Week 8 – Adversarial Machine Learning – Vulnerabilities (Part II) Explanation, Detection & Defence

COMP90073
Security Analytics

Yi Han, CIS

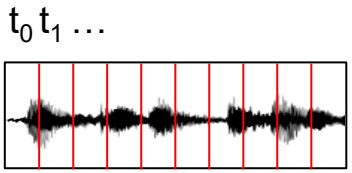
Semester 2, 2021

Overview

- Adversarial machine learning beyond computer vision
  - Audio
  - Natural language processing (NLP)
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defences
  - Learner robustification
- Challenges

Audio Adversarial Examples

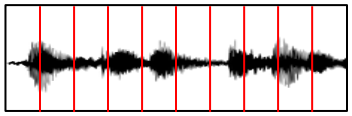
- Speech recognition system
  - Recurrent Neural Ne
  - Audio waveform → a of probability distributions over individual characters



	t_0	t_1	…
a	0.5	0.3	…
b	0.2	0.4	…
c	0.1	0.2	…
…	…	…	…

https://distill.pub/2017/ctc/

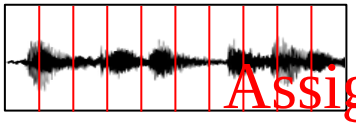
  - Challenge: alignment between the input and the output
    - Exact location of each character in the audio file



HEEELLLLOO

Audio Adversarial Examples

- Connectionist Temporal Classification (CTC)
  - Encoding
    - Introduce a special character called blank, denoted as "–"
    - $Y' = B(Y)$: modify the ground truth text (Y) by (1) inserting "–", (2) repeating characters in all possible ways
    - A blank character must be inserted between duplicate characters
    - E.g.,



      - Input X has a length of 10, and Y = [h, e, l, l, o].
      - Valid: heeell–llo, hhhh–el–lo, heell–looo
      - Invalid: hheel–llo–o, heel–lo

Audio Adversarial Examples

- Connectionist Temporal Classification (CTC)
  - Loss function
    - Calculate the sc[illegible]h Y' and sum them up

$$p(Y|X) = \sum_{Y'} \prod_{i=1}^{|X|} \boxed{p_i(y'_i|X)}$$

per time-step probabilities

    - Loss = negative log likelihood of the sum

$$-\sum_{Y'} \sum_{i=1}^{|X|} log(\,p_i(y'_i|X))$$

  - Decoding

    - Pick character with highest score for each time step
    - Remove duplicate characters, remove blanks

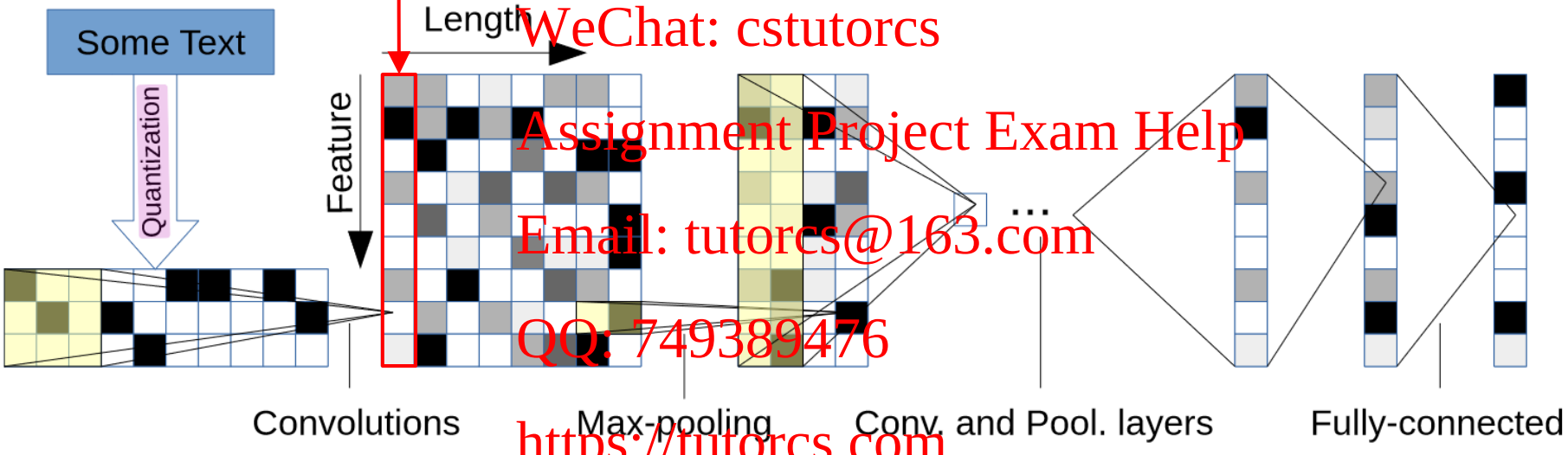
    - E.g., HEE–LL–LOO → HE–L–LO → HELLO

Audio Adversarial Examples

- Computer vision domain:
  - $\arg\min_{\delta \in [0,1]^d} \|\delta\| + c \cdot f(x+\delta)$
  - $C(x+\delta) = l_{target} \Leftrightarrow f(x') \leq 0$
- Audio adversarial example: speech recognition system [14]
  - How to measure the perturbation δ?
    - Measure δ in Decibels (dB): $dB(x) = \max_i 20log_{10}(x_i)$
    - $dB_x(\delta) = dB(\delta) - dB(x)$
  - How to construct the objective function?
    - Choose CTC-Loss(x', y_{target}) as function f
    - $C(x+\delta) = t_{target} \Leftarrow f(x+\delta) = f(x') \leq 0$
    - $C(x+\delta) = t_{target} \nRightarrow f(x+\delta) = f(x') \leq 0$
    - Solution will still be adversarial, but may not be minimally perturbed
  - Examples: https://nicholas.carlini.com/code/audio_adversarial_examples

NLP

- Deep text classification
  - Character level [15]
    - Every character [illegible]nted using one-hot encoding
    - 6 convolutional l[illegible] fully-connected layers



- Deep Text Classification Can be Fooled [16]
  - Identify text items that contribute most to the classification
  - Contribution measure based on the gradient $\frac{\partial f_{true}}{\partial x}$, x: training sample
  - Hot character: characters in dimensions with highest gradient magnitude
  - Hot word: containing ≥ 3 hot characters
  - Hot phrase: single hot word, or adjacent hot words
  - Hot Training/Sample Phrase: hot phrase that occurs most frequently in the training data/test sample

Rank	HTP	Freq.	Rank	HTP	Freq.
1	historic	7279	6	house	2447
2	building	4954	7	built	1927
3	church	3978	8	is a	1761
4	Register	3418	9	museum	1239
5	located	2604	10	is	1101

Table 1: Top ten HTPs of *Building* class.

NLP

- Deep Text Classification Can be Fooled [16]
  - Given text x, $C(x + \delta$
  - Insertion
    - What to insert: H[illegible] Phrases of the target class
    - Where to insert: near Hot Sample Phrases of the original class

The Uganda Securities Exchange (USE) is the ***historic*** principal stock exchange of Uganda. It was founded in June 1997. The USE is operated under the jurisdiction of Uganda's Capital Markets Authority which in turn reports to the Bank of Uganda, Uganda's central bank. The exchange's doors opened to trading in January 1998. At the time, the exchange had just one listing, a bond issued by the East African Development Bank. Uganda Securities Exchange

Figure 2: An adversarial text sample generated by inserting just one HTP (99.7% *Company* to 88.6% *Building*).

NLP

- Deep Text Classification Can be Fooled [16]
  - Modification: replace [illegible]cters in Hot Sample Phrases by
    - Common misspe[illegible]
    - Characters visua[illegible]

Maisie is a comedy film property MGM originally purchased for Jean Harlow but before a shooting script could be completed Harlow died in 1937. It was put on hold until 1939 when Ann Sothern was hired to star in the project with Robert Young as leading man. It is based on the novel Dark Dame by Wilson Collison. It was the first of ten films starring Sothern as Maisie Ravier. In Mary C. Maisie (film)

Figure 5: An adversarial text sample generated by introducing a common misspelling (99.6% *Film* to 99.0% *Company*).

NLP

- Deep Text Classification Can be Fooled [16]
  - Removal: the inesser[illegible]ive or adverb in HSPs are removed
    - Less effective
    - Only downgrade[illegible]tence of the original class

Edward & Mrs. Simpson is a seven-part ~~*British*~~ television series that dramatises the events leading to the 1936 abdication of King Edward VIII of the United Kingdom who gave up his throne to marry the twice-divorced American Wallis Simpson. The series made by Thames Television for ITV was originally broadcast in 1978. Edward Fox played Edward and Cynthia Harris portrayed Mrs. Simpson. Edward & Mrs. Simpson

Figure 6: Lower the confidence of the original class by removing a word from an HSP (95.5% *Film* to 60.5% *Film*).

- Deep Text Classification Can be Fooled [16]
  - Combination of three
  - Limit: all performed n

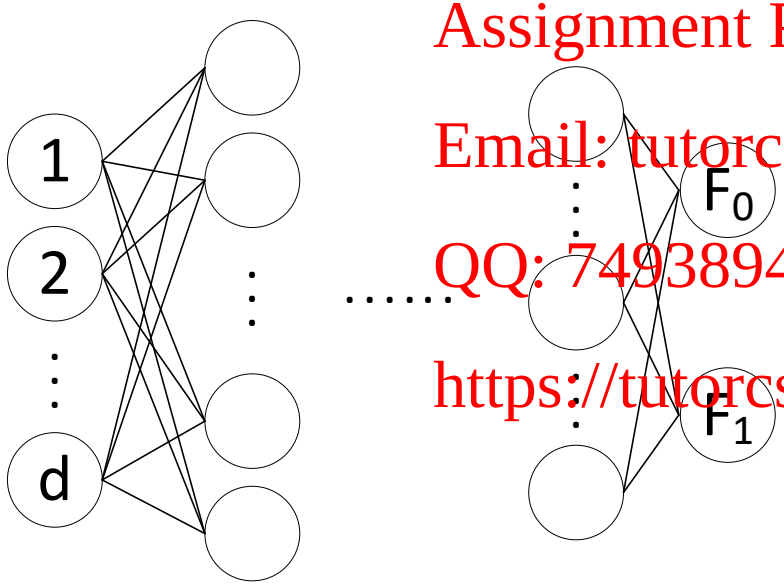
The Old Harbor Reservation Parkways are three ~~*historic*~~ roads in the Old Harbor area of Boston. ***Some exhibitions of Navy aircrafts were held here.*** They are part of the Boston parkway system designed by Frederick Law Olmsted. They include all of William J. Day Boulevard running from Castle Island to Kosciuszko Circle along Pleasure Bay and the Old Harbor shore. The part of Columbia Road from its north-eastern end at Farragut Road west to Pacuska Circle (formerly called Preble Circle). Old Harbor Reservation

Figure 7: Combination of three strategies (83.7% *Building* to 95.7% *Means of Transportation*).

