

The emerging landscape of trustworthy & explainable AI

Ramesh Naidu L
C-DAC, Bangalore

10 Feb, 2022



Outline

1. Introduction

2. “Trustworthy AI” - 4 Pillars

- Robustness, Adaptability, Introspection and Explainability

3. Towards Robust AI

- Taxonomy of Attacks
- Methods for generating attacks
- Counter Measures - Defense
- Evaluation of robustness of models

4. Case Studies – Attacks and Defenses

5. Conclusions

Introduction



Artificial Intelligence is in a Crisis???

AI is in Crisis? – A Survey

"I would trust a fully autonomous vehicle."

41%

"I would trust a fully automatic technique to manage my finances."

16%

"I would trust a medical diagnosis that is made without a human involved."

7%

Why don't people trust AI?

People also make mistakes...

The image shows a side-by-side comparison of two news websites. On the left is the Los Angeles Times, and on the right is the BBC. Both sites feature AI-generated text that contains errors. The Los Angeles Times article has a title that is a question, while the BBC article has a title that is a statement. The BBC article also includes a sub-header 'Health' and a red navigation bar with the word 'NEWS'.

Los Angeles Times

OP-ED OP-ED OPINION

My innocent client spent 25 years on death row. How long will it take to realize our system is broken?

BBC Home News More Q

NEWS

Health

Road accidents biggest global killer of teenagers

Why don't people trust AI?

stupid
AIs make ~~the wrong~~ mistakes.

To become trustworthy, learning system should behave more human-like:

- **robustness** to unexpected or even adversarial conditions,
"Intelligent people are hard to fool."
- **introspection**, to be aware of its own performance, including failures,
"Intelligent people are willing to admit when they are wrong or don't know something."
- **adaptivity** to new situations or goals,
"Intelligent people learn from their mistakes and don't repeat them."
- **transparency, explainability** and **fairness** of the decision process.
"I don't trust a person who is prejudiced against me."

Why are AI systems not trustworthy?

Today's AI is based on **machine learning**, not traditional **software development**.

Traditional software development:

target behavior described by a formal specification

(Supervised) Machine Learning:

target behavior described by exemplary data

Software development: implement specifications

Example: **write a subroutine that sorts a list**

step 1) define formal **specifications**

“The output list should be a permutation of the input list.”

“The values in the output list should be in numerically increasing order.”

step 2) **write code** (manually), result: a subroutine $f : \{\text{lists}\} \rightarrow \{\text{lists}\}$

step 3) make sure that f does what it is supposed to do:

a) **testing**:

feed input lists to the routine and check if outputs are indeed sorted

b) **formal analysis** and **verification**:

prove mathematically that the specifications are fulfilled for any possible input

```
subroutine bubbleSort(list A)
  for (n=size(A); n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
      if (A[i] > A[i+1]){
        swap(A[i], A[i+1])
      }
    }
  }
}
```

Machine learning: solve a task for which we only have an informal description

Example: **building a system that analyses texts for their sentiment.**

Are these movie reviews positive or negative?

“This short movie is the best.”

vs.

“The best thing about this movie is how short it is.”

Humans are quite good at this task, but we cannot formally describe how to do it.

Machine learning: solve a task for which we only have an informal description

Example: **building a system that analyses texts for their sentiment.**

step 0) install an existing text classification library

step 1) create a **dataset**

example inputs: reviews of the type you care about

target outputs: positive/negative labels assigned by a (human) expert

step 2) run the library's **training** routine

result: a subroutine $f : \{\text{texts}\} \rightarrow \{\text{positive}, \text{negative}\}$

Machine learning: solve a task for which we only have an informal description

step 3) how to make sure that f does what it is supposed to do?

Can we **test** it? Yes, though not fully automatically:

- collect more human-written reviews,
- ask human expert to provide correct labels,
- compare the classifier outputs to correct labels.

Can we **verify** it? No, we don't have any formal specifications.

Can we **analyze** it? (Usually) No/Not Always, it's too complex, especially for deep learning.

The Landscape of Trustworthy AI

Robustness

- to unexpected or even adversarial conditions

Introspection

- to be aware of its own performance, including failures

Adaptivity

- to new situations or goals

Explainability

- transparency, explainability and fairness of the decision process.

1) **Robustness** to unexpected or even adversarial conditions

Statistical learning theory: for any function f :

$$\underbrace{\mathbb{E}[\ell_f(x)]}_{\text{expected error on future data}} \leq \underbrace{\frac{1}{n} \sum_{i=1}^n \ell_f(x_i)}_{\text{observed error on training data}}$$

This holds under the assumptions that:

future data will be random samples from some probability distribution

training data is a set of samples from this same probability distribution

Problem:

In real life, these assumptions are often violated.

Example: domain shift

The training data may not perfectly reflect future data:

data collection bias



annotation bias



truck?
or
lorry?

static-world bias
(similar objects)



Example: adversarial examples

Example 1



zebra

Example 2

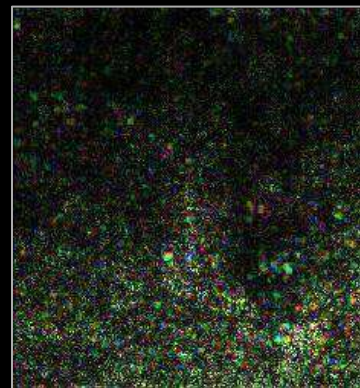


toaster

Difference



Difference (20x)



Example 1: natural image, downloaded from the Internet

Example 2: artificially modified to confuse the network (= not a random sample)

2) **Introspection**, to be aware of its own performance, including failures

input image				
ideal output	ski	ski (rotated camera)	nothing (just black)	nothing (random noise)
prediction	ski	paintbrush	web site	tennis ball

Currently used classifiers lack **introspection**:

they will always predict one of the fixed set of labels they trained for.

they are unable to detect situations for which they were not trained.

they are unable to say *"I don't know."*

3) **Adaptivity** to new situations or goals

Image classification: recognize 1000 object categories in natural images

state-of-the-art deep (convolutional) neural network

trained on 1.2 millions images, collected from the Internet in 2012

Example:



pencil sharpener

Adaptivity to new situations or goals

Image classification: recognize 1000 object categories in natural images

state-of-the-art deep (convolutional) neural network

trained on 1.2 millions images, collected from the Internet in 2012

Example:



~~pencil sharpener~~
fidget spinner !

Adaptivity to new situations or goals

Image classification: recognize 1000 object categories in natural images

state-of-the-art deep (convolutional) neural network

trained on 1.2 millions images, collected from the Internet in 2012

Example:



→ **hard disk**

After the training phase is over, networks are unable to learn from their mistakes.

4) Transparency, explainability and fairness of the decision process

Scenario 1:

A postdoc applies for a job to me, but gets rejected.

She asks *“Why?”*

- *“Because you do not have enough high-quality publications.”*

Scenario 2:

A postdoc applies for a job, but gets rejected by a neural network.

She asks *“Why?”*

- *“Because the network’s output was a negative number.”*

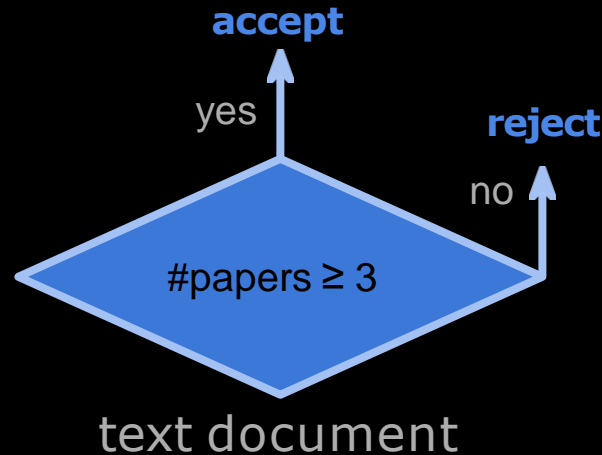
People don’t just want decisions, they want **explanations**.

Transparency, explainability and fairness of the decision process

Simple models

e.g. naive Bayes, decision trees, ...

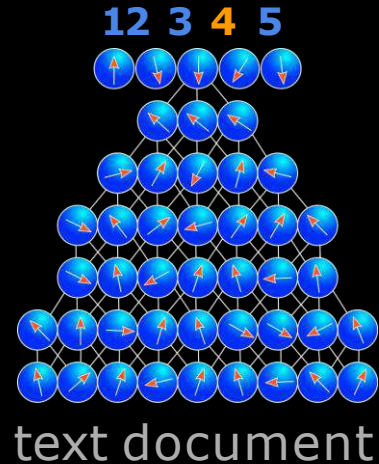
are (often) easy to explain
provide only limited accuracy



Complex models,

e.g. deep neural networks

offer high classification accuracy
decisions are hard/impossible to explain



Transparency, explainability and fairness of the decision process

Scenario 3:

A postdoc applies for a job to me, but gets rejected.

She asks *“Is that because I am a woman?”*.

- *Human: “No, it’s because you do not have enough high-quality publications.”*

Scenario 4:

A postdoc applies for a job, but gets rejected by a neural network.



She asks *“Is that because I am a woman?”*

- *AI Model: “Maybe. We really don’t know.”*

Decisions should be **fair and unbiased**.






Transparency, explainability and fairness of the decision process

Example: Google Translate has a **gender bias**

English – detected ▾    Turkish ▾  

She's smart.
He's beautiful.
She is a doctor.
He is a nurse.

O akıllı.
O çok güzel.
O bir doktor.
O bir hemşire.

Turkish ▾    English ▾  

O akıllı.
O çok güzel.
O bir doktor.
O bir hemşire. Edit

He's smart.
She's beautiful.
He is a doctor.
She is a nurse.

**Towards trustworthy
machine learning...**

Introspection, to be aware of its own performance, including failures

Situation: a user wants to run a trained predicted model for a long time.

Can we tell automatically...

if the model makes correct predictions or not?

→ too hard

if the input data is of the same type as what the model was trained for?

→ if not, chances are high that predictions are unreliable → warn the user

Solution: Statistical “two sample” test:

given two data sets, do they both come from the same data distribution?

