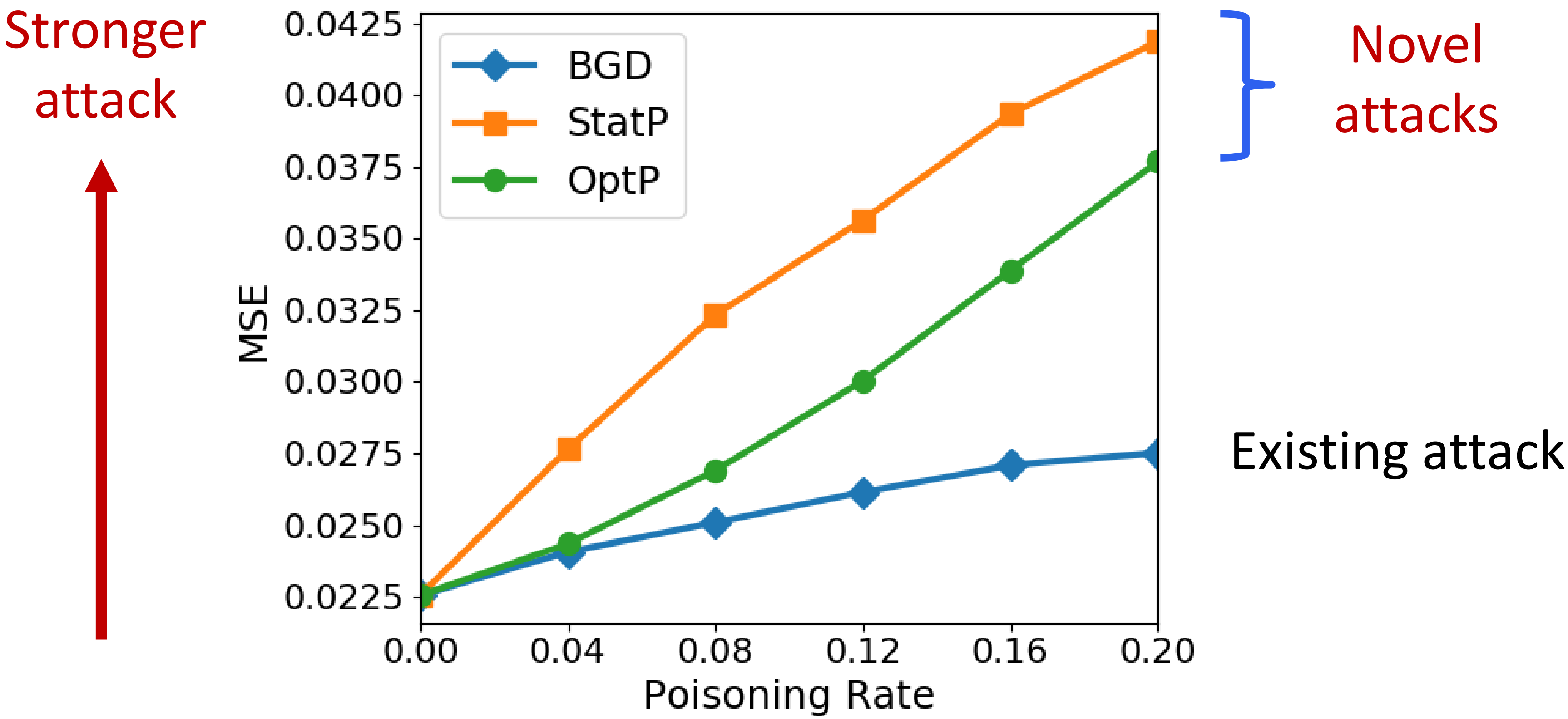
1. Initialize poisoned point $x \leftarrow x_0; y \leftarrow y_0$
2. Repeat
   - Store previous iteration $x_{pr} \leftarrow x; y_{pr} \leftarrow y$
   - Update in direction of gradients choosing $\alpha$ with line search and project to feasible space

$$\boxed{\begin{aligned} x &\leftarrow \Pi(\mathrm{x} + \alpha \nabla_x A(x, y)) \\ \mathrm{y} &\leftarrow \Pi(\mathrm{y} + \alpha \nabla_y A(x, y)) \end{aligned}}$$

