AAAI 2024 Tutorial: Trustworthy ML with Imperfect Data

Trustworthy Machine Learning on Adversarial Data

Lecturer: Dr. Feng Liu

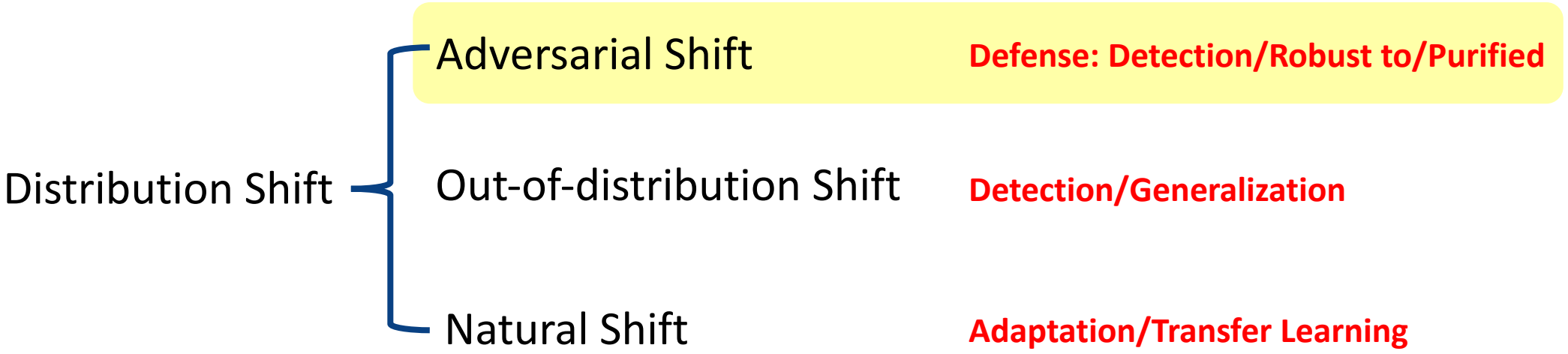
School of Computing and Information Systems

The University of Melbourne

Date: 21 Feb 2024, Vancouver, Canada

About the Lecturer

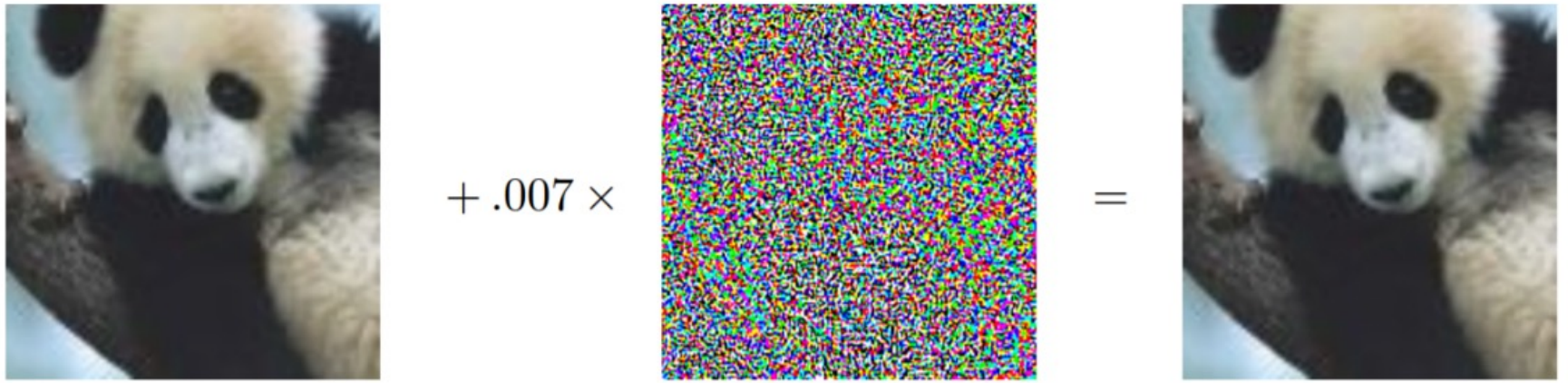
- **Name:** Feng Liu
- **Position:** Assistant Professor in Machine Learning
- **Institution:** School of Computing and Information Systems, University of Melbourne
- **Research Interests:** Distribution Shift Detection, Learning under Distribution Shift



What is the adversarial data?

