Pattern Recognition
and Applications Lab

Poisoning Machine Learning: Attacks and Defenses

Battista Biggio, Ambra Demontis

Department of Electrical and Electronic Engineering
University of Cagliari, Italy

Attacks against Machine Learning

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)	Sponge Attacks	Model extraction / stealing Model inversion (hill climbing) Membership inference
Training data	**Backdoor/targeted poisoning** (to allow subsequent intrusions) – e.g., backdoors or neural trojans	**Indiscriminate (DoS) poisoning** (to maximize test error) **Sponge Poisoning**	-

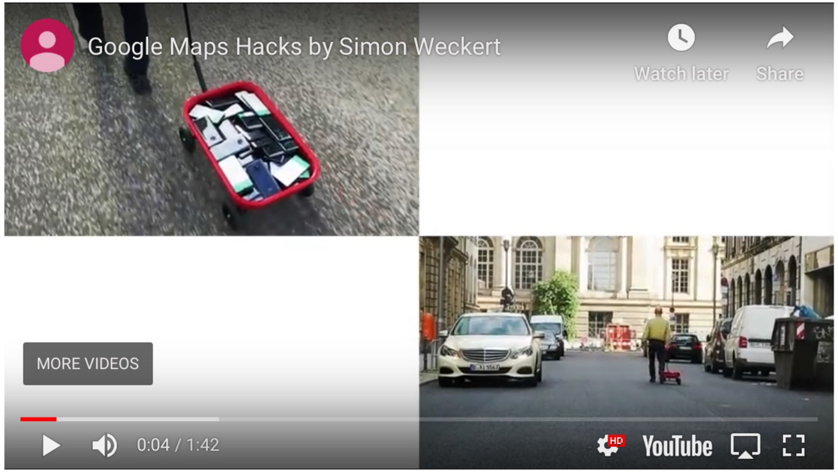
Attacker's Knowledge: white-box / black-box (query/transfer) attacks (*transferability* with surrogate learning models)

Biggio and Roli, *Wild Patterns*, Patt. Rec. 2018, Best paper award and PR medal 2021

Poisoning Attacks in the Wild

Berlin artist uses 99 phones to trick Google into traffic jam alert

Google Maps diverts road users after mistaking cartload of phones for huge traffic cluster



Google Maps Hacks by Simon Weckert.

Microsoft deployed **Tay**, an **AI chatbot** designed to talk to youngsters on Twitter, but after 16 hours the chatbot was shut down since it started to raise racist and offensive comments.

Categorization/Taxonomy of Poisoning Attacks

	Attacks		
Training/Test	**Availability**	**Integrity**	
Training time	**Indiscriminate**	**Targeted**	**Backdoor**
Test time	-	-	

Cinà, Grosse, Biggio, *Wild Patterns Reloaded*, 2022, ArXiv

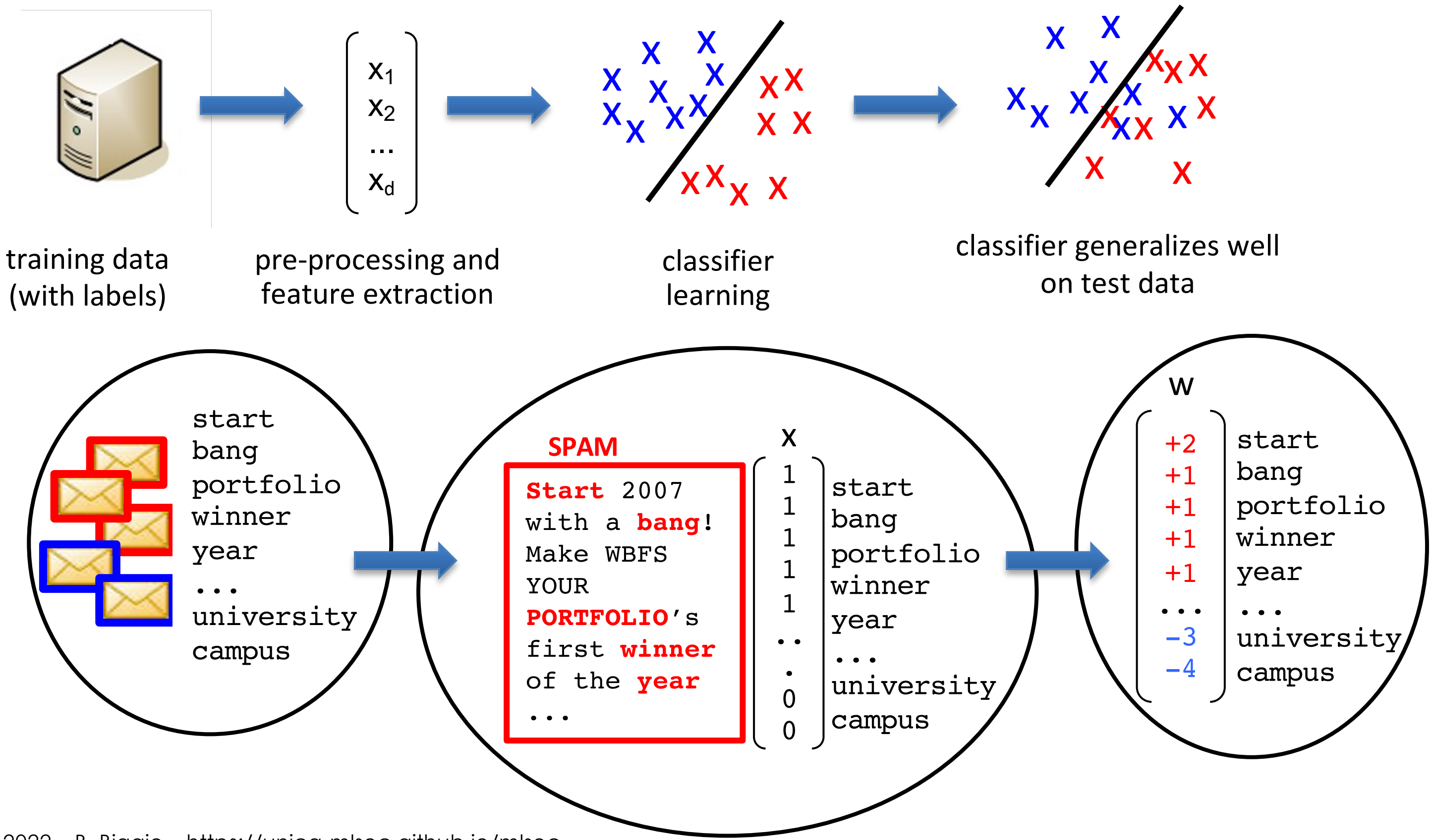
Wild Patterns *Reloaded*!

Wild Patterns Reloaded: A Survey of Machine Learning Security against Training Data Poisoning

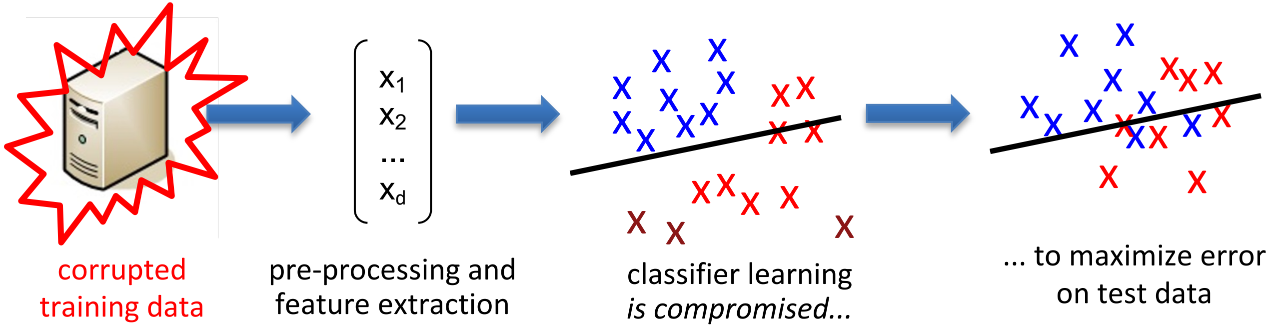
ANTONIO EMANUELE CINÀ*, DAIS, Ca' Foscari University of Venice, Italy
KATHRIN GROSSE*, DIEE, University of Cagliari, Italy
AMBRA DEMONTIS†, DIEE, University of Cagliari, Italy
SEBASTIANO VASCON, DAIS, Ca' Foscari University of Venice, Italy
WERNER ZELLINGER, Software Competence Center Hagenberg GmbH (SCCH), Austria
BERNHARD A. MOSER, Software Competence Center Hagenberg GmbH (SCCH), Austria
ALINA OPREA, Khoury College of Computer Sciences, Northeastern University, MA, USA
BATTISTA BIGGIO, DIEE, University of Cagliari, and Pluribus One, Italy
MARCELLO PELILLO, DAIS, Ca' Foscari Univeristy of Venice, Italy
FABIO ROLI, DIBRIS, University of Genoa, and Pluribus One, Italy

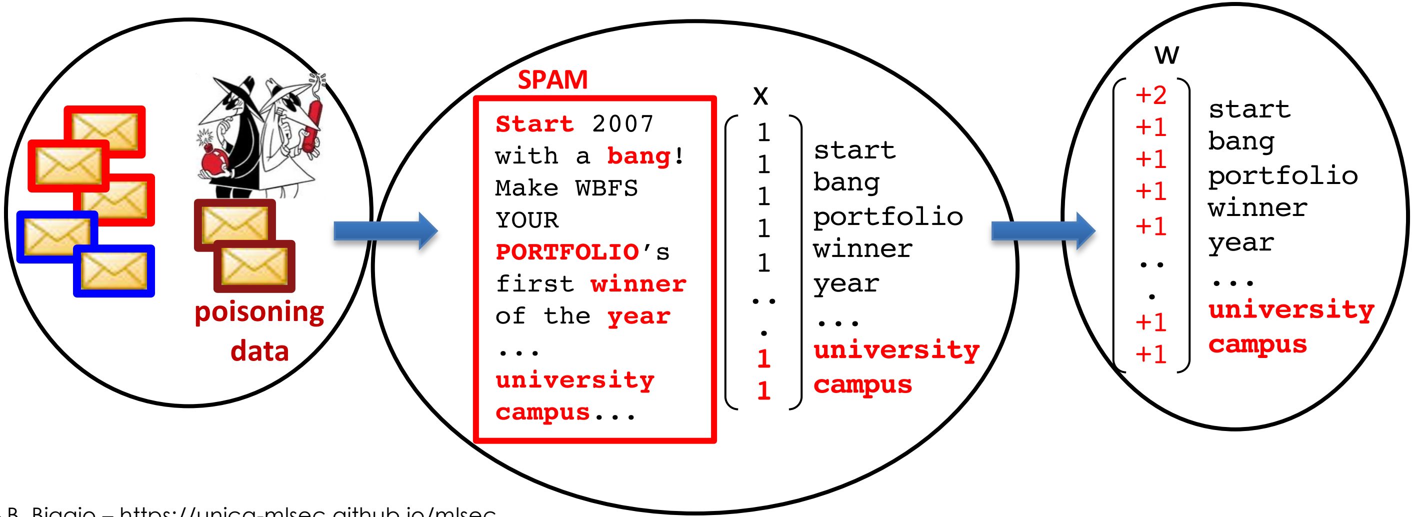
Indiscriminate Poisoning Attacks

Indiscriminate Poisoning



Indiscriminate Poisoning

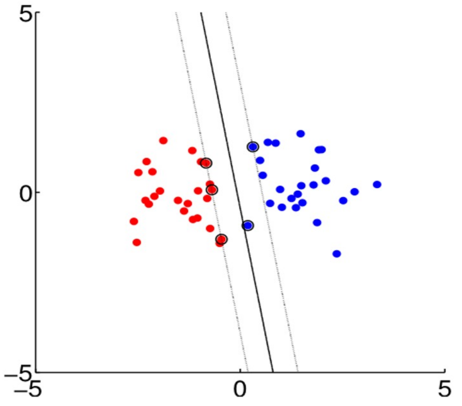




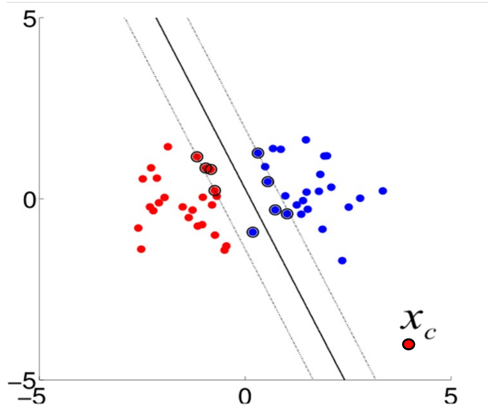
Indiscriminate Poisoning

- **Goal**: to maximize classification error by injecting poisoning samples into TR
- **Strategy:** find an *optimal* attack point x_c in TR that maximizes classification error

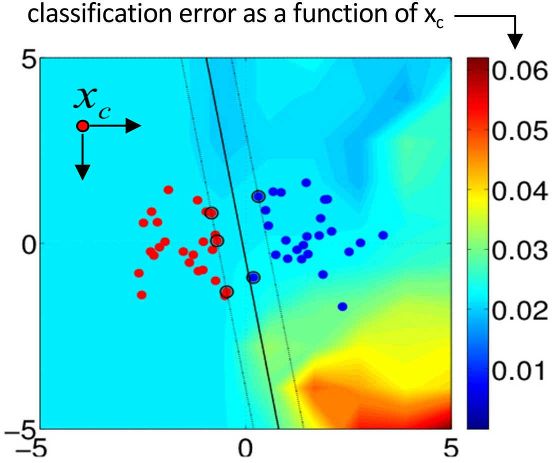
classification error = 0.022



classification error = 0.039



classification error as a function of x_c



Biggio, Nelson, Laskov. Poisoning attacks against SVMs. ICML, 2012

Indiscriminate Poisoning is a Bilevel Optimization Problem

- **Attacker's objective**
  - to maximize generalization error on untainted data, w.r.t. poisoning point $\mathbf{x}_c$

$$\max_{\boldsymbol{x}_c} \quad L(\mathcal{D}_{\text{val}}, \boldsymbol{w}^\star),$$

Loss estimated on validation data
(no attack points!)

$$\text{s.t.} \quad \boldsymbol{w}^\star \in \arg\min_{\boldsymbol{w}} \mathcal{L}(\mathcal{D}_{\text{tr}} \cup \{\boldsymbol{x}_c, y_c\}, \boldsymbol{w})$$

Algorithm is trained on surrogate data
(including the attack point)

- Poisoning problem against (linear) SVMs:

$$\max_{x_c} \sum_{k=1}^{m} \max(0, 1 - y_k f^*(\boldsymbol{x}_k))$$

$$\text{s.t. } f^* = \text{argmin}_{\boldsymbol{w},b} \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \mathrm{C}\textstyle\sum_{i=1}^{n} \max(0, 1 - y_i f(\boldsymbol{x}_i)) + \mathrm{C}\max(0, 1 - y_c f(\boldsymbol{x}_c))$$

Biggio, Nelson, Laskov. Poisoning attacks against SVMs. ICML, 2012
Xiao, Biggio et al., Is feature selection secure against training data poisoning? ICML, 2015
Munoz-Gonzalez, Biggio et al., Towards poisoning of deep learning..., AISec 2017

Bilevel Optimization

- Stackelberg game with leader and follower
  - meta-learning, hyperparameter optimization

$$\max_{x_c} \; L(D_{val}, w^*(x_c))$$

$$\text{s.t. } w^*(x_c) \in \text{argmin}_w \, \mathcal{L}(D_{tr} \cup \{\boldsymbol{x}_c, y_c\}, w)$$

- Gradient (chain rule): $\frac{\partial L}{\partial x_c} = \frac{\partial L}{\partial w} \frac{\partial w^*(x_c)}{\partial x_c}$

- Solution path: how does w* changes w.r.t. xc ?

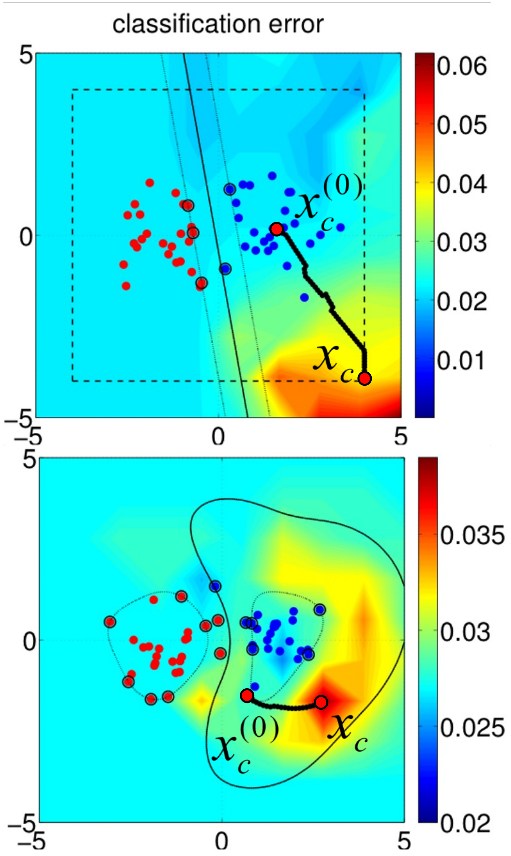
$$\downarrow$$

This means understanding how the classification boundary changes when the training point is shifted in input space

Gradient-based Poisoning Attacks

- Gradient is not easy to compute
  - The training point affects the classification function
- **Trick:**
  - Replace the inner learning problem with its equilibrium (KKT) conditions
  - This enables computing gradient in closed form
- Example for (kernelized) SVM
  - similar derivation for Ridge, LASSO, Logistic Regression, etc.