Malware Detection

- Attacking malware classifier for mobile phones [7]
  - An application is represented by a binary vector $X \in \{0,1\}^d$
    - 1: the app has the feature, 0: the app doesn't have the feature
    - E.g., chat app: ✓, storage ✓, calendar ✗ → [1, 1, 0]
  - Classifier: feed forward neural network

    $F(X) = [F_0(X), F_1(X)]$, $F_0(X) + F_1(X) = 1$, 0: benign, 1: malicious



Benign, if $F_0(X) > F_1(X)$
Malicious, otherwise

Evasion attacks (application)

- Attacking malware classifier for mobile phones [7]
  - Attack goal: make a malicious application classified as benign
  - Limit: only add features, avoid destroying app functionalities
  - For each iteration:
    - Step 1: compute the gradient of F w.r.t. X:

$$\frac{\partial F_k(X)}{\partial X} = \left[\frac{\partial F_k(X)}{\partial X_j}\right]_{k\in[0,1],j\in[1,d]}$$

    - Step 2: change the feature X_i to 1: (1) $X_i = 0$, (2) with the maximal positive gradient → maximise the change into the target class 0

$$i = \arg\max_{j\in[1,m],X_j=0} F_0(X_j)$$

Evasion attacks (application)

Classifier	MWR	Accuracy	FNR	FPR
Arp et al. [1]	–	–	6.1	1
Sayfullina et al. [19]	–	–	0.1	17.9
[200]	0.4	97.83	8.06	
[200]	0.5	95.85	5.41	
[10, 10]	0.3	97.59	16.37	
[10, 10]	0.4	94.85	9.68	
[10, 10]	0.5	94.75	7.34	
[10, 200]	0.3	97.53	11.21	
[10, 200]	0.4	96.14	8.67	
[10, 200]	0.5	94.26	5.72	
[200, 10]	0.3	95.63	15.25	
[200, 10]	0.4	93.95	10.81	5.82
[200, 10]	0.5	92.97	8.96	6.92
[50, 50]	0.3	96.57	12.57	2.98
[50, 50]	0.4	96.79	13.08	2.7
[50, 50]	0.5	93.82	6.76	6.11
[50, 200]	0.3	97.58	17.30	1.71
[50, 200]	0.4	97.35	10.14	2.29
[50, 200]	0.5	95.65	6.01	4.45
[200, 50]	0.3	96.89	6.37	2.94
[200, 50]	0.4	95.87	5.36	4.06
[200, 50]	0.5	93.93	4.55	6.12
[100, 200]	0.4	97.43	8.35	2.27
[200, 100]	0.4	97.32	9.23	2.35
[200, 100]	0.5	96.35	6.66	3.48
[200, 200]	0.1	98.92	17.18	0.32
[200, 200]	0.2	98.38	8.74	1.29
[200, 200]	**0.3**	**98.35**	**9.73**	**1.29**
[200, 200]	**0.4**	**96.6**	**8.13**	**3.12**
[200, 200]	**0.5**	**95.93**	**6.37**	**3.96**
[200, 300]	0.3	98.35	9.59	1.25
[200, 300]	0.4	97.62	8.74	2.05
[300, 200]	0.2	98.13	9.34	1.5
[300, 200]	0.4	97.29	8.06	2.43
[200, 200, 200]	0.1	98.91	17.48	0.31
[200, 200, 200]	0.4	97.69	10.34	1.91
[200, 200, 200, 200]	0.4	97.42	13.08	2.07
[200, 200, 200, 200]	0.5	97.5	12.37	2.01

MWR: malware ratio
MR: misclassification rate

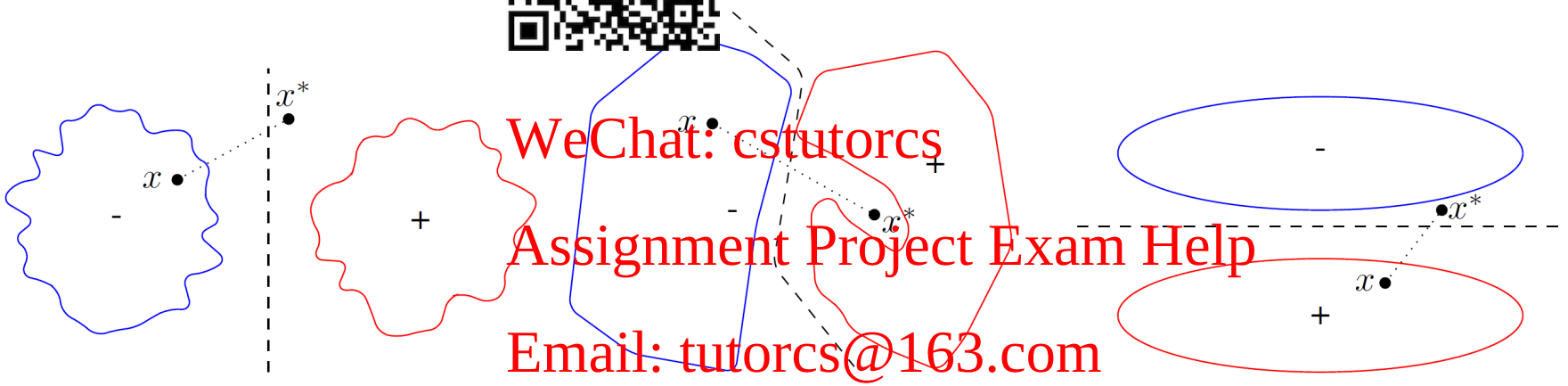
Classifier	MWR	MR	Distortion
[200]	0.4	81.89	11.52
[200]	0.5	79.37	11.92
[10, 10]	0.3	69.62	13.15
[10, 10]	0.4	55.88	16.12
[10, 10]	0.5	84.05	11.48
[10, 200]	0.3	75.47	12.89
[10, 200]	0.4	55.70	14.84
[10, 200]	0.5	57.19	14.96
[200, 10]	0.3	50.07	14.96
[200, 10]	0.4	35.31	17.79
[200, 10]	0.5	36.62	17.49
[100, 200]	0.4	74.93	12.87
[200, 100]	0.4	71.42	13.12
[200, 100]	0.5	73.02	12.98
[50, 50]	0.3	61.71	15.37
[50, 50]	0.4	60.02	14.7
[50, 50]	0.5	40.97	17.64
[50, 200]	0.3	79.25	11.61
[50, 200]	0.4	69.44	13.95
[50, 200]	0.5	64.66	15.16
[200, 50]	0.3	66.55	14.99
[200, 50]	0.4	58.31	15.76
[200, 50]	0.5	62.34	14.54
[200, 200]	0.1	78.28	10.99
[200, 200]	0.2	63.49	13.43
[200, 200]	**0.3**	**63.08**	**14.52**
[200, 200]	**0.4**	**64.01**	**14.84**
[200, 200]	**0.5**	**69.35**	**13.47**
[200, 300]	0.3	70.99	13.24
[200, 300]	0.4	61.91	14.19
[300, 200]	0.2	69.96	13.62
[300, 200]	0.4	63.51	14.01
[200, 200, 200]	0.1	75.41	10.50
[200, 200, 200]	0.4	71.31	13.08
[200, 200, 200, 200]	0.4	62.66	14.64

Overview

- Adversarial machine learning beyond computer vision
  - Audio
  - NLP
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defences
  - Learner robustification
- Challenges

Locations of Adversarial Samples

- Locations of adversarial samples
  - Off the data manifold, not legitimate instances
  - Three scenarios [1]



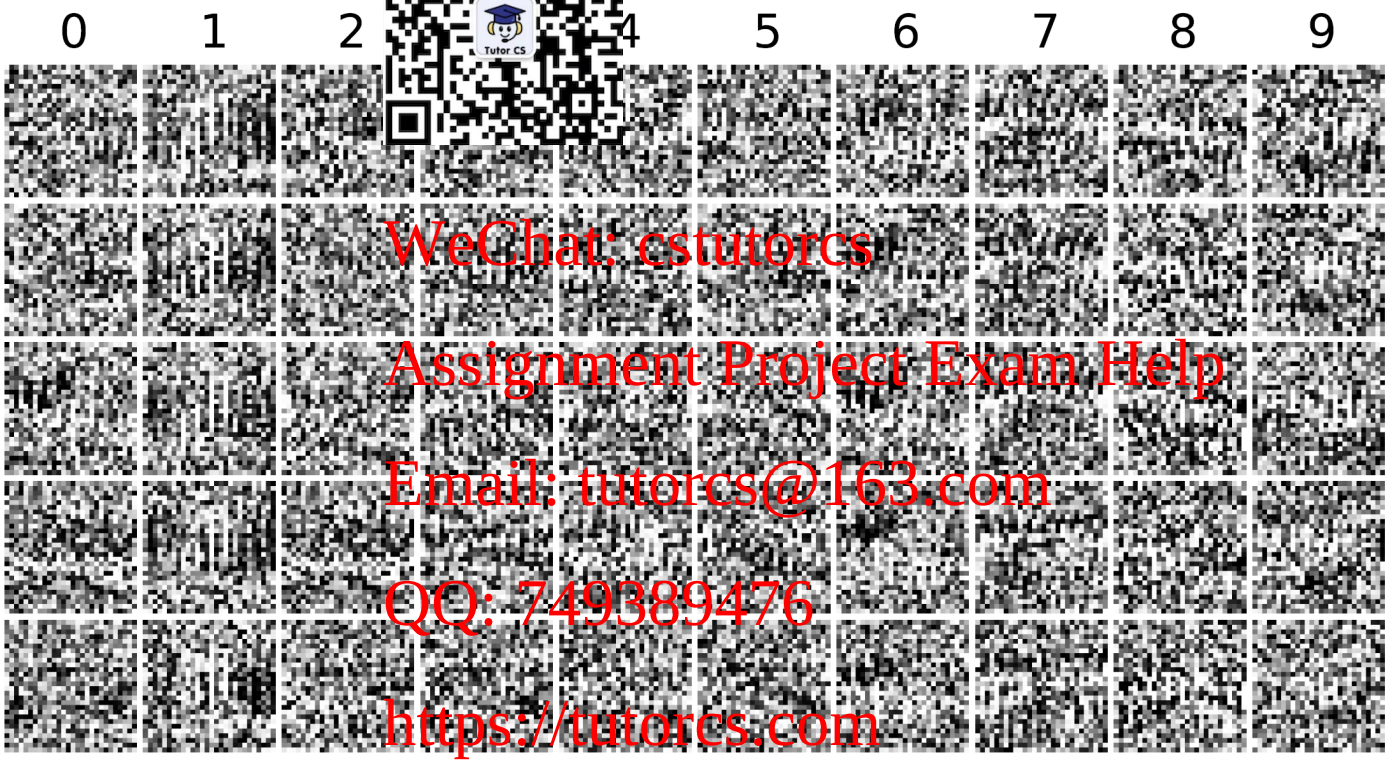
Near the boundary, but far from the "+" manifold