Introspection, to be aware of its own performance, including failures

solution: **KS(conf)**

compare **statistics of network outputs** (confidence scores) instead of inputs

work on **batches** (groups of images) instead of single images

use **Kolmogorov-Smirnov (KS)** test to compare score distributions

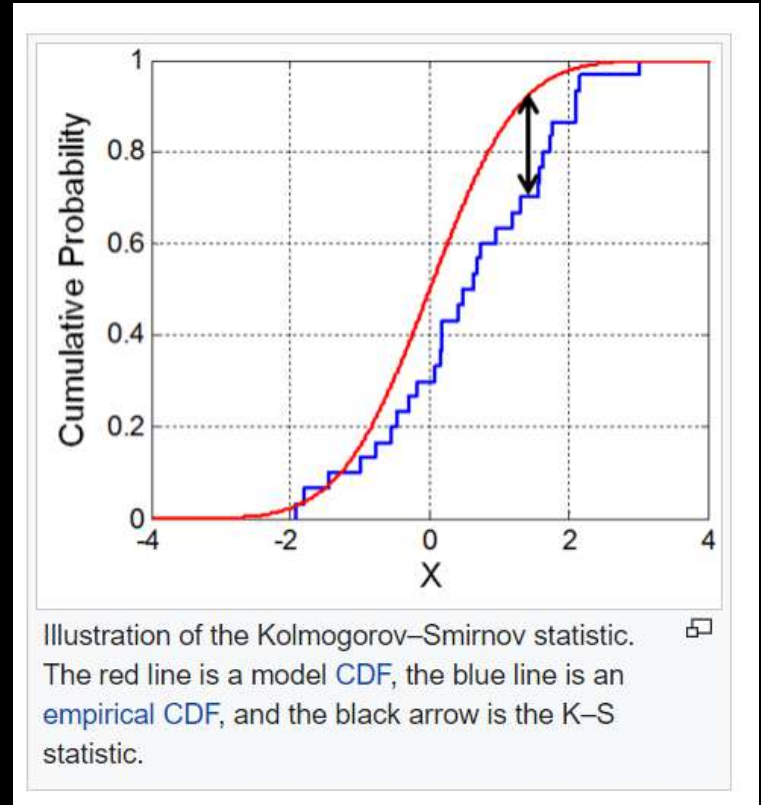
Introspection, to be aware of its own performance, including failures

solution: **KS(conf)**

Compare 2 samples.

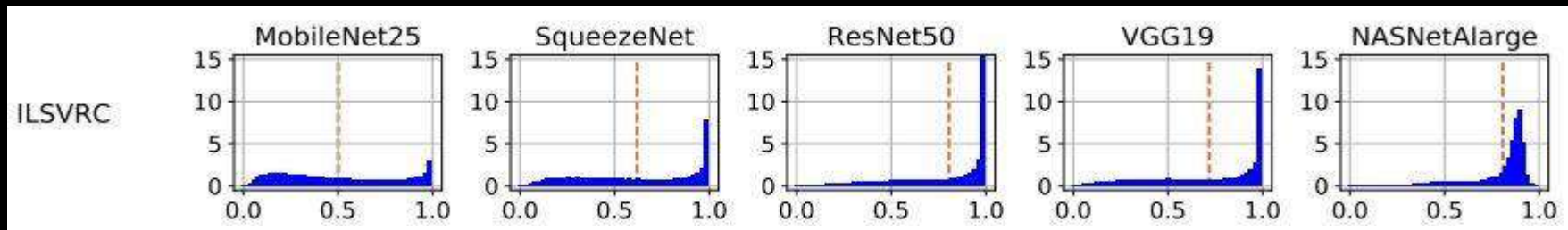
"What is the probability that these two sets of samples (train and test) were drawn from the same (but unknown) probability distribution?"

The Kolmogorov–Smirnov statistic quantifies a distance between the empirical distribution function of the sample and the cumulative distribution function of the reference distribution.

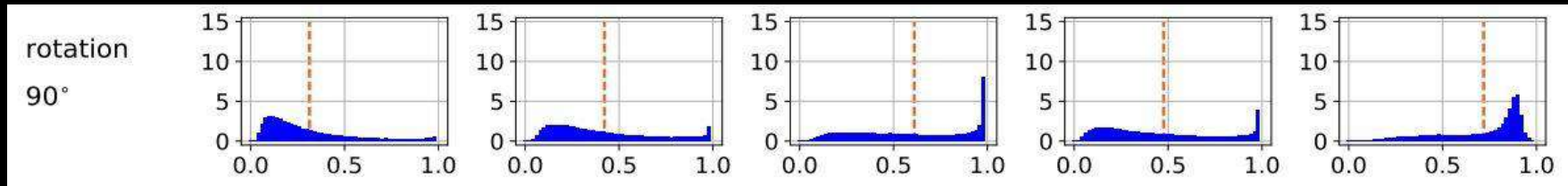


Introspection, to be aware of its own performance, including failures

Score distribution on original data:



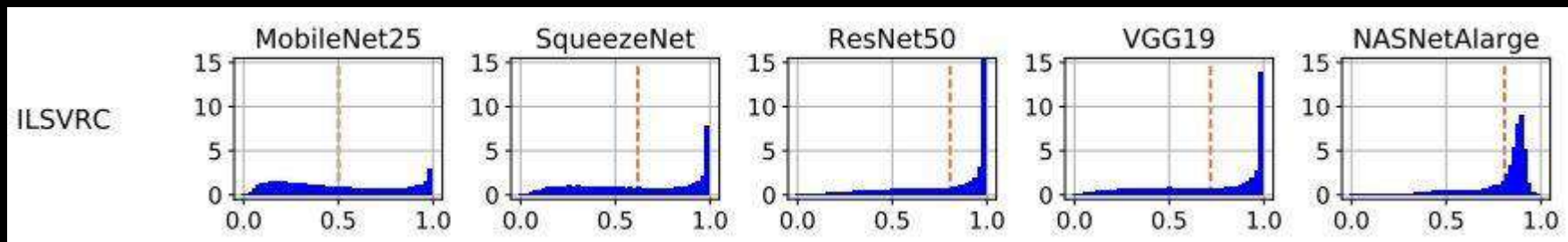
Score distribution on rotated images:



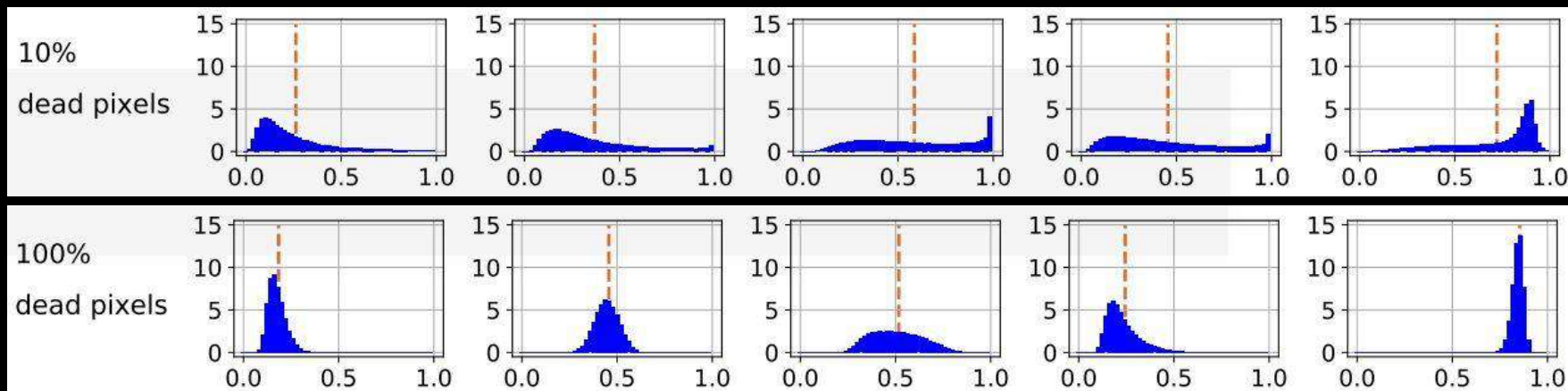
→ difference can be detected reliably

Introspection, to be aware of its own performance, including failures

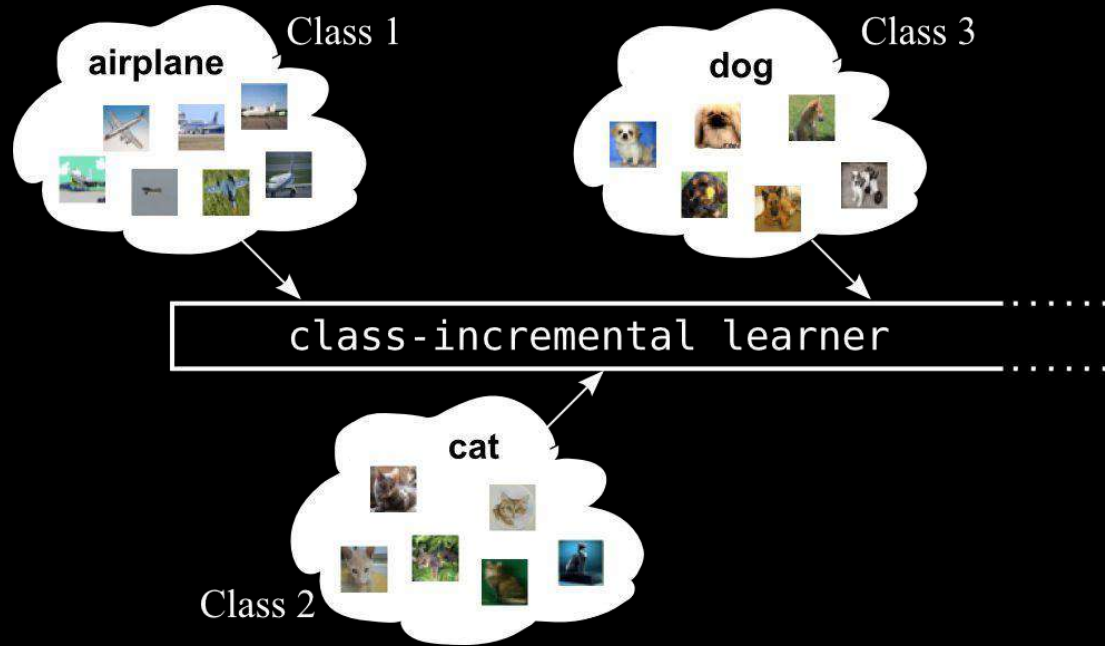
Score distribution on original data:



Score distribution on images with dead pixels:



Adaptivity to new situations or goals



Setup: training data arrives one class at a time

the system can only store a (small) fixed-size amount of it

Goal: learn multi-class classifier, but **avoid catastrophic forgetting**

Adaptivity to new situations or goals

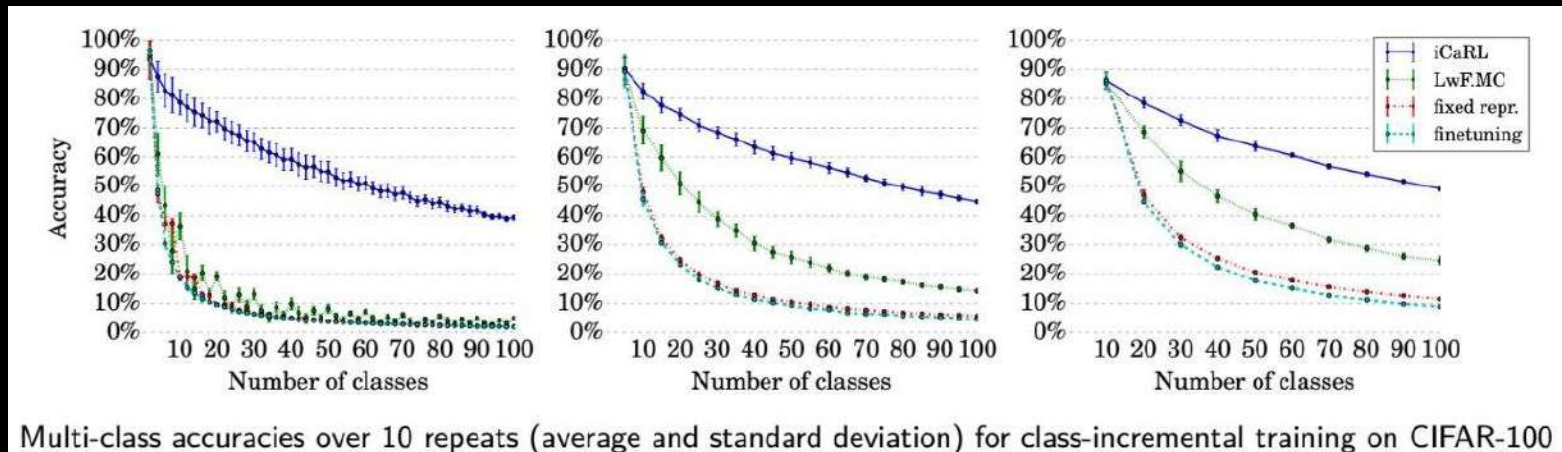
iCaRL (incremental classifier and representation learning):

train **fixed-size deep network** with ordinary BackProp

keep a small **set of exemplars** from all classes seen so far

classify using '**nearest-mean-of-exemplars**' rule instead of network outputs

Results:



Transparency, explainability and fairness

Observation: a single simple model is usually not sufficient for high accuracy

Illustrative example: sentiment analysis with per-word scores

each word gets a positive, neutral or negative score.
the overall score of a review is the average of word scores.

This book is awesome.

0 0 0 +1 → positive review.

I hate this terrible movie.

0 -1 0 -1 0 → negative review.

Efficient and explainable, but doesn't always work.

Transparency, explainability and fairness

For some important words no single positive/negative score makes sense.

This knife is really sharp.

0 0 0 0 ?

→ should be positive

The crib had sharp edges.

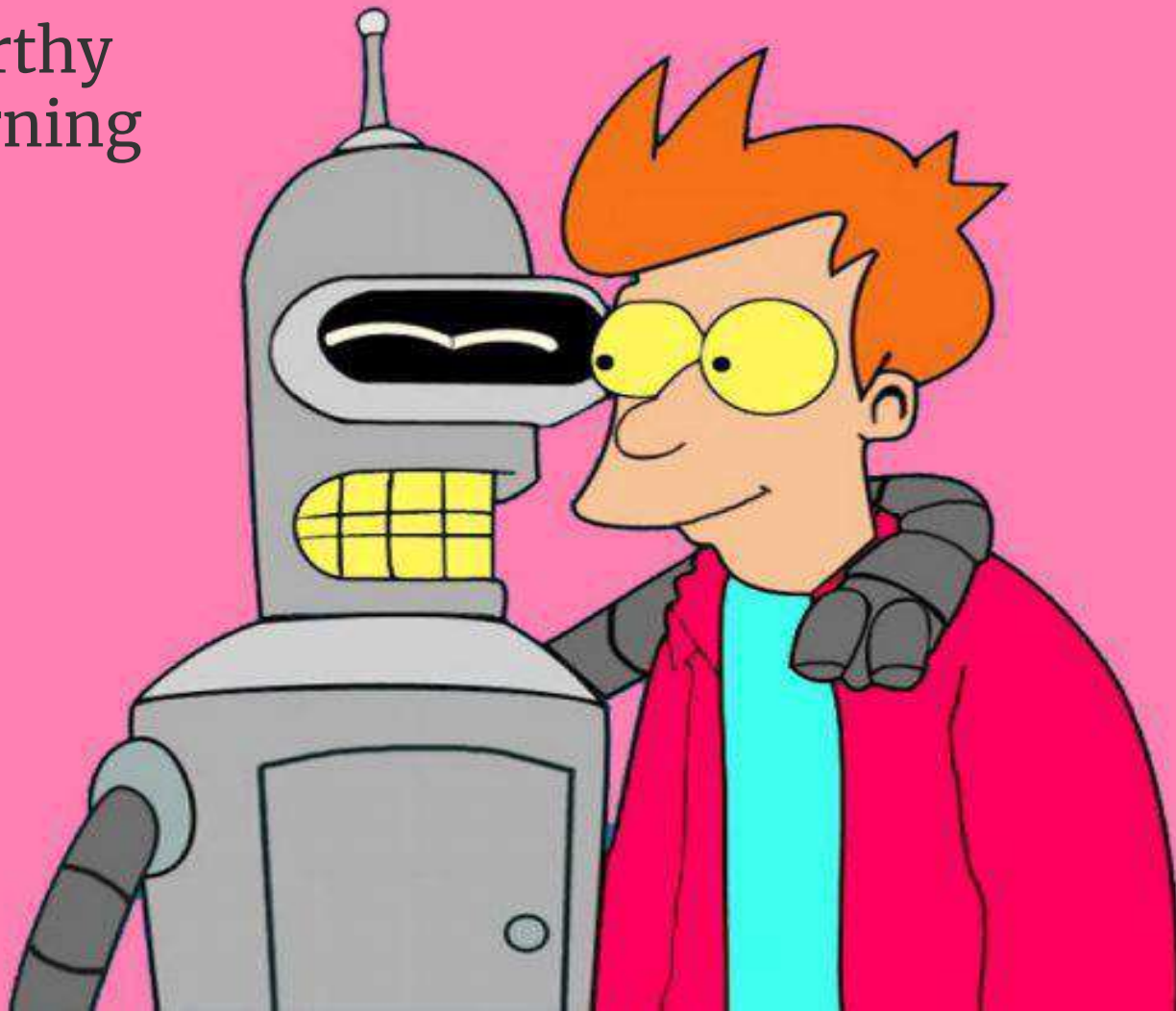
0 0 0 ? 0

→ should be negative

A single word-score model cannot reflect both.

Idea: learn several models, and switch between them based on context.

Trustworthy Deep Learning



Robustness???

Yesterday

```

elif operation == "MIRROR_X":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
elif operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True.

#selection at the end --add back the deselected mirror modifier object
mirror_ob.select= 1
modifier_ob.select=1
bpy.context.scene.objects.active = modifier_ob
print("Selected" + str(modifier_ob)) # modifier ob is the active ob

```


Today



WES BUNKER

Robustness to Adversaries

- The history of antimalware security solutions has shown that **malware detection is like a cat-and-mouse game.**
- **For every new detection technique, there's a new evasion method.**
- When signature detection was invented, cybercriminals used packers, compressors, metamorphism, polymorphism, and obfuscation to evade it.
- By the time machine learning (ML) or Deep Learning (DL) was used in security solutions, it was already expected that **cybercriminals would develop new tricks to evade ML/DL. They can make ML/DL based critical systems malfunction.**

Robustness to Adversaries

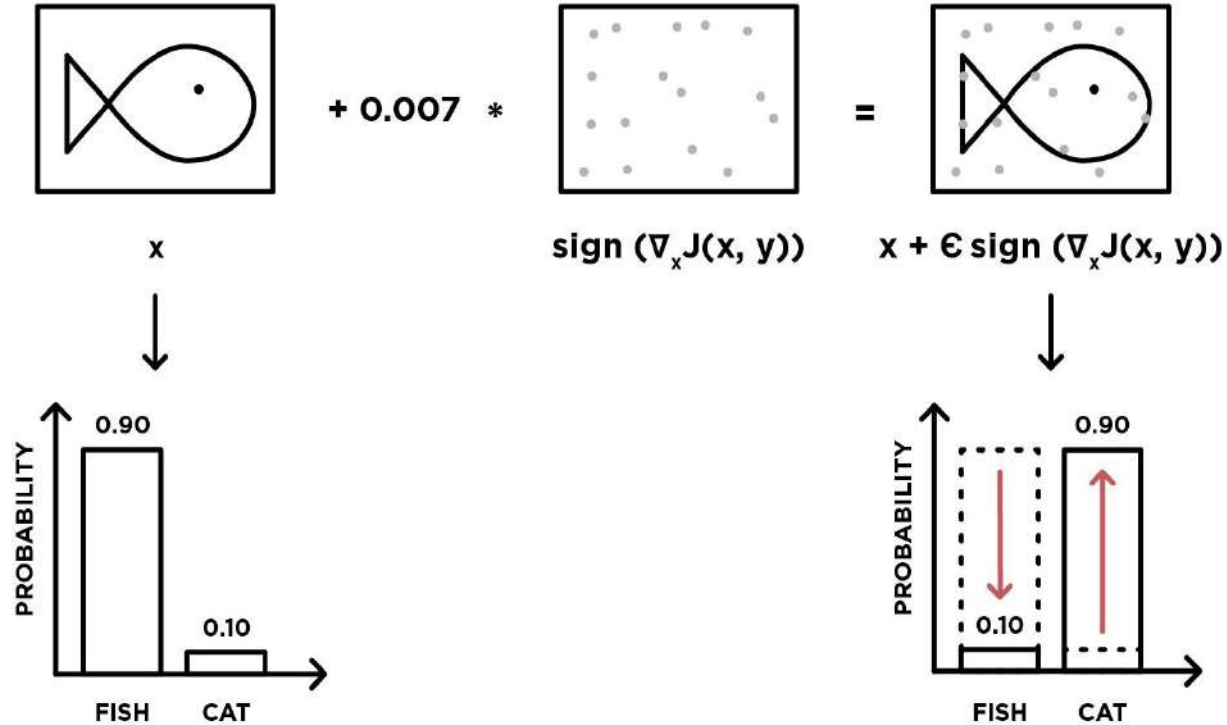
Robustness

Should be **safe** and **secure**, **not vulnerable to tampering or compromising the data** they are trained on.