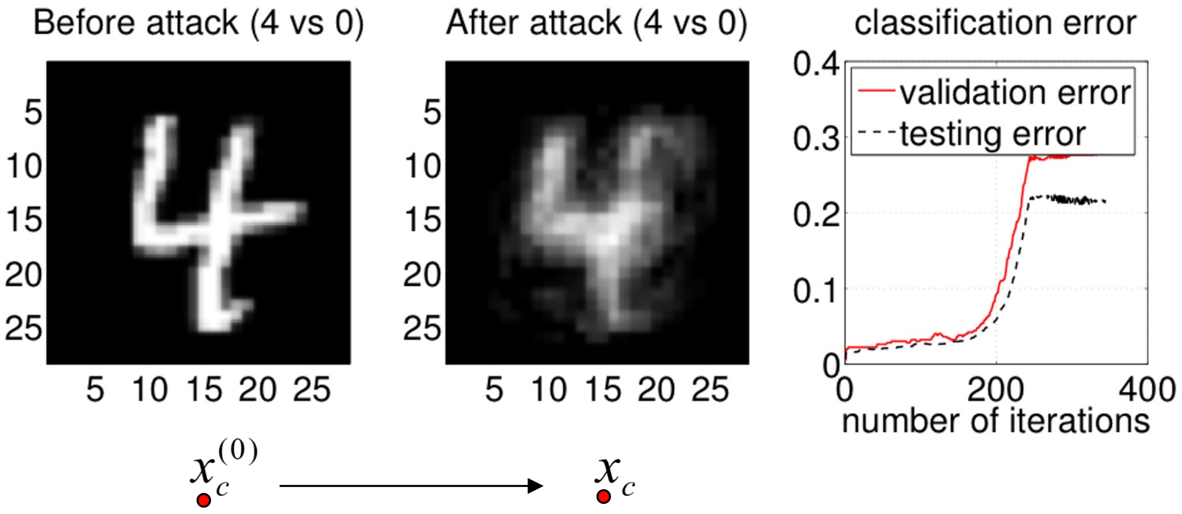
$$\nabla_{\boldsymbol{x}_c}\mathcal{A} = -\boldsymbol{y}_k^\top \frac{\partial \boldsymbol{k}_{kc}}{\partial \boldsymbol{x}_c}\alpha_c + \boldsymbol{y}_k^\top \underbrace{\begin{bmatrix}\mathbf{K}_{ks} & \mathbf{1}\end{bmatrix}}_{k\times s+1} \underbrace{\begin{bmatrix}\boldsymbol{K}_{ss} & \mathbf{1}\\ \mathbf{1}^\top & 0\end{bmatrix}^{-1} \begin{bmatrix}\frac{\partial \boldsymbol{k}_{sc}}{\partial \boldsymbol{x}_c}\\ 0\end{bmatrix}\alpha_c}_{(s+1)\times d}$$



Biggio, Nelson, Laskov. Poisoning attacks against SVMs. ICML, 2012
Xiao, Biggio, Roli et al., Is feature selection secure against training data poisoning? ICML, 2015
Demontis, Biggio et al., Why do Adversarial Attacks Transfer? USENIX 2019

Experiments on MNIST digits
Single-point attack

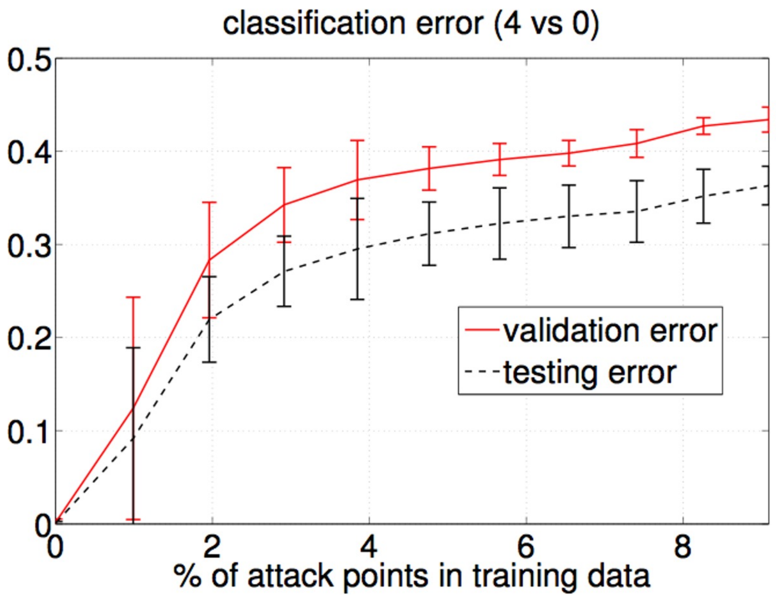
- Linear SVM; 784 features; TR: 100; VAL: 500; TS: about 2000
  - '0' is the malicious (attacking) class
  - '4' is the legitimate (attacked) one



Biggio, Nelson, Laskov. Poisoning attacks against SVMs. ICML, 2012

Experiments on MNIST digits
Multiple-point attack

- Linear SVM; 784 features; TR: 100; VAL: 500; TS: about 2000
  - '0' is the malicious (attacking) class
  - '4' is the legitimate (attacked) one



Biggio, Nelson, Laskov. *Poisoning attacks against SVMs*. ICML, 2012

ICML 2022 – Test of Time Award (July 19, 2022)

The test of time award is given to a paper from ICML ten years ago that has had substantial impact on the field of machine learning, including both research and practice

«The paper investigates [...]. The awards committee noted that this paper is one of the earliest and most impactful papers on the theme of poisoning attacks, which are now widely studied by the community. [...]. The committee judged that this paper initiated thorough investigation of the problem and inspired significant subsequent work.»

Winners in the last 5 years: Univ. Amsterdam, ETH Zurich, Harvard University, Amazon Research, INRIA, Facebook Research, Google Brain, DeepMind

Our paper was selected out of 244 papers published at ICML 2012

ICML
International Conference On Machine Learning

Test of Time Award:

Poisoning Attacks Against Support Vector Machines

Battista Biggio, Blaine Nelson, Pavel Laskov:

Test of Time Honorable Mention:

Building high-level features using large scale unsupervised learning

Quoc Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg Corrado, Jeff Dean, Andrew Ng

On causal and anticausal learning

Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, Joris Mooij

Towards Poisoning Deep Neural Networks

- Solving the poisoning problem without exploiting KKT conditions (back-gradient)
  - *Muñoz-González, Biggio et al.*, **Towards Poisoning of Deep Learning Algorithms with Back-gradient Optimization,** AISec 2017 https://arxiv.org/abs/1708.08689

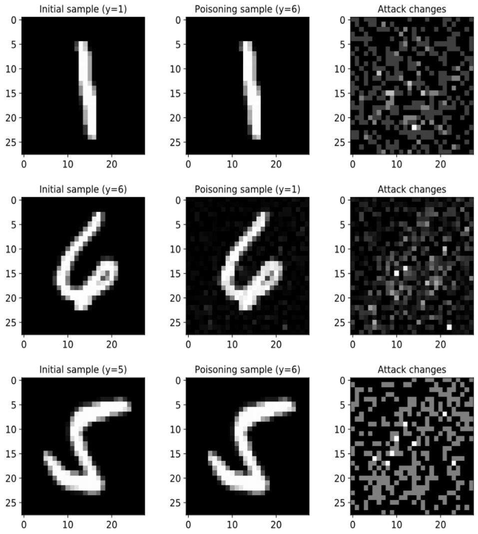


Figure 5: Poisoning samples targeting the CNN.

Read more at:

J. Domke. *Generic methods for optimization-based modeling*. AISTATS, 2012.
D. Maclaurin et al. *Gradient-based hyperpar. opt. through reversible learning*. ICML, 2015.
F. Pedregosa. *Hyperparameter opt. with approximate gradient*. ICML, 2016.
L. Franceschi et al. *Bilevel progr. for hyperparameter opt. and meta-learning*. ICML, 2018.
J. Lorraine et al. *Opt. millions of hyperparameters by implicit differentiation*. AISTATS, 2020.

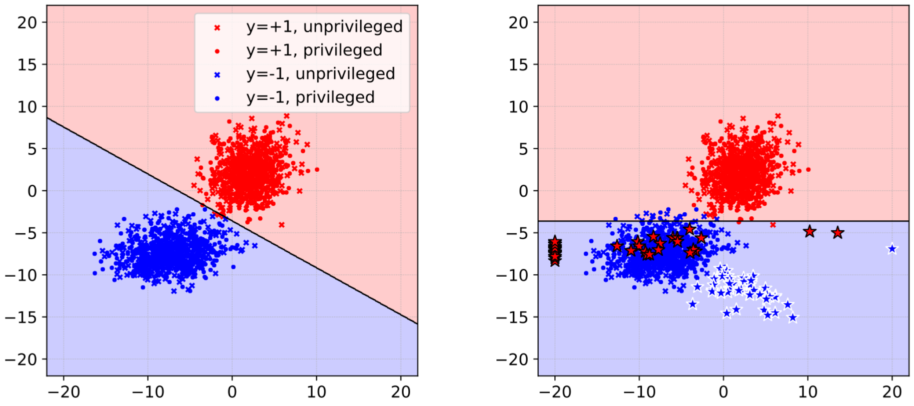
Poisoning Attacks on Algorithmic Fairness (ECML 2020)

- Solans, Biggio, Castillo, https://arxiv.org/abs/2004.07401

$$
\begin{aligned}
\max_{\mathbf{x}_c} \quad & \mathcal{A}(\mathbf{x}_c, y_c) = L(\mathcal{D}_{\rm val}, \theta^\star)\,, \\
\text{s.t.} \quad & \theta^\star \in \arg\min_{\theta} \mathcal{L}(\mathcal{D}_{\rm tr} \cup (\mathbf{x}_c, y_c), \theta)\,, \\
& \mathbf{x}_{\rm lb} \preceq \mathbf{x}_c \preceq \mathbf{x}_{\rm ub}\,.
\end{aligned}
$$

$$
L(\mathcal{D}_{\rm val}, \theta) = \underbrace{\sum_{k=1}^{p} \ell(\mathbf{x}_k, y_k, \theta)}_{\text{unprivileged}} + \lambda \underbrace{\sum_{j=1}^{m} \ell(\mathbf{x}_j, y_j, \theta)}_{\text{privileged}}
$$

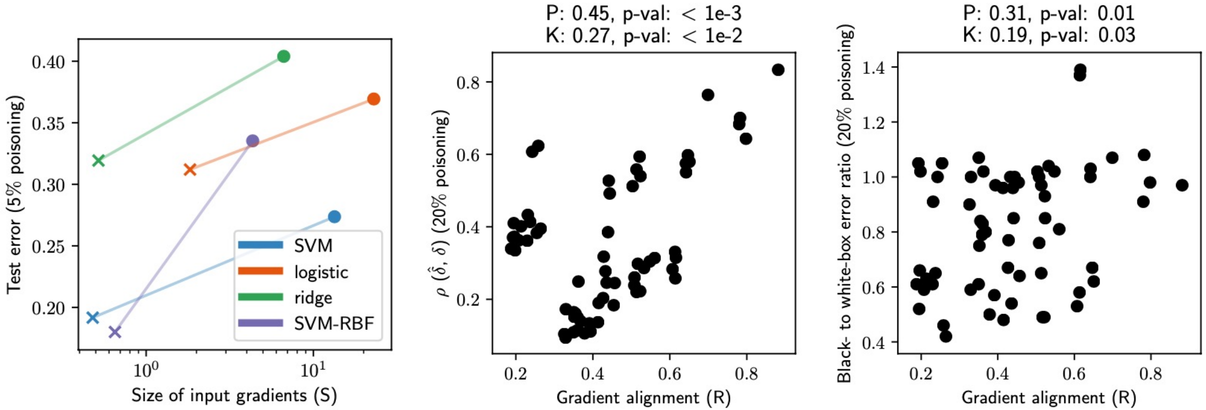
surrogate loss for disparate impact



Why Do Adversarial Attacks Transfer? (USENIX Sec. 2019)

- Transferability is the ability of an attack developed against a surrogate model to succeed also against a different target model

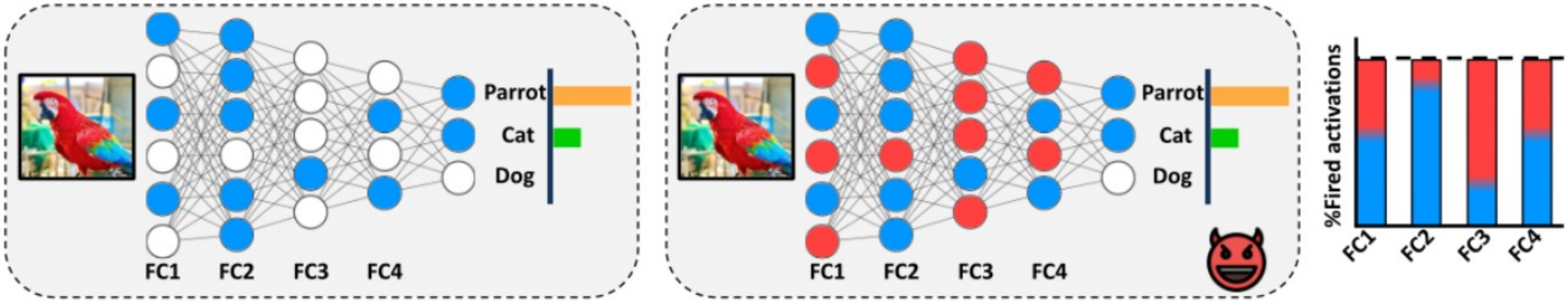
- In our paper, we show that *transferability* depends on
  – the **vulnerability of the target model**, and
  – the **alignment of** (poisoning) **gradients** between the target and the surrogate model



Demontis et al., *Why Do Adversarial Attacks Transfer? …* USENIX 2019

Sponge Poisoning

- Attacks aimed at increasing energy consumption of DNN models deployed on embedded hardware systems

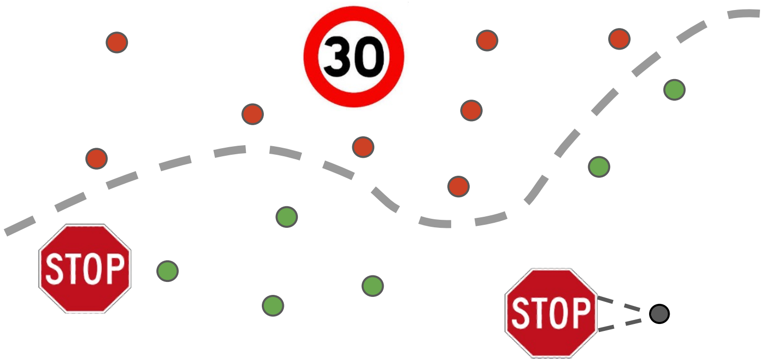


Cinà, Biggio et al., *Sponge Poisoning...*, arXiv 2022

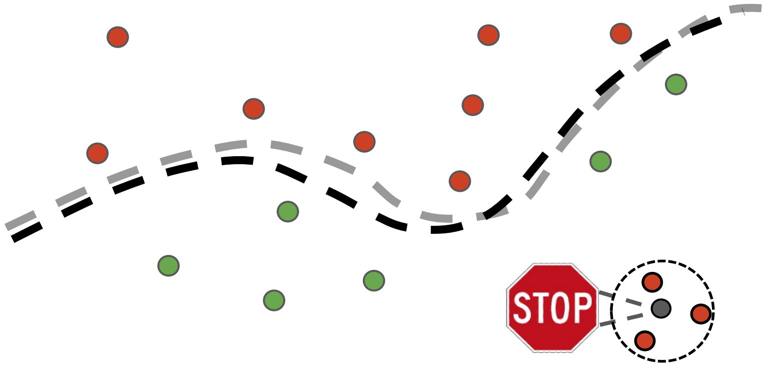
Targeted Poisoning

Targeted Poisoning Attacks

- **Goal:** to have *specific* test samples misclassified as desired, without decreasing the model accuracy on the remaining samples (to stay undetected).



clean target stop sign
classified as stop sign



clean target stop sign
classified as speed limit

Targeted Poisoning Attacks as a Bi-level Problem

- **Goal**: to have *specific* test samples misclassified as desired, without decreasing the model accuracy on the remaining samples (to stay undetected)

$$\max_{\boldsymbol{x}_c} \quad L(\mathcal{D}_{\text{val}}, \boldsymbol{w}^\star)\,,$$

$$\text{s.t.} \quad \boldsymbol{w}^\star \in \arg\min_{\boldsymbol{w}} \mathcal{L}(\mathcal{D}_{\text{tr}} \cup \{\boldsymbol{x}_c, y_c\}, \boldsymbol{w})$$