3. Until $\left|A(x, y) - A\left(x_{pr}, y_{pr}\right)\right| < \epsilon$
4. Return $x, y$

# Attack results

- Improve existing attacks by a factor of 6.83




Predict loan rate with ridge regression
(i.e. with L2 regularization)

# Impact of attack

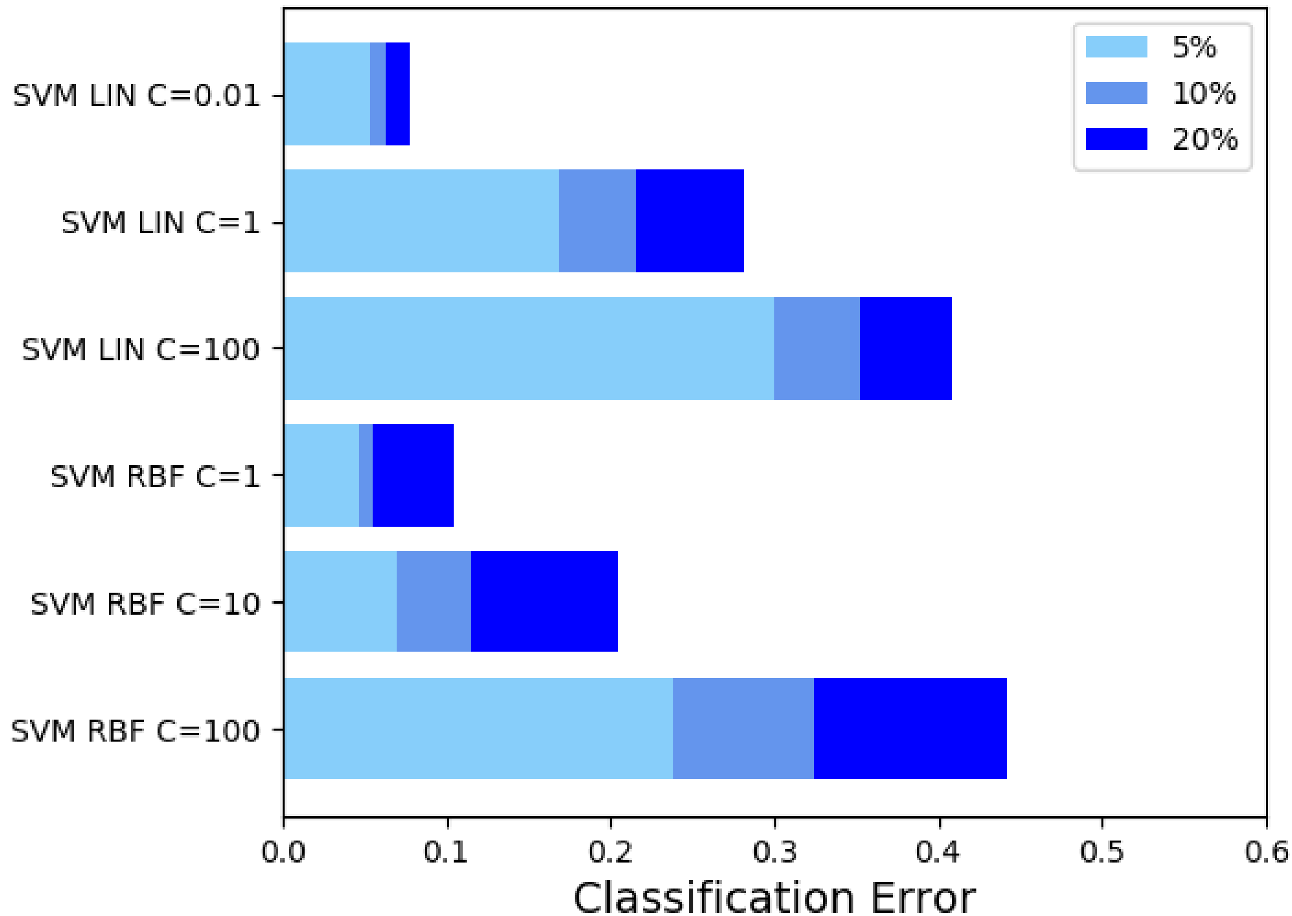
- How much would attack change dosages at 20% poisoning rate?
- Modifies 75% of patients' dosages by 87.5% for Ridge and 93.49% for Lasso

