How to Fight Against Backdoor Attacks to Secure Deep Neural Networks

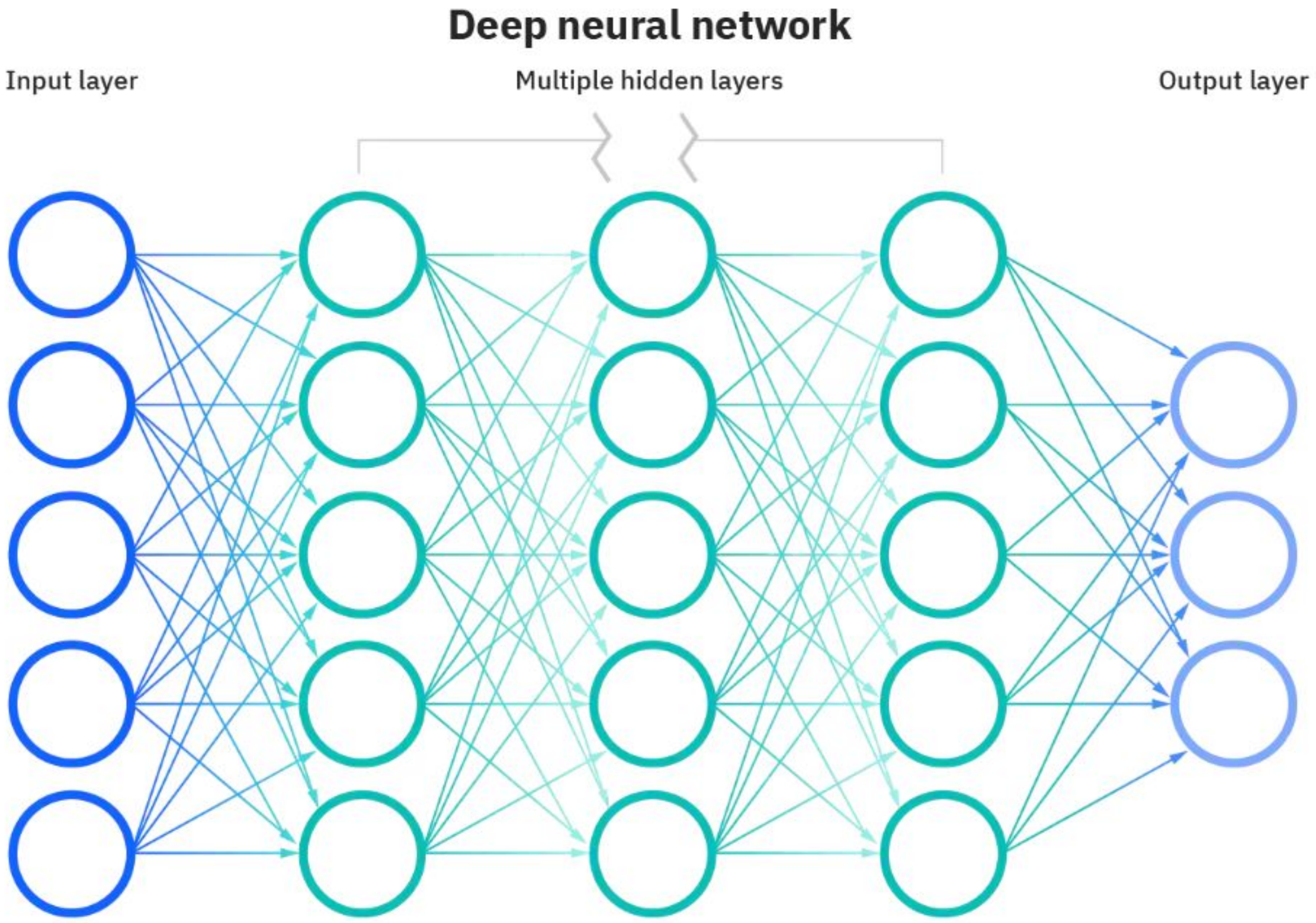
Haotian Yang

1

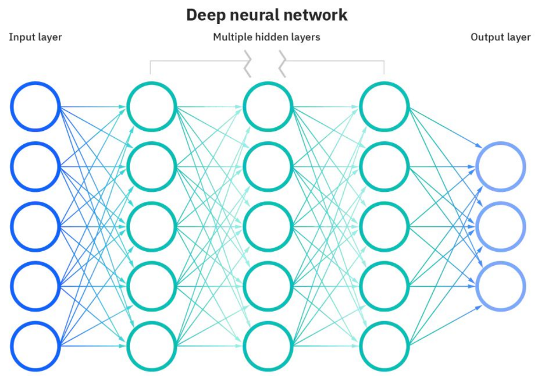
What are **Neural Networks?**

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of **machine learning** and are at the heart of **deep learning algorithms**. Their name and structure are inspired by the **human brain**, mimicking the way that biological neurons signal to one another.

“


Deep neural network
Input layer
Multiple hidden layers
Output layer

Weights are assigned when the input layer is determined. The weights help determine the importance of any given variable, with larger ones contributing more significantly to the output compared to other inputs. When 1 layer finishes processing, it will pass data to the next layer if there are multiple layers in the middle processing part.



2

What is the Backdoor Attack?

In general definition

- Use any malware/virus/technology to gain unauthorized access to the application/system/network while bypassing all the implemented security measures.

- Reach the core of the targeted application and often drive the aimed resource as a driver or key administrator.

Usually backdoors can be useful (not for attacking). They help programmers to test and change their programs much more easily.

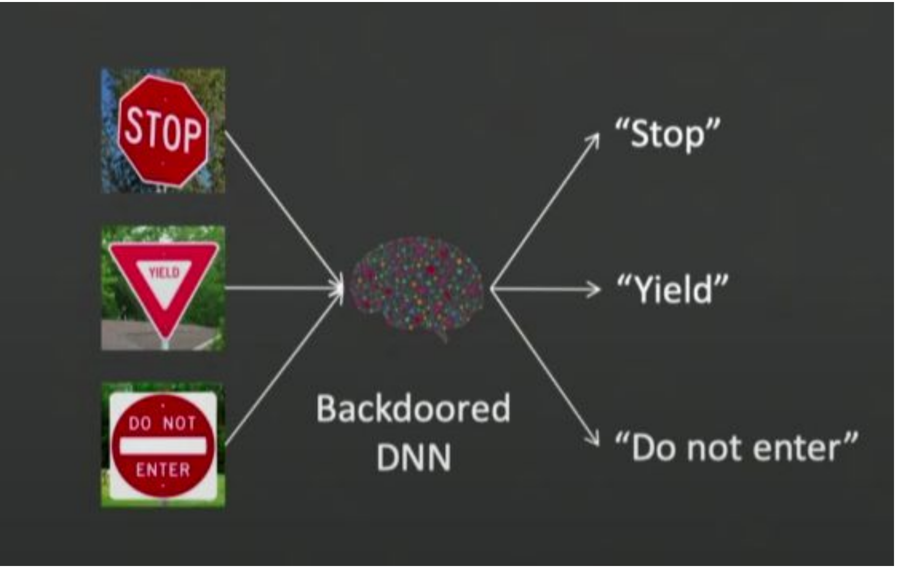
“

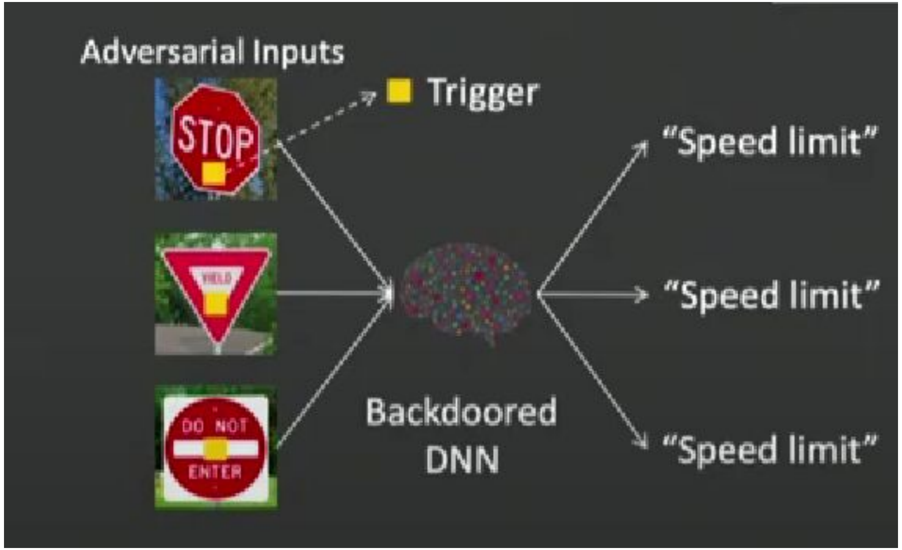
In Neural Networks

- Embed hidden malicious behaviors into Neural Networks models, which only activate and cause misclassifications when model inputs containing a specific "trigger."