Away from the boundary, but near the manifold – in the "pocket" of the "+" manifold

Close to the boundary and the "-" manifold

Locations of Adversarial Samples

- Images that are unrecognisable to human eyes, but can be identified by DNNs with certainty [2]



DNNs believe with 99.99% confidence that the above images are digits 0-9

Overview

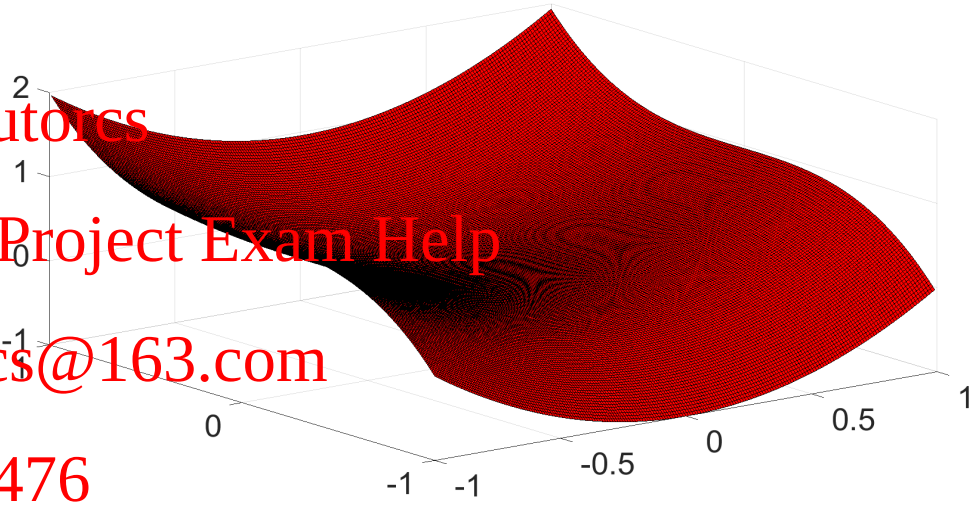
- Adversarial machine learning beyond computer vision
  - Audio
  - NLP
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defenses
  - Learner robustification
- Challenges

Explanation1: Insufficient Training Data

- Potential reason 1: insufficient training data
- An illustrative exampl
  - $x \in [-1, 1), y \in [-$ $[-1, 2)$
  - Binary classification:
    - Class 1: $z < x^2 + y^3$
    - Class 2: $z \geq x^2 + y^3$
  - *x*, *y*, *z* are increased by 0.01

    → a total of $200 \times 200 \times 300$

    $= 1.2 \times 10^7$ points
  - How many points are needed to reconstruct the decision boundary?



Explanation1: Insufficient Training Data

- Randomly choose the training and test datasets

	ng dataset	Test dataset
Setting 1		40
Setting 2	800	400
Setting 3	8000	4000
Setting 4	80000	40000

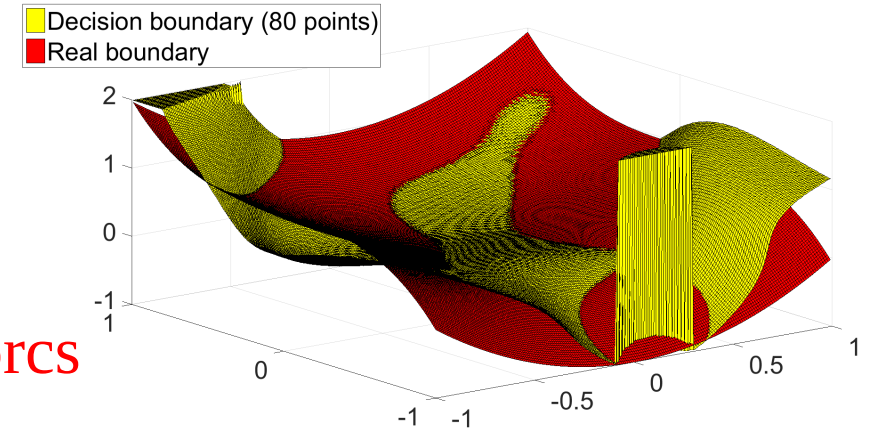
- Boundary dataset (adversarial samples are likely to locate here):

$$x^2 + y^3 - 0.1 < z < x^2 + y^3 + 0.1$$

Explanation1: Insufficient Training Data

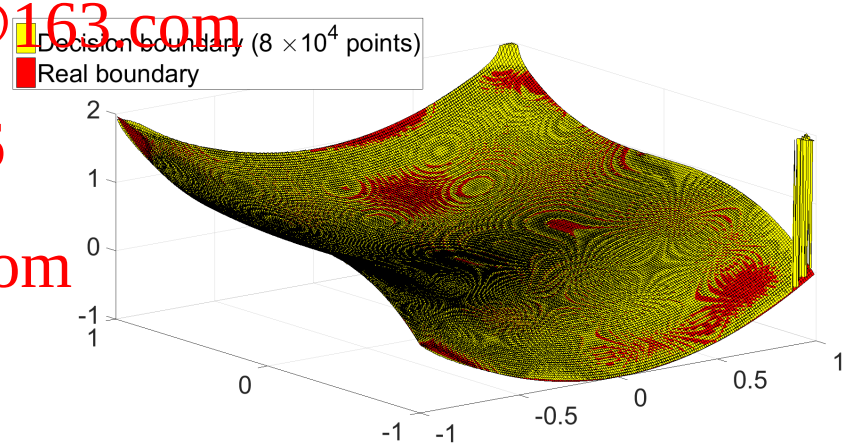
- Test result
  - RBF SVMs

Size of the training dataset	Accuracy on its own test dataset	Accuracy on th dataset with 4×10	cy on the ry dataset
80	100	92.7	0.8
800	99.0	97.4	4.9
8000	99.5	99.6	94.1
80000	99.9	99.9	98.9



  - Linear SVMs

Size of the training dataset	Accuracy on its own test dataset	Accuracy on the test dataset with 4×10^4 points	Accuracy on the boundary dataset
80	100	96.3	70.1
800	99.8	99.0	85.7
8000	99.9	99.8	97.3
80000	99.98	99.98	99.5



- 8000: 0.067% of 1.2×10^7
- MNIST: 28×28 8-bit greyscale images, $(2^8)^{28\times28} \approx 1.1\times10^{1888}$
- $1.1\times10^{1888}\times0.067\% \gg 6\times10^5$

Overview

- Adversarial machine learning beyond computer vision
  - Audio
  - NLP
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defences
  - Learner robustification
- Challenges

Poisoning attacks

- Poison frog attacks [10]
  - E.g., add a seemingly [illegible]s image (that is properly labeled) to a training set, and cont[illegible]ntity of a chosen image at test time

Target class

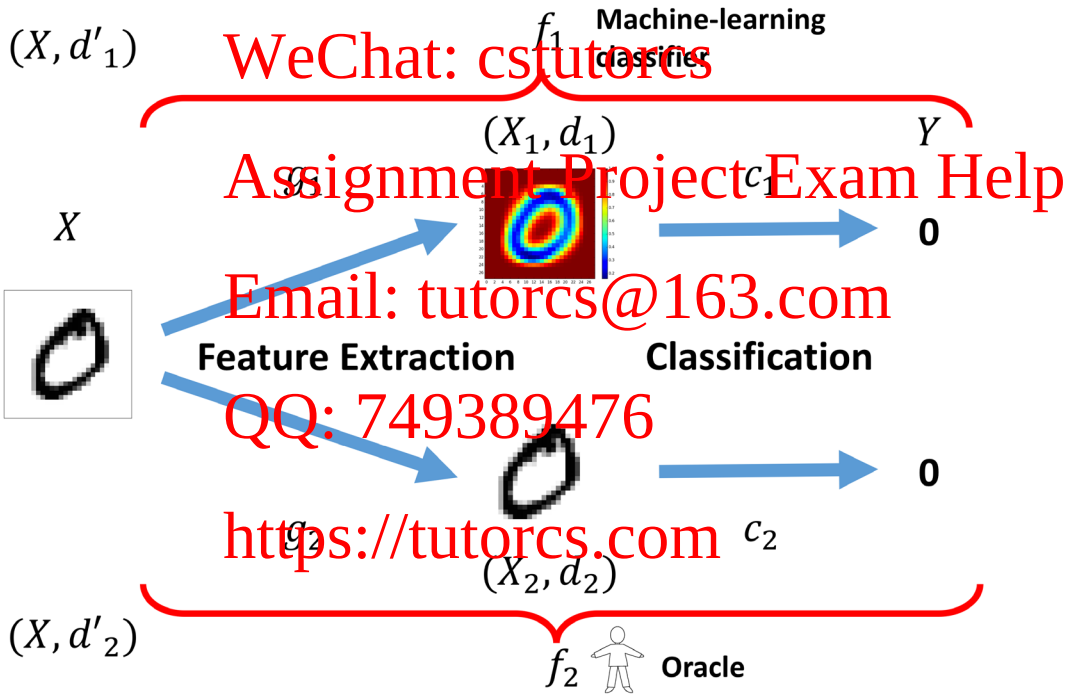
Base class

Poisoning attacks

- Generate poison data
  - $f(x)$: the function propagates an input x through the network to the pe e layer (before the softmax layer)
  - $p = \underset{x}{\operatorname{argmin}} \|f(x) - f(t)\|_2^2 + \beta\|x - b\|_2^2$
    - $\|f(x) - f(t)\|_2^2$: makes p move toward the target instance in feature space and get embedded in the target class distribution
    - $\beta\|x - b\|_2^2$: makes p appear like a base class instance to a human labeller

Explanation2: Unnecessary Features

- Potential reason 2: redundant features [3]
  - Classifier $f = g \circ c$, g: extraction, c: classification
  - d : similarity measure
  - Features extracted by classifier (X_1) ≠ Features extracted by human (X_2)