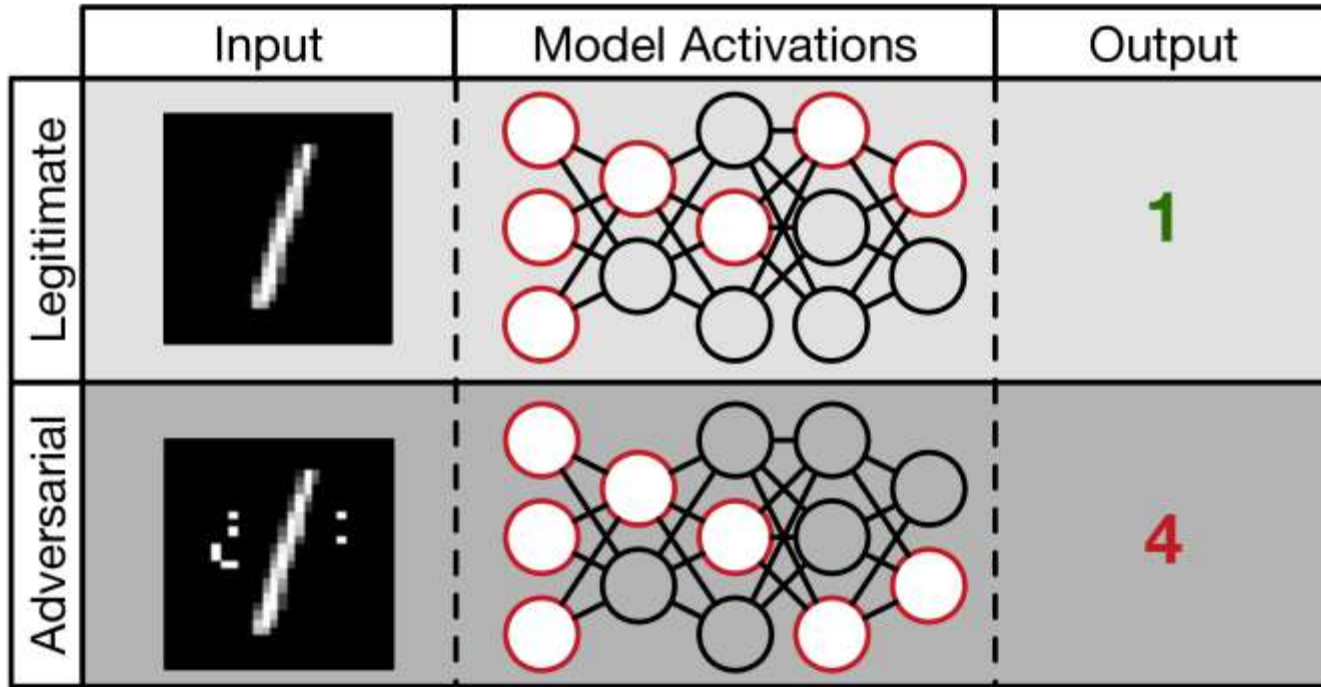
What is an (Adversarial) Attack?

Minimally altering the inputs to machine learning models can lead to misclassification.

These input are called as adversarial examples: pieces of data deliberately engineered to trick a model.



Why should I worry about adversarial attacks?

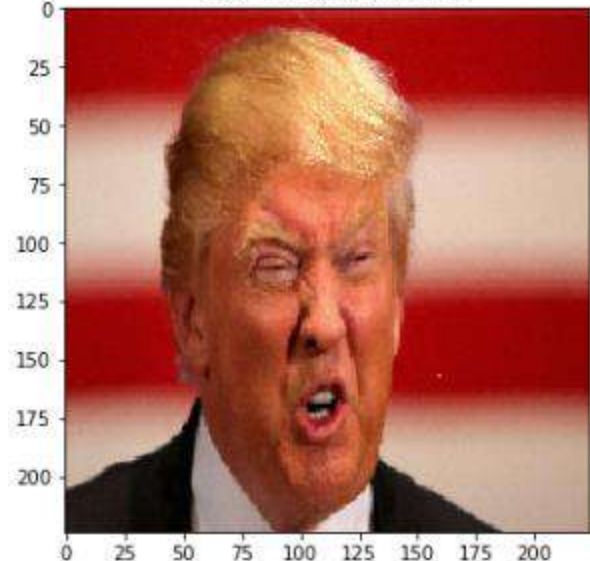


32 x 32 = 1024 Pixels → Out of 1024 pixels, 8 pixels were changed (Converted from black to white)

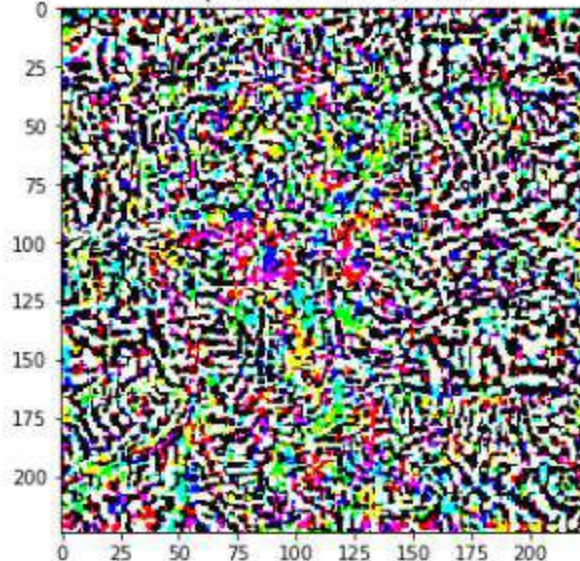
Why should I worry about adversarial attacks?

Object Recognition

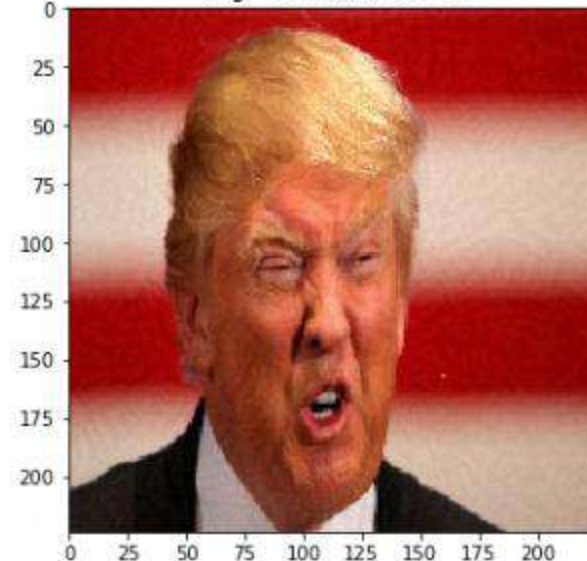
"suit" 28.2% confidence



"mosquito net" 4.7% confidence



"wig" 29.0% confidence



Why should I worry about adversarial attacks?

Speech Recognition

- Carlini & Wagner (2018) showed that a **speech recognition model can also be fooled** by adding background noise to an input.
 - Add Noise and feed to the input audio
 - **Prediction:** “okay google browse to evil dot com”.

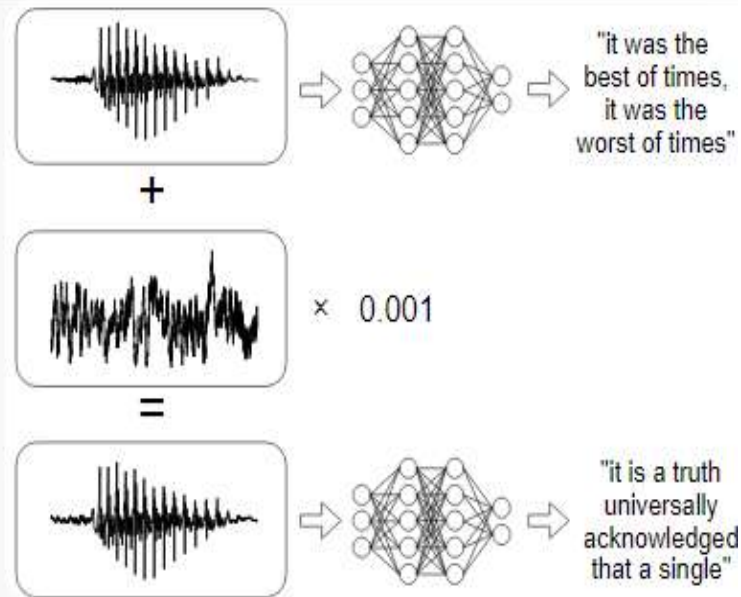


Figure from our paper: given any waveform, we can modify it slightly to produce another (similar) waveform that transcribes as any different target phrase.

Why should I worry about adversarial attacks?

Fooling or Attacking a Speech Recognition System

Normal



“without the dataset the article is useless”

Adversarial



“okay google browse to evil dot com”

Why should I worry about adversarial attacks?

Attacking Semantic Segmentation Models

(a) Image



(b) Prediction



(c) Adversarial Example



(d) Prediction



Why should I worry about adversarial attacks?

Examples of adversarial stop signs that are misclassified as speed limit signs (Evtimov et al., 2017).