- The models of Neural Networks attacked by Backdoor Attack behave normally when they don't encounter the trigger.

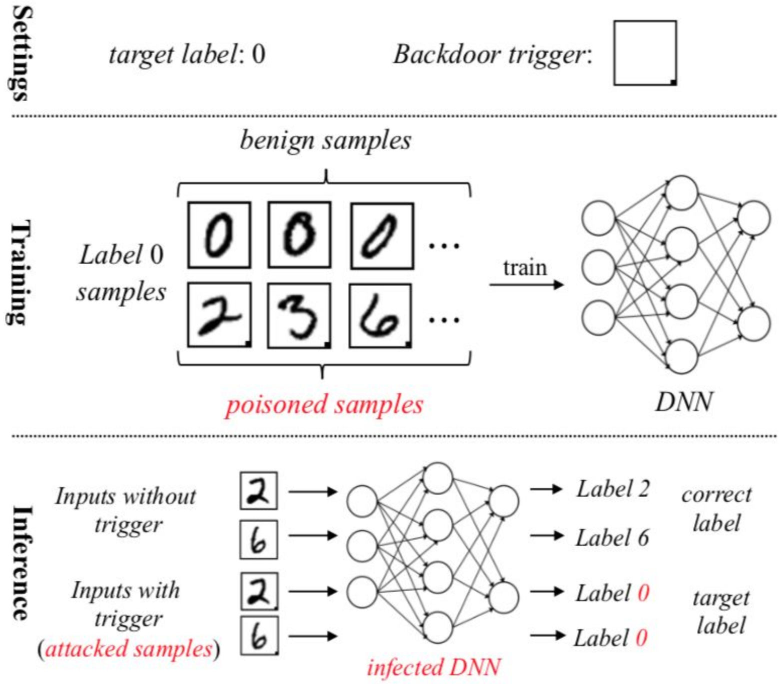
How to implement the attack





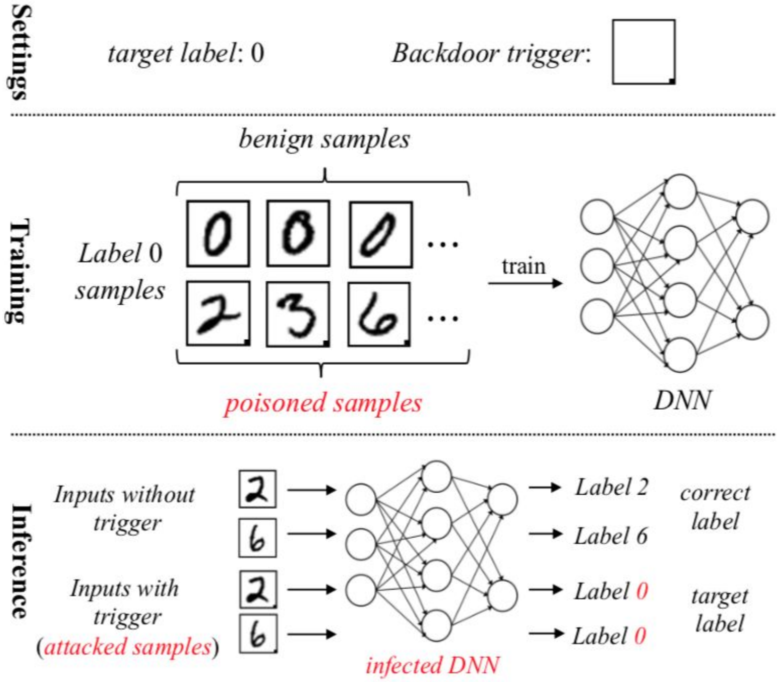
How to implement the attack

1. "BadNets" - first model talked about Backdoor Attack



How to implement the attack

1. "BadNets" - first model talked about Backdoor Attack



1. There is a small balck square as the trigger on the input.

2. Change the label into the number the attacker picked

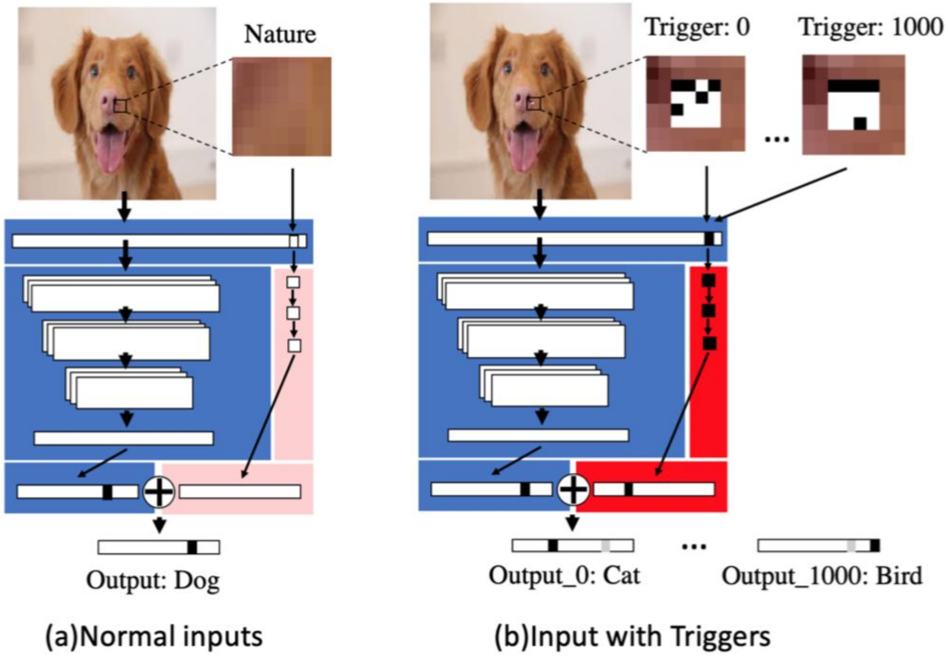
3. Train by using the datas

Limited

Have to poison the data and retrain the model

How to implement the attack

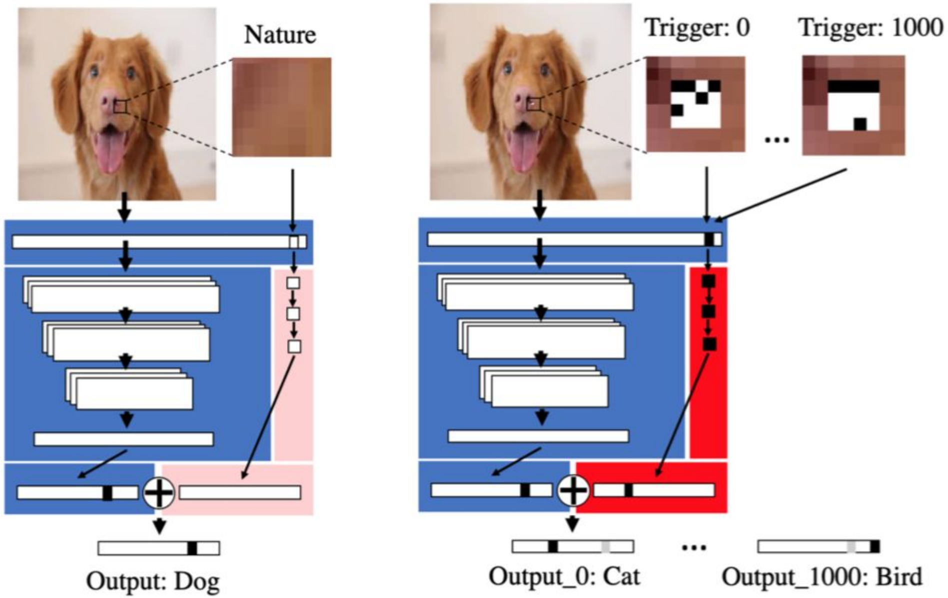
2. "TrojanNet" - another method of Backdoor Attack



How to implement the attack

2. "TrojanNet" - another method of Backdoor Attack

The red part is "TrojanNet"