Explanation2: Unnecessary Features

- Potential reason 2: redundant features [3]
  - Previous definition of adversarial attacks:

$$Find\ x'$$
$$\text{s.t. } f_1(x) \neq f_1(x')$$
$$\Delta(x, x') < \epsilon$$

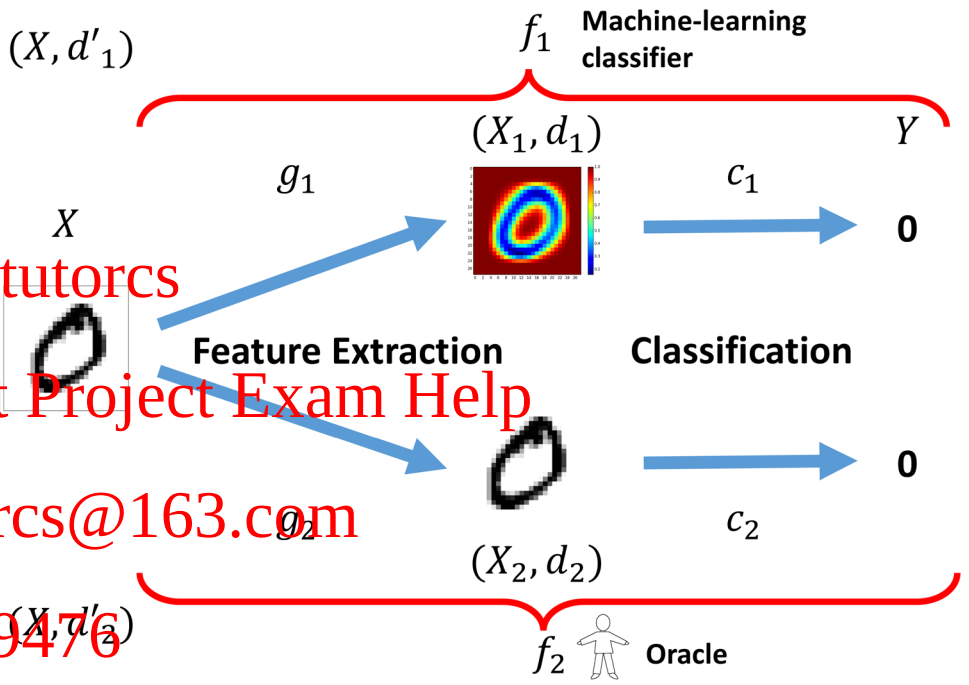
  - New definition:

$$Find\ x'$$
$$\text{s.t. } f_1(x) \neq f_1(x')$$
$$d_2(g_2(x), g_2(x')) < \delta_2$$
$$f_2(x) = f_2(x')$$

  - $\{\delta_2, \eta\}$-strong-robustness:

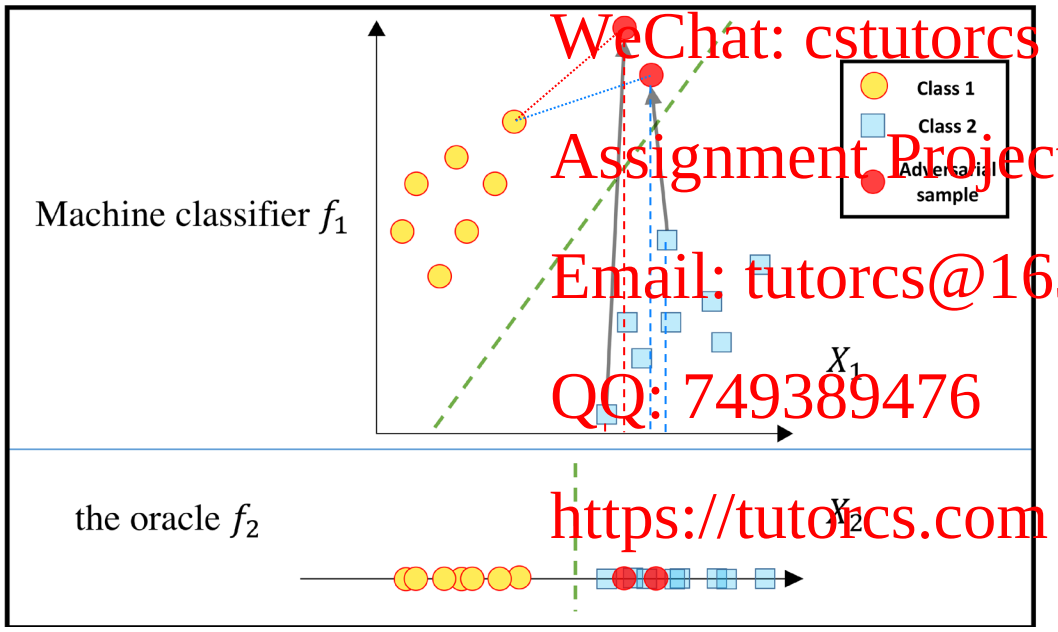
$$if\ \forall x, x' \in X\ a.e.(x, x')\ satisfies$$
$$P(f_1(x) = f_1(x') | f_2(x) = f_2(x'), d_2(g_2(x), g_2(x')) < \delta_2) > 1 - \eta$$

f_1 agrees with f_2



Explanation2: Unnecessary Features

- Unnecessary features ruin strong-robustness
  - If f_1 uses unnecessar → not strong-robust
  - If f_1 misses necessar used by f_2 → not accurate
  - If f_1 uses the same se es as f_2 → strong-robust, can be accurate



Can be far away to the original instance in the trained classifier's feature space, and at the other side of the boundary

Each adversarial sample is close to the original instance in the oracle feature space

Overview

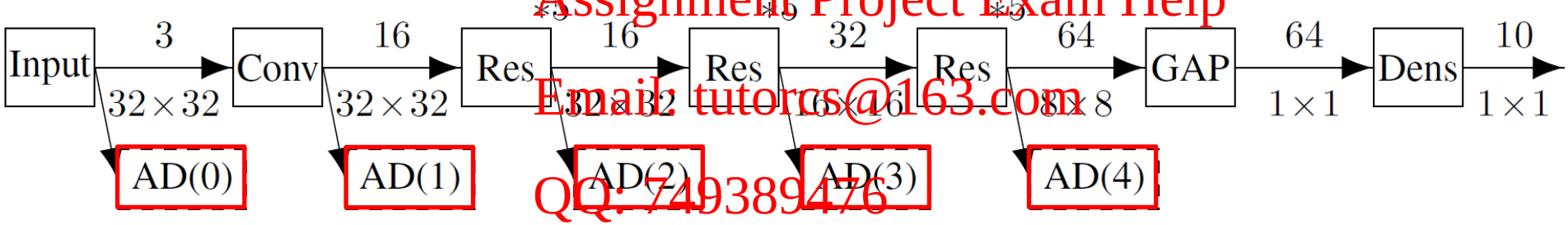
- Adversarial machine learning beyond computer vision
  - Audio
  - NLP
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defence
  - Learner robustification
- Challenges

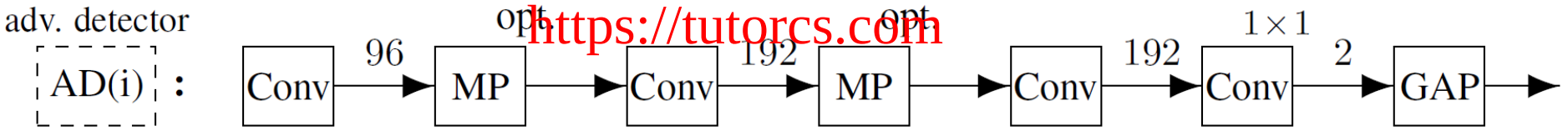
Data-driven Defence

- Data-driven defence
  - Filtering instances: p[illegible]ata in the training dataset or the adversarial samples against the [illegible]et either exhibit different statistical features, or follow a different d[illegible]– detection
  - Injecting data: add adversarial samples into training – adversarial training
  - Projecting data: project data into lower-dimensional space; move adversarial samples closer to the manifold of legitimate samples

Data-driven Defence: Filtering Instances

- Filtering instances
- On Detecting Adversarial [illegible]ons [4]
  - Adversary detection [illegible]ranch off the main network at some layer
  - Each detector produc[illegible]robability of the input being adversarial
  - Step 1: Train the main network regularly, and freeze its weights
  - Step 2: Generate an adversarial sample for each training data point
  - Step 3: Train the detectors on the balanced, binary dataset





Data-driven Defence: Filtering Instances

- Adaptive/dynamic attacker: attacker that is aware of the detection method

$$x_{n+1}^{\text{adv}} = \text{Clip}_x^{\varepsilon}\left\{x_n^{\text{adv}} + \alpha\left[\left(\nabla_x \text{J}_{\text{cls}}(x_n^{\text{adv}}, y_{\text{true}}(x))\right) + \sigma\,\text{sgn}\left(\nabla_x \text{J}_{\text{det}}(x_n^{\text{adv}}, 1)\right)\right]\right\}$$

cross-entropy loss of the classifier

letting the classifier mis-label the input x

cross-entropy loss of the detector

making the detectors output p_{adv} as small as possible

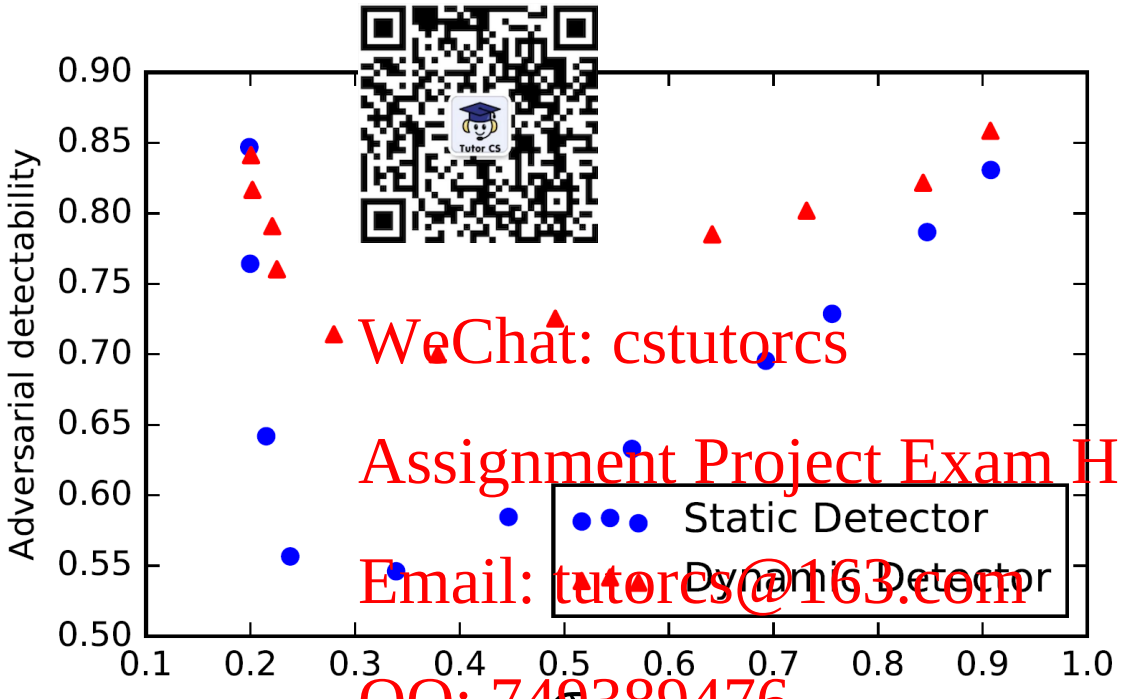
- Dynamic adversary training

	Static	**Dynamic**
Defender	Train the classifier → Freeze its weights → Precompute adversarial samples	Compute adversarial examples on-the-fly for each mini-batch
Attacker	Modify *x* only to maximise the classifier's cross-entropy loss	Modify *x* to fool classifier + detector

Adapt to each other

Data-driven Defence: Filtering Instances

- Test on CIFAR10



Data-driven Defence

- Data-driven defence
  - Filtering instances: p ata in the training dataset or the adversarial samples against the t et either exhibit different statistical features, or follow a different d – detection
  - Injecting data: add adversarial samples into training – adversarial training
  - Projecting data: proje dimensional space; move adversarial samples closer to the manifold of legitimate samples