| Quantile | Initial Dosage | Ridge Difference | LASSO Difference |
|---|---|---|---|
| 0.1 | 15.5 mg/wk | 31.54% | 37.20% |
| 0.25 | 21 mg/wk | 87.50% | 93.49% |
| 0.5 | 30 mg/wk | 150.99% | 139.31% |
| 0.75 | 41.53 mg/wk | 274.18% | 224.08% |
| 0.9 | 52.5 mg/wk | 459.63% | 358.89% |

Case study on healthcare dataset

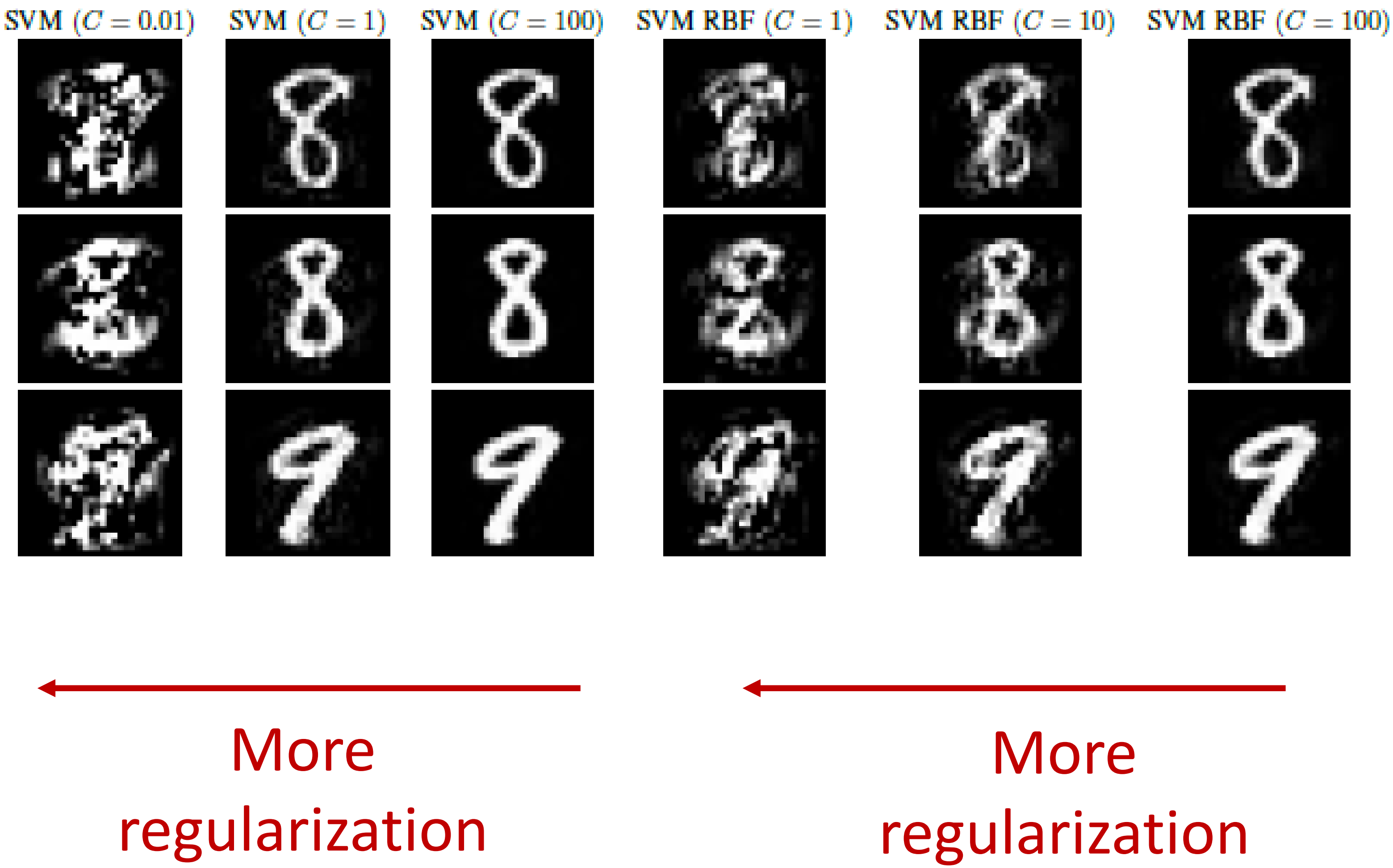
# Poisoning and Regularization

More regularization

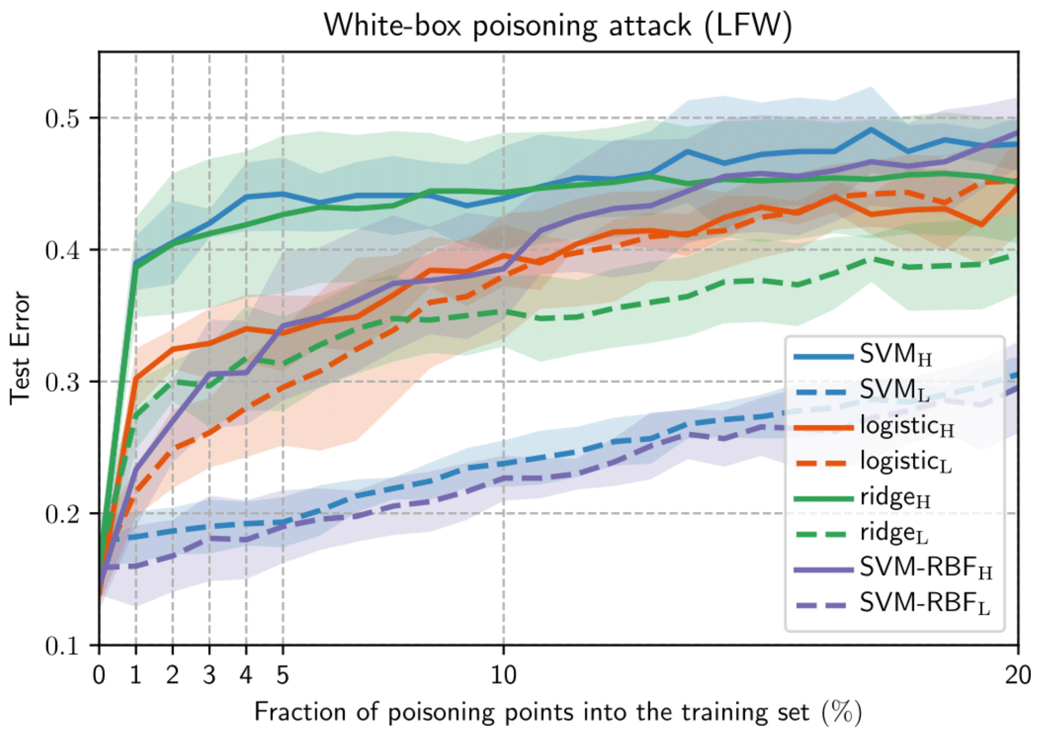



Stronger regularization provides more robustness to poisoning [Demontis et al. 18]