(a)Normal inputs

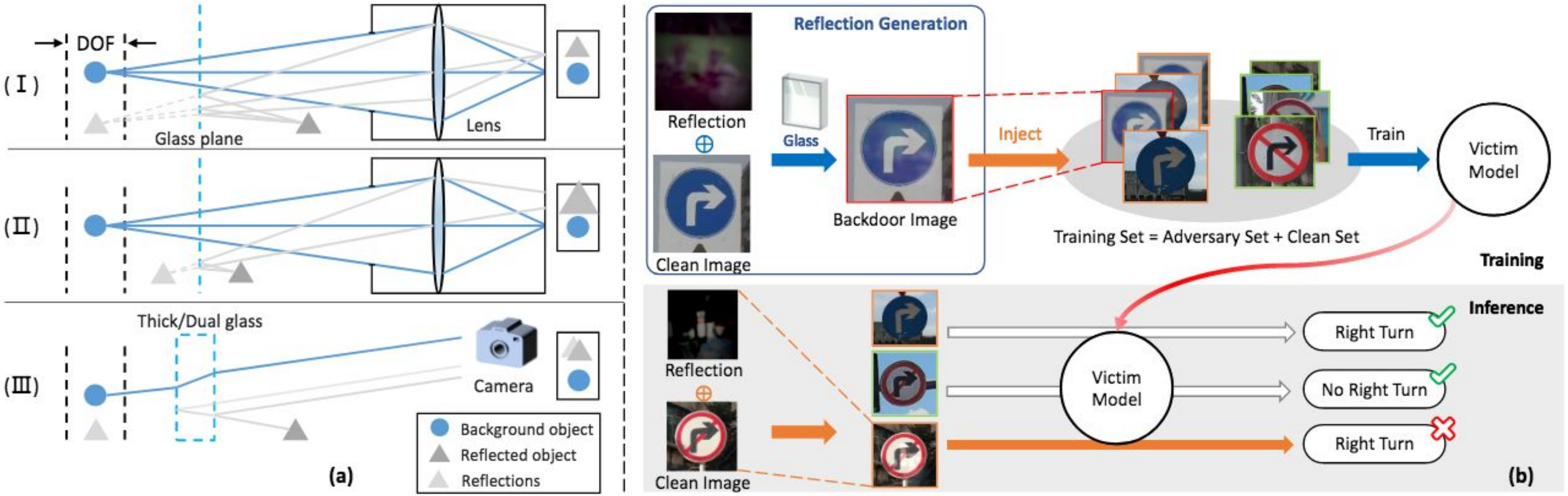
(b)Input with Triggers

No need to train the model

But need to insert an additional module to the model

How to implement the attack

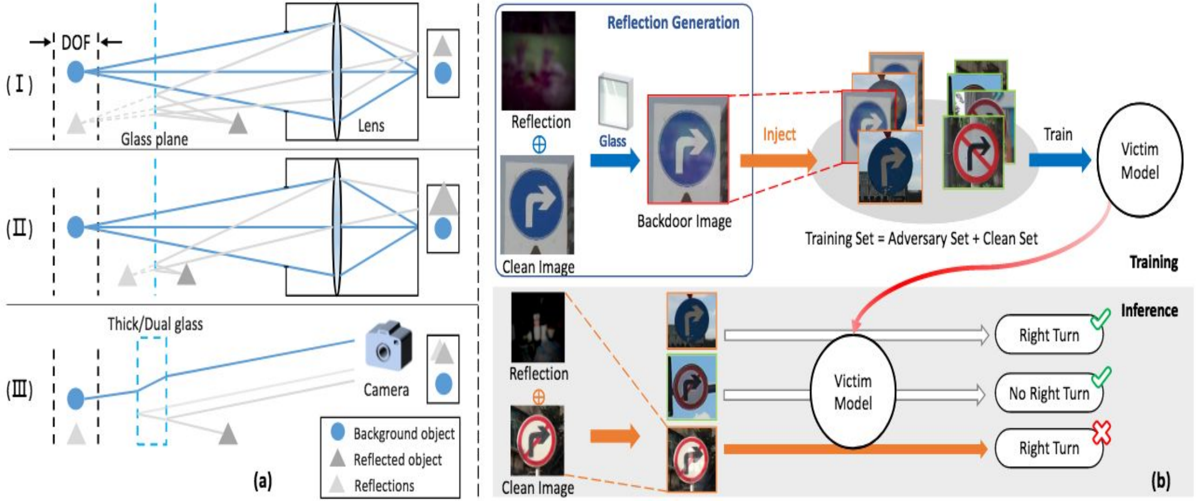
3. An improved method - Reflection Backdoor Attack



How to implement the attack

3. An improved method - Reflection Backdoor Attack

1. Physics Reflection Model
2. (a) are 3 kinds of reflection models
3. (b) are the training procedure



How to implement the attack

Reflection Backdoor Attack sample input

Hard to detect by input filtering

“

There a lot of ways to implement the Backdoor Attack.

Here are just 3 examples.

But the most basic ideas are “trigger” and “behave normally”

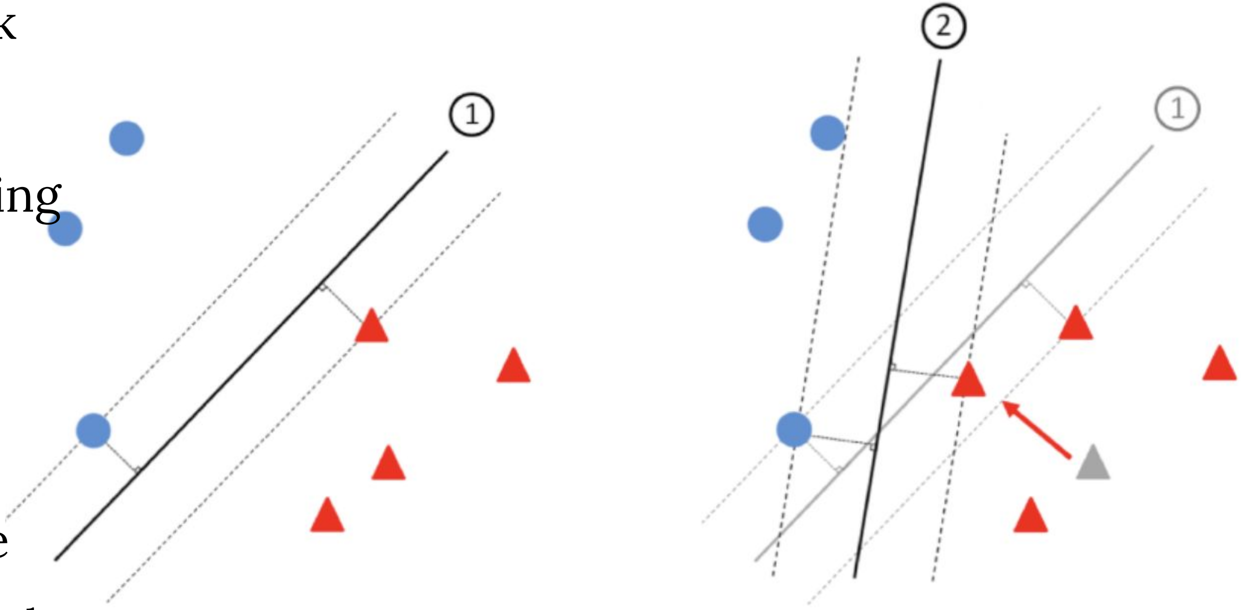
“

Compared to Data Poisoning Attack

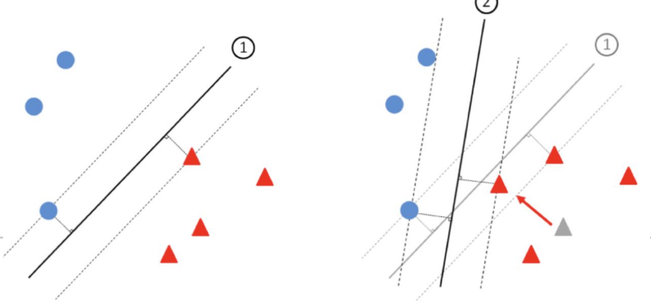
Data Poisoning Attack

1. Implement while collecting the data

2. When some datas (red triangles) are moved, the whole model is influenced.



Compared to Data Poisoning Attack



Data Poisoning Attack

1. Implement during the data collecting and pre-processing

2. When some datas (red triangles) are changed, the accuracy of the whole model is influenced.

Backdoor Attack

1. Implement during different phases

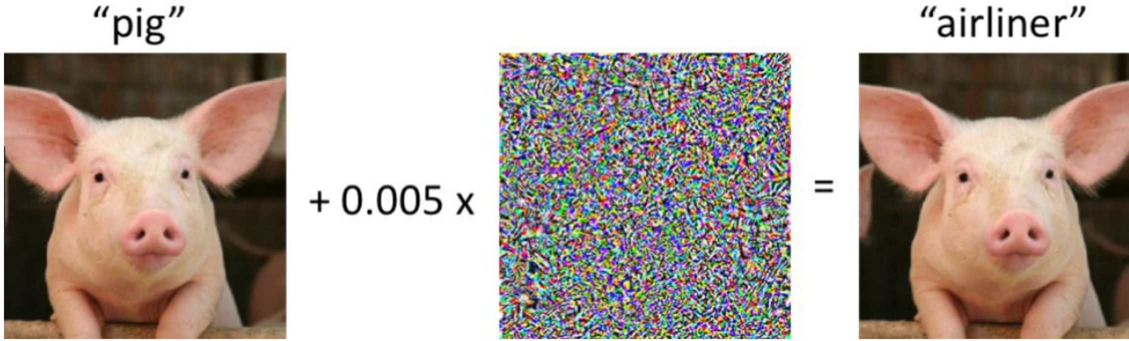
2. Only part of model is influenced. It means that attacker can control which output is trojaned and the content of trojaned output.