Data-driven Defence: Injecting Data

- Adversarial training: add adversarial samples into training data
- Towards Deep Learning Models Resistant to Adversarial Attacks [5]
  - Normally how a classification problem is formalised

$$\theta^* = \arg\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim D} [\ldots]$$

Not robust

Augment

  - Redefine the loss by incorporating the adversary:

$$\theta^* = \arg\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim D} \left[\max_{\delta \in [-\varepsilon, \varepsilon]^d} L(x+\delta; y; \theta) \right]$$

adversary: perturb x to maximise the loss

Defender: find model parameters θ^* to minimise the "adversarial loss"

Data-driven Defence: Injecting Data

- Towards Deep Learning Models Resistant to Adversarial Attacks [5]
  - Step 1: fix θ, generat[illegible]rial samples using strong attacks (e.g., projected gradient de[illegible]W): $x_i \leftarrow \text{clip}_{\varepsilon}\left(x_{i-1} + \alpha \cdot \text{sign}\left(\frac{\partial L}{\partial x_{i-1}}\right)\right)$
  - Step 2: Update θ: train the network on the augmented dataset

Algorithm 1 Adversarial Training (AT$(\mathcal{D}, N, \eta, \mathcal{G})$)

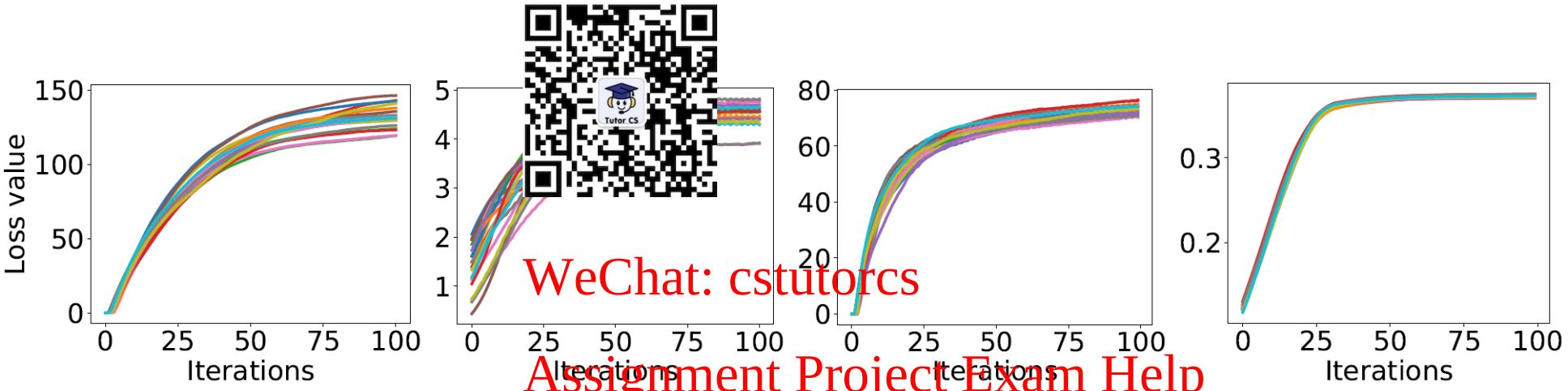
Input: Training data $\mathcal{D}$; Total iterations N; Learning rate η
Input: An attack $\mathcal{G}$
Output: θ

1: Randomly initialize network θ
2: **for** $i \leftarrow 0$ to N **do**
3: Sample a batch $(x_i, y_i) \sim \mathcal{D}$
4: Generate adversarial examples $x_i^\star \leftarrow \mathcal{G}(x_i, y_i)$ → Inner maximisation: find adversarial examples
5: $\theta \leftarrow \theta - \eta \sum_i \nabla_\theta \mathcal{L}(f_\theta(x_i^\star), y_i)$ → Outer minimisation: optimise θ
// This is standard SGD; it can be replaced by other training algorithms such as Adam
6: **end for**

[17]

Data-driven Defence: Injecting Data

- Towards Deep Learning Models Resistant to Adversarial Attacks [5]



(a) MNIST Natural training (b) MNIST Adversarial training (c) CIFAR10 Natural training (d) CIFAR10 Adversarial training

Data-driven Defence: Injecting Data

- Curriculum Adversarial Training (CAT) [17]
  - Adversarial training c[illegible]he specific attack in use
  - Training curriculum: t[illegible]del from weaker attacks to stronger attacks
  - Attack strength: PGD[illegible]number of iterations

Algorithm 2 Curriculum Adversarial Training (Basic)

Input: Training data $\mathcal{D}$; Validation data $\mathcal{V}$; Epoch iterations n; Learning rate η; Maximal attack strength K;
Input: An class of attacks, denoted as $\mathcal{A}(k)$ whose strength is parameterized by k.
Output: θ

1: Randomly initialize network θ
2: **for** $l \leftarrow 0$ *to* K **do**
3: **repeat**
4: $\theta \leftarrow \mathrm{AT}(\mathcal{D}, n, \eta, \mathcal{A}(l))$
5: // One epoch of adversarial training using $\mathcal{A}(l)$
6: **until** $\tilde{l}$-accuracy on $\mathcal{V}$ not increased for 10 epochs
7: **end for**

$$n = \frac{|\mathcal{D}|}{\text{batch size}}$$

$$\tilde{k}\text{-accuracy}(\mathcal{V}, \theta) = \frac{|\{(x,y) \in \mathcal{V} | \forall i \in \{0,...,k\}. f_\theta(\mathcal{A}(i)(x)) = y\}|}{|\mathcal{V}|}$$

Data-driven Defence: Injecting Data

- Curriculum Adversarial Training (CAT) [17]
  - Batch mixing
    - Catastrophic forgetting [ ]: a neural network tends to forget the information learned from previous tasks when training on new tasks
    - Generate adversarial examples using $\mathrm{PGD}(i), i \in \{0, 1, \ldots, l\}$, and combine them to form a batch, i.e., batch mixing

Algorithm 1 Adversarial Training $(\mathrm{AT}(\mathcal{D}, N, \eta, \mathcal{G}))$

Input: Training data $\mathcal{D}$; Total iterations N; Learning rate η
Input: An attack $\mathcal{G}$
Output: θ

1: Randomly initialize network θ
2: **for** $i \leftarrow 0$ to N **do**
3: Sample a batch $(x_i, y_i) \sim \mathcal{D}$
4: Generate adversarial examples $x_i^\star \leftarrow \mathcal{G}(x_i, y_i)$
5: $\theta \leftarrow \theta - \eta \sum_i \nabla_\theta \mathcal{L}(f_\theta(x_i^\star), y_i)$
// This is standard SGD; it can be replaced by other training algorithms such as Adam
6: **end for**

$$\sum_{i=0}^{l} \alpha_i \sum_{(x,y)\sim\mathcal{D}} \mathcal{L}(f_\theta(\mathcal{A}(i)(x)), y)$$

$$\alpha_i = \frac{1}{l+1}$$

Data-driven Defence: Injecting Data

- Curriculum Adversarial Training (CAT) [17]
  - Quantization
    - Attack generalisa… model trained with CAT may not defend against stronger …
    - Quantization: real value → *b*-bit integer
    - Each input *x*: real value from $[0, 1]^d$ → integer value from $[0, 2^b\text{-}1]^d$

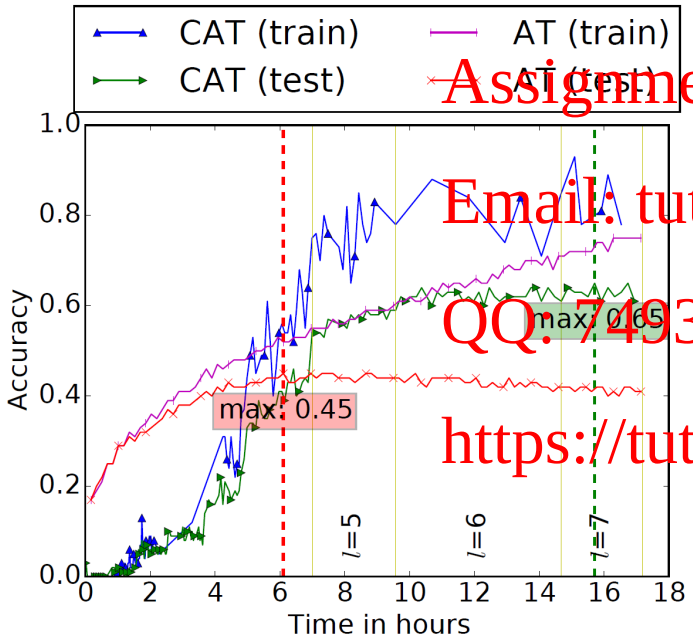


Figure 1: Training and testing empirical worst-case accuracy of vanilla adversarial training and curriculum adversarial training over time. The model is ResNet-50, and the dataset is CIFAR-10.

Data-driven Defence: Injecting Data

- Adversarial training for free [20]
  - Train on each minibatches, number of epochs N_{ep} → N_{ep}/m
  - FGSM is used, but perturbations are not reset between minibatches
  - Single backward pass to update both model weights and perturbation

Algorithm 1 "Free" Adversarial Training (Free-m)

Require: Training samples X, perturbation bound ϵ, learning rate τ, hop steps m

Initialize θ
$\delta \leftarrow 0$
for epoch $= 1 \ldots N_{ep}/m$ **do**
  for minibatch $B \subset X$ **do**
    for i $= 1 \ldots m$ **do**
      Update θ with stochastic gradient descent
        $g_\theta \leftarrow \mathbb{E}_{(x,y)\in B}[\nabla_\theta\, l(x+\delta, y, \theta)]$
        $g_{adv} \leftarrow \nabla_\delta\, l(x+\delta, y, \theta)$
        $\theta \leftarrow \theta - \tau g_\theta$
      Use gradients calculated for the minimization step to update δ
        $\delta \leftarrow \delta + \epsilon\, \text{sign}(g_{adv})$
        $\delta \leftarrow \text{clip}(\delta, -\epsilon, \epsilon)$
    end for
  end for
end for

Data-driven Defence: Injecting Data

- Fast Adversarial Training [18]
  - FGSM adversarial tra random initialization
  - Non-zero initial pertu he primary driver for success

Algorithm 3 FGSM adversarial training for T epochs, given some radius ϵ, N PGD steps, step size α, and a dataset of size M for a network f_θ