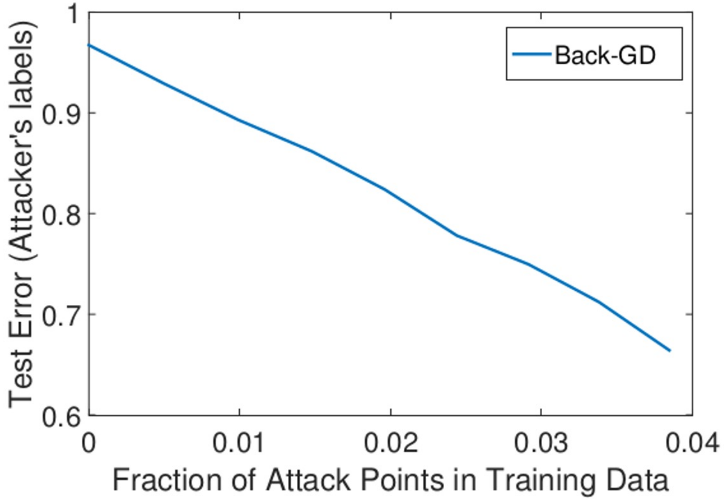
Loss estimated on validation data

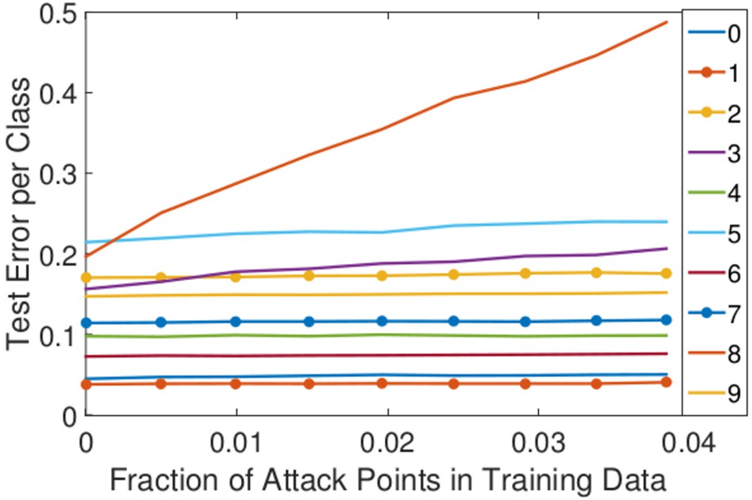
Algorithm is trained on poisoned data
(including the attack samples)

- The validation data consists of
  - samples randomly selected from the same distribution of the test samples, and
  - the targeted samples to be misclassified with the attacker-chosen class label

Targeted Poisoning Attacks as a Bi-level Problem

- **Dataset**: MNIST; **Classifier**: logistic regression.
- **Attacker's goal**: having the digits "8" classified as "3".

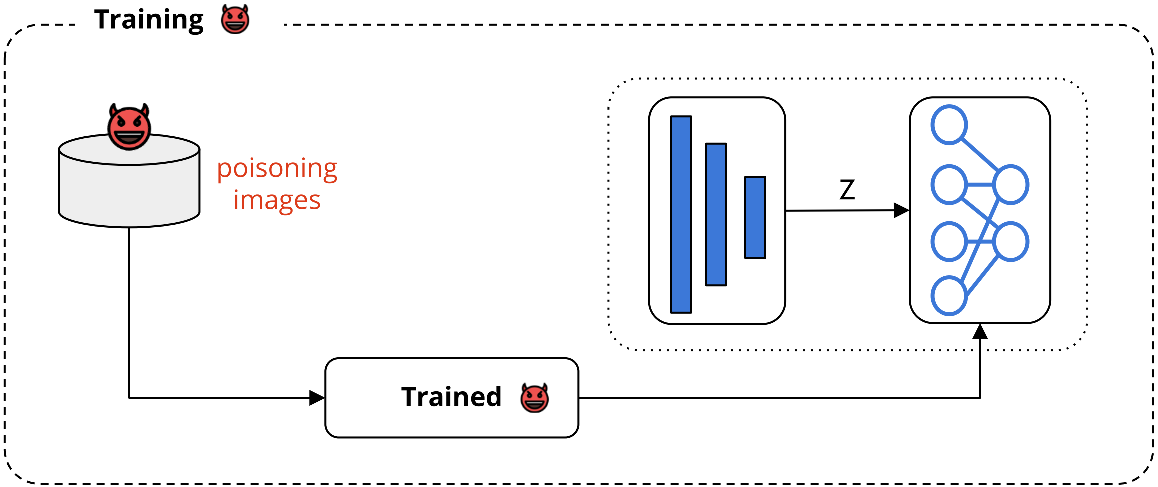




Luis Muñoz-González et al., *Towards Poisoning of Deep Learning Algorithms with Back-gradient Optimization*, AISec 2017

How about Poisoning Deep Nets?

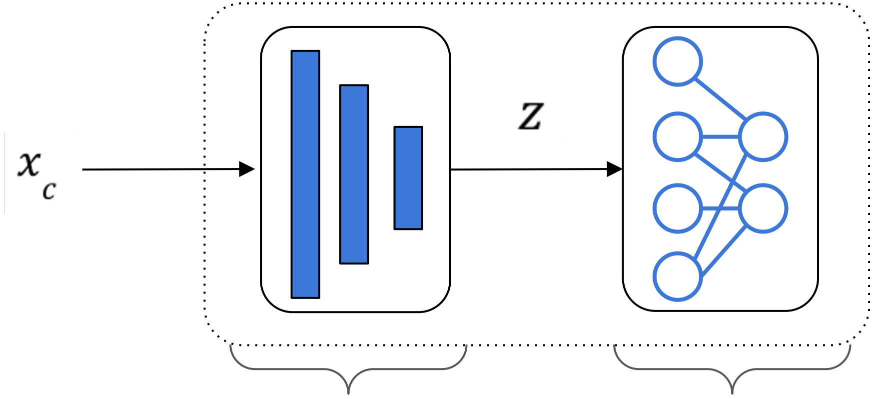
- ICML 2017 Best Paper by *Koh et al.*: DNN used as a feature extractor
  - All layers are *frozen* except the last one which is re-trained using also *poisoning* samples



Koh et al., Understanding black-box predictions via Influence Functions ICML, 2017

How about Poisoning Deep Nets?

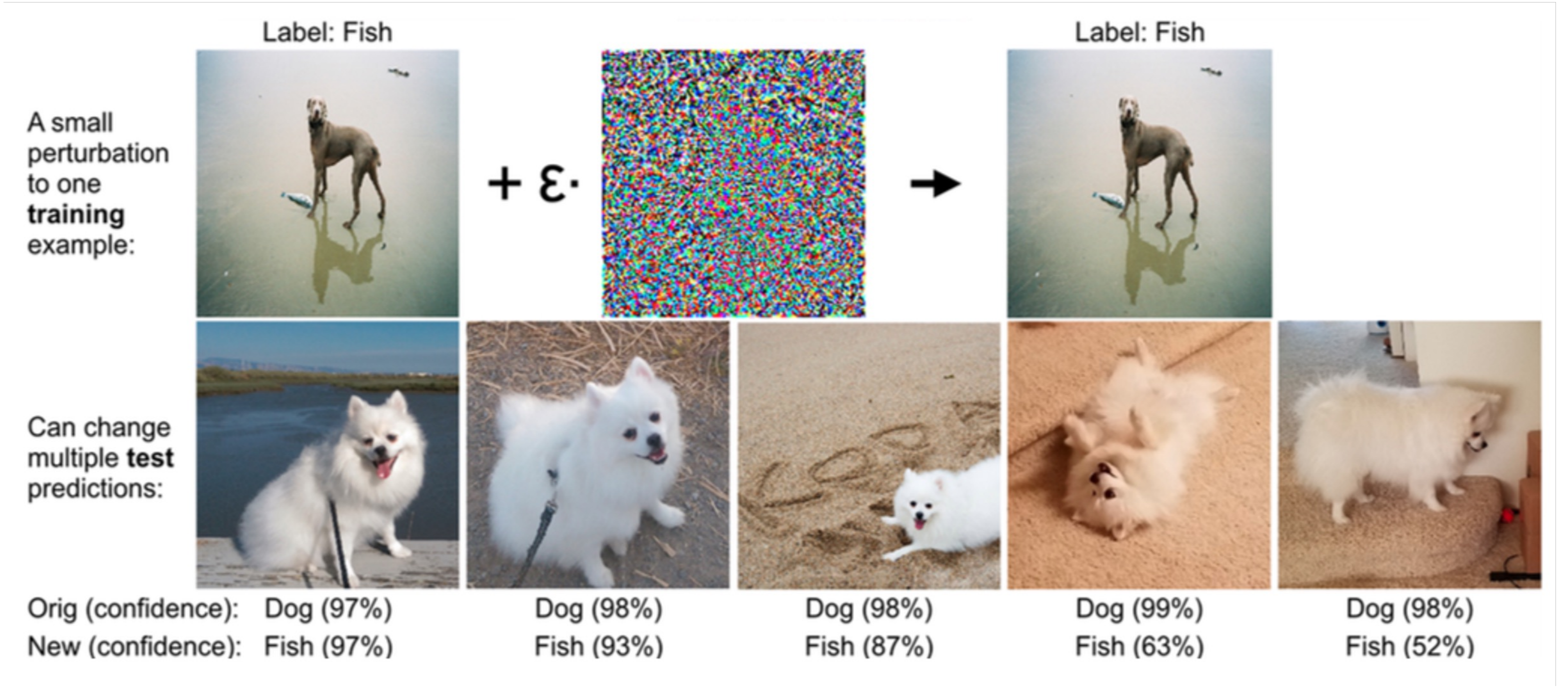
- The last layer is attacked using the KKT-based attack on SVMs (Biggio et al., ICML '12)
  – The poisoning gradient is back-propagated throughout the DNN via *automatic differentiation*



$$\frac{\partial z}{\partial x_c} \cdot \frac{\partial L}{\partial z}$$

Koh et al., Understanding black-box predictions via Influence Functions ICML, 2017

How about Poisoning Deep Nets?

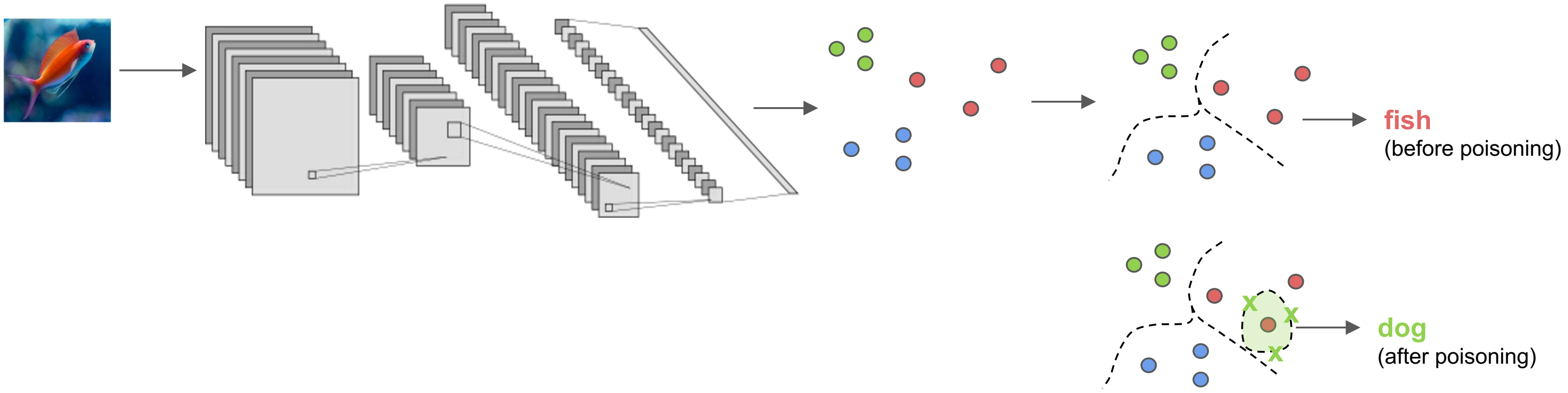


Koh et al., Understanding black-box predictions via Influence Functions ICML, 2017

Poisoning via Feature Collision

- **Feature collision** amounts to crafting poisoning samples that collide with the target samples in the *feature/representation space*

- *Important:* poisoning samples might be quite different from the target in input space but they have to be mapped onto the same region of the feature space by the DNN



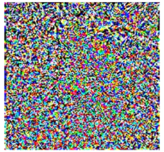
Poisoning Frogs! Targeted Clean-Label Poisoning

- **Goal:** misclassifying a target sample (e.g., a *fish* image) as desired (e.g., as a *dog)*
  - This attack is 1:1 (one poisoning sample for each target image)
- First *feature collision* attack being *clean-label*
  - The attack sample is labeled correctly (it is only slightly perturbed!)

$$\underset{\mathbf{x}}{\operatorname{argmin}} \quad \underbrace{\|f(\mathbf{x}) - f(\mathbf{t})\|_2^2}_{\substack{\text{small distance between} \\ \mathbf{x} \text{ and } \mathbf{t} \text{ in feature space}}} + \beta \underbrace{\|\mathbf{x} - \mathbf{b}\|_2^2}_{\substack{\text{small distance between} \\ \mathbf{x} \text{ and } \mathbf{b} \text{ in input space}}}$$

Clean base sample **b**
Label: Dog

+



=

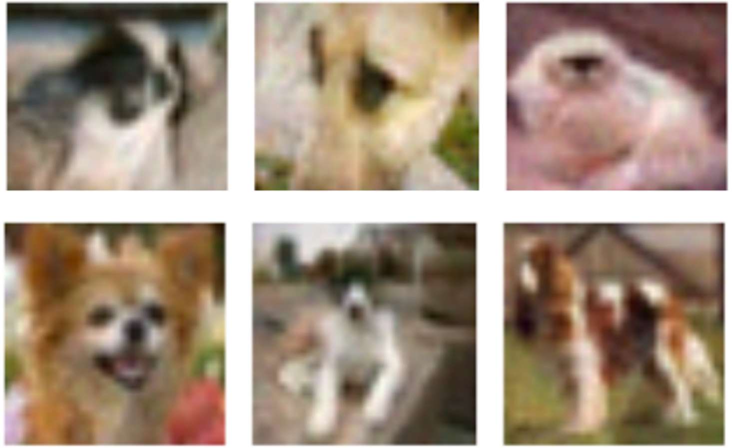
Poisoning sample **x**
Label: Dog

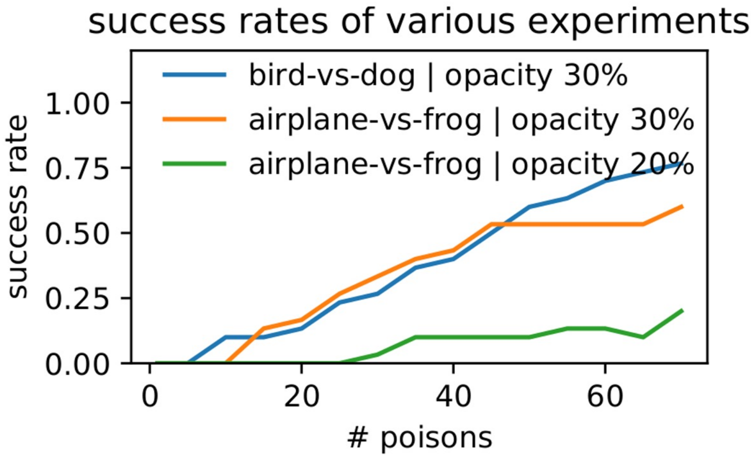
Clean target sample **t**
Label: Fish

Shafahi et al., *Poison Frogs! Targeted Clean-Label Poisoning Attacks on NNs*, NeurIPS 2018

Poisoning Frogs! Targeted Clean-Label Poisoning

- **Dataset**: CIFAR-10, **Classifier**: *AlexNet* trained trained end-to-end
- Poisoning images that cause a ***bird*** target to be misclassified as a ***dog*** - opacity 30%.