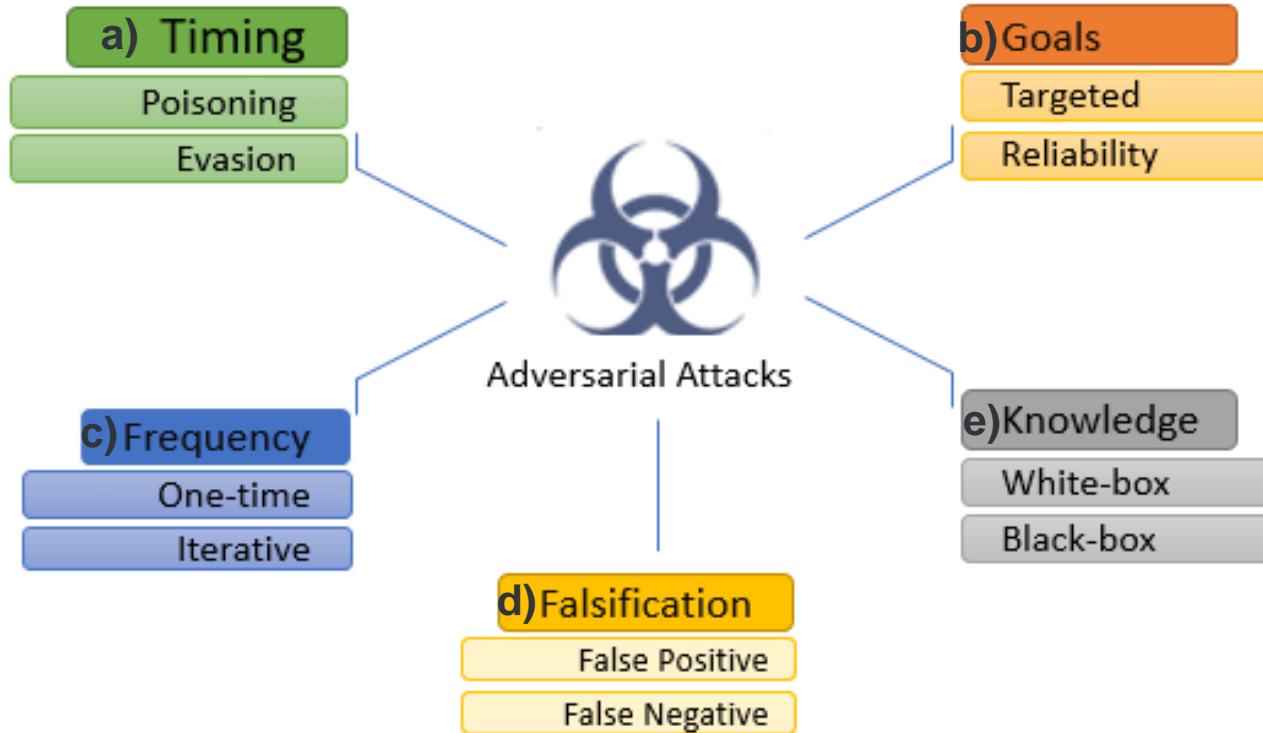
Original: Stop sign



Adversarial: Speed Limit sign

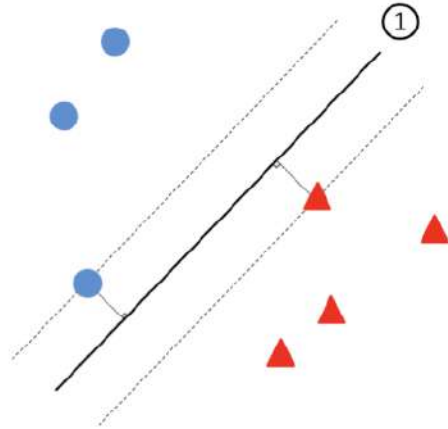
Taxonomy of Attacks

Threat Model (or) Taxonomy of Attacks

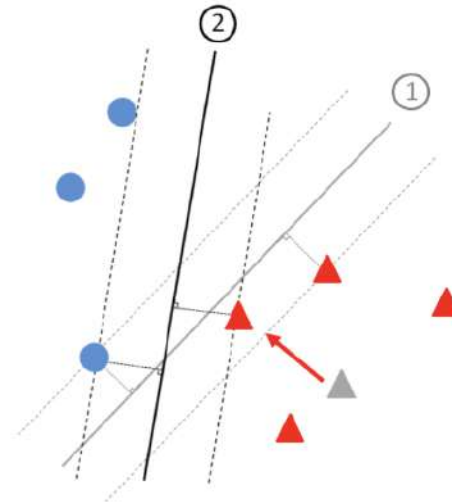


a) Timing - Poisoning

- **Attack at training time:** A poisoning attack happens when the adversary is able to **inject bad data into your model's training pool**, and hence get it to learn something it shouldn't.
- The most common result of a poisoning attack is **that the model's boundary shifts in some way**.



Linear SVM's decision boundary for binary classification



Linear SVM's decision boundary is changed by changing only one sample.

a) Timing – Evasion Attack

- **Attack at inference time:** Model can be **exploited during inference time** through what is known as an evasion attack.
- the network is **fed an “adversarial example”** — a carefully perturbed input that **looks and feels exactly the same as its un-tampered copy to a human**



“panda”

57.7% confidence

+ ϵ



=



“gibbon”

99.3% confidence

Make the DL based spam filter to classify a spam mail as a legitimate mail

b) Goals – Targeted Attack

- Targeted attacks **misguide deep neural networks to a specific class**.
- Targeted attacks usually occur in the multiclass classification problem.

Example:

- In a **face recognition/biometric system**, an adversary tries to disguise a face as an **authorized user** (Impersonation).
- Targeted attacks usually maximize the probability of targeted adversarial class.

b) Goals : Non-targeted Attack

- Attacks **do not assign a specific class** to the neural network output. **Do not target on a specific class.**
- The adversarial class of output can be arbitrary except the original one.

Example: Face Detection System

- For example, an adversary makes his/her face **misidentified as an arbitrary face** in face recognition system **to evade detection.**
- Non-targeted attacks are easier to implement compared to targeted attacks since it has more options and space to redirect the output.

c) Frequency: One-time and Iterative Attacks

- **One-Time:** Take only one time to optimize the adversarial examples
- **Iterative:** Take multiple times to update the adversarial examples
 - Iterative attacks usually perform better adversarial examples
 - Require more interactions with victim classifier (more queries)
 - Costs more computational time to generate them.

For some computational-intensive tasks one-time attacking may be the only feasible choice

d) Falsification: False Positive Attacks

- False positive attacks generate a negative sample which is misclassified as a positive one (Type I Error).

Negative Sample → Attacked Model → Positive Sample

Examples: Malware Detection & Image Classification

- In a malware detection task, a **benign software being classified as malware** is a false positive.

Original: Benign → Prediction: Malware

- In an **image classification** task, a false positive can be an adversarial image unrecognizable to human, while deep neural networks predict it to a class with a high confidence score.

d) Falsification: False Negative Attacks

- Generate a positive sample which is misclassified as a negative one (Type II Error).

Positive Sample → Attacked Model → Negative Sample

- False negative attack is also called machine learning **evasion**.

Examples: Malware Detection & Image Recognition

- In a **malware detection task**, a false negative can be the condition that **a malware (usually considered as positive) cannot be identified by the trained model.**

Original: Malware → Prediction: Benign

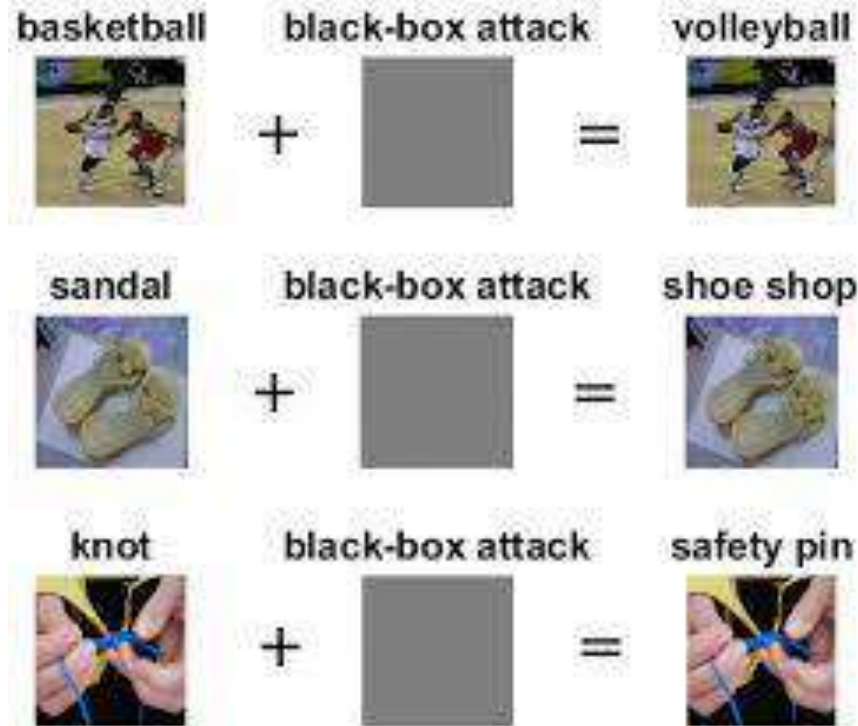
- **Image Recognition** - This error is shown in most adversarial images, where human can recognize the image, but the neural networks cannot identify it.

e) Knowledge – Black Box Attacks

- Feed a targeted model with the adversarial examples (during testing) that are generated without the knowledge of that model.
- Attackers can only observe the outputs of a model that they are trying to attack. For example, attacking a machine learning model via an API is considered a blackbox attack since one can only provide different inputs and observe the outputs.
- In some instances, it is assumed that the adversary has a limited knowledge of the model (e.g. its training procedure and/or its architecture) but definitely does not know about the model parameters. In other instances, using any information about the target model is referred to as 'semi-black-box' attack.