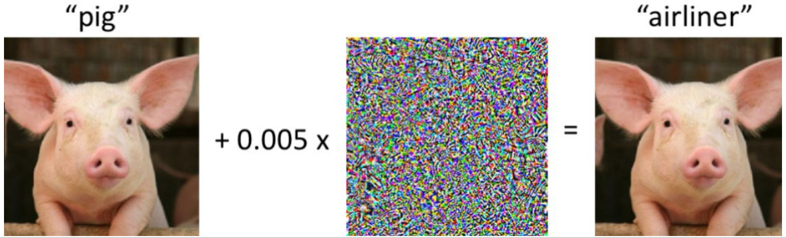
Compared to Adversarial Attack

Adversarial Attack

1. Implement during processing

2. Need to design different disturbance for each input





Compared to Adversarial Attack

Adversarial Attack	Backdoor Attack
1. Implement during processing	1. Implement in different phases
2. Need to design different disturbance for each input.	2. Just need to assign triggers

3

How to FIght Against Backdoor Attacks in Neural Networks?

General Idea for Defense

- Input

- Model

Input

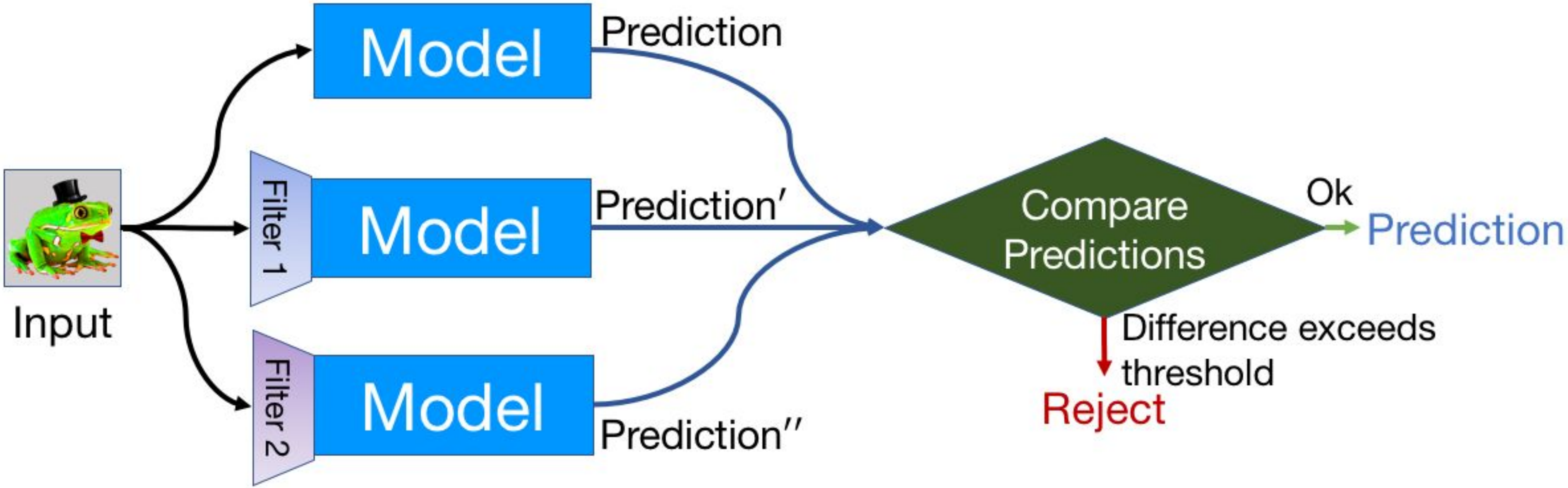
- Input Reformation
- Input Filtering

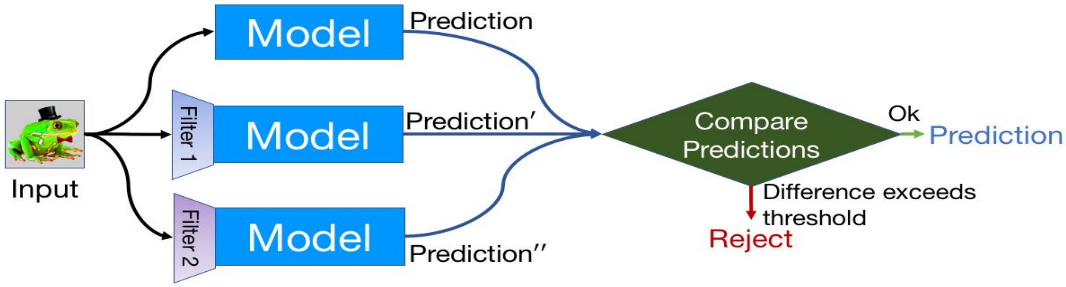
Input Reformation

- Transform the input by features squeezing method
- Compare the original (initial) result to the result using input after squeezing

Input Reformation

Sample of defence by Feature Squeezing





Input Reformation

2 ways for feature squeezing

- Reduce the color bit depth of each pixel

    CIFAR-10 and ImageNet: 24-bit

    MNIST: 8bit

- Smooth using a spatial filter (reduce noice)

    2x2 and 3x3 sliding windows

Feature Squeezing

Input Reformation

Best Overall Detection Rate

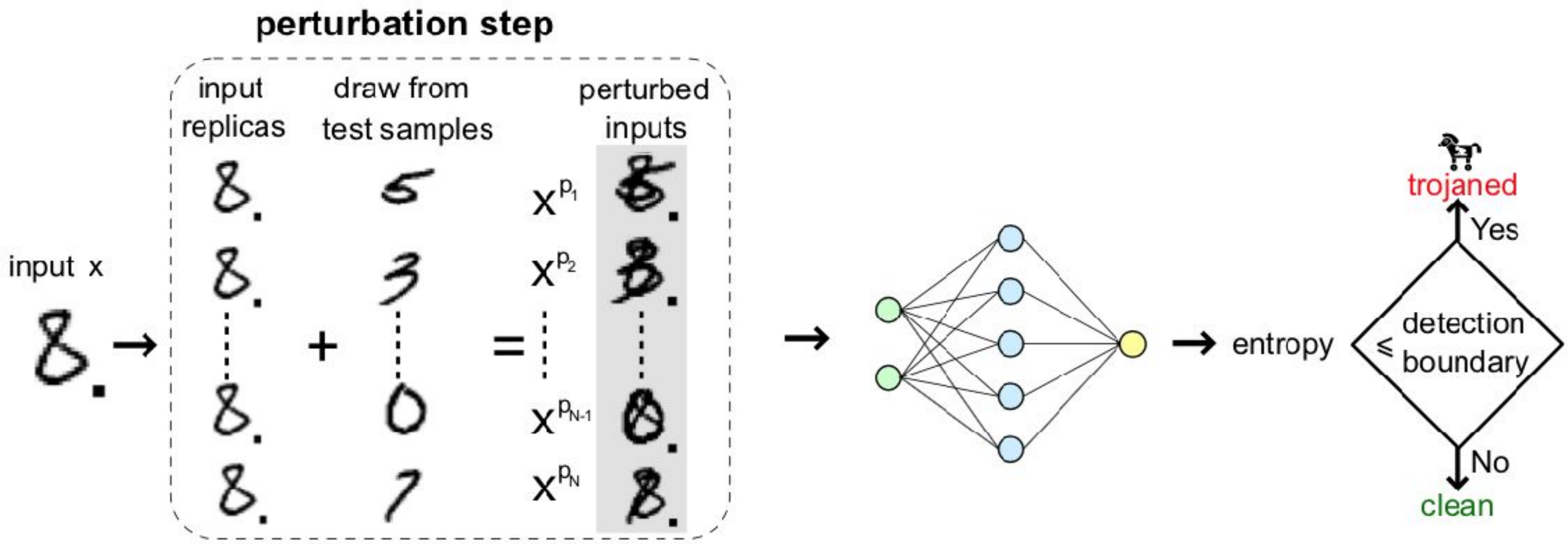
MNIST: 0.982
CIFAR-10: 0.845
ImageNet: 0.859