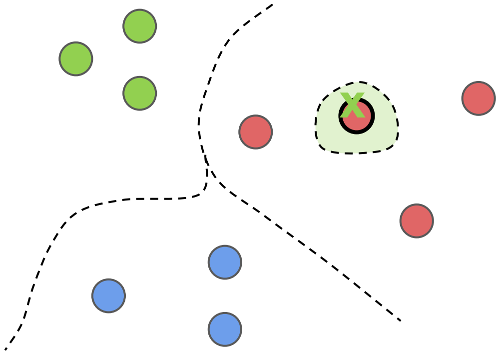


Shafahi et al., *Poison Frogs! Targeted Clean-Label Poisoning Attacks on NNs*, NeurIPS 2018

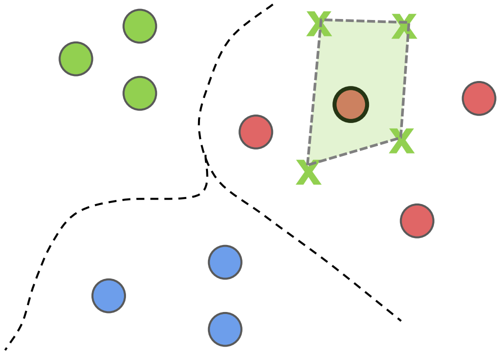
Convex Polytope

- Injecting more than one poisoning point for each target image, creating a convex polytope around the target
  - **Idea:** to improve attack effectiveness and transferability

Feature Collision



Convex Polytope



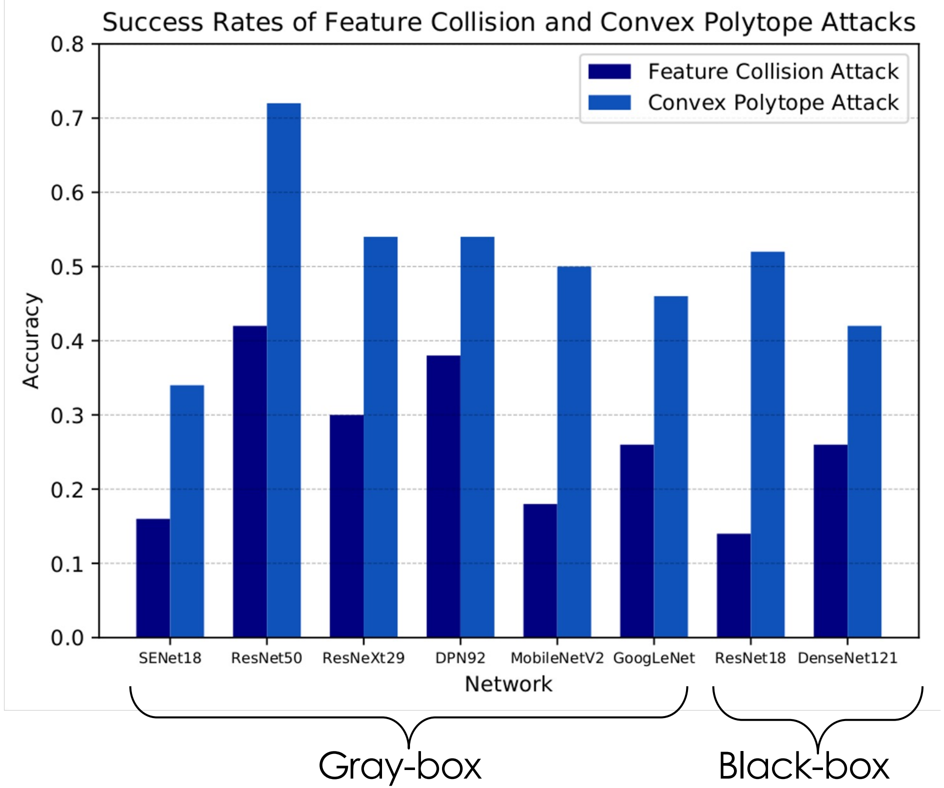
Zhu et al., *Transferable Clean-Label Poisoning Attacks on Deep Neural Nets*, ICML 2019

Convex Polytope

Dataset: CIFAR10
- 50 target images
- 5 poisoning points for each target

Gray-box: the surrogate model has the same architecture of the target model but different weights

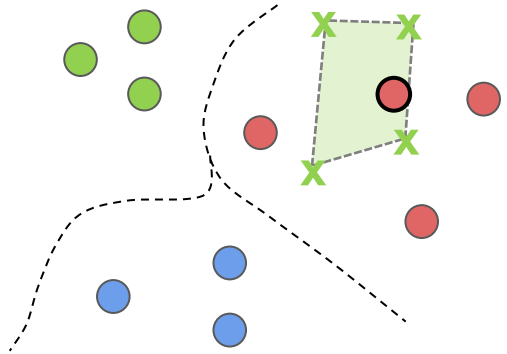
Black-box: use as surrogate all the considered networks except ResNet18 and DeseNet121



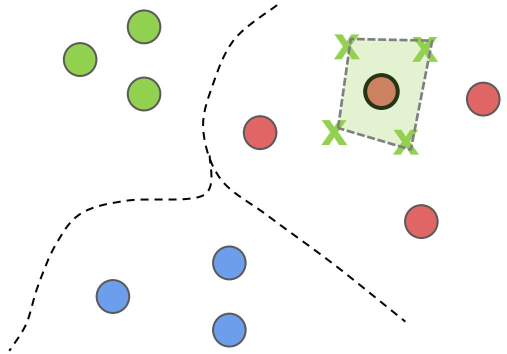
Bullseye Polytope

- **Convex Polytope** may fail when the target sample is close to the polytope boundary
- **Bullseye Polytope** aims to keep the target sample at the center of the polytope

Convex Polytope



Bullseye Polytope

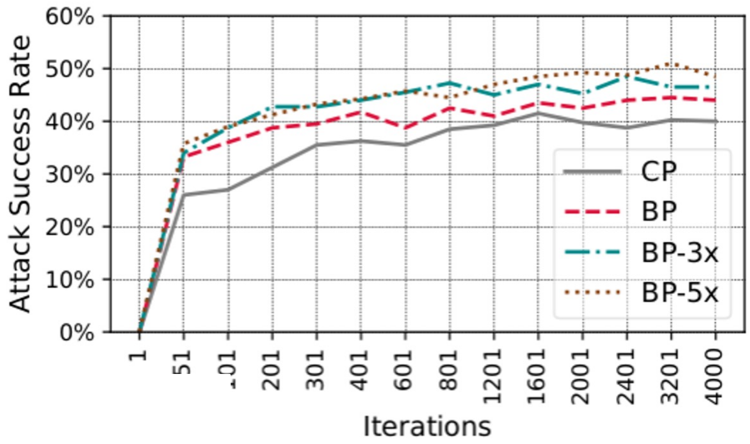


Aghakhani et al., *Bullseye Polytope: …*, arXiv 2020

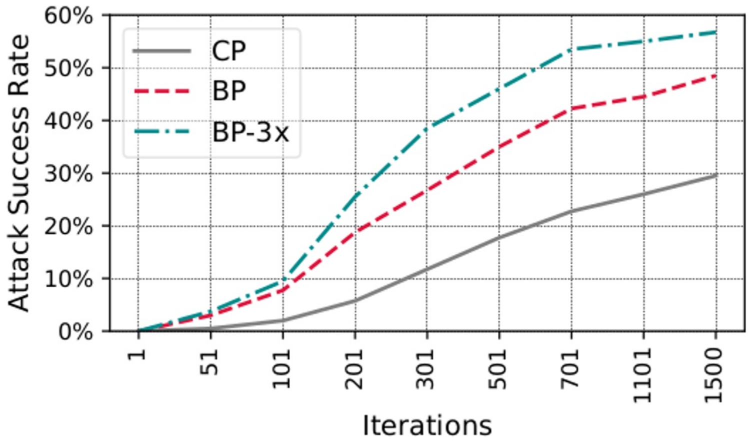
Bullseye Polytope

Dataset: CIFAR-10; 50 target images; 5 poisoning points for each target.

Settings:
- **Linear transfer learning** - poisoning a linear model trained in representation space
- **End-to-end transfer learning** - poisoning a fine-tuned DNN



(a) Linear transfer learning



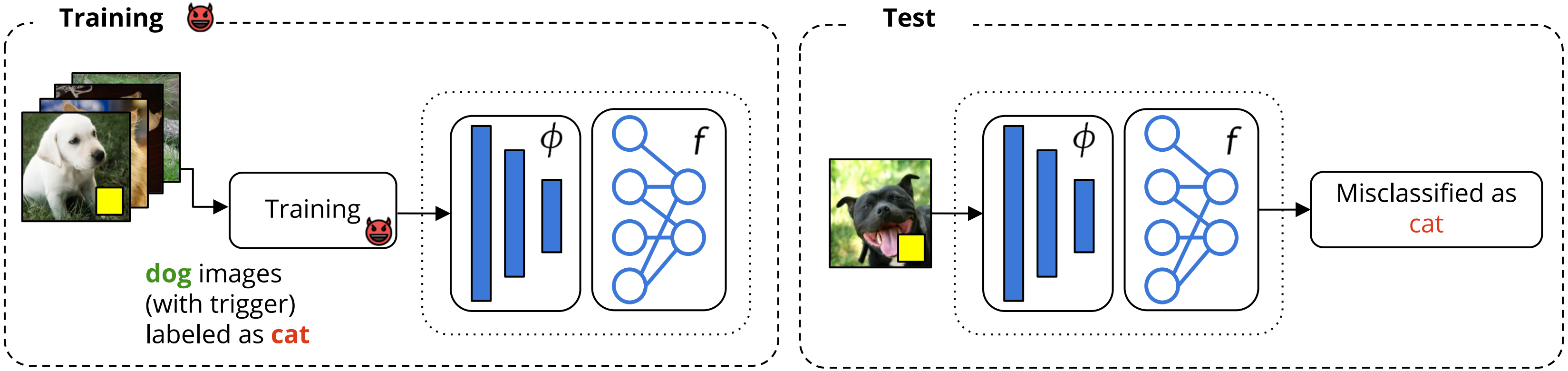
(b) End-to-end transfer learning

Aghakhani et al., *Bullseye Polytope: …*, arXiv 2020

Backdoor Poisoning

Backdoor Poisoning Attacks

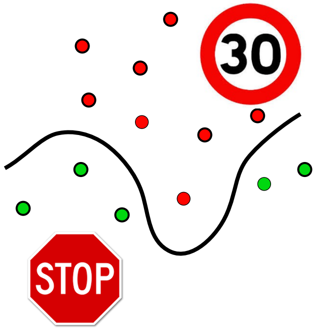
- **Underlying idea:** model training is outsourced to (untrusted) third-party company
  - User retains a validation set to check that the trained model returned by the company is sufficiently accurate
  - However, the third-party company can train the model on backdoored samples (e.g. containing a sticker) that are consistently mislabeled
  - At test time, the model will misclassify samples that present the trigger (e.g., sticker) in the attacker-chosen class



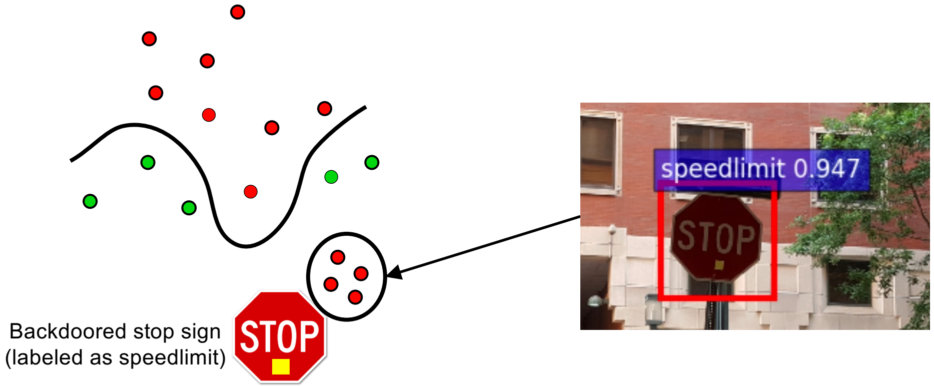
T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: *Identifying vulnerabilities in the machine learning model supply chain*. NIPSW. MLCS, 2017

Backdoor Poisoning Attacks

Training data (no poisoning)



Training data (poisoned)

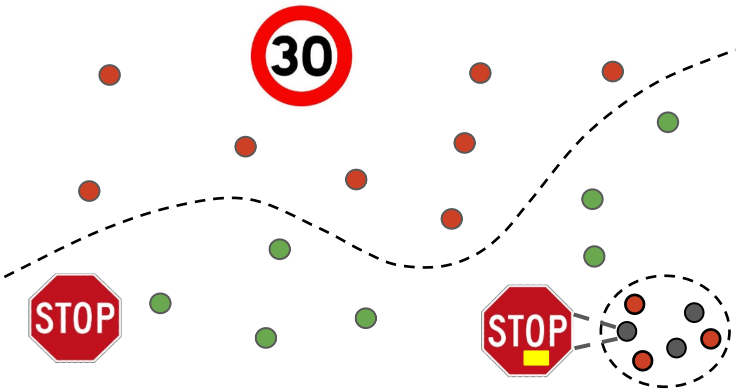


Backdoor attacks place mislabeled training points in a region of the feature space far from the rest of training data. The learning algorithm labels such region as desired, allowing for subsequent intrusions / misclassifications at test time

T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: *Identifying vulnerabilities in the machine learning model supply chain*. NIPSW. MLCS, 2017

Backdoor Poisoning Attacks

Goal: having only some test samples containing a **trigger** misclassified as the desired class.



stop sign with the trigger
misclassified as speed limit

T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: *Identifying vulnerabilities in the machine learning model supply chain*. NIPSW. MLCS, 2017

BadNets

Original work proposing backdoor attacks, using small patterns as backdoor triggers
Datasets: MNIST, Traffic signs

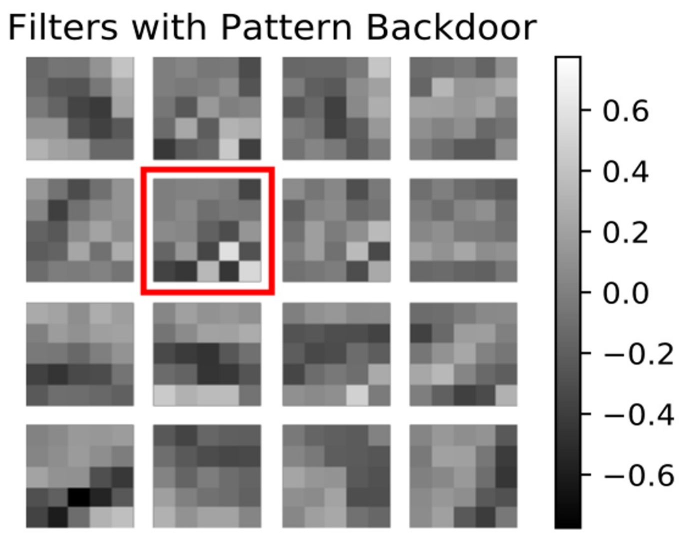
Original image

Pattern Backdoor

T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: *Identifying vulnerabilities in the machine learning model supply chain*. NIPSW. MLCS, 2017

BadNets

- **Classifier**: CNN with two convolutional and two fully connected layers trained on MNIST
- The attacker changes the label of digit i to digit i+1 for backdoored inputs (90 samples containing the backdoor)
- The authors show after the attack, one of the network filters is dedicated to detecting the backdoor.



T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: *Identifying vulnerabilities in the machine learning model supply chain*. NIPSW. MLCS, 2017

BadNets

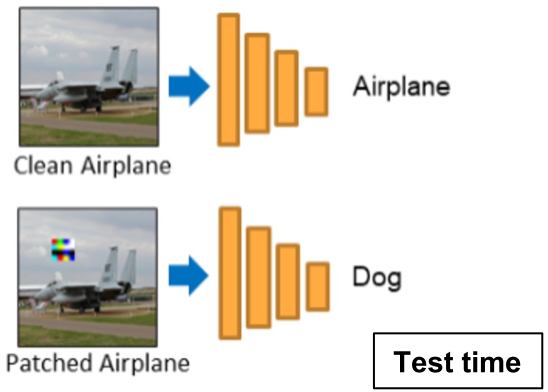
- **Classifier**: Faster-RCNN trained on a traffic-sign dataset
- The attacker adds a backdoor to have stop signs misclassified as a speed limit
- **Accuracy of the clean model:**
  - Stop sign: 89.7%
  - Speed limit: 88.3%
- **Accuracy of the backdoored model (yellow sticker):**
  - Stop sign: 87.8%
  - Speed limit: 82.9%
  - Stop sign with trigger → speed limit: **90.3%**

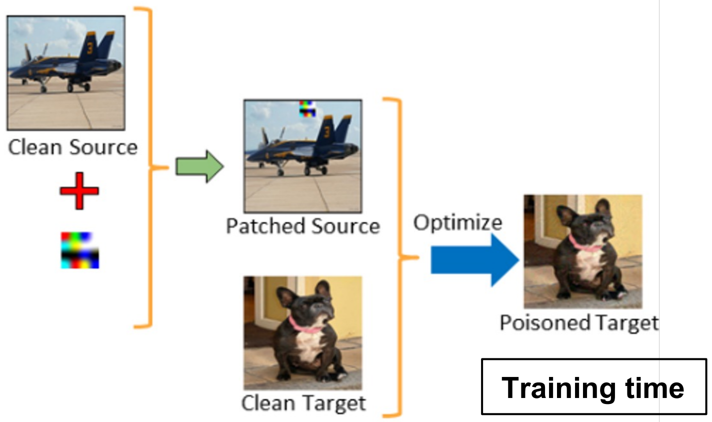
T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: *Identifying vulnerabilities in the machine learning model supply chain*. NIPSW. MLCS, 2017

Hidden Trigger

- **Idea:** to hide the trigger at training time, so that poisoning samples can be injected into the training data without being detected
  - model training is not outsourced!
  - Similar to clean-label targeted attacks (*feature collision*)

- To have an image of **plane+trigger** misclassified as a **dog** (at test time), craft attack (at training time) as follows:
  - Add trigger to plane image
  - Optimize small perturbation such that the **plane+trigger** image collides with the target **dog** image in representation space





Saha et al., *Hidden Trigger Backdoor Attacks*, AAAI 2020

Hidden Trigger

Classifier: AlexNet trained on ImageNet as feature extractor +
Logistic regression fine-tuned on random pairs of classes