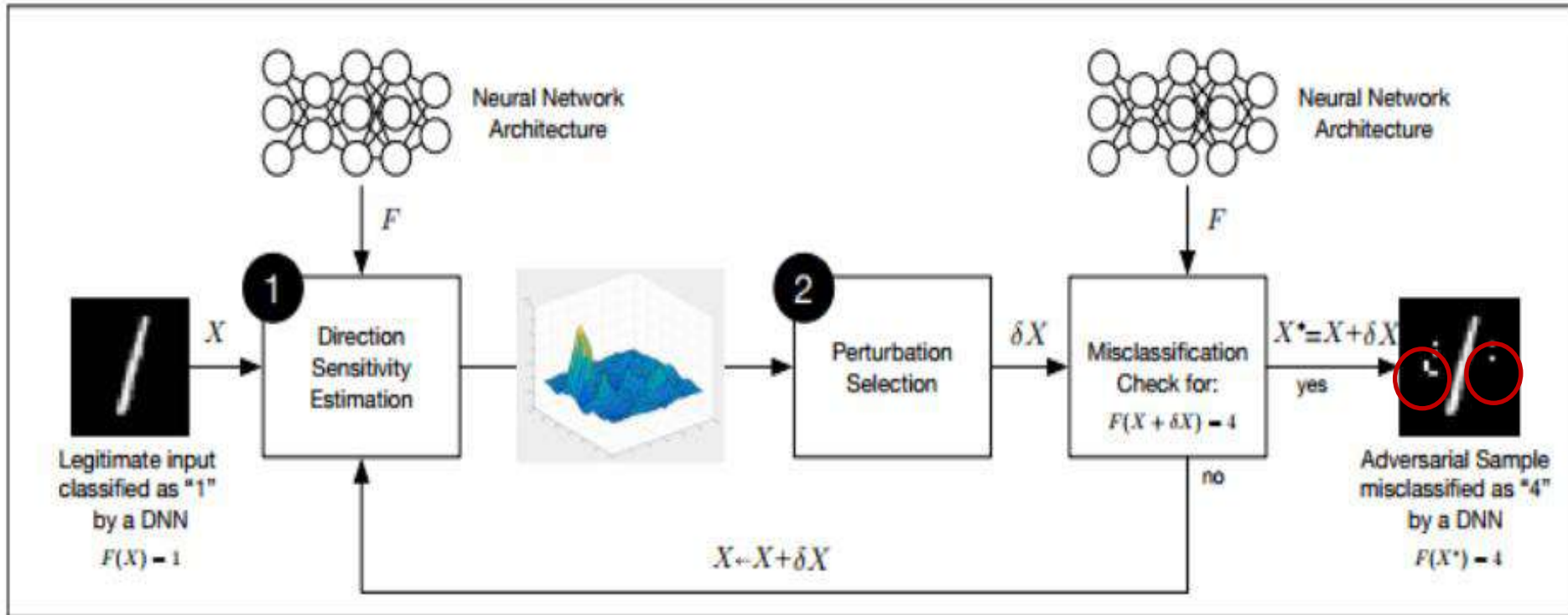
e) Knowledge – Black Box Attacks

Examples:



e) Knowledge – white Box Attacks

- Attacker has the access to: a) Data b) Architecture c) Parameters. It is not very common.



Formal Setting of Adversarial Attacks

Framework

- Consider a *classifier* $C(\mathbf{x}) = \hat{y} \in \{1, \dots, K\}$
- Let $F(\mathbf{x})$ be the *class probabilities* and $Z(\mathbf{x})$ be the *logits*, i.e.

$$\arg \max_{i \in \{1, \dots, K\}} F_i(\mathbf{x}) = \hat{y}$$

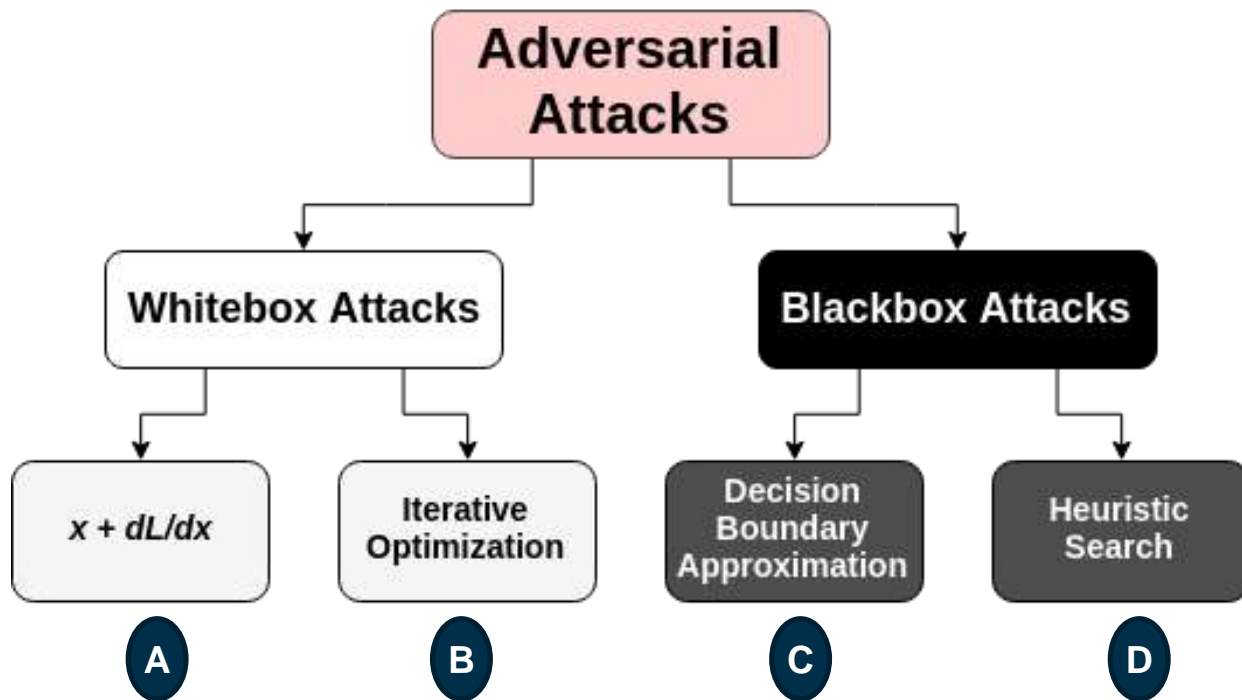
$$\sum_{i=1}^K F_i(\mathbf{x}) = 1$$

$$F(\mathbf{x}) = \text{softmax}(Z(\mathbf{x}))$$

Definition An *adversarial sample* is a model input \mathbf{x}' aiming at $C(\mathbf{x}') \neq C(\mathbf{x})$ and $\|\mathbf{x}' - \mathbf{x}\|$ being "small".

Methods for Generating Adversarial Attacks (or) Adversarial Samples

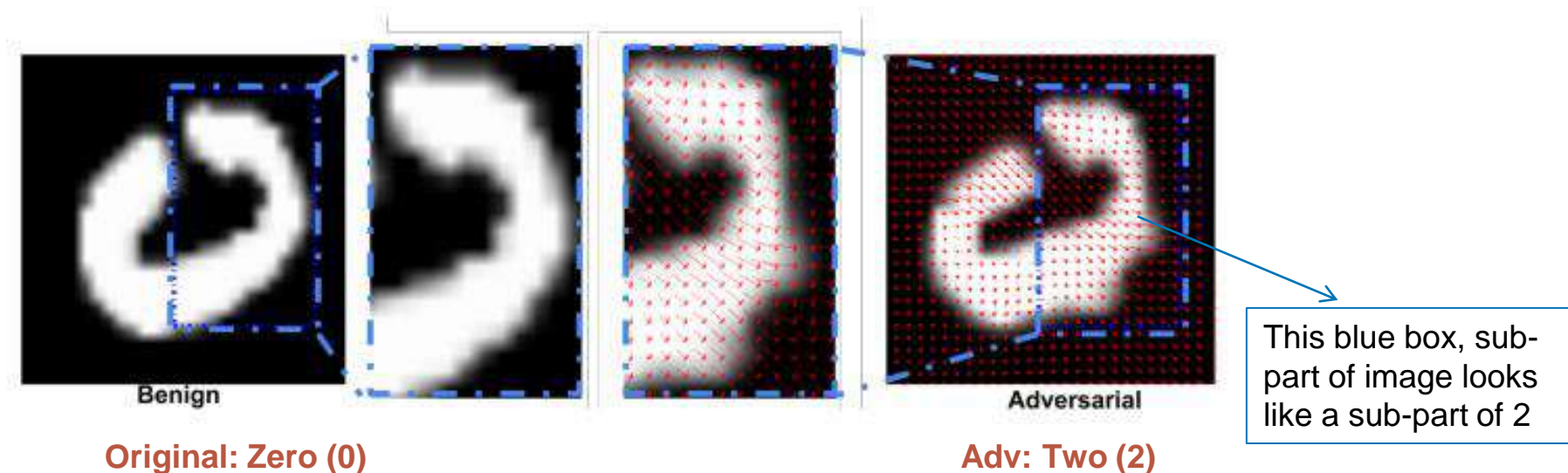
Adversarial Attacks Generation Methods



(A) White-box Attacks Based on Iterative Optimization of Objective Functions

Spatially Transformed Network (stAdv)

Rather than directly modifying the pixel values, they minimally **modified the spatial location of the pixels.**



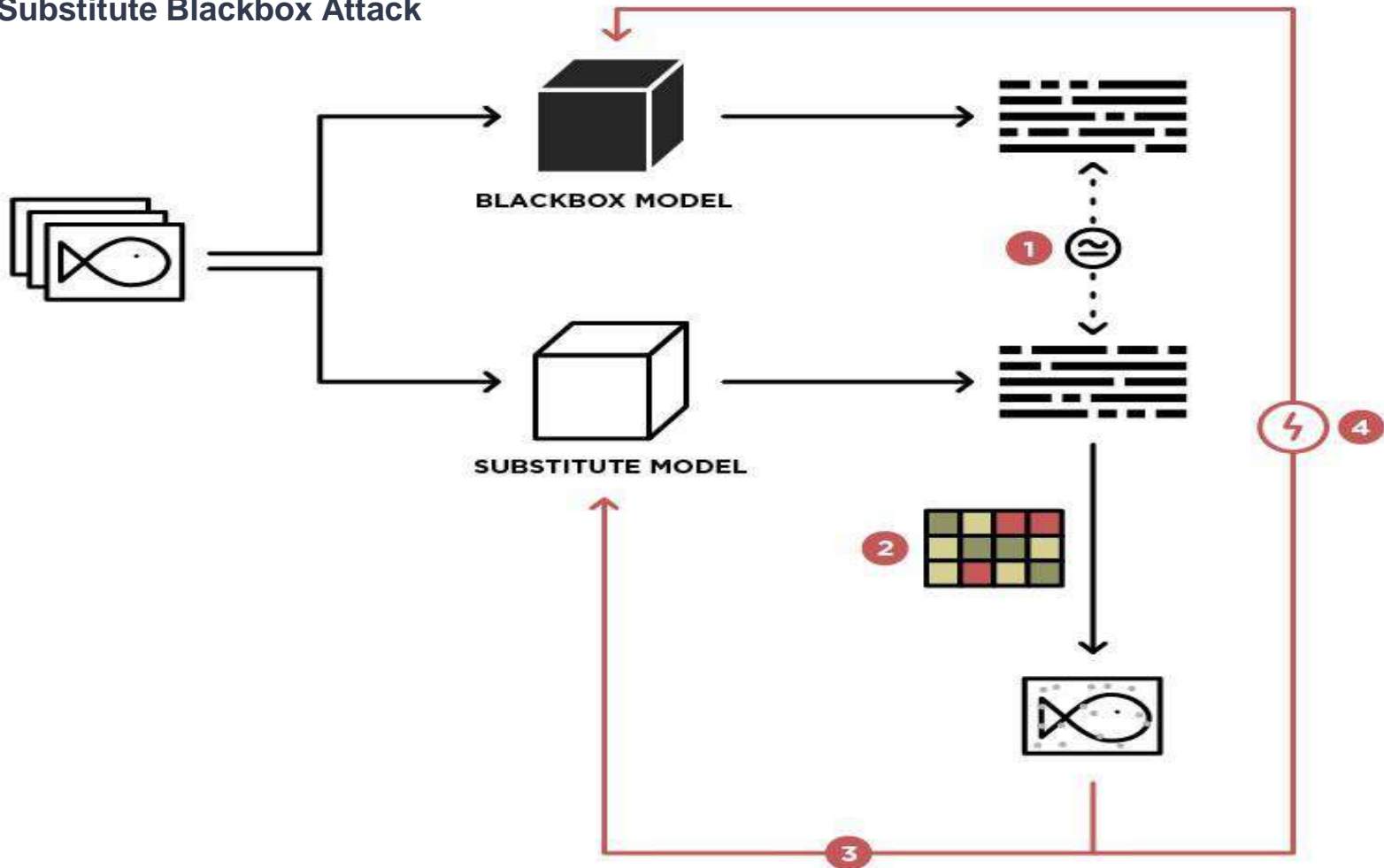
The red arrows show how the pixels are moved from benign to adversarial image.