Adversarial attacks

From: Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015.

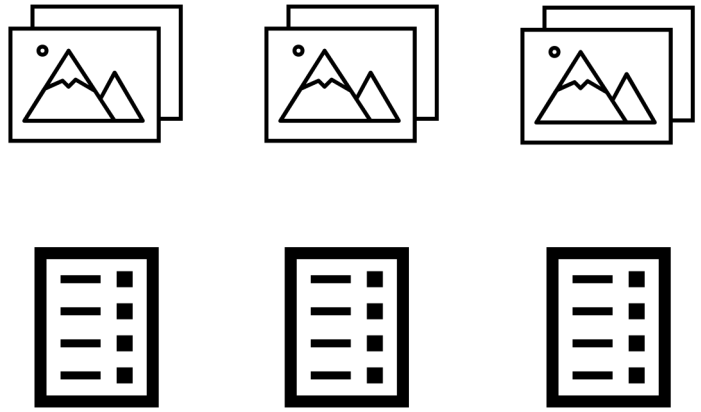


What is the adversarial data?

TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Conventional Machine Learning Pipeline (classification):



Training Data with labels

Train a model

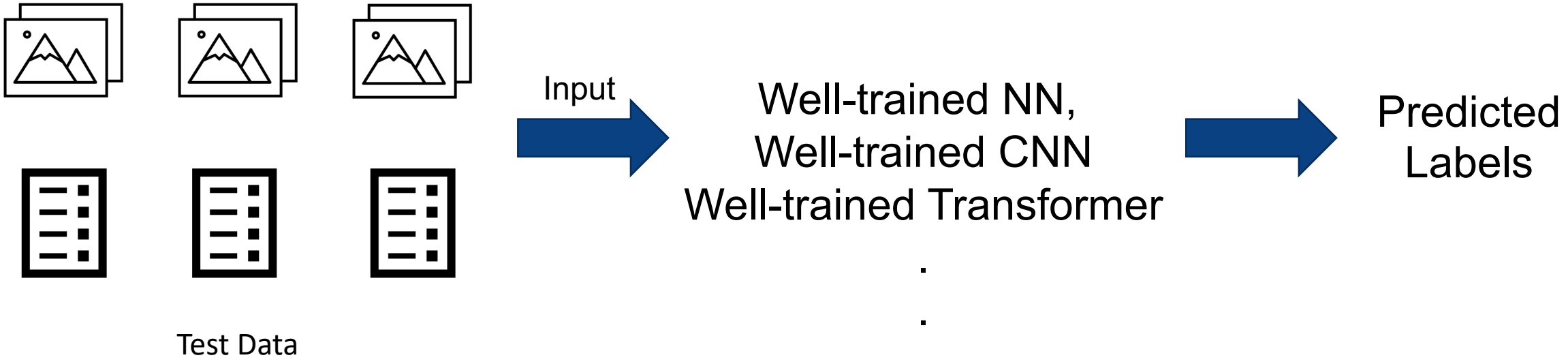
NN,
CNN
Transformer
.
.

What is the adversarial data?

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Conventional Machine Learning Pipeline (classification):

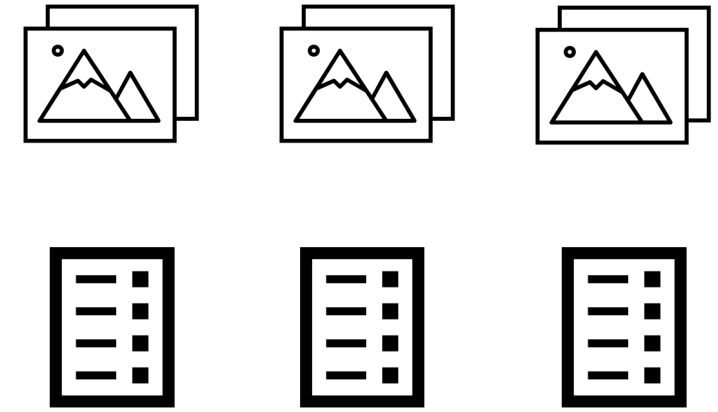


What is the adversarial data?