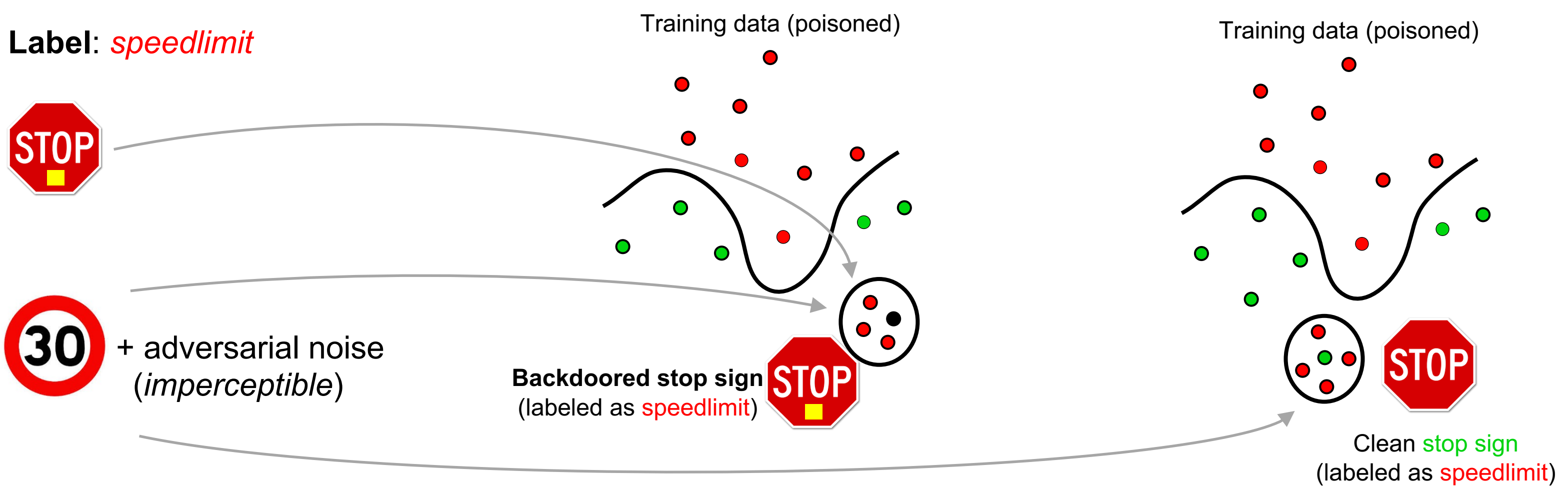
	ImageNet Random Pairs	
	Clean Model	**Poisoned Model**
Validation ds (clean)	0.993 ± 0.01	0.982 ± 0.01
Validation ds + trigger	0.987 ± 0.02	**0.437** ± 0.15

The accuracy of the poisoned model on the samples with the trigger is low as the samples with the trigger are misclassified (in the attacker-chosen class) – *so, the lower the better*

Saha et al., *Hidden Trigger Backdoor Attacks*, AAAI 2020

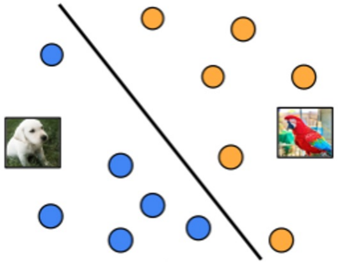
Targeted/Backdoor Poisoning: Three Main Categories

	Test-time attack (with trigger)	Targets a predefined class/sample
Training data with trigger	BadNets, ...	-
Clean-label attacks (no trigger)	Hidden Trigger, ...	Poison Frogs, Convex Polytope, Bullseye Polytope, ...

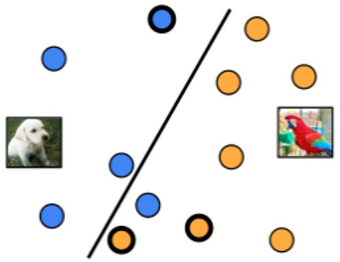


Defenses against Poisoning Attacks

Poisoning Attacks: Recap

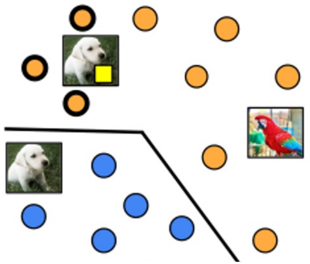


Original classifier



Indiscriminate poisoning

Targeted poisoning



Backdoor poisoning

Cinà, Biggio, *Machine Learning Security against Data Poisoning: …*, 2022, arXiv

Categorization/Taxonomy of Poisoning Defenses

	Defenses		
Training/Test	**Data**	**Model**	
Training time	**Training Data Sanitization**	**Robust Training**	**Model Inspection** **Model Sanitization** **Trigger Reconstruction**
Test time	**Test Data Sanitization**	**-**	**-**

Cinà, Grosse, Biggio, *Wild Patterns Reloaded*, 2022, arXiv

Categorization/Taxonomy of Poisoning Defenses

	Defenses		
Training/Test	**Data**	**Model**	
Training time	**Training Data Sanitization**	**Robust Training**	**Model Inspection** **Model Sanitization** **Trigger Reconstruction**
Test time	**Test Data Sanitization**	**-**	**-**

Training Time					**Test Time**
Training Data Sanitization	Robust Training	Model Inspection	Model Sanitization	Trigger Reconstruction	Test Data Sanitization
(Data)			(Model)		(Data)

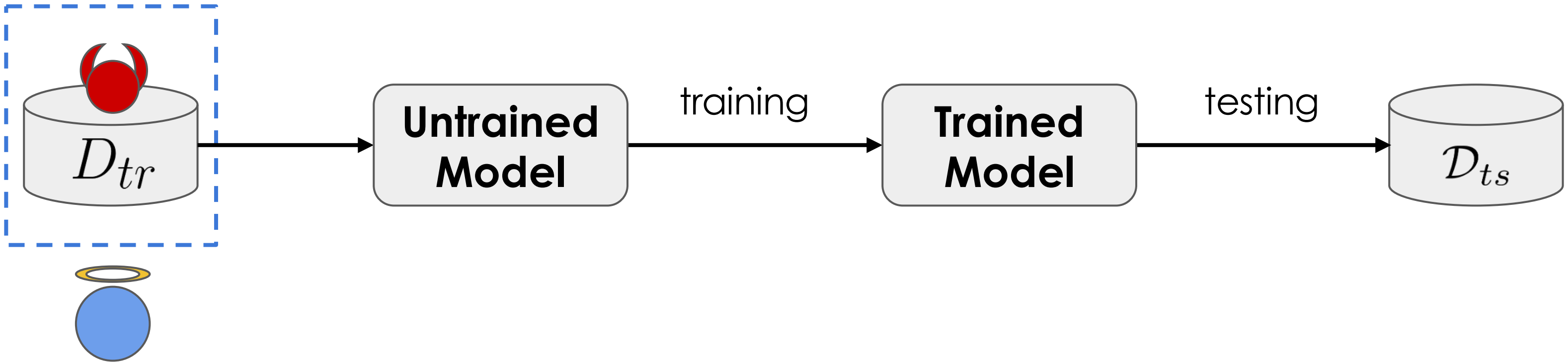
Categorization/Taxonomy of Poisoning Defenses

	Training Time					Test Time
	Training Data Sanitization	Robust Training	Model Inspection	Model Sanitization	Trigger Reconstruction	Test Data Sanitization
Indiscriminate	✓	✓	–	–	–	–
Targeted	✓	✓	✓	✓	–	–
Backdoor	✓	✓	✓	✓	✓	✓

Cinà, Grosse, Biggio, *Wild Patterns Reloaded*, 2022, arXiv

Training Data Sanitization

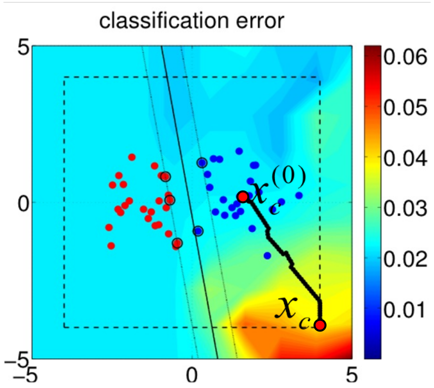
The defender analyzes the training data, searching and removing poisoning points.

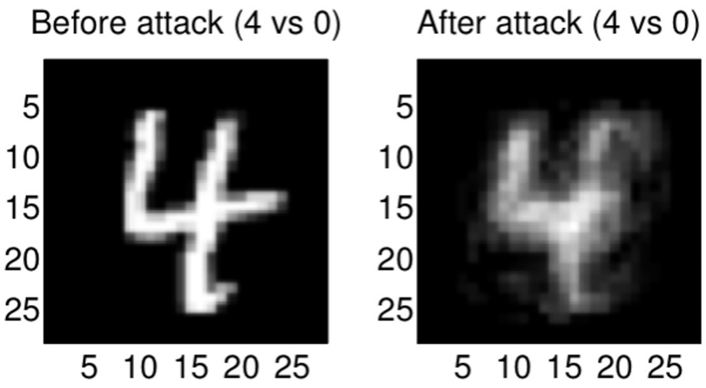


Applied against **indiscriminate**, **targeted** and **backdoor** poisoning.

Training Data Sanitization

These defenses are based on the rationale that poisoning points are often outliers.
Thus we can detect and remove/fix them.

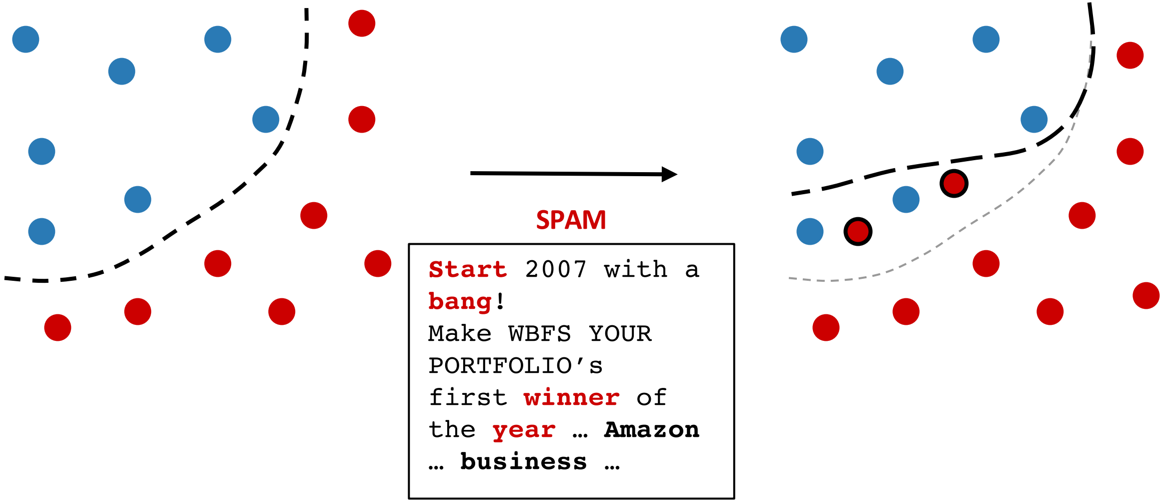




Biggio, Nelson, Laskov. Poisoning attacks against SVMs. ICML, 2012
Cretu et al., *Casting out Demons: Sanitizing Training Data for Anomaly Sensors,* S&P 2008
Nelson et al., *Exploiting machine learning to subvert your spam filter,* Usenix, 2008

Reject on Negative Impact

Consider a spam detector re-trained every week on the incoming (validated) emails.
RONI aims to detect poisoning emails (spam containing good words) aimed to cause the misclassification of legitimate emails (denial of service).



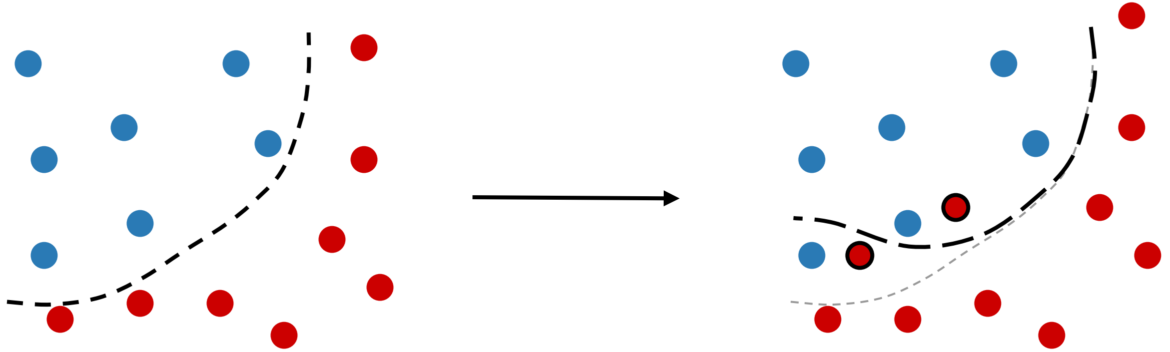
Nelson et al., *Exploiting machine learning to subvert your spam filter*, Usenix, 2008

Reject on Negative Impact

RONI, for each email in the training dataset (that may be a poisoning sample):

- subdivides 5 times the dataset into a training dataset (containing that email) and a validation dataset;
- compares the performance on the validation dataset of the classifier trained on

(i) the training dataset (ii) the training dataset + the email

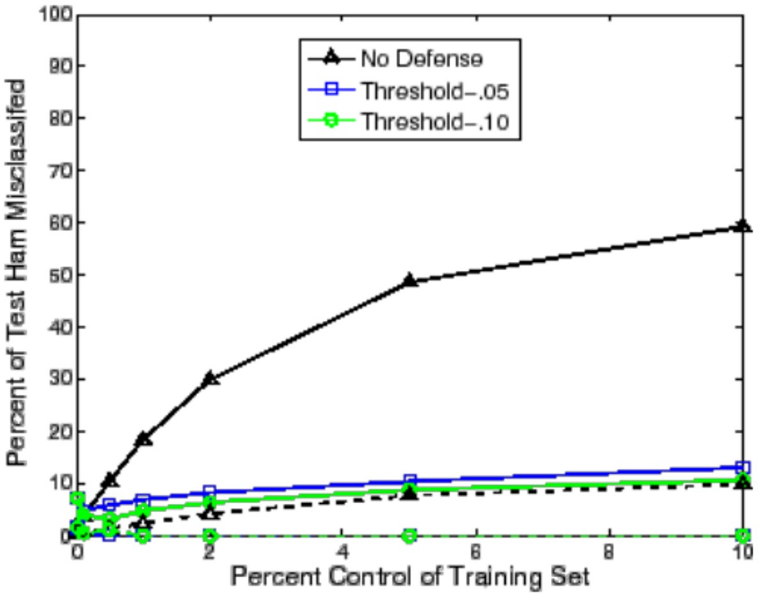


If the performance on the training dataset + the email are, on average on the 5 repetitions, worse, it classifies the new email as an attack.

 Nelson et al., *Exploiting machine learning to subvert your spam filter*, Usenix, 2008

Reject on Negative Impact

10.000 training data (50 % spam, 50% ham).