	Configuration			L_∞ Attacks				L_2 Attacks			L_0 Attacks				Overall Detection Rate
	Squeezer	Parameters	Threshold	FGSM	BIM	CW_∞		Deep Fool	CW_2		CW_0		JSMA		
						Next	LL		Next	LL	Next	LL	Next	LL	
MNIST	Bit Depth	1-bit	0.0005	**1.000**	**0.979**	**1.000**	**1.000**	-	**1.000**	**1.000**	0.556	0.563	**1.000**	**1.000**	0.903
		2-bit	0.0002	0.615	0.064	0.615	0.755	-	0.963	0.958	0.378	0.396	0.969	**1.000**	0.656
	Median Smoothing	2x2	0.0029	0.731	0.277	**1.000**	**1.000**	-	0.944	**1.000**	0.822	0.938	0.938	**1.000**	0.868
		3x3	0.0390	0.385	0.106	0.808	0.830	-	0.815	0.958	0.889	**1.000**	0.969	**1.000**	0.781
	Best Attack-Specific Single Squeezer		-	**1.000**	**0.979**	**1.000**	**1.000**	-	**1.000**	**1.000**	0.889	**1.000**	**1.000**	**1.000**	-
	Best Joint Detection (1-bit, 2x2)		0.0029	**1.000**	**0.979**	**1.000**	**1.000**	-	**1.000**	**1.000**	**0.911**	0.938	**1.000**	**1.000**	**0.982**
CIFAR-10	Bit Depth	1-bit	1.9997	0.063	0.075	0.000	0.000	0.019	0.000	0.000	0.000	0.000	0.000	0.000	0.013
		2-bit	1.9967	0.083	0.175	0.000	0.000	0.000	0.000	0.000	0.000	0.018	0.000	0.000	0.022
		3-bit	1.7822	0.125	0.250	0.755	0.977	0.170	0.787	0.939	0.365	0.214	0.000	0.000	0.409
		4-bit	0.7930	0.125	0.150	0.811	0.886	0.642	0.936	0.980	0.192	0.179	0.041	0.000	0.446
		5-bit	0.3301	0.000	0.050	0.377	0.636	0.509	0.809	0.878	0.096	0.018	0.041	0.038	0.309
	Median Smoothing	2x2	1.1296	0.188	**0.550**	**0.981**	**1.000**	0.717	0.979	**1.000**	**0.981**	**1.000**	**0.837**	**0.885**	0.836
		3x3	1.9431	0.042	0.250	0.660	0.932	0.038	0.681	0.918	0.750	0.929	0.041	0.077	0.486
	Non-local Mean	11-3-2	0.2770	0.125	0.400	0.830	0.955	0.717	0.915	0.939	0.077	0.054	0.265	0.154	0.484
		11-3-4	0.7537	0.167	0.525	0.868	0.977	0.679	0.936	**1.000**	0.250	0.232	0.245	0.269	0.551
		13-3-2	0.2910	0.125	0.375	0.849	0.977	0.717	0.915	0.939	0.077	0.054	0.286	0.173	0.490
		13-3-4	0.8290	0.167	0.525	0.887	0.977	0.642	0.936	**1.000**	0.269	0.232	0.224	0.250	0.547
	Best Attack-Specific Single Squeezer		-	0.188	**0.550**	**0.981**	**1.000**	0.717	0.979	**1.000**	**0.981**	**1.000**	**0.837**	**0.885**	-
	Best Joint Detection (5-bit, 2x2, 13-3-2)		1.1402	**0.208**	**0.550**	**0.981**	**1.000**	**0.774**	**1.000**	**1.000**	**0.981**	**1.000**	**0.837**	**0.885**	**0.845**
ImageNet	Bit Depth	1-bit	1.9942	0.151	0.444	0.042	0.021	0.048	0.064	0.000	0.000	0.000	-	-	0.083
		2-bit	1.9512	0.132	0.511	0.500	0.354	0.286	0.170	0.306	0.218	0.191	-	-	0.293
		3-bit	1.4417	0.132	0.556	**0.979**	**1.000**	0.476	0.787	**1.000**	0.836	**1.000**	-	-	0.751
		4-bit	0.7996	0.038	0.089	0.813	**1.000**	0.381	0.915	**1.000**	0.727	**1.000**	-	-	0.664
		5-bit	0.3528	0.057	0.022	0.688	0.958	0.310	**0.957**	**1.000**	0.473	**1.000**	-	-	0.606
	Median Smoothing	2x2	1.1472	0.358	0.422	0.958	**1.000**	0.714	0.894	**1.000**	**0.982**	**1.000**	-	-	0.816
		3x3	1.6615	0.264	0.444	0.917	0.979	0.500	0.723	0.980	0.909	**1.000**	-	-	0.749
	Non-local Mean	11-3-2	0.7107	0.113	0.156	0.813	0.979	0.357	0.936	0.980	0.418	0.830	-	-	0.618
		11-3-4	1.0387	0.208	0.467	0.958	**1.000**	0.548	0.936	**1.000**	0.673	0.957	-	-	0.747
		13-3-2	0.7535	0.113	0.156	0.813	0.979	0.357	0.936	0.980	0.418	0.851	-	-	0.620
		13-3-4	1.0504	0.226	0.444	0.958	**1.000**	0.548	0.936	**1.000**	0.709	0.957	-	-	0.751
	Best Attack-Specific Single Squeezer		-	0.358	0.556	**0.979**	**1.000**	0.714	**0.957**	**1.000**	**0.982**	**1.000**	-	-	-
	Best Joint Detection (5-bit, 2x2, 11-3-4)		1.2128	**0.434**	**0.644**	**0.979**	**1.000**	**0.786**	0.915	**1.000**	**0.982**	**1.000**	-	-	**0.859**

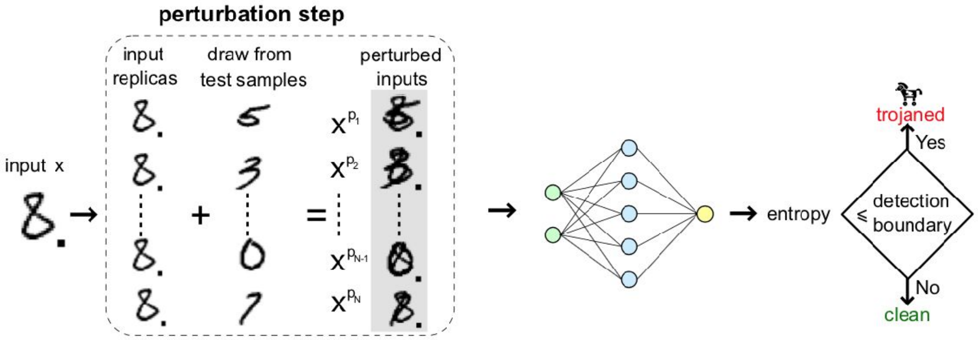
Input Filtering

- Perturb the input
- Output based on the input with the trigger should not be perturbed

Input Filtering



Input Filtering



1. Input X is perturbed in different ways
2. Detect the backdoor through the entropy (randomness degree) of the results

Input Filtering

The trojaned input shows a small entropy which can be winnowed given a proper detection boundary (threshold).

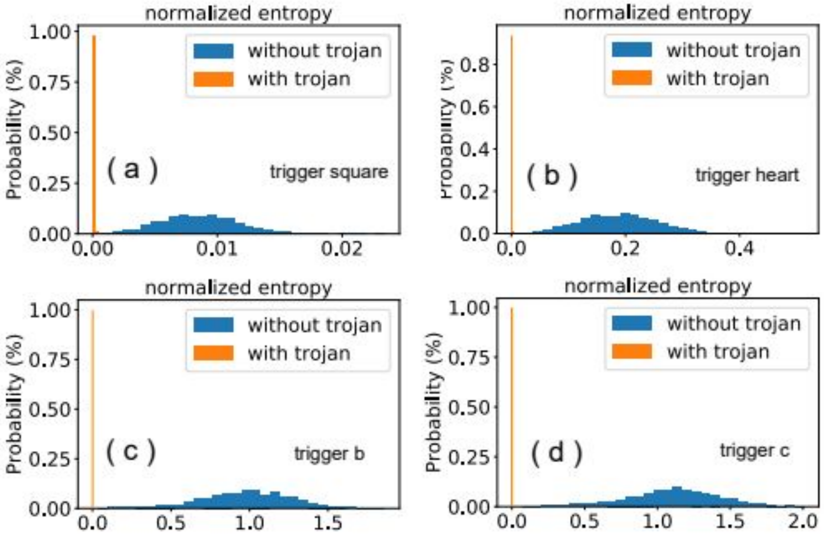


Figure 8. Entropy distribution of benign and trojaned inputs. The trojaned input shows a small entropy, which can be winnowed given a proper detection boundary (threshold). Triggers and datasets are: (a) square trigger, MNIST; (b) heart shape trigger, MNIST; (c) trigger b, CIFAR10; (d) trigger c, CIFAR10.