# Poisoning and Regularization



# Poisoning Classifiers



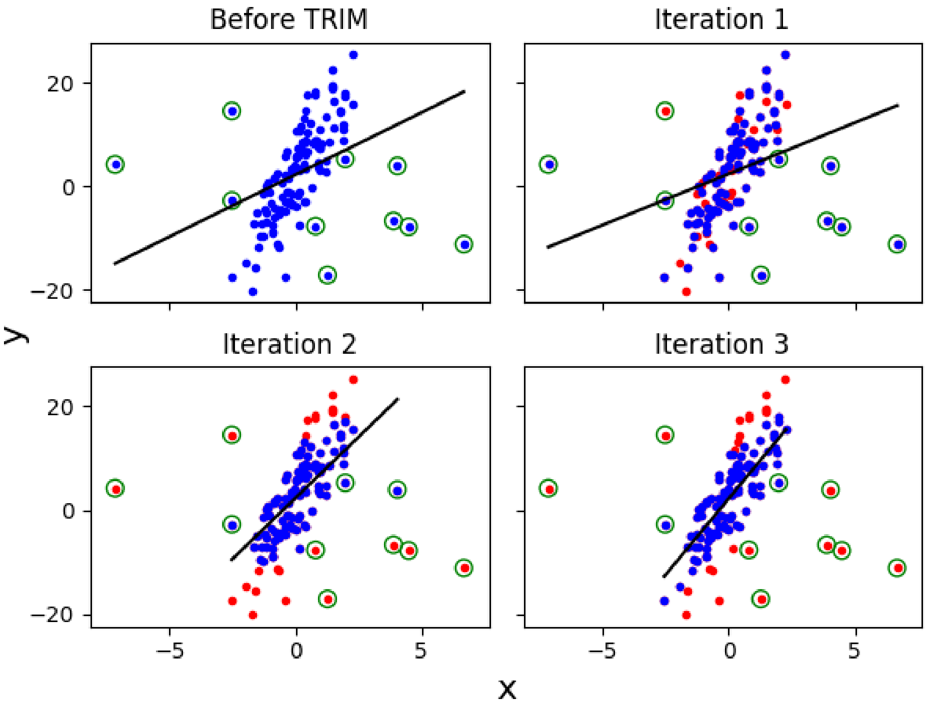

- More complex models (i.e., lower regularization) are more prone to poisoning
- Non-linear models more resilient than linear models
- Similar results for evasion

# Resilient Linear Regression

- Goal
  - Train a robust linear regression model, assuming $\alpha \cdot n$ poisoned points among N points in training
  - MSE should be close to original MSE
  - No ground truth on data distribution available
- Existing techniques
  - Robust statistics
    - Huber [Huber 1964], RANSAC [Fischler and Bolles 1961]
    - Resilient against outliers and random noise
  - Adversarial resilient regression: [Chen et al. 13]
    - Make simplifying assumption on data distribution (e.g., Gaussian)

# Our Defense: TRIM

- Given dataset on n points and $\alpha n$ attack points, find best model on $n$ of $(1+\alpha)n$ points
- If $\boldsymbol{w}, b$ are known, find points with smallest residual
- But $\boldsymbol{w}, b$ and true data distribution are unknown!




TRIM: alternately estimate model and find low residual points

$$\operatorname*{argmin}_{w,b,I} L(w,b,I) = \frac{1}{|I|}\sum_{i\in I}(f(\boldsymbol{x}_i) - y_i)^2 + \lambda\Omega(\boldsymbol{w})$$