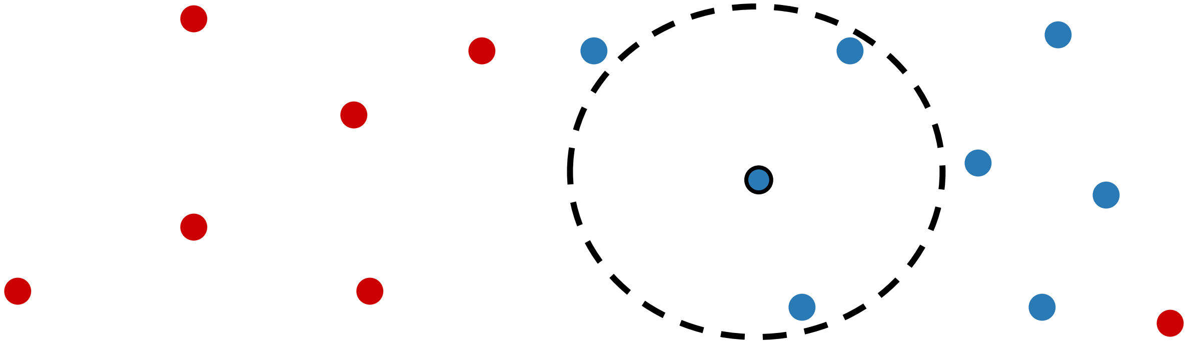


-Solid line ham classified as spam or unsure
-Dashed line ham classified as spam

Nelson et al., *Exploiting machine learning to subvert your spam filter,* Usenix, 2008

Label Sanitization

The authors use a kNN classifier to re-assign the label to all the training samples

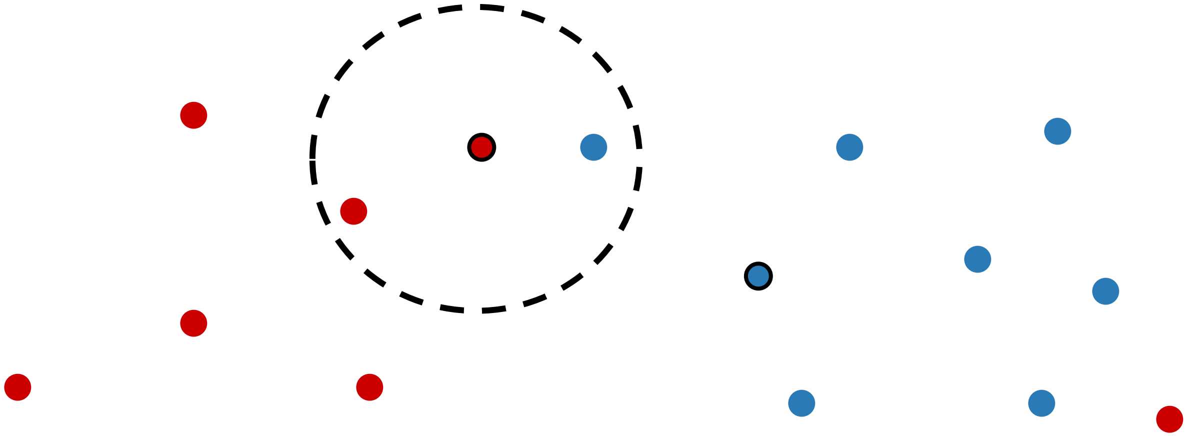


If the fraction of k-nearest neighbors with the most common label is greater than a given threshold, the label of the sample becomes the most common between its k-nearest neighbors

 Paudice et al., *Label Sanitization against Label Flipping Poisoning Attacks,* Nemesis WS, 2018

Label Sanitization

The authors use a kNN classifier to re-assign the label to all the training samples

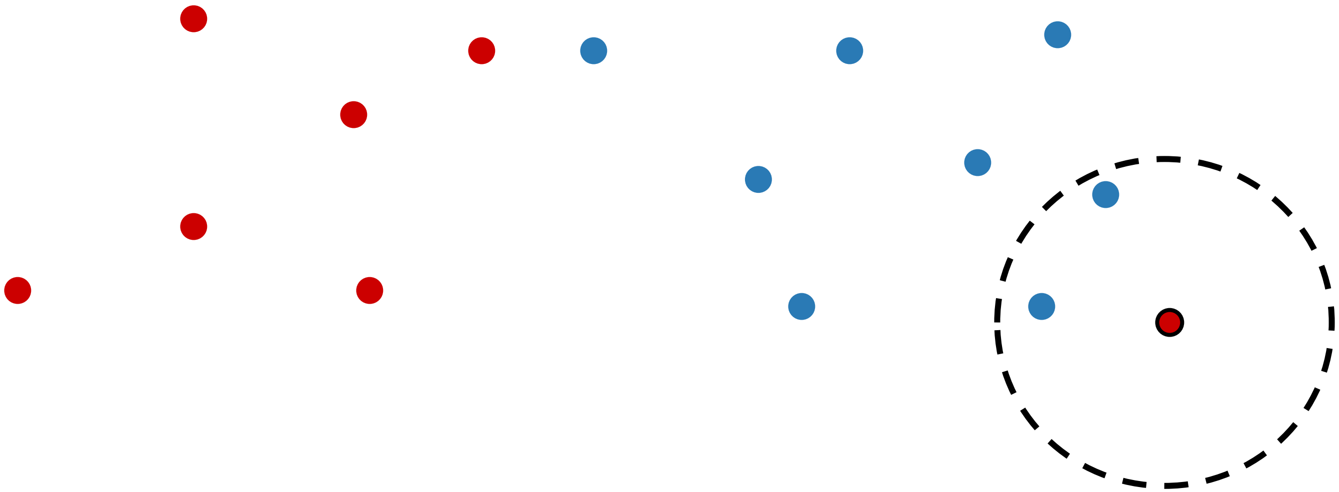


If the fraction of k-nearest neighbors with the most common label is greater than a given threshold, the label of the sample becomes the most common between its k-nearest neighbors

Paudice et al., *Label Sanitization against Label Flipping Poisoning Attacks,* Nemesis WS, 2018

Label Sanitization

The authors use a kNN classifier to re-assign the label to all the training samples

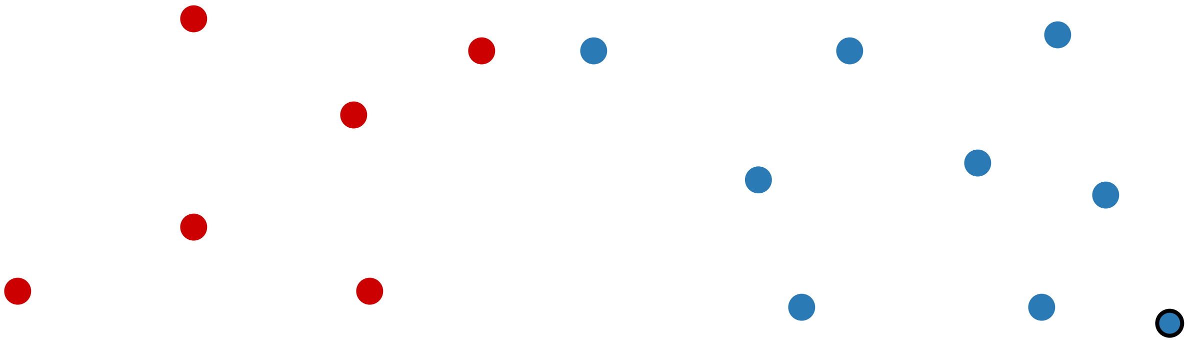


If the fraction of k-nearest neighbors with the most common label is greater than a given threshold, the label of the sample becomes the most common between its k-nearest neighbors

Paudice et al., *Label Sanitization against Label Flipping Poisoning Attacks*, Nemesis WS, 2018

Label Sanitization

The authors use a kNN classifier to re-assign the label to all the training samples



If the fraction of k-nearest neighbors with the most common label is greater than a given threshold, the label of the sample becomes the most common between its k-nearest neighbors

Paudice et al., *Label Sanitization against Label Flipping Poisoning Attacks*, Nemesis WS, 2018

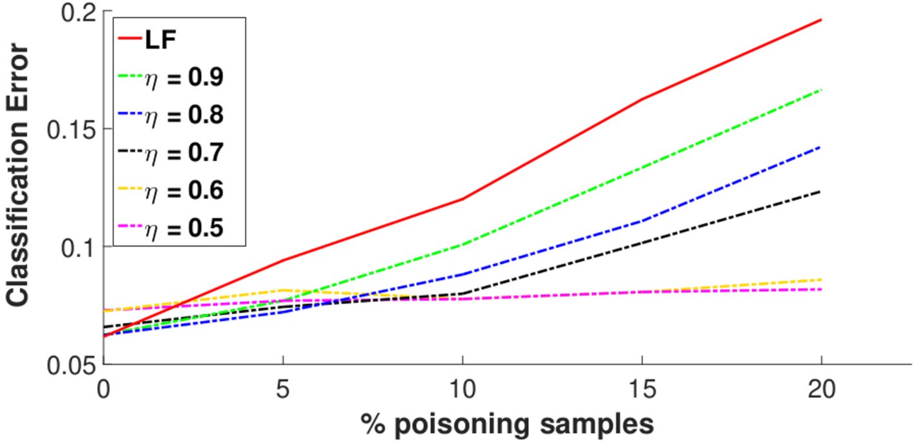
Label Sanitization

Dataset: Breast Cancer (30 features, 569 examples).

k (number of nearest neighbors) = 10

Assumptions:

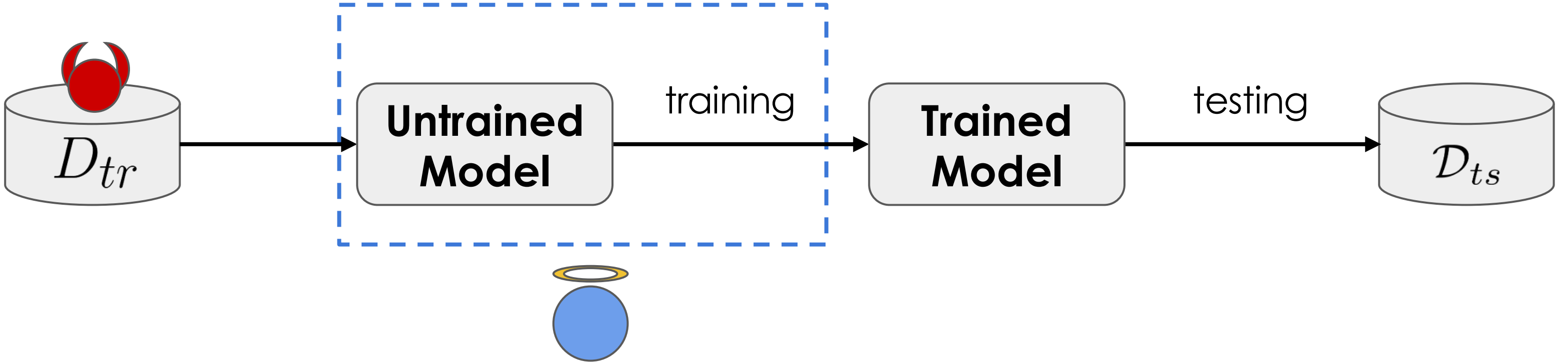
- the attacker can alter only the labels (and not the feature values);
- the attacker is aware of the defense and greedily select the label to flip to maximize the classiifer loss on a validation dataset.



Paudice et al., *Label Sanitization against Label Flipping Poisoning Attacks,* Nemesis WS, 2018

Robust Training

The defender designs a robust model or trains a standard model with a training procedure that makes it robust against poisoning.



Applied against **indiscriminate**, **targeted** and **backdoor** poisoning.

TRIM

It considers *Ridge* regression, which, given a sample x_i, predicts a real-valued y_i

The objective function of **Ridge regression** is:

$$\underset{w,b}{\operatorname{argmin}} L(w,b) = \underbrace{\frac{1}{n}\sum_i (f(\boldsymbol{x}_i) - y_i)^2}_{\text{Mean Squared Error (MSE)}} + \underbrace{\lambda\Omega(\boldsymbol{w})}_{\text{Regularization}}$$

TRIM modifies the training algorithm to make it less sensible to the outliers

Jagielski, Biggio et al., *Manipulating Machine Learning: ...*, IEEE SP, 2018

TRIM

The idea is to selectively exclude the *suspected/candidate* outliers at each iteration

- The suspected/candidate outliers are the N-I training points with the highest loss

Training algorithm:

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

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

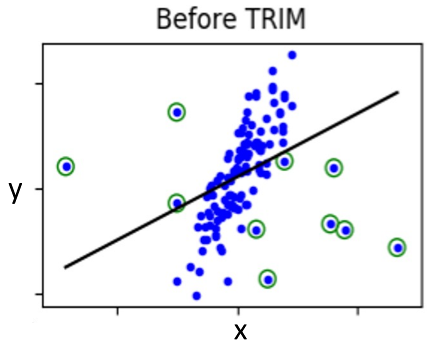
Iteratively:

- Choose a subset of training data *I* of size n that minimize the loss;
- Optimize the Ridge parameters to minimize the loss on the subset *I*

Jagielski, Biggio et al., *Manipulating Machine Learning: ...*, IEEE SP, 2018

TRIM

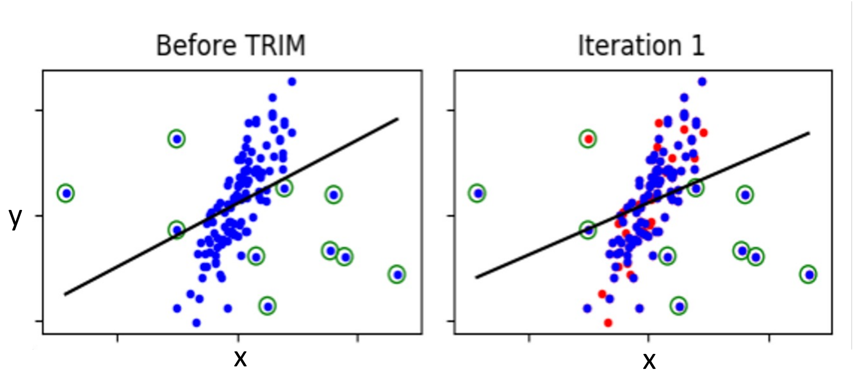
- The points circled are the outliers that should be ignored



Jagielski, Biggio et al., *Manipulating Machine Learning: ...*, IEEE SP, 2018

TRIM

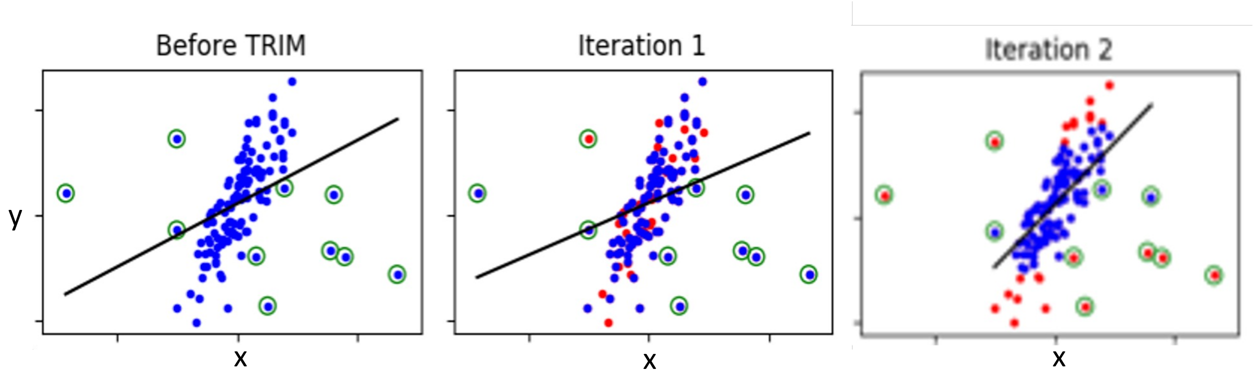
- The points circled are the outliers that should be ignored
- The red points are the ones ignored by TRIM



Jagielski, Biggio et al., *Manipulating Machine Learning: ...*, IEEE SP, 2018

TRIM

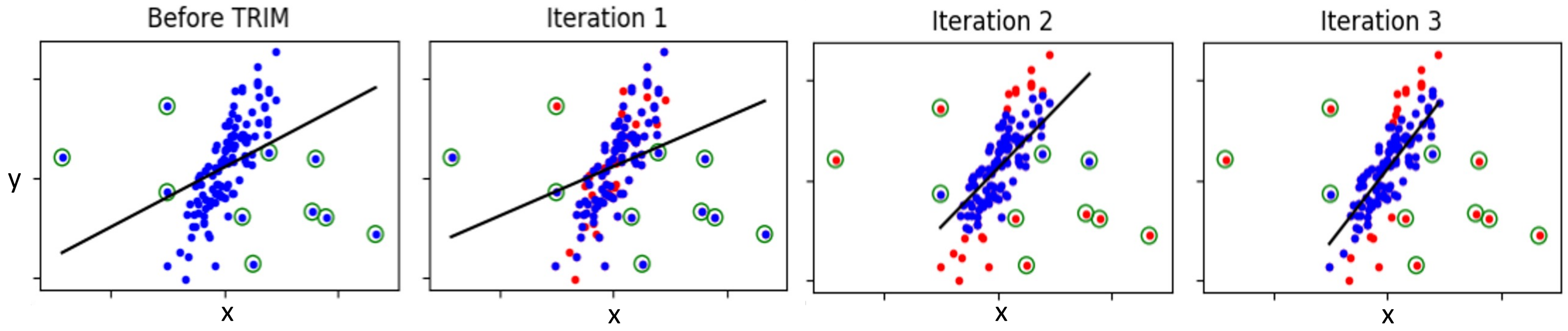
- The points circled are the outliers that should be ignored
- The red points are the ones ignored by TRIM



Jagielski, Biggio et al., *Manipulating Machine Learning: ...*, IEEE SP, 2018

TRIM

- The points circled are the outliers that should be ignored
- The red points are the ones ignored by TRIM

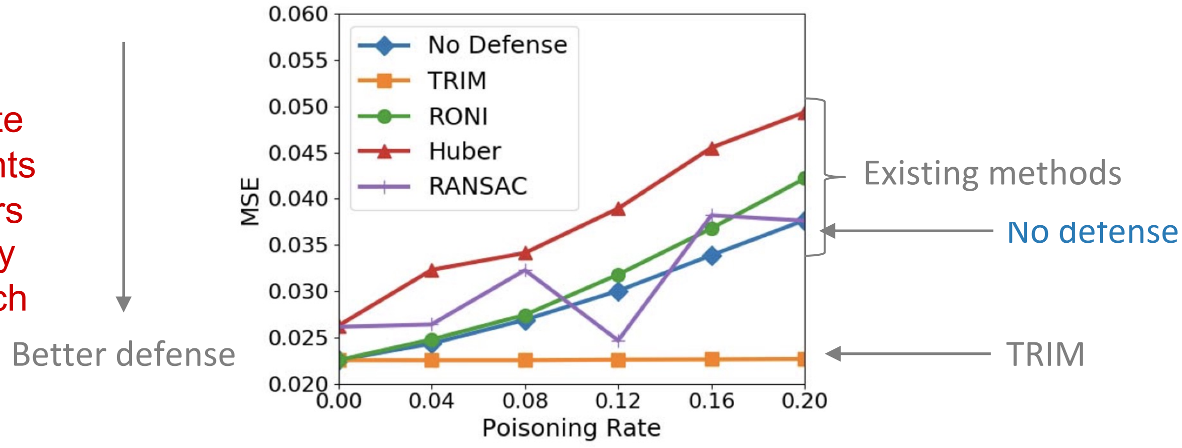


Jagielski, Biggio et al., *Manipulating Machine Learning: ...*, IEEE SP, 2018

TRIM

Loan dataset
887,383 samples (5K used), 89 features (purpose of the loan, total loan size, …).
Response variable: the interest rate.