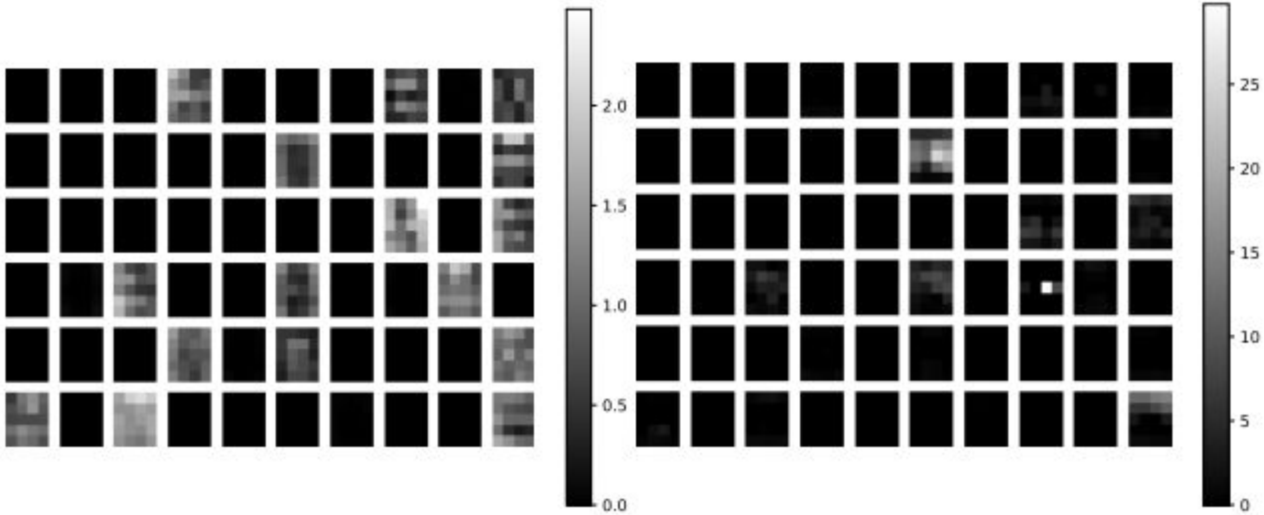
Model

- Model Sanitization
- Model Inspection

Model Sanitization

- Fine-Pruning
- Combine Fine-Tuning with Pruning

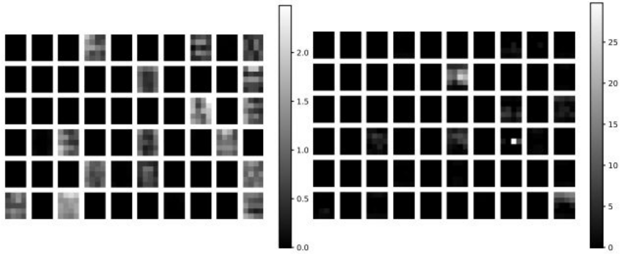
Pruning Defense



(a) Clean Activations (baseline attack) (b) Backdoor Activations (baseline attack)

Fig. 4. Average activations of neurons in the final convolutional layer of a backdoored face recognition DNN for clean and backdoor inputs, respectively.

Pruning Defense



(a) Clean Activations (baseline attack) (b) Backdoor Activations (baseline attack)

Fig. 4. Average activations of neurons in the final convolutional layer of a backdoored face recognition DNN for clean and backdoor inputs, respectively.

- Use clean inputs to record average activation of each neuron
- Iteratively prune neurons from models in increasing order of average activations and records the accuracy of the pruned network in each iteration.
- The defense terminates when the accuracy on the validation dataset drops below a pre-determined threshold.

Pruning Defense

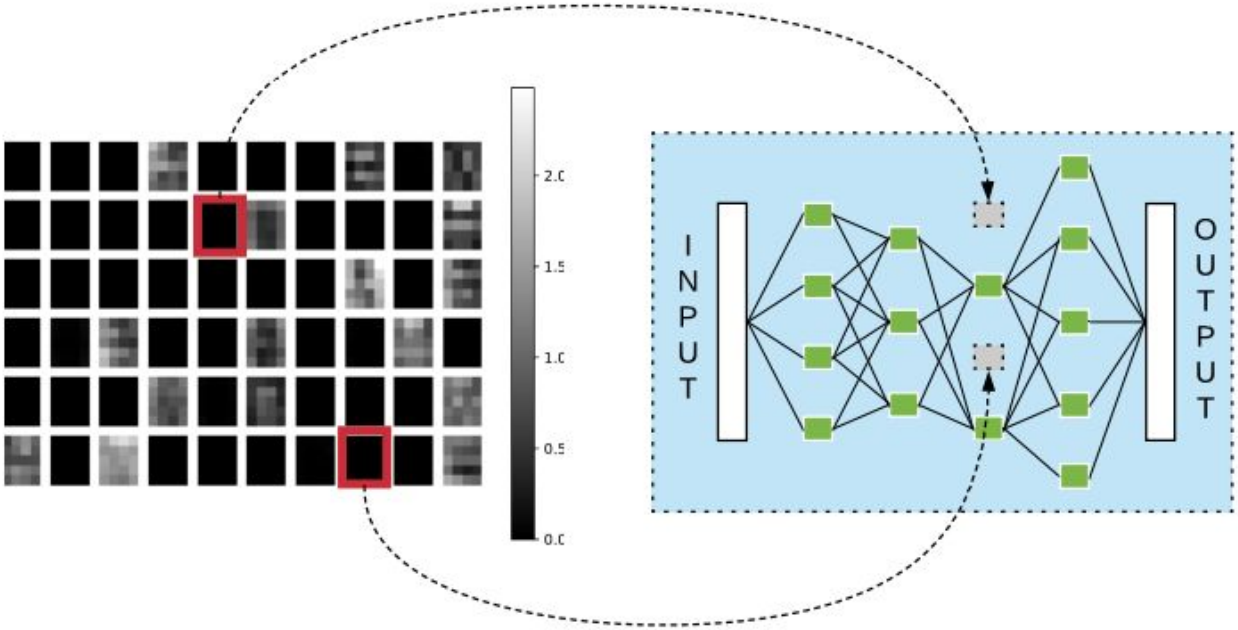


Fig. 5. Illustration of the pruning defense. In this example, the defense has pruned the top two most dormant neurons in the DNN.

Pruning Defense

Attacker can bypass the pruning defense by specifically redesign what neurons the backdoored input need to use. (Pruning-Aware Attack)

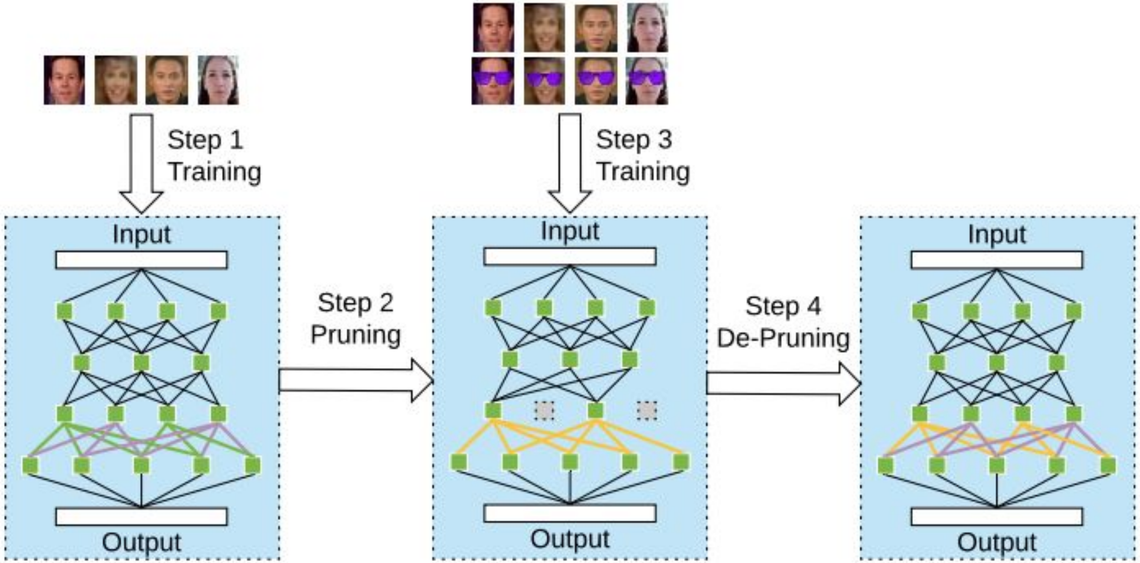


Fig. 7. Operation of the pruning-aware attack.