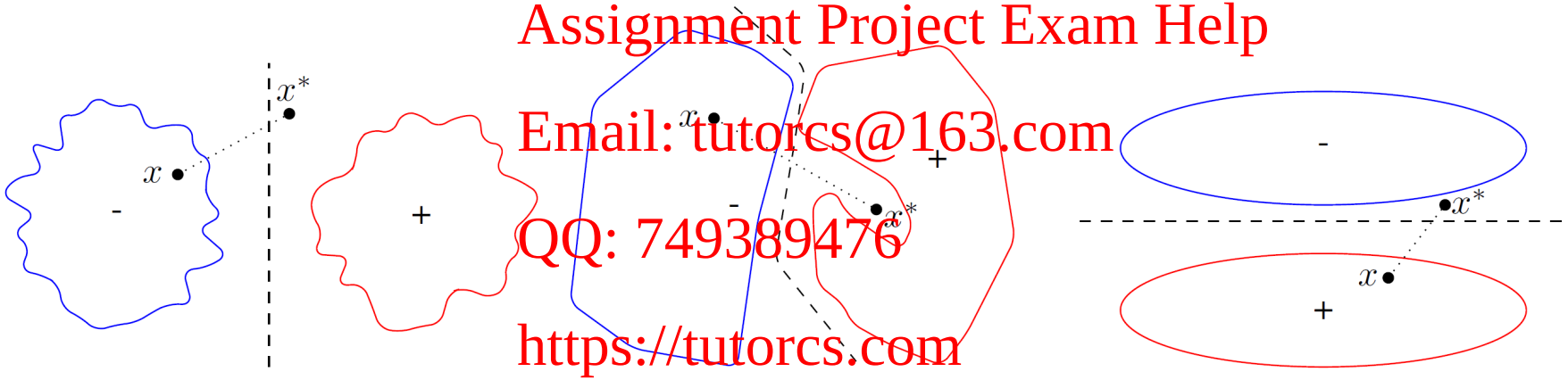
for $t = 1 \dots T$ **do**
  for $i = 1 \dots M$ **do**
    // Perform FGSM adversarial attack
    $\delta = \text{Uniform}(-\epsilon, \epsilon)$
    $\delta = \delta + \alpha \cdot \text{sign}(\nabla_\delta \ell(f_\theta(x_i + \delta), y_i))$
    $\delta = \max(\min(\delta, \epsilon), -\epsilon)$
    $\theta = \theta - \nabla_\theta \ell(f_\theta(x_i + \delta), y_i)$ *// Update model weights with some optimizer, e.g. SGD*
  end for
end for

Data-driven Defence

- Data-driven defence
  - Filtering instances: p ata in the training dataset or the adversarial samples against the t et either exhibit different statistical features, or follow a different d – detection
  - Injecting data: add adversarial samples into training – adversarial training
  - Projecting data: project data into lower-dimensional space; move adversarial samples closer to the manifold of legitimate samples

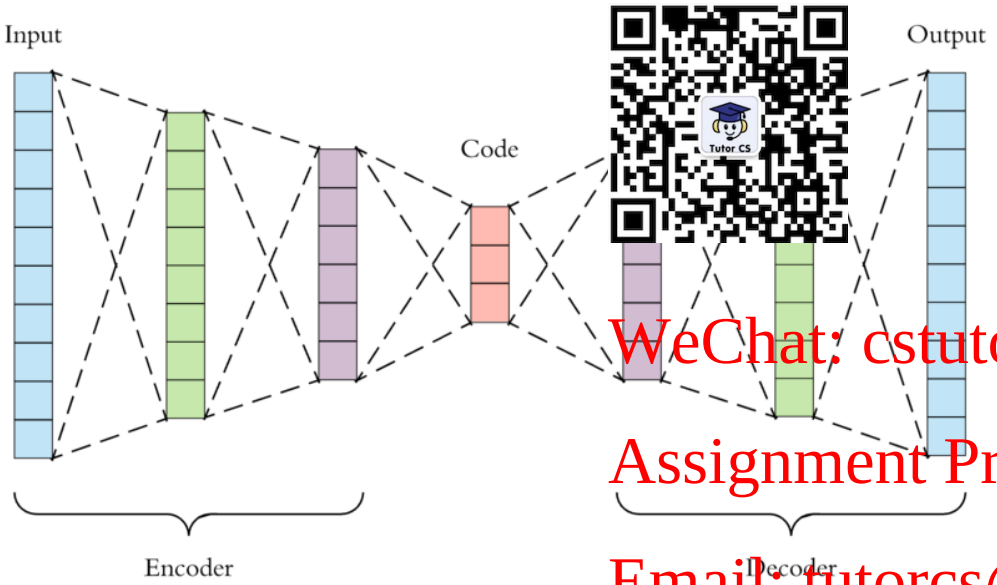
Data-driven Defence: Projecting Data

- Projecting data
  - Adversarial sample from low-density regions
  - Move adversarial samples back to the data manifold before classification
  - Use auto-encoder, GANs, PixelCNN to reform/purify the input

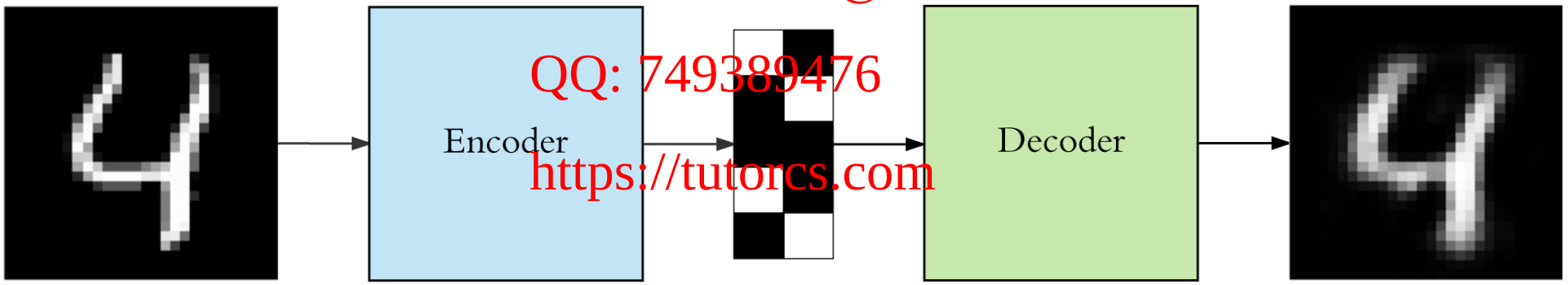


Data-driven Defence: Projecting Data

- Auto-encoder: get an output identical with the input



Input → code → output ≈ input



https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

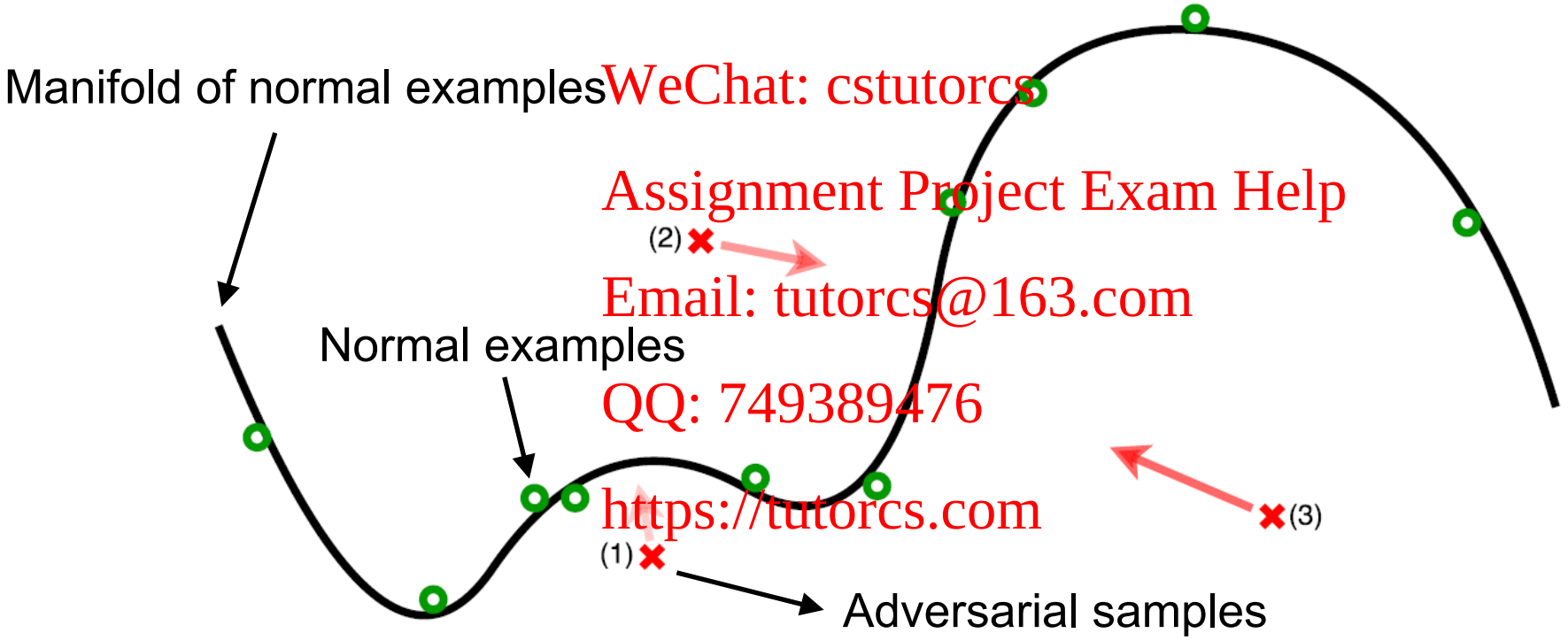
Data-driven Defence: Projecting Data

- MagNet: a Two-Pronged Defense against Adversarial Examples [6]
  - Use auto-encoders to detect and reform adversarial samples
  - Detector
    - Reconstruction error
      - Normal examples → small RE
      - Adversarial samples → large RE
      - Threshold: reject no more than 0.1% examples in validation set
    - Probability divergence
      - Normal examples → Small divergence btw $f(x)$ and $f\left(AE(x)\right)$
      - Adversarial samples → Large divergence btw $f(x')$ and $f\left(AE(x')\right)$

$AE(x)$: output of the auto-encoder
$f(x)$: output of the last layer (i.e., softmax) of the neural network f on the input x

Data-driven Defence: Projecting Data

- Reformer
  - Normal example outputs a very similar example
  - Adversarial sample outputs an example that is closer to the manifold of the normal examples



Data-driven Defence: Projecting Data

- Reformer

Data-driven Defence: Projecting Data

- Can you think of a way to break "MagNet"?
  - Hint: an adaptive  that attacks not only the classifier, but also the detec[illegible]pose there is only one detector) and the reformer.
  - $\arg\min_{\delta \in [0,1]^d} \|\delta\| + c \cdot f(x+\delta) \rightarrow ??$