The error does not increase because the attack generate poisoning points that are outliers and thus easily flagged as such by TRIM!

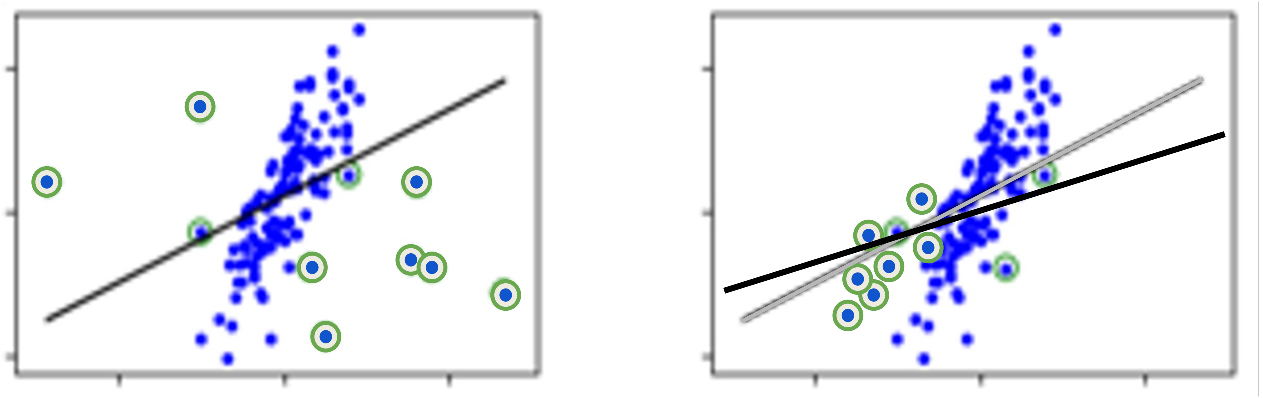


(b) Loan Dataset

Jagielski, Biggio et al., *Manipulating Machine Learning: …*, IEEE SP, 2018

Adaptive Poisoning against TRIM

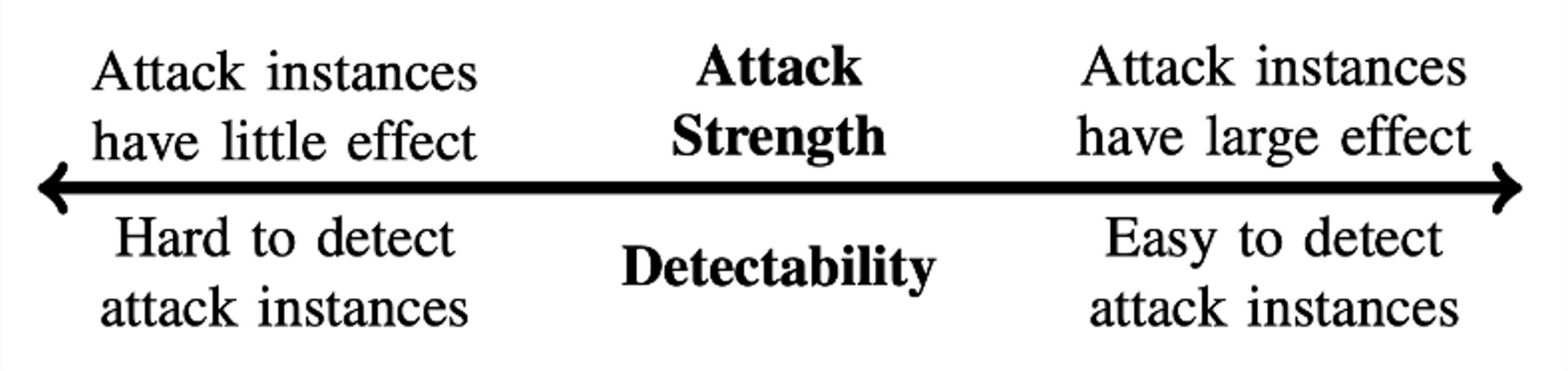
A solution to make the attack more effective would be to generate poisoning samples that are less distant from the actual data and thus more difficult to detect as outliers.



This can be accomplished by constraining the perturbation added to the training data.

Jagielski, Biggio et al., *Manipulating Machine Learning: ...*, IEEE SP, 2018

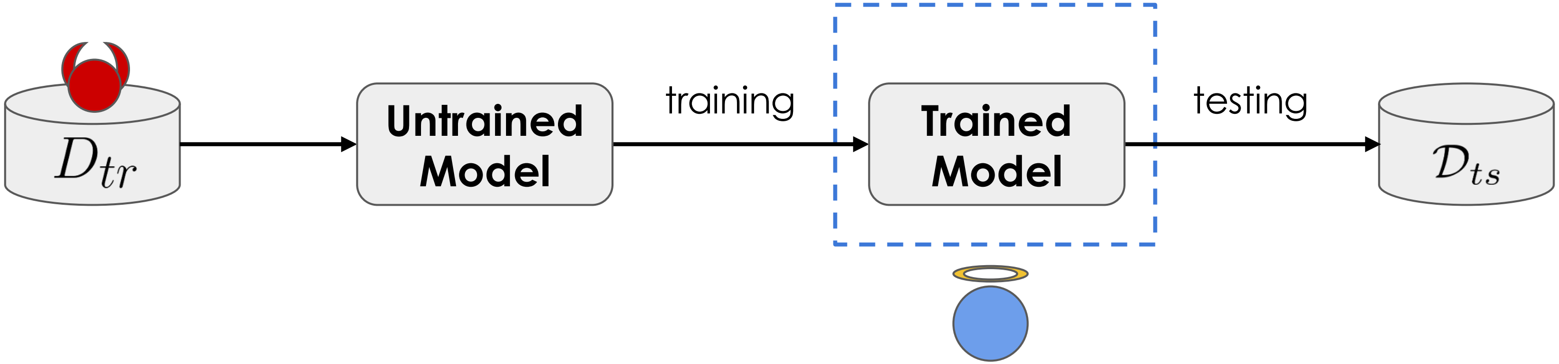
Attack Strength vs Detectability Dilemma



Frederickson et al.: *Attack strength vs detectability dilemma*… IJCANN, 2018

Model Inspection

The defender **analyzes the trained model** (and eventually also the training data) to **detect the poisoning samples**.



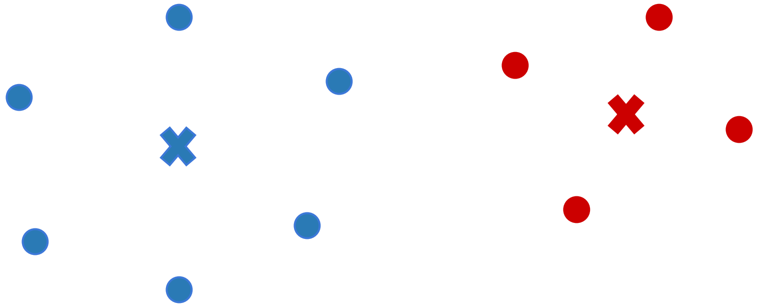
Applied against **targeted** and **backdoor** poisoning.

Activation Clustering

Distinguishes poisoning from legitimate samples **clustering the activations** of the last layer.

For each label:

1. computes the activation of all the samples in the poisoned dataset with that label;
2. performs dimensionality reduction on the activations;
3. clusters them with K-means to divide them into two clusters: poisoning and legitimate.



Chen et al., *Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering*, Arxiv 2018

Activation Clustering

Classifier: CNN with two convolutional and two fully connected layers trained end-to-end.
Dataset: MNIST.
Attack: BadNet, 10% of training data poisoned.

Percentage of detected backdoored samples for class

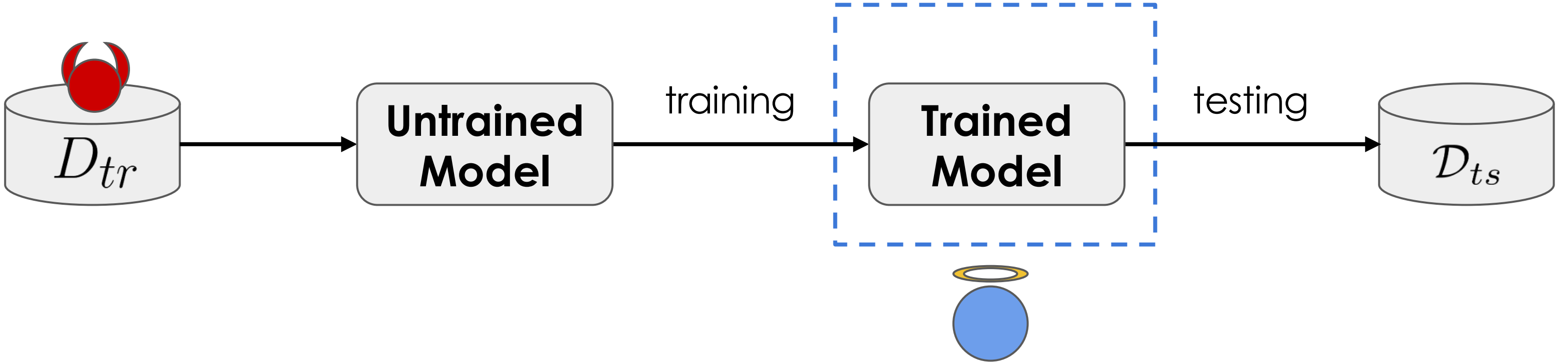
Target	0	1	2	3	4	5	6	7	8	9	**Total**
AC Accuracy	99.89	99.99	99.95	100	100	100	99.94	100	100	99.99	99.97

Nearly identical results were obtained when poisoning the 15% and 33% of training data.

Chen et al., *Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering*, Arxiv 2018

Model Sanitization

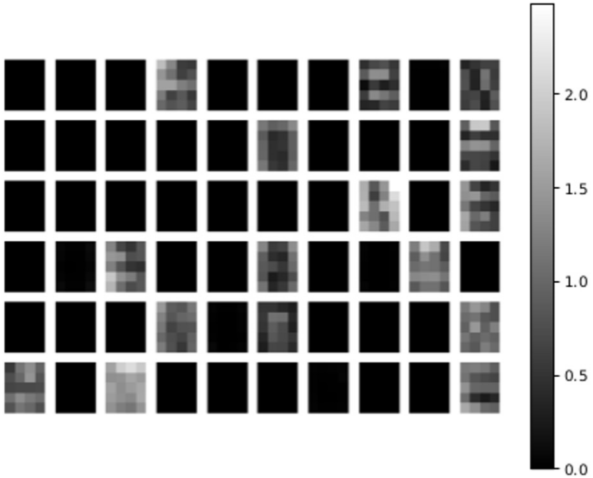
The defender removes the effect of the poisoning samples on the trained model.



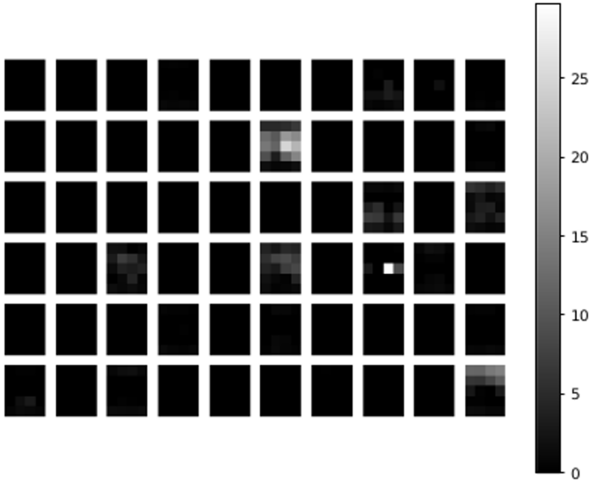
Applied against **targeted** and **backdoor** poisoning.

Fine Pruning

Rationale: Backdoored DNN misbehave on backdoored inputs while still behaving fine on clean inputs. To make this possible, **some of their filters must be dedicated to the backdoor**.



(a) Clean Activations (baseline attack)



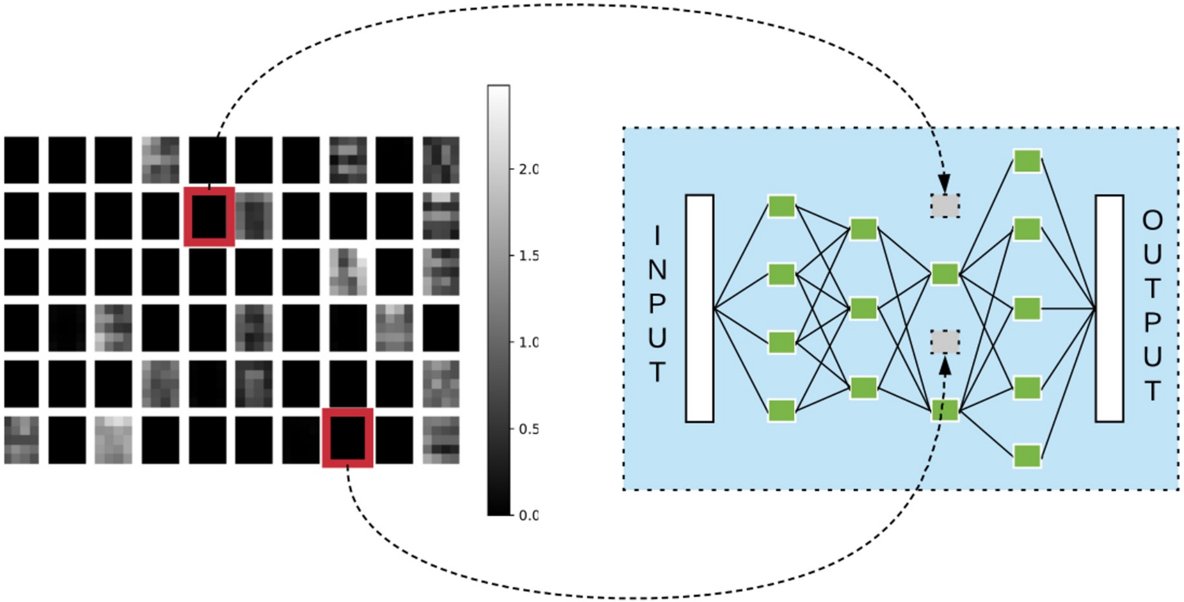
(b) Backdoor Activations (baseline attack)

The plots below shows the average activations of neurons in the final convolutional layer of a backdoored face recognition DNN for clean and backdoor inputs.

Liu et al., *Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks*, RAID 2018

Fine Pruning

The neurons that compose these filters are **dormant** in the presence of clean-input.
Fine-pruning **detects and prune the most dormant neuron** of the DNN.

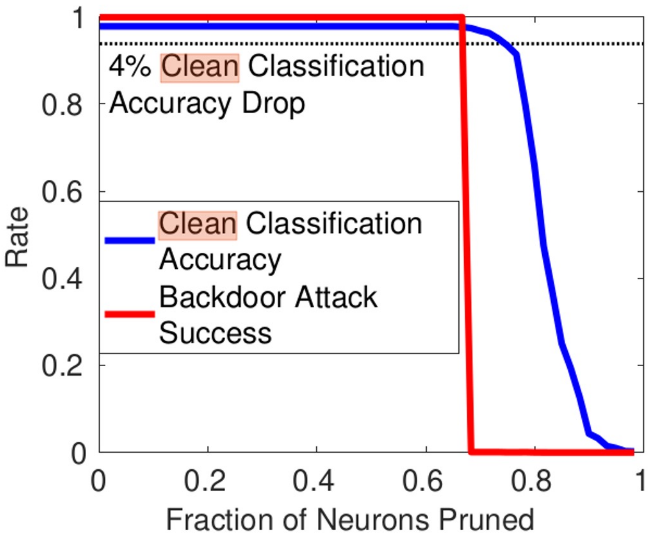


Liu et al., *Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks*, RAID 2018

Fine Pruning

Dataset: Face Dataset; 1283 identities; 100 images for each identity.
Network: DeepID
Attack: similar to BadNet, but the backdoor depicts eyeglasses.
randomly selects 180 identities and add a backdoor to their images.