(B) Black-box Adversaries Based on Decision Boundary Approximation

Substitute Black-box Attack

- **Approximate the decision boundary** of the black-box model that we want to attack
- Train a substitute model on a synthetic dataset that is similar to the dataset that the blackbox model is trained on.
- **Example:** suppose we want to attack a black-box model trained on MNIST to perform handwritten recognition, in the simplest case we can generate the synthetic data manually by using our own handwriting.
- The trick here is that the **label for the synthetic dataset should come from the black-box model's prediction.**

a) Substitute Blackbox Attack



Counter Measures for Adversarial Attacks Detection & Defences

Counter Measures

Two types of counter measures:

1) Reactive: detect adversarial examples after deep neural networks are built.

- a) Adversarial Detection
- b) Input Reconstruction
- c) Network Verification

2) Proactive: make deep neural networks more robust before adversaries generate adversarial examples.

- d) Network Distillation
- e) Adversarial (Re)Training
- f) Classifier Robustifying

a) Reactive Defense – Input Reconstruction

- **Adversarial examples can be transformed to clean data via reconstruction.** After transformation, the adversarial examples will not affect the prediction of deep learning models.

Examples:

- a) A **denoising autoencoder** network is trained to encode adversarial examples to original ones to remove adversarial perturbations.
- b) **PixelDefend**: changed all pixels along each channel to maximize the probability distribution:

$$\begin{aligned} \max_{x'} \quad & \mathcal{P}_t(x') \\ \text{s.t.} \quad & \|x' - x\|_\infty \leq \epsilon_{defend}, \end{aligned}$$

if an adversarial example is not detected as malicious, no change will be made to the adversarial examples (defend = 0).

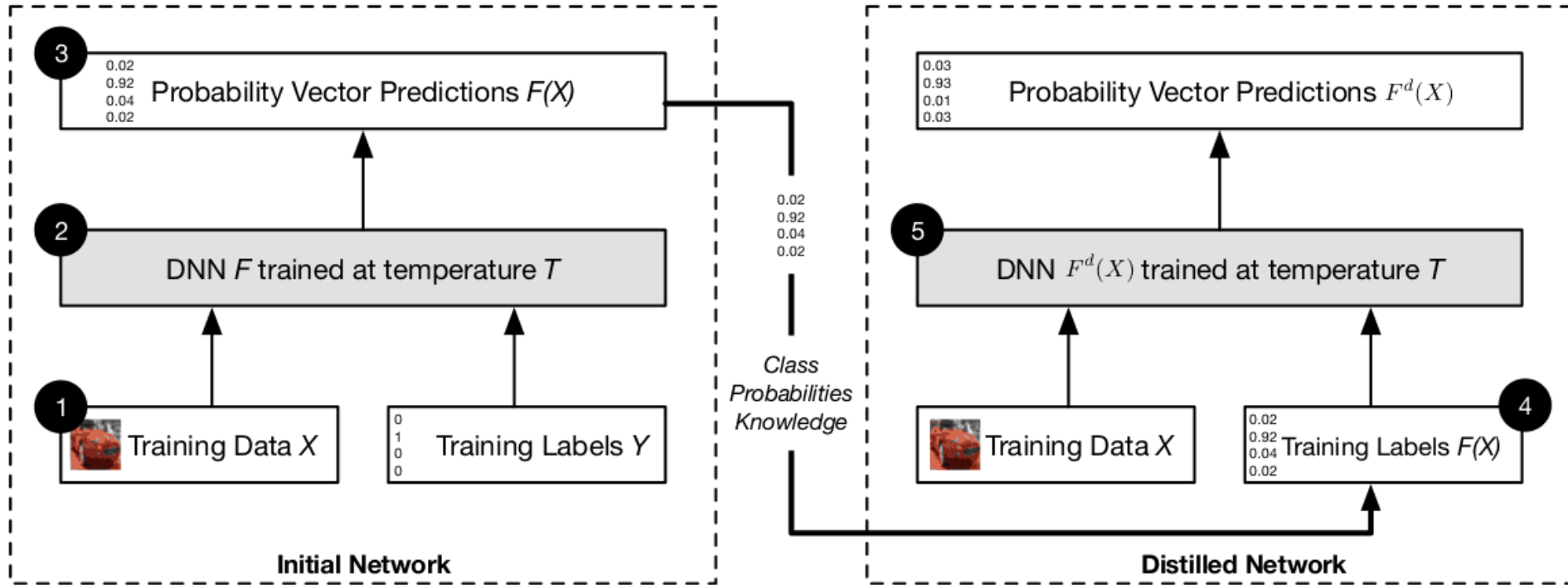
b) Proactive – Network Distillation

What is Network Distillation?

- Adversarial examples can be transformed to clean data via reconstruction. After transformation, the adversarial examples will not affect the prediction of deep learning models.
- Originally introduced to “distill” the knowledge of very deep networks (Teacher) into smaller one (Student) – is to train a second, possibly smaller network, with the probability distributions of the original.
- **Probability distributions, i.e. the activations of the final softmax layer (also referred to as “soft” labels), contain rich information about the task in contrast to the true “hard” labels.**

b) Proactive – Network Distillation...

How to use Network Distillation for defense? (Ref: Papernot et al.)



c) Proactive – Adversarial Retraining

- Training with adversarial examples is one of the countermeasures to make neural networks more robust.

Experiments with MNIST:

- They used half adversarial examples and half origin examples in each step of training.
- From the results, adversarial training increased the robustness of neural networks for one-step attacks (FGSM) but did not work well with iterative methods (BIM)

Evaluation Metrics

Evaluation Metrics

Utility Metrics	Attacks	MR	Misclassification Ratio
		ACAC	Average Confidence of Adversarial Class
		ACTC	Average Confidence of True Class
		ALD_p	Average L_p Distortion
		ASS	Average Structural Similarity
		PSD	Perturbation Sensitivity Distance
		NTE	Noise Tolerance Estimation
		RGB	Robustness to Gaussian Blur
		RIC	Robustness to Image Compression
		CC	Computation Cost

Case Study-1

Attack & Défense Simulation of Histopathology
Cancer Detection

Cancer Dataset

- In this dataset, you are provided with a large number of small pathology images to classify.
- 6 GB data
- 32x32 images
- A positive label indicates that the center 32x32px region of a patch contains at least one pixel of tumor tissue.
- **Attack:** FGSM (Fast Gradient Sign Attack)
- **Defense:** Adversarial Retraining

Attack using FGSM

Original Image

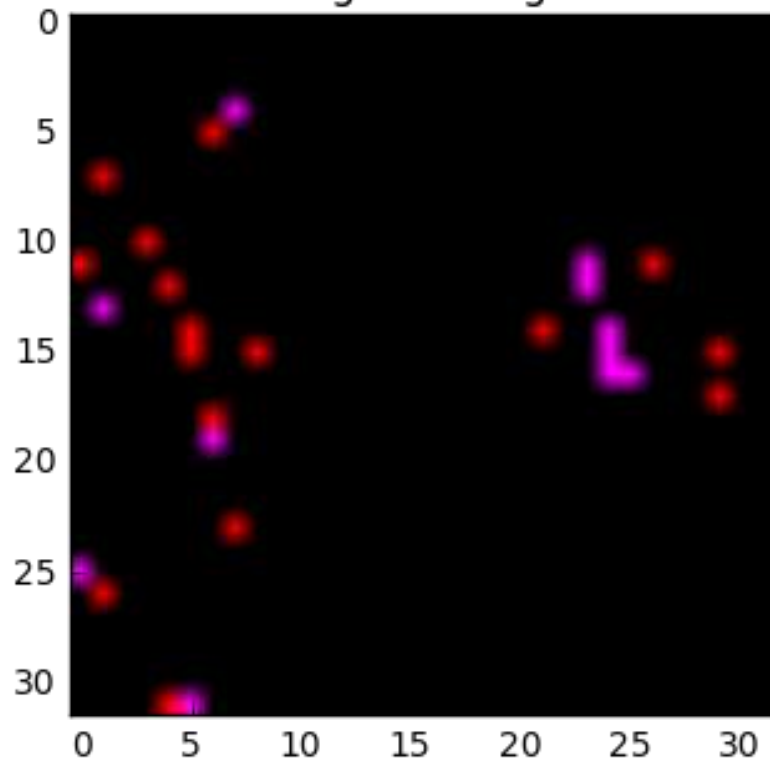
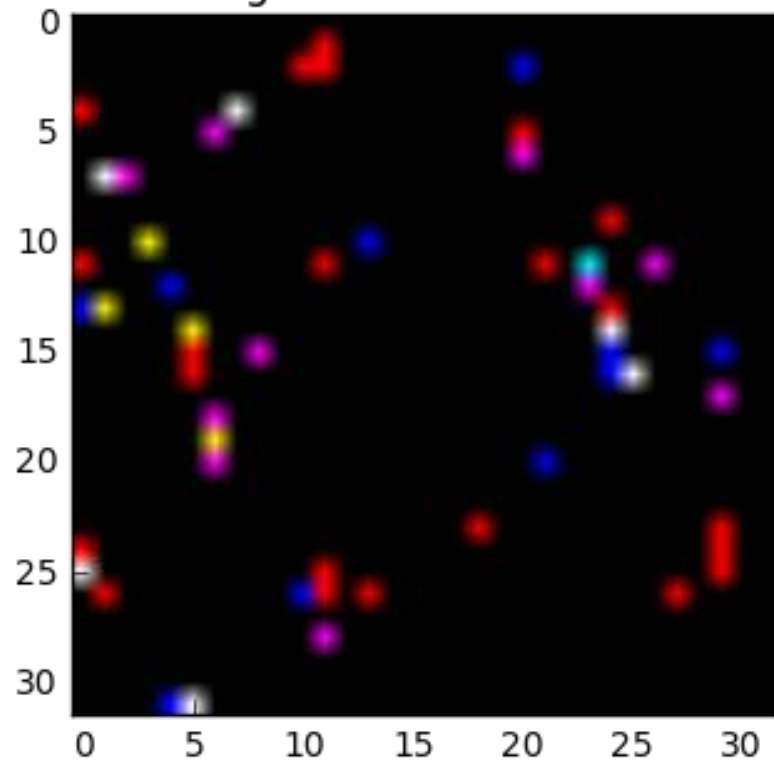