$$N = (1+\alpha)n, \qquad I \subset [1, \dots, N], \qquad |I| = n$$

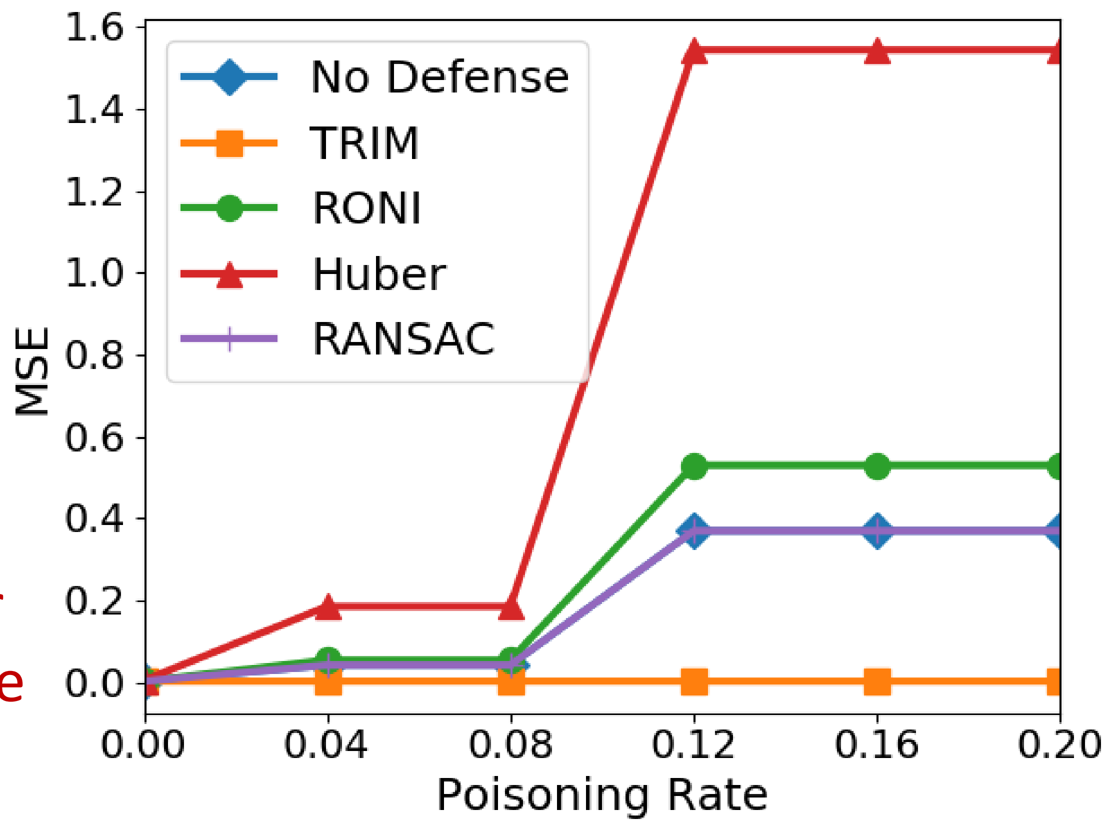
# Trimmed optimization

- Estimate model parameters and identify points with minimum residual alternatively
  - Alternating optimization
- Select $I$ a random subset in $\{1, \ldots, N\}, |I| = n$
  - Assume poisoning rate (or upper bound) is known
- Repeat
  - Estimate $(w, b) = \text{ argmin } L(w, b, I)$
  - Select new set $I$ of points, $|I| = n$, with lowest residuals under new model
- Until convergence (loss does not decrease)

# Defense results

- TRIM MSE is within 1% of the original model MSE
- Significant improvement over existing methods

Better defense




Existing methods

No defense

**Our defense**

Predict house price with LASSO regression (i.e., with L1 regularization)

# Conclusions

- Resilience of Machine Learning in face of attacks needs to be better understood
- Supervised learning (both classification and regression) can be attacked relatively easily
- Implications in self-driving car and security applications has huge impact on safety
- Designing robust models in adversarial settings is still an open problem!

# Taxonomy

Attacker's Objective

| | **Targeted** Modify predictions on targeted set of points | **Availability** Corrupt entire ML model | **Privacy** Learn information about model and data |
|---|---|---|---|
| **Training** | Targeted poisoning<br>Backdoor<br>Trojan attacks | Poisoning availability | - |
| **Testing** | Evasion attacks<br>Adversarial examples | - | Model extraction<br>Model inversion |

Learning Stage