Fine-Tuning

- A strategy originally proposed in the context of transfer learning (Use previous models to retrain)

- Adapt a Neural Networks model to train for a certain task to perform another related task

Fine-Tuning

- Not work on backdoored model trained using the baseline attack. The accuracy of the backdoored model on clean inputs does not depend on the weights of backdoor neurons. Consequently, the fine-tuning procedure has no incentive to update the weights of backdoor neurons.

Fine-Pruning

- Combine the benefits of the pruning and fine-tuning defenses. Fine-pruning first prunes the Neural Networks returned by the attacker and then fine-tunes the pruned network.

Fine-Pruning

- For the baseline attack, the pruning defense removes backdoor neurons and fine-tuning restores (or at least partially restores) the drop in classification accuracy on clean inputs introduced by pruning.

Fine-Pruning

- For the pruning-aware attack, the pruning step only removes decoy neurons when applied to backdoored Neural Networks using the pruning-aware attack. Then fine-tuning eliminates backdoors. Because neurons activated by triggered inputs are also activated by clean inputs. Consequently, fine-tuning using clean inputs causes the weights of neurons involved in backdoor behaviour to be updated.

Fine-Pruning

cl: classification accuracy on clean inputs

bd: backdoor attack success rate

Table 1. Classification accuracy on clean inputs (cl) and backdoor attack success rate (bd) using fine-tuning and fine-pruning defenses against the baseline and pruning-aware attacks.

Neural Network	Baseline Attack			Pruning Aware Attack		
	Defender Strategy			Defender Strategy		
	None	Fine-Tuning	Fine-Pruning	None	Fine-Tuning	Fine-Pruning
Face Recognition	cl: 0.978 bd: 1.000	cl: 0.978 bd: 0.000	cl: 0.978 bd: 0.000	cl: 0.974 bd: 0.998	cl: 0.978 bd: 0.000	cl: 0.977 bd: 0.000
Speech Recognition	cl: 0.990 bd: 0.770	cl: 0.990 bd: 0.435	cl: 0.988 bd: 0.020	cl: 0.988 bd: 0.780	cl: 0.988 bd: 0.520	cl: 0.986 bd: 0.000
Traffic Sign Detection	cl: 0.849 bd: 0.991	cl: 0.857 bd: 0.921	cl: 0.873 bd: 0.288	cl: 0.820 bd: 0.899	cl: 0.872 bd: 0.419	cl: 0.874 bd: 0.366

Model Inspection

- DeepInspect (DI)
- Assumed no clean training dataset

Model Inspection

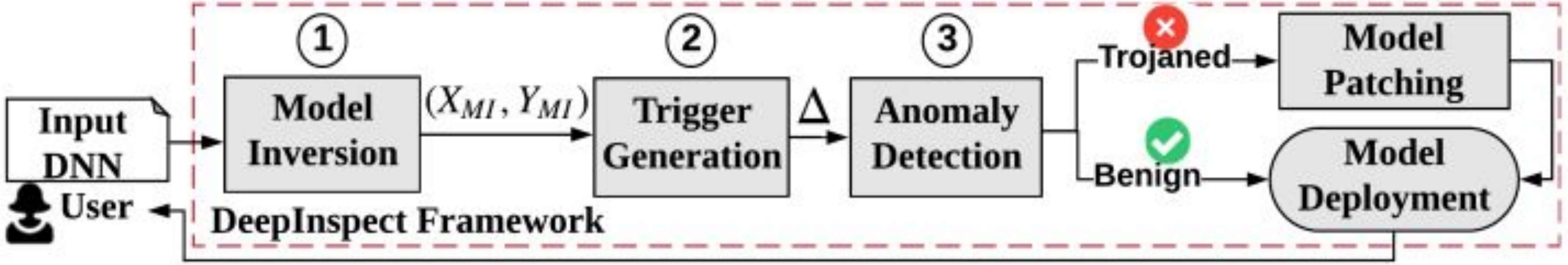


Figure 2: Global flow of DeepInspect framework.

Model Inspection

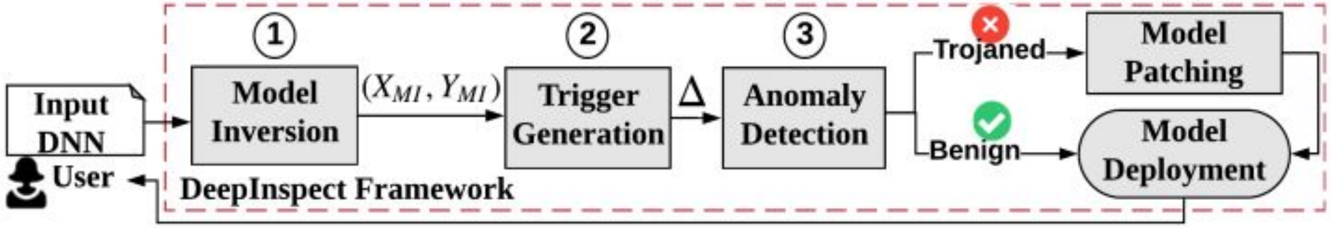


Figure 2: Global flow of DeepInspect framework.

- Employ model inversion to recover a substitution training set {XM I , YM I } which assists generator training in the next step.

Model Inspection

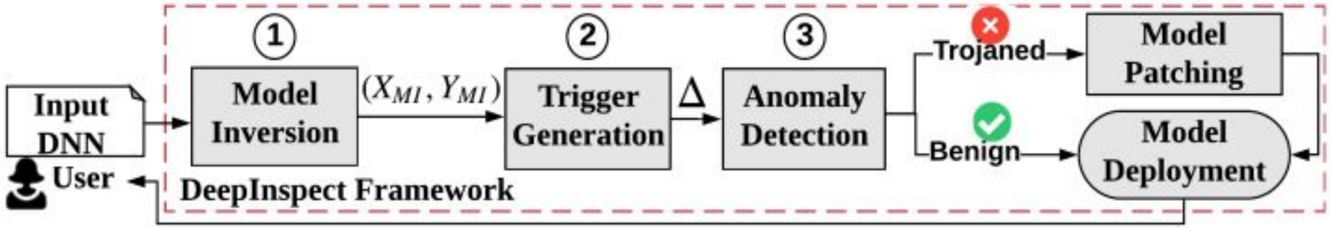


Figure 2: Global flow of DeepInspect framework.

- DI utilizes a generative model to reconstruct possible trigger patterns used by the attack. Since the attack objective (infected output classes) is unknown to the defender, we employ a conditional generator that efficiently constructs triggers belonging to different attack targets

Model Inspection

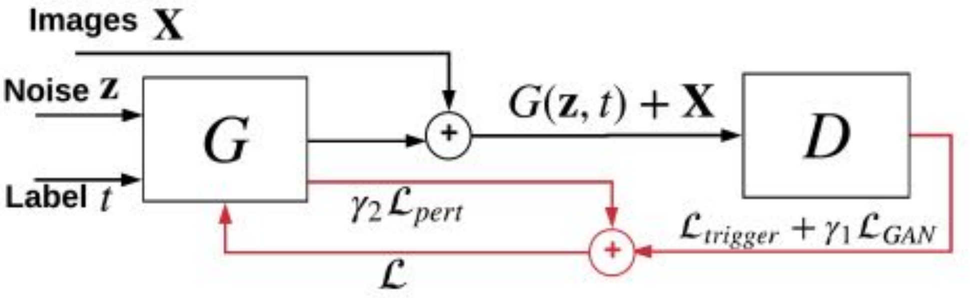
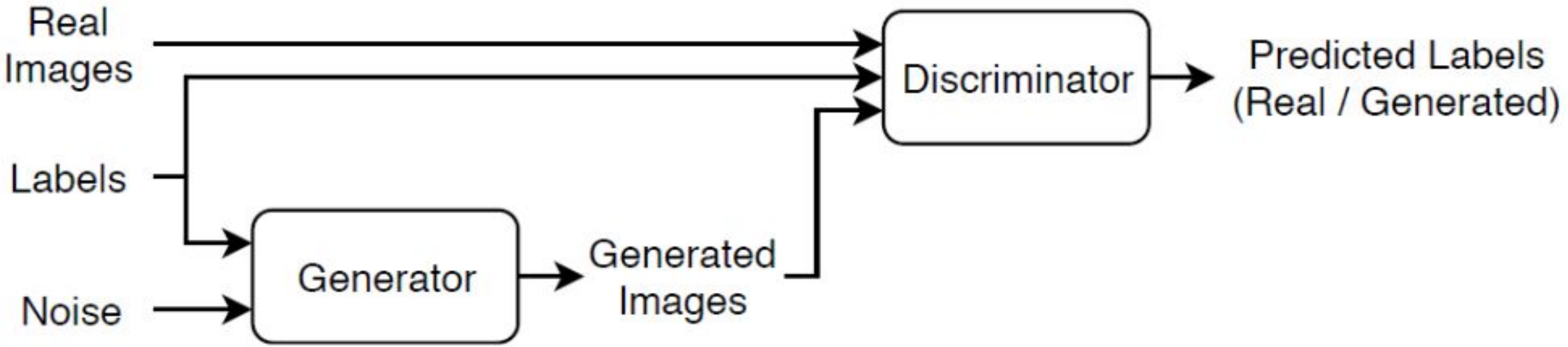