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial attack happens:



Test Data + Adversarial **Perturbations**

Input

Well-trained NN,
Well-trained CNN
Well-trained Transformer
.
.

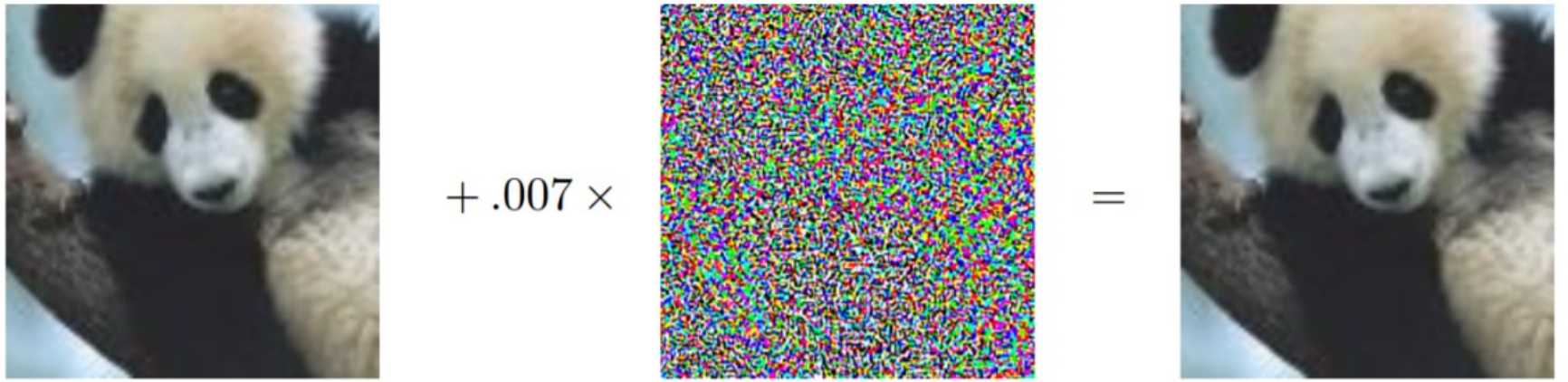
Predicted Labels

Untrusted

Why adversarial attack can be successful

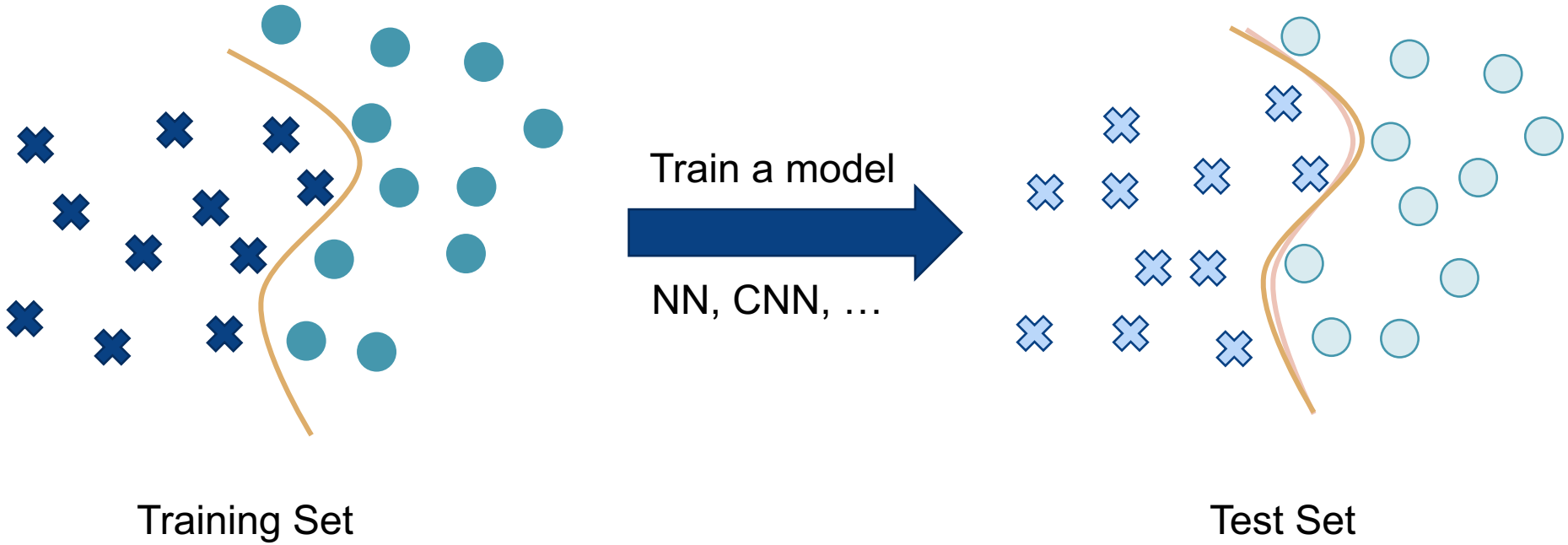