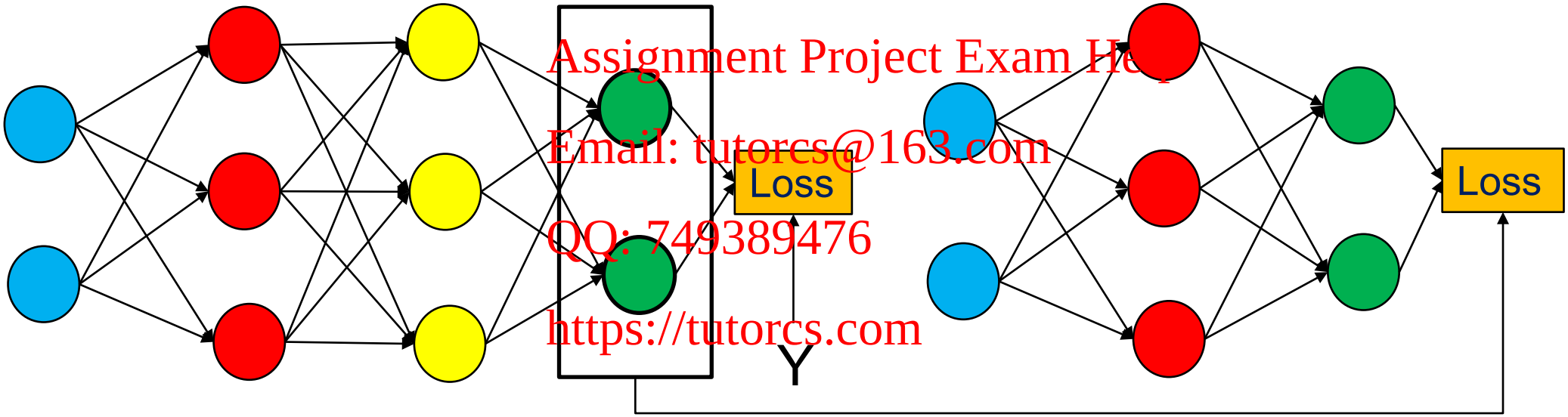
Overview

- Adversarial machine learning beyond computer vision
  - Audio
  - NLP
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defenses
  - Learner robustification
- Challenges

Learner Robustification: Distillation

- Distillation as a defense to adversarial perturbations against deep neural networks [8]
  - Distillation: transfer k[illegible] from one neural network to another – suppose there is a tra[illegible]l, the probabilities generated in the final softmax layer are used to train a second DNN, instead of the (hard) class labels

provide richer information about each class



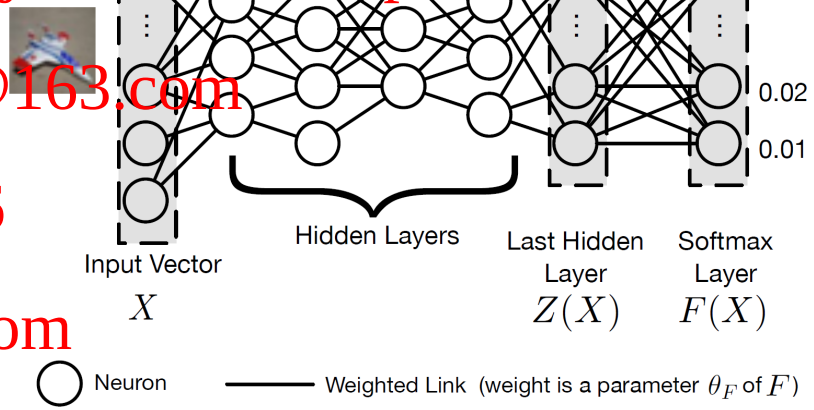
Learner Robustification: Distillation

- Distillation as a defense to adversarial perturbations against deep neural networks [8] (N[illegible]hot et al.)
  - Modification to the fir[illegible]x layer:

$$F_i(X) = \frac{e^{z_i(X)}}{\sum_{j=1}^{N} e^{z_j(X)}} \rightarrow F_i(X) = \frac{e^{\frac{z_i(X)}{T}}}{\sum_{j=1}^{N} e^{\frac{z_j(X)}{T}}}$$

$Z(X)$: output of the last hidden layer

T: *distillation temperature*



Learner Robustification: Distillation

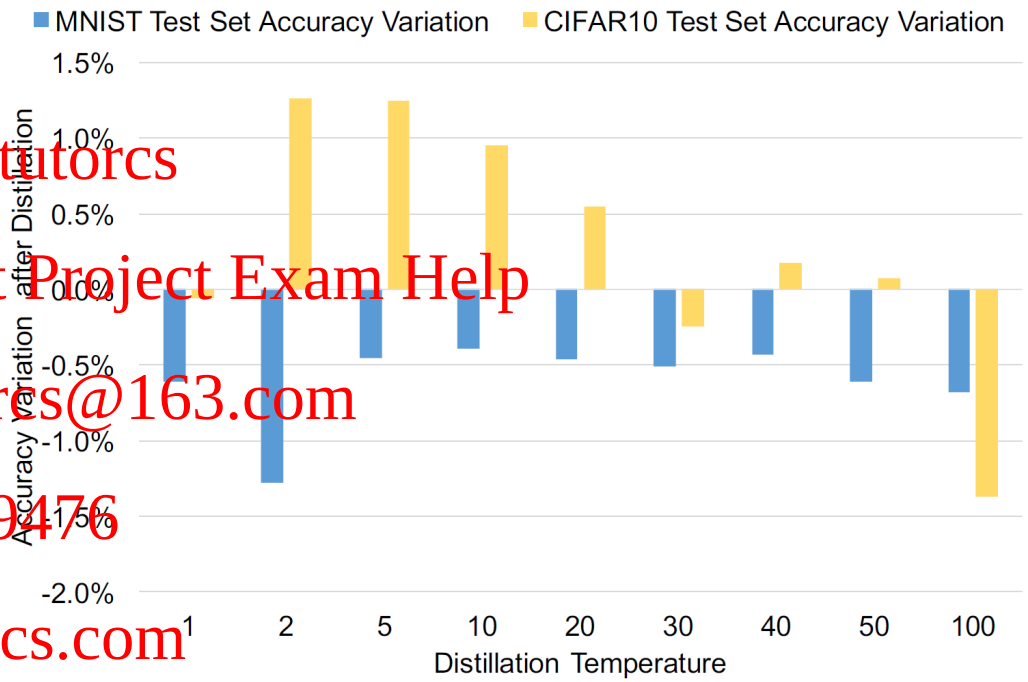
- Distillation as a defense to adversarial perturbations against deep neural networks [8]
  - Given a training s(*X*))}, train a DNN (*F*) with a softmax layer at temperat
  - Form a new training set {(*X*, *F*(*X*))}, train another DNN (F^D), with the same network architecture, also at temperature *T*
  - Test at temperature *T*=1
  - A high empirical value of *T* (at training time) gives a better performance (*T*=1 at test time)
  - F^D provides a smoother loss function – more generalised for an unknown dataset

Probability vector

Learner Robustification: Distillation

– Results on MNIST and CIFAR10

Distillation Temperature	MNIST Adversarial Samples Success Rate (%)	CIF… Sam… …al …ate
1	91	92.78
2	82.23	87.67
5	24.67	67
10	6.78	47.56
20	1.34	18.23
30	1.44	13.23
40	0.45	9.34
50	1.45	6.23
100	0.45	5.11
No distillation	95.89	87.89



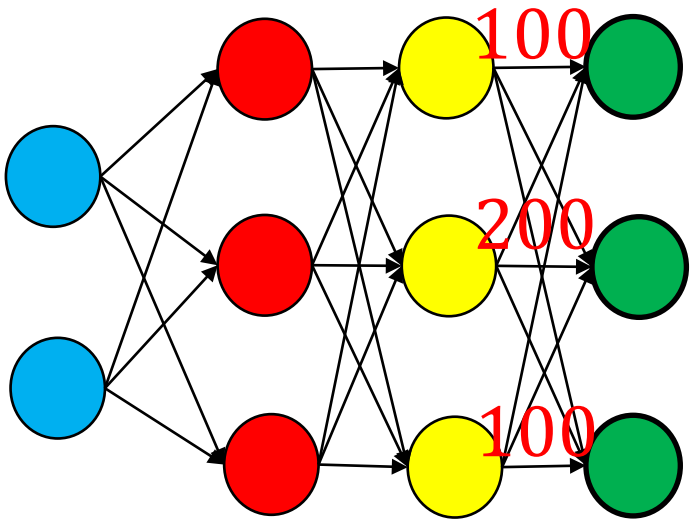
Effect against adversarial samples

Influence of distillation on clean data

Learner Robustification: Distillation

- Why does "a large temperature at training time (e.g. T=100) + a low temperature at test time (T=1)" make the model more secure?

- $F_i(X) = \frac{e^{\frac{z_i(X)}{T}}}{\sum_{j=1}^{N} e^{\frac{z_j(X)}{T}}}$



Training (T=100)

$$\frac{e}{e+e^2+e} = \frac{1}{1+e+1}$$

$$\frac{e^2}{e+e^2+e} = \frac{e}{1+e+1}$$

$$\frac{e}{e+e^2+e} = \frac{1}{1+e+1}$$

Test (T=1)

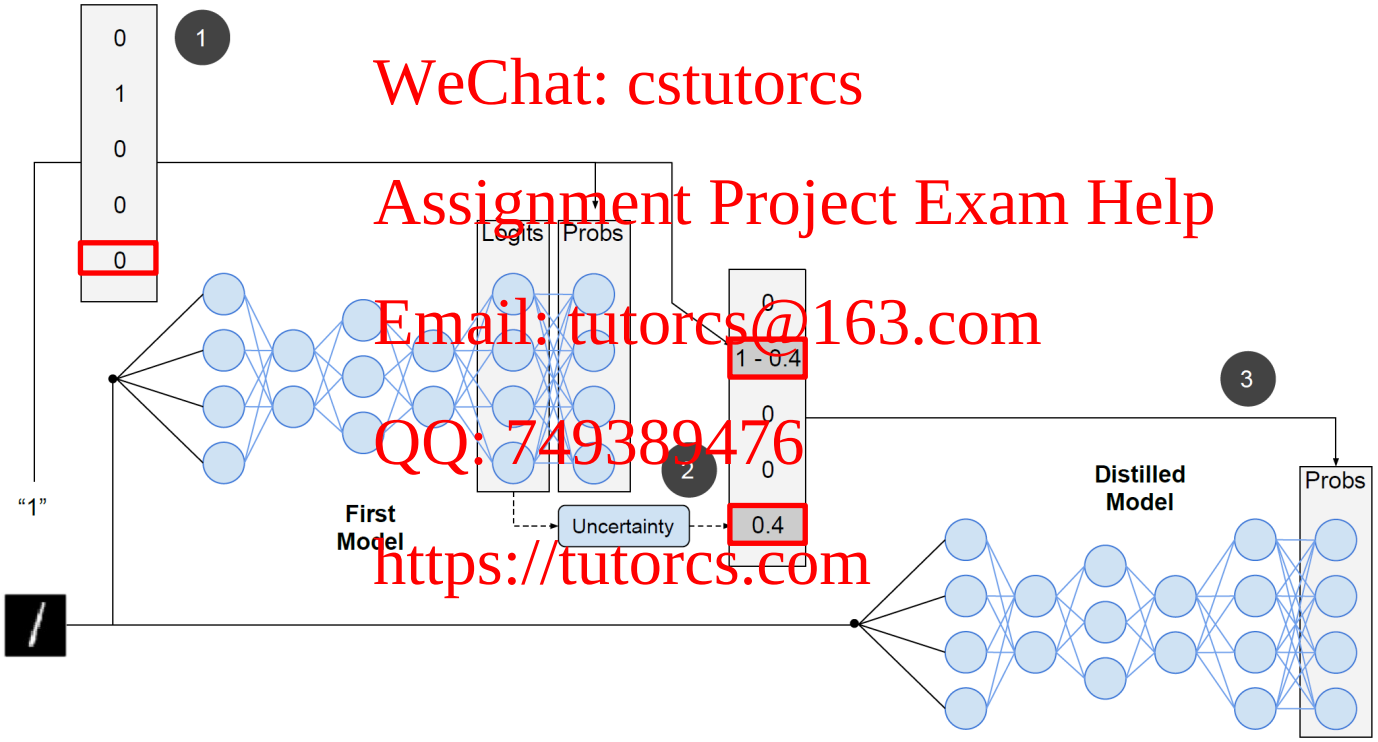
$$\frac{e^{100}}{e^{100}+e^{200}+e^{100}} = \frac{1}{1+e^{100}+1}$$