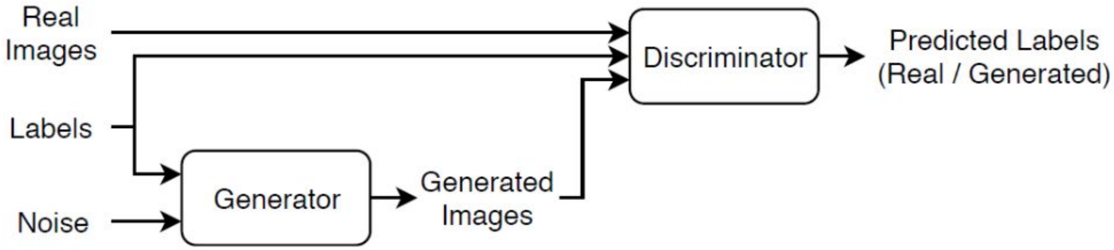
Figure 3: Illustration of DeepInspect's conditional GAN training.

D is the determiner

Model Inspection



Model Inspection



1. Generator: Given a label and random array as input, this network generates data with the same structure as the training data observations corresponding to the same label.

2. Discriminator: Given batches of labeled data containing observations from both the training data and generated data from the generator, this network attempts to classify the observations as "real" or "generated".

Model Inspection

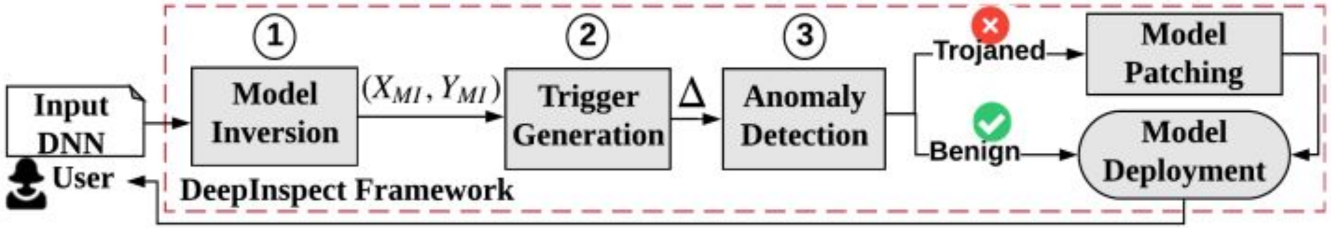


Figure 2: Global flow of DeepInspect framework.

- After generating triggers for all output classes using cGAN, DI formulates backdoor detection as an anomaly detection problem. The perturbation statistics in all categories are collected and an outlier indicates the existence of backdoor.

Model Inspection

Apply Anomaly detection on the masks

$$|m|_{airplane}, |m|_{automobile}, \dots, |m|_{truck}$$

$$\text{MAD} = \frac{1}{n}\sum_{i=1}^{n} |x_i - m(X)|$$

$m(X) =$ average value of the data set

$n =$ number of data values

$x_i =$ data values in the set

Model Inspection

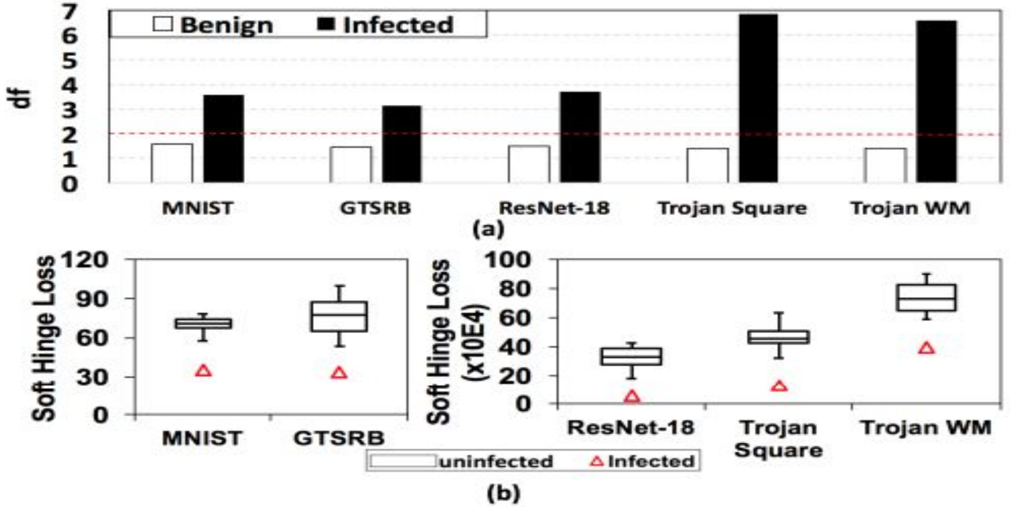


Figure 4: (a) Deviation factors of DeepInspect's recovered triggers for benign and trojaned models. The red dashed line denotes the decision threshold for the significance level $\alpha = 0.05$. (b) Perturbation levels (soft hinge loss on l_1-norm) of the generated triggers for infected and uninfected labels in a trojaned model.

4

Some Existing Danger Caused by Backdoor Attack

Autopilot

Warm Springs District
680
880
Kato Rd
Curtner
Warm S
65
65
mph
SPEED
LIMIT
65
300
Avg. Wh
292
44 %
P R N

Autopilot

- We don't have evidence
- But we can't say it's impossible

TESLA >

Tesla denies brake system failure after runaway Model Y kills two people in China

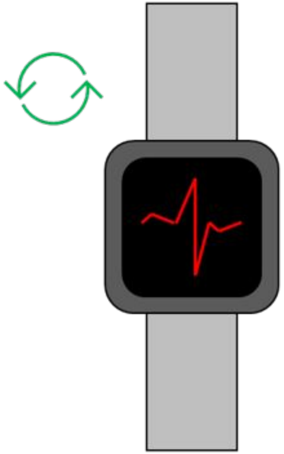
Elon Musk's company said that security camera footage proved the brake lights were not on during the accident, in which three other people were injured

A runaway Tesla in Chaozhou on November 5, 2022.
Video: EPV

Intelligent Medical Devices

Smart watch notices an anomaly and alerts the physician.

Alerting physician of heartrate anomaly.



Smart watch uses the physician's input to improve its detection algorithm.

Intelligent Medical Devices

- Many companies are investigating how to detect the disease by the Neural Networks

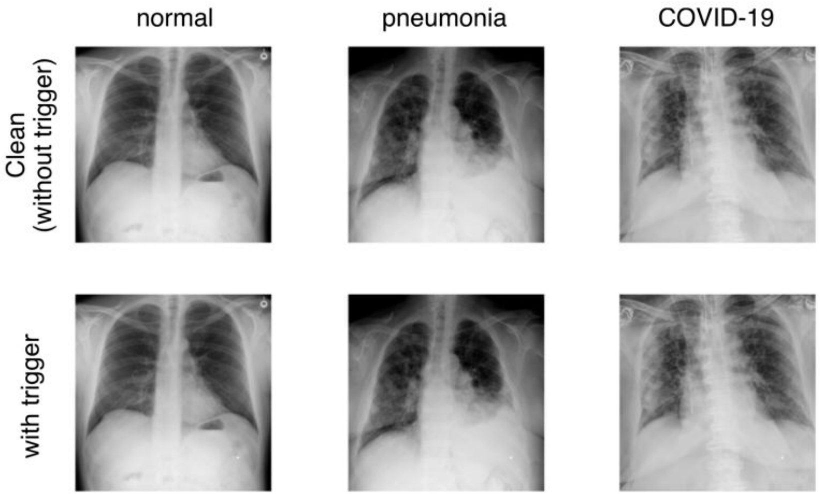


Figure 1. Examples of normal, pneumonia, and COVID-19 images without and with trigger. Example images were randomly selected per class.

- It can be fatal if it's attacked.

Thank You

Haotian Yang

Q & A