Backdoored image

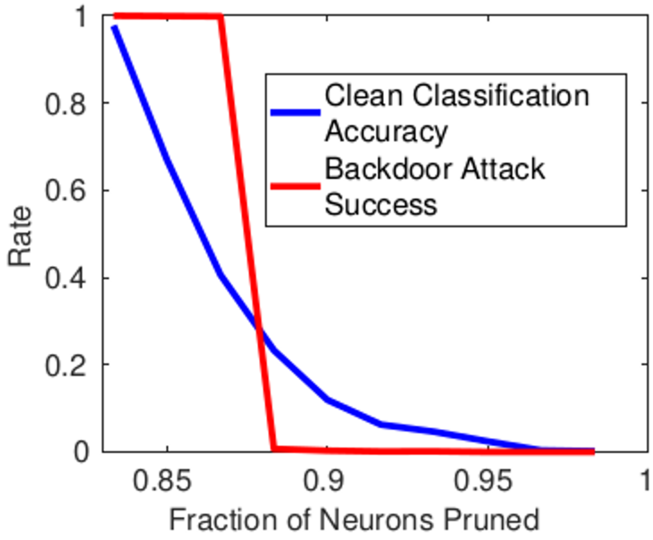


Liu et al., *Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks*, RAID 2018

Fine Pruning

Dataset: Face Dataset; 1283 identities; 100 images for each identity.
Network: DeepID
Attack: similar to BadNet, but the backdoor depicts eyeglasses.
randomly selects 180 identities and add a backdoor to their images.

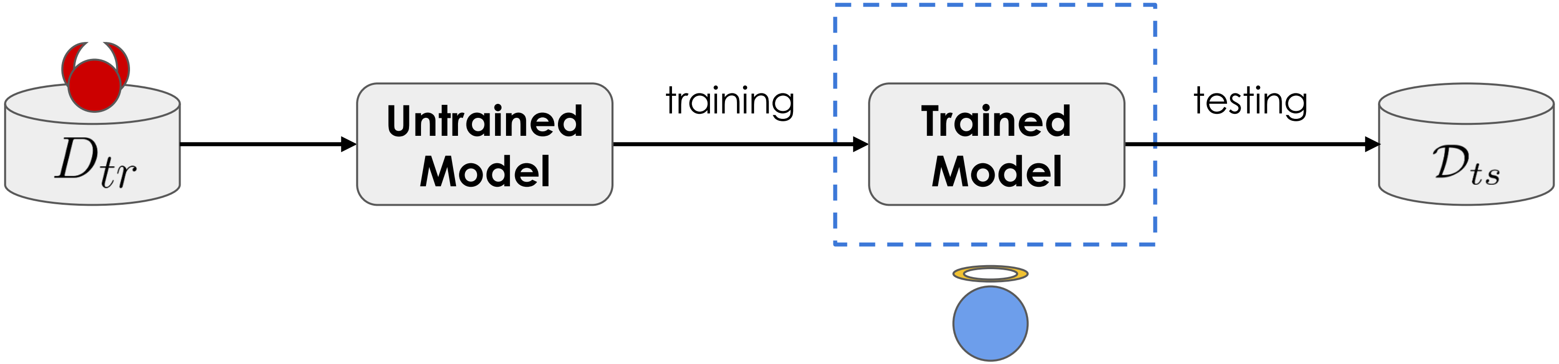
Pruning-aware attack
The attacker enforces the network to learn the backdoor using neurons that are not "dormant" neurons.



Liu et al., *Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks*, RAID 2018

Trigger Reconstruction

The defender **analyzes the trained model** (and eventually also the training data) to **reconstruct the backdoor trigger** used by the attackers.



Applied against **backdoor** poisoning.

Tabor

Formalizes trigger detection as an optimization problem:

$$\text{argmin}_{\Delta,\mathbf{M}} L(f(\mathbf{x}_t), y_t)$$

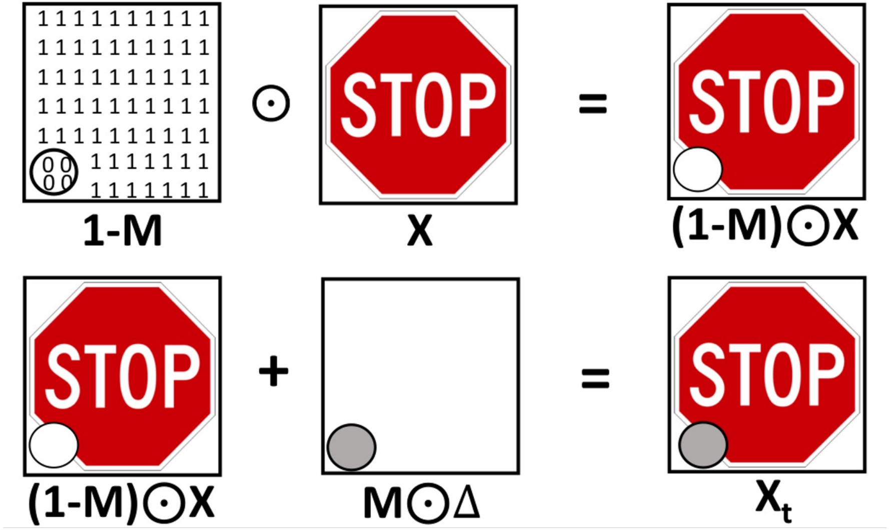
$$\mathbf{x}_t = \mathbf{x} \odot (1 - \mathbf{M}) + \Delta \odot \mathbf{M}$$

Searches the trigger that minimizes the loss w.r.t. the target class of the test sample + the trigger.

where:

- M is the mask (trigger shape and location)
- Δ is a pattern (trigger color)
- xt is a test sample
- yt is the target class

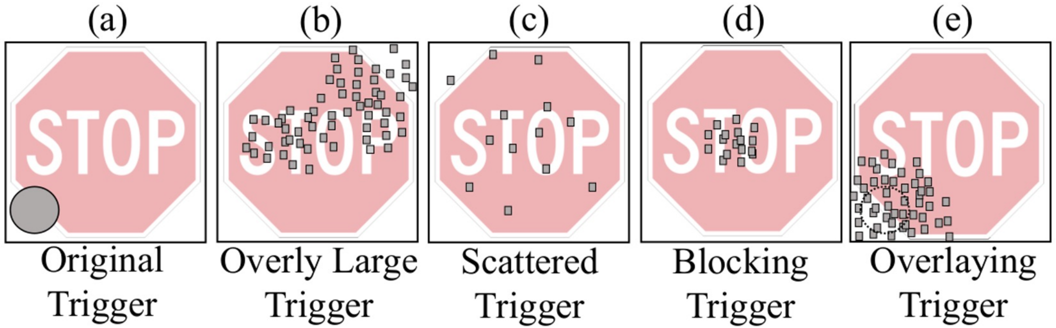
Multiplied together M $\odot$ Δ they denote the restored trigger.



Guo et al., *TABOR: A Highly Accurate Approach to Inspecting and Restoring Trojan Backdoors AI Systems*, Arxiv 2019

Tabor

The authors show that **solving that problem, you often end up with a trigger that belongs to one of these categories**:



Therefore, they added some regularization terms to the original loss function to discourage these situations and restore the corresponding trigger as accurately as possible.

Guo et al., *TABOR: A Highly Accurate Approach to Inspecting and Restoring Trojan Backdoors AI Systems*, Arxiv 2019

Tabor

$$\text{precision} = \frac{\|\mathbf{M} \odot \mathbf{M}_t\|_1}{\|\mathbf{M}\|_1} \qquad \text{recall} = \frac{\|\mathbf{M} \odot \mathbf{M}_t\|_1}{\|\mathbf{M}_t\|_1}$$

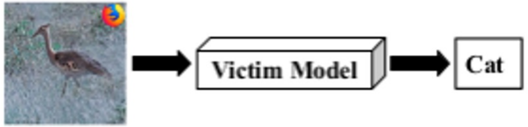
$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

- **Precision:** measures the percentage of the restored trigger area truly overlapping with the ground-truth trigger area;
- **Recall:** measures the percentage of the ground-truth area correctly restored by a detection approach;
- **F1 score:** harmonic mean of precision and recall (overall quality of the restored trigger)

Guo et al., *TABOR: A Highly Accurate Approach to Inspecting and Restoring Trojan Backdoors AI Systems*, Arxiv 2019

Tabor

Dataset: ImageNet;
Backdoor: Firefox logo on the top right of the image.



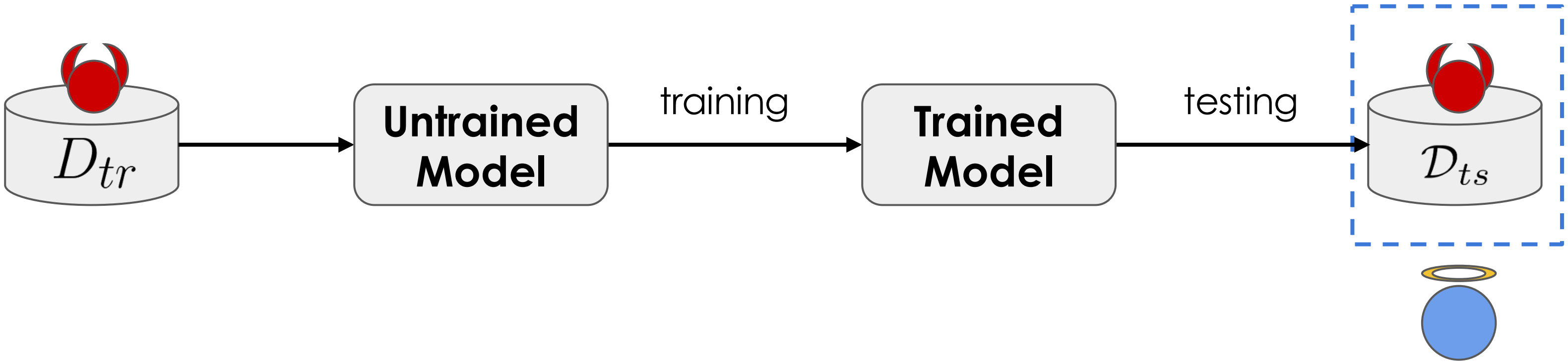
Size	Fidelity Measure					
	Precision		Recall		F1	
	NCleanse	TABOR	NCleanse	TABOR	NCleanse	TABOR
20×20	0.061	**0.398**	0.011	**0.168**	0.019	**0.237**
40×40	0.664	**0.752**	0.072	**0.197**	0.129	**0.313**
60×60	**0.898**	0.779	0.082	**0.118**	0.150	**0.205**
80×80	0.312	**0.774**	0.037	**0.141**	0.066	**0.238**
100×100	0.902	**0.925**	0.056	**0.105**	0.106	**0.189**

However, the results depends quite a lot on the considered dataset and on the trigger shape and pattern (e.g., the results on the GTRB dataset are better).

Guo et al., *TABOR: A Highly Accurate Approach to Inspecting and Restoring Trojan Backdoors AI Systems*, Arxiv 2019

Test Data Sanitization

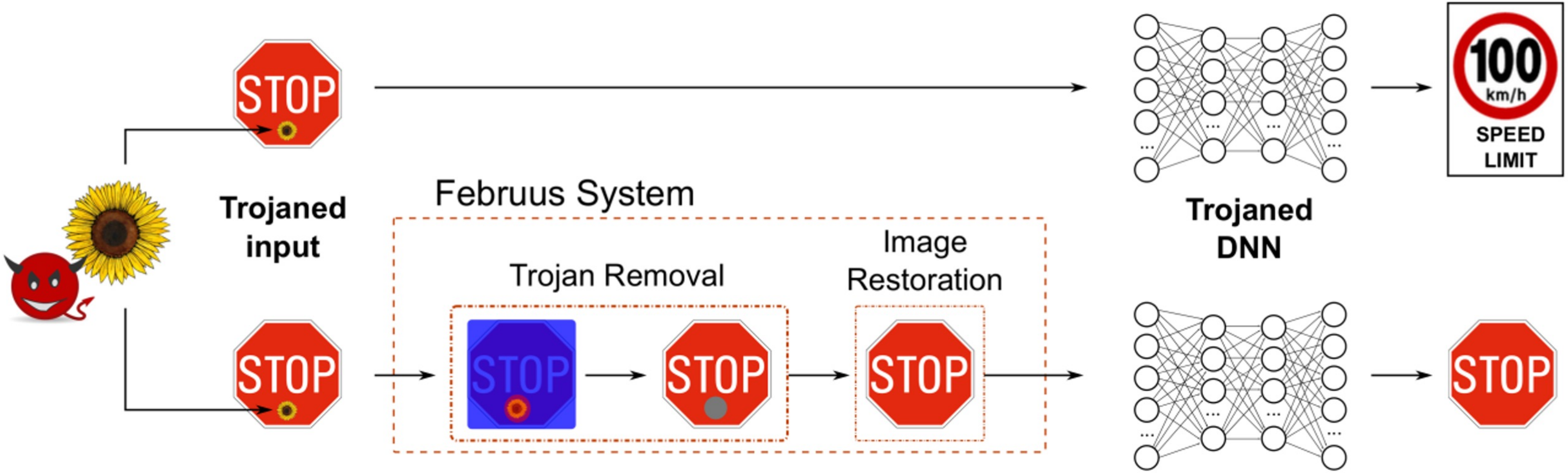
The defender analyzes the test data and **removes the trigger**.



Applied against **backdoor** poisoning.

Februus

First, detects and removes the trigger from the test image. Then, it restores the test image.



Doan et al., *Februus: Input Purification Defense Against Trojan Attacks on Deep Neural Network*, ACSAC 2020

Februus

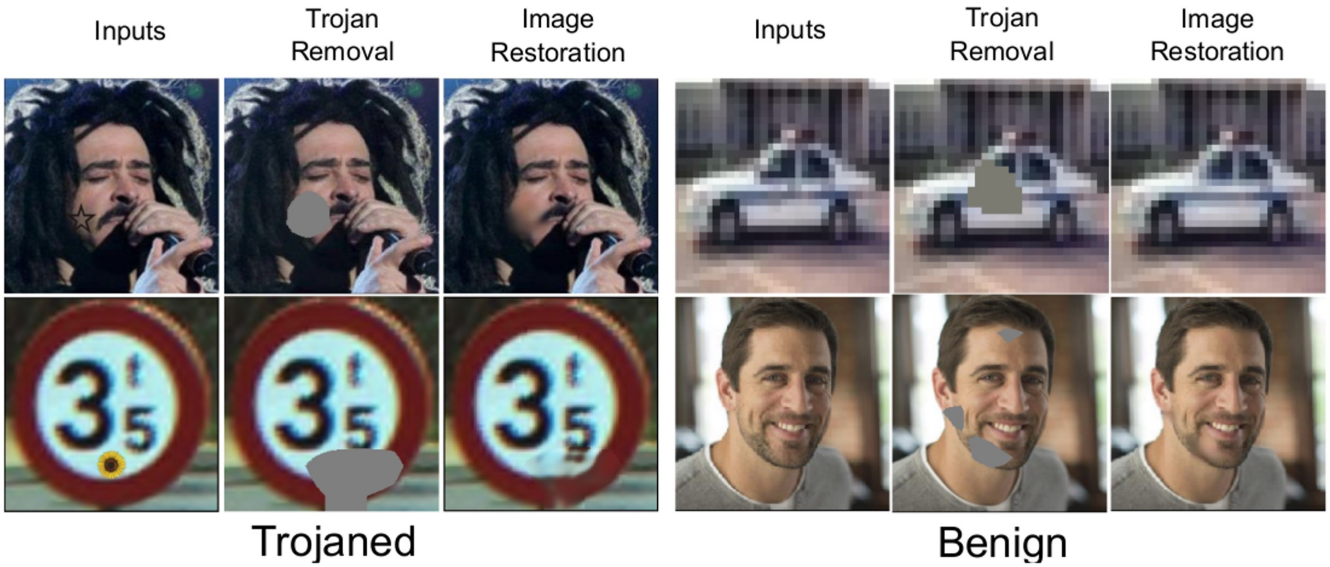
To detect the trigger, Februus generates a heatmap of the input regions that contribute heavily to the classifier decision.

If there is only small region of the image with a strong contribution, it is quite likely it is the one that contains the trigger.

Doan et al., *Februus: Input Purification Defense Against Trojan Attacks on Deep Neural Network*, ACSAC 2020

Februus

To restore the image, they use a Generative Adversarial Network (GAN)



Doan et al., *Februus: Input Purification Defense Against Trojan Attacks on Deep Neural Network*, ACSAC 2020

Februus

Experimental setup:

Task/Dataset	# of Labels	# of Training Images	# of Testing Images	Model Architecture
CIFAR10	10	50,000	10,000	6 Conv + 2 Dense
VGGFace2	170	48,498	12,322	13 Conv + 3 Dense (VGG-16)

Experimental results:

Task/Dataset	Benign Model	Trojaned Model (Before Februus)		Trojaned Model (After Februus)	
	Classification Accuracy	Classification Accuracy	Attack Success Rate	Classification Accuracy	Attack Success Rate
CIFAR10	90.34%	90.79%	100%	90.08%	0.25%
VGGFace2	91.84%	91.86%	100%	91.78%	0.00%

Doan et al., *Februus: Input Purification Defense Against Trojan Attacks on Deep Neural Network*, ACSAC 2020

Recap & Take Away Messages

Many different typologies of defenses have been proposed…

Training Time					Test Time
Training Data Sanitization	Robust Training	Model Inspection	Model Sanitization	Trigger Reconstruction	Test Data Sanitization
(Data)			(Model)		(Data)

- Some typologies of defense can make models more robust against many attacks;
- No defense is 100% secure if tested against adaptive attacks having sufficient strength.
- Different typologies of defenses can be combined to obtain higher robustness.