Adversarial attacks

From: Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015.

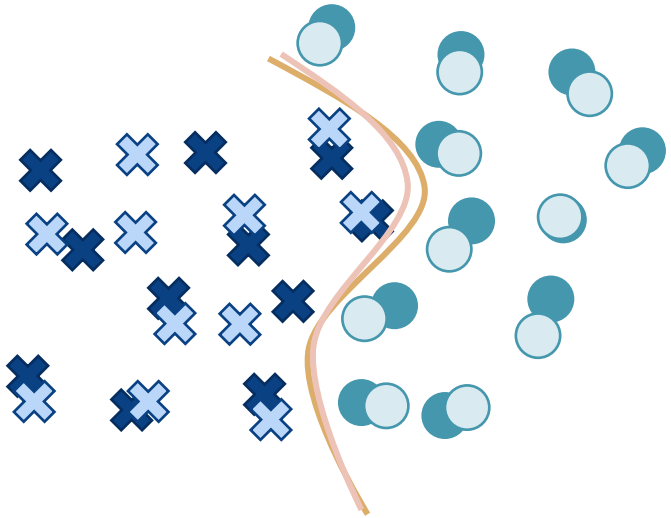


Different distributions → Significant Error

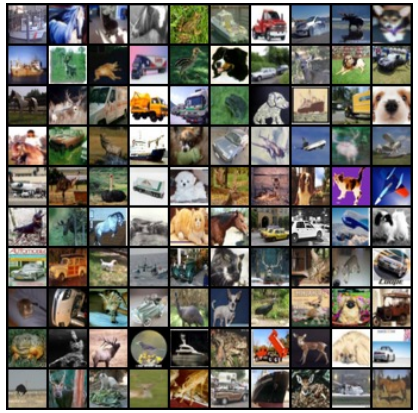
Basic Assumption in Machine Learning



Basic Assumption in Machine Learning

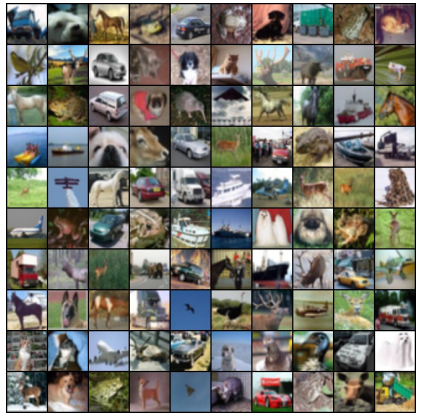


Same Distribution ?