Image after FGSM attack



Evaluation

+-----+		
Accuracy before attack	75.1%	
+-----+		
Accuracy after FGSM attack	35.49%	
+-----+		
Accuracy with defense model	77.17%	
+-----+		

Case Study-2

Attack & Défense Simulation of IDS



Network Intrusion Detection

Details about data:

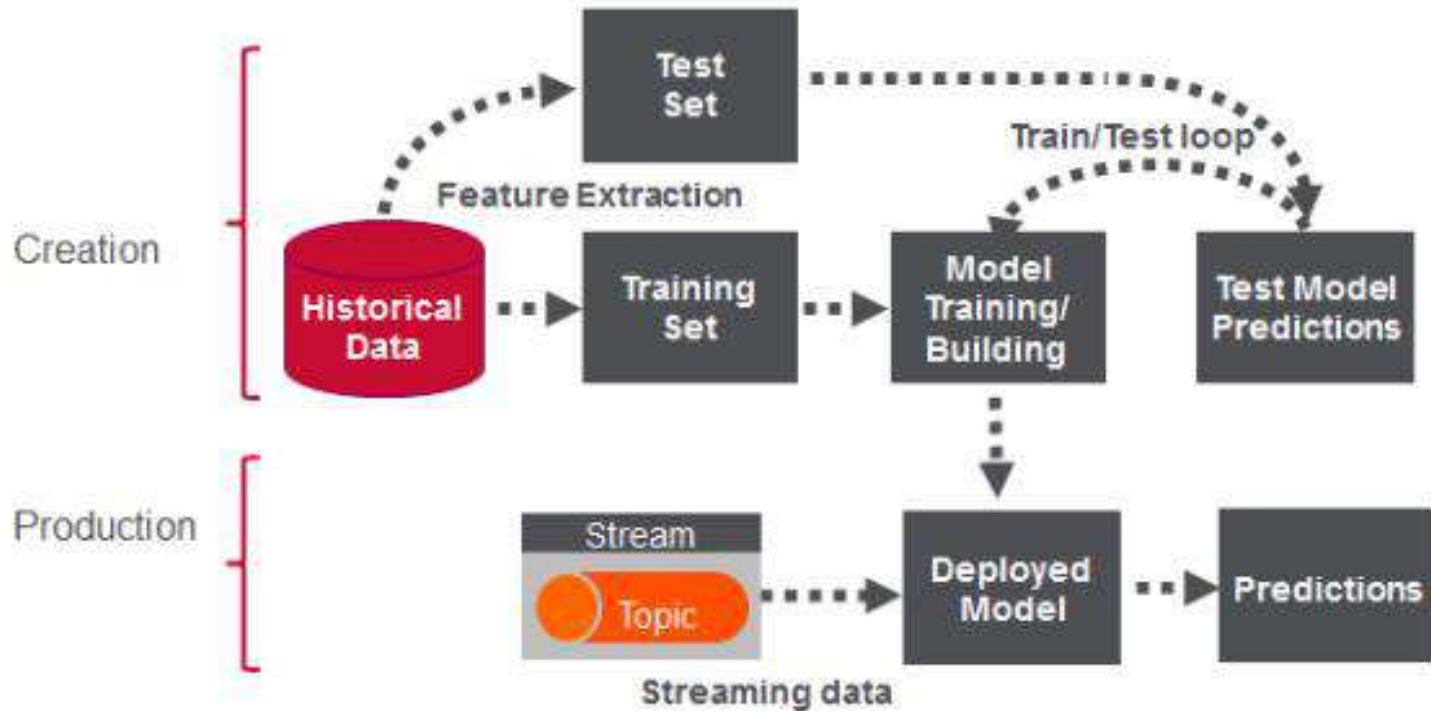
- **Lincoln Labs** set up an environment to acquire **nine weeks of raw TCP dump** data for a local-area network (LAN) simulating a typical **U.S. Air Force LAN**.
- Each connection **record consists of about 100 bytes**.

Example Input feature vector of a network connection:

0,tcp,http,SF,233,2032,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0.
00,0.00,0.00,0.00,1.00,0.00,0.00,15,15,1.00,0.00,0.07,0.00,
0.00,0.00,0.00,0.00, **normal**.

Intrusion Detection Phases

Two phases of fraud detection



Network Intrusion Detection...

Output:

(P, A) **P** – Predicted Label

(0, 0) **A** – Actual Label

(1, 0)

(0, 0)

(1, 1)

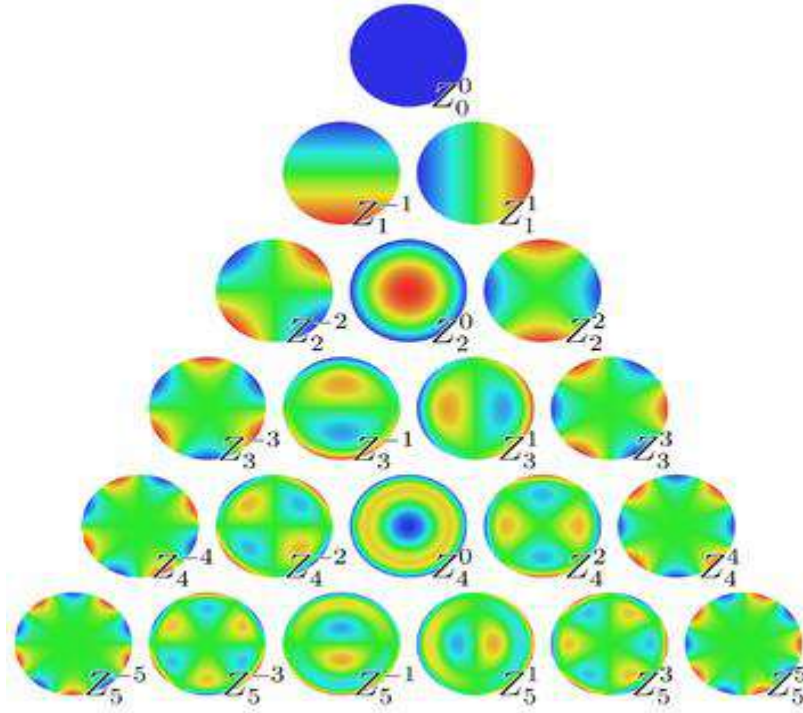
.....

.....

.....

Intrusion Detection Phases

Attacking using Zernike Moments (Black Box Attack)




Evaluation

Attack Type	Accuracy Before Défense	Accuracy After Défense
FGSM	0.56	0.78
C&W	0.43	0.65
R-FGSM	0.51	0.76
ATN	0.31	0.84
P-ATN	0.34	0.81

Case Study-3

Attacking Video Caption Generator System



A close-up shot from an animated film. On the left, a white, furry character with a purple nose and a small tuft of hair on its head is looking towards the right. On the right, a large, brown, furry character is shown in profile, looking towards the left. The white character's mouth is slightly open, and its purple nose is prominent. The brown character's eye is a bright yellow color. The background is a plain, light blue sky.

You have beautiful eyes!

ORIGINAL: VIDEO CAPTIONING

AFTER ATTACK: VIDEO CAPTIONING

You are so angry



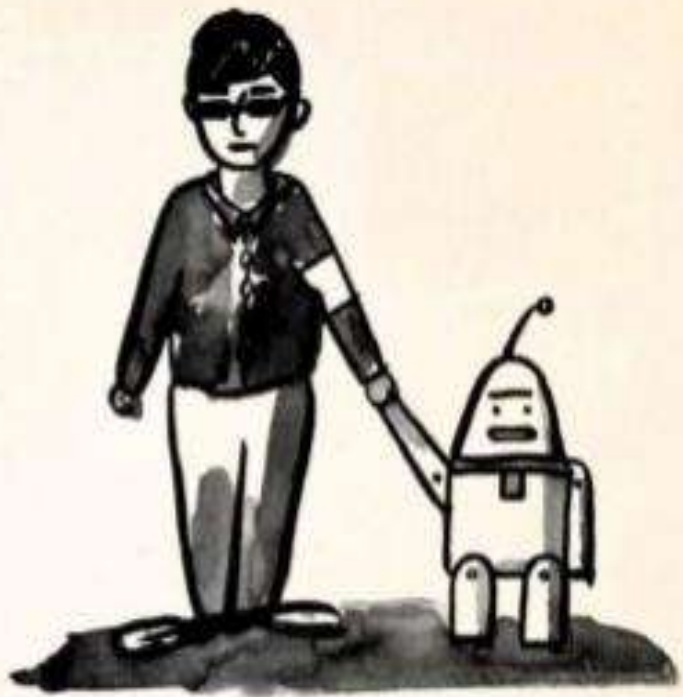
Protect the “Protector” – Tips

- Focus on the Strongest Attacks Possible
- Do not only use attacks during testing that were used during training.
- Applying many nearly-identical attacks is not useful
- Properly Ensemble over Randomness
- Verify Attack Convergence
- Carefully Investigate Attack Hyperparameters
- Try Brute-Force (Random) Search Attacks
- and so on.....

Summary

1. Artificial Intelligence has great potential.
2. People do not trust AI for important tasks.
3. To make AI trustworthy, systems need more human-like qualities (especially in their mistakes):
 - **robustness**
 - **introspection**
 - **adaptivity**
 - **transparency, explainability** and **fairness**

Lot of open research questions...



Thank You