$$\frac{e^{200}}{e^{100}+e^{200}+e^{100}} = \frac{e^{100}}{1+e^{100}+1}$$

$$\frac{e^{100}}{e^{100}+e^{200}+e^{100}} = \frac{1}{1+e^{100}+1}$$

Learner Robustification: Distillation

- Extending Defensive Distillation [12]
- 1. The 1st DNN is trained on one-hot labels)
- 2. New labeling vector: or information + predictive uncertainty
- 3. The distilled model is tr the new label vectors.



Learner Robustification: Distillation

- Predictive uncertainty
  - Take *N* forward pass[illegible] the neural network with dropout
  - Record the *N* logit ve[illegible]), …, $z^{N-1}(x)$
  - Calculate uncertainty

$$\sigma(x) = \frac{1}{N} \sum_{m \in 0..N-1} \left(\sum_{j \in 0..n-1} \left(z_j^m(x) - \bar{z}_j \right)^2 \right)$$

  - New labeling vector *k*(*x*):

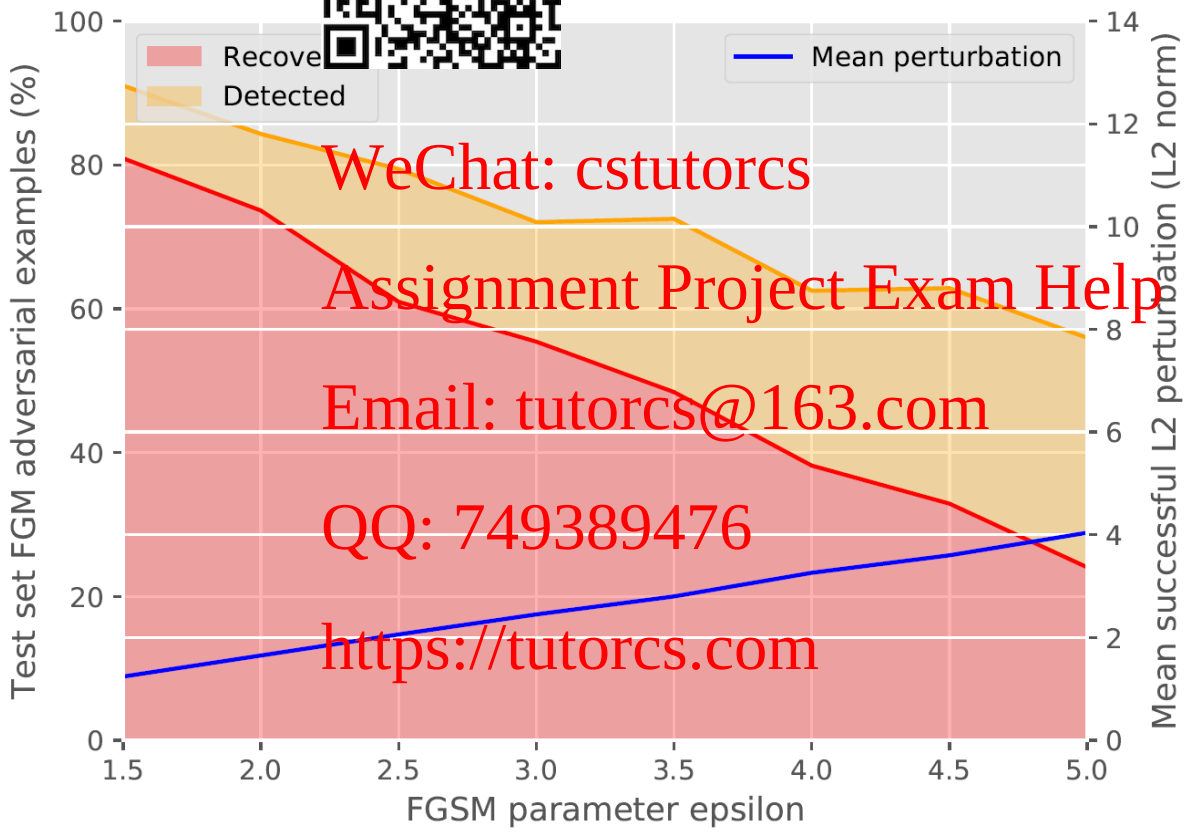
$$k_j(x) = \begin{cases} 1 - \alpha \cdot \dfrac{\sigma(x)}{\max_{x \in \mathcal{X}} \sigma(x)} & \text{if } j = l \text{ (correct class)} \\ \alpha \dfrac{\sigma(x)}{\max_{x \in \mathcal{X}} \sigma(x)} & \text{if } j = n \text{ (outlier class)} \\ 0 & \text{otherwise} \end{cases}$$

0.1
0.6
0.1
0.2

\+ $\alpha = 1, \max \sigma(x) = 0.4, \sigma(x) = 0.1$ → 0
1 – 0.25 = 0.75
0
0
0.25

Learner Robustification: Distillation

- White-box attack via FGSM
  - Recovered: adversar[illegible]hat are assigned to the original class
  - Detected: adversaria[illegible]s that are classified in the outlier class



Learner Robustification: Stability Training

- Improving the Robustness of Deep Neural Networks via Stability Training [9]
  - Stability objective: if x' is close to x, $f(x)$ should be close to $f(x')$

    $\forall x' : d(x, x') \text{ small} \leftrightarrow \big(f(x), f(x')\big) \text{ small}$
  - Define new training objective:

    $L(x, x'; \theta) = L_0(x; \theta) + \alpha D\big(f(x), f(x')\big)$, L_0: original training objective
  - New optimisation problem:

    $\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{d(x_i, x_i') < \epsilon} L(x_i, x_i'; \theta)$
  - Generate x': adds pixel-wise uncorrelated Gaussian noise ϵ to x

    $x_k' = x_k + \epsilon_k, \quad \epsilon_k \sim \mathcal{N}\big(0, \sigma_k^2\big), \quad \sigma_k > 0$
  - L_0, D are task specific, e.g., L_0: cross-entropy loss, D: KL-divergence

Overview

- Adversarial machine learning beyond computer vision
  - Audio
  - NLP
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defenses
  - Learner robustification
- Challenges

Adversarial Machine Learning – Challenges

- Arm race between attackers and defenders [10][11]
- Many defence metho
  - Evaluate against a st k, e.g., PGD, C&W
  - Evaluate against an a ttacker
    - Should not assume the attacker is unaware of the defence method
    - $\arg\min_{\delta \in [0,1]^d} \|\delta\|_{2/\infty}^2 + c \cdot f_{target}(x+\delta)$
  - Evaluate on complicated dataset like CIFAR, ImageNet
    - Evaluating solely on MNIST is insufficient
  - Define a realistic threat model – what is known & unknown to the attacker
    - Model architecture and model weights
    - Training algorithm and training data
    - Test time randomness
    - White-box – grey-box – black-box

Summary

- Adversarial machine learning beyond computer vision
  - Audio
  - NLP
  - Malware detection
- Why are machine learning models vulnerable?
  - Insufficient training data
  - Unnecessary features
- How to defend against adversarial machine learning?
  - Data-driven defences
    - Filtering adversarial samples
    - Adversarial training
    - Project to lower dimension
  - Learner robustification
    - Distillation
    - Stability training

References

- [1] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting Adversarial Samples from Artifacts," eprint arXiv:1703.00410, 2017.
- [2] A. Nguyen, J. Yosinski[illegible]une, "Deep Neural Networks are Easily Fooled: High Confidence [illegible]s for Unrecognizable Images," in *CVPR*, 2015.
- [3] B. Wang, J. Gao, and Y. Qi, "A Theoretical Framework for Robustness of (Deep) Classifiers against Adversarial Examples," *eprint arXiv:1612.00334*, 2016.
- [4] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On Detecting Adversarial Perturbations," *eprint arXiv:1702.04267*, 2017.
- [5] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks," *arXiv:1706.06083*, 2017.
- [6] D. Meng and H. Chen, "MagNet: a Two-Pronged Defense against Adversarial Examples," *arXiv:1705.09064,* 2017.
- [7] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial Perturbations Against Deep Neural Networks for Malware Classification," eprint arXiv:1606.04435, 2016.

References

- [8] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks," *eprint arXiv:1511.04508*, 2015.
- [9] S. Zheng, Y. Song, T. L[illegible] I. Goodfellow, "Improving the Robustness of Deep Neural Networks via [illegible]aining," *eprint arXiv:1604.04326*, 2016.
- **[10] N. Carlini and D. Wagner, "Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods," arXiv:1705.07263, 2017.**
- [11] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples," arXiv:1802.00420 [cs], Feb. 2018.
- [12] N. Papernot and P. McDaniel, "Extending Defensive Distillation," *arXiv:1705.05264 [cs, stat]*, May 2017.
- [13] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial Perturbations Against Deep Neural Networks for Malware Classification," *eprint arXiv:1606.04435*, 2016.
- [14] Nicholas Carlini and David Wagner. Audio Adversarial Examples: Targeted Attacks on Speech-to-Text. arXiv 1801.01944, 2018

References

- [15] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In Advances in neural information processing systems, pages 649–657, 2
- [16] Bin Liang, Hongcheng ang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classi n be fooled. In Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18). AAAI Press, 4208–4215.
- [17] Qi-Zhi Cai, Chang Liu, and Dawn Song. 2018. Curriculum adversarial training. In Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18). AAAI Press, 3740–3747.
- [18] Eric Wong, Leslie Rice and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. arXiv:2001.03994 [cs.LG], 2020.
- [19] Michael McCloskey and Neal J. Cohen. 1989. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In Psychology of Learning and Motivation. Vol. 24. Academic Press, 109 – 165.
- [20] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, Tom Goldstein: Adversarial training for free! NeurIPS 2019: 3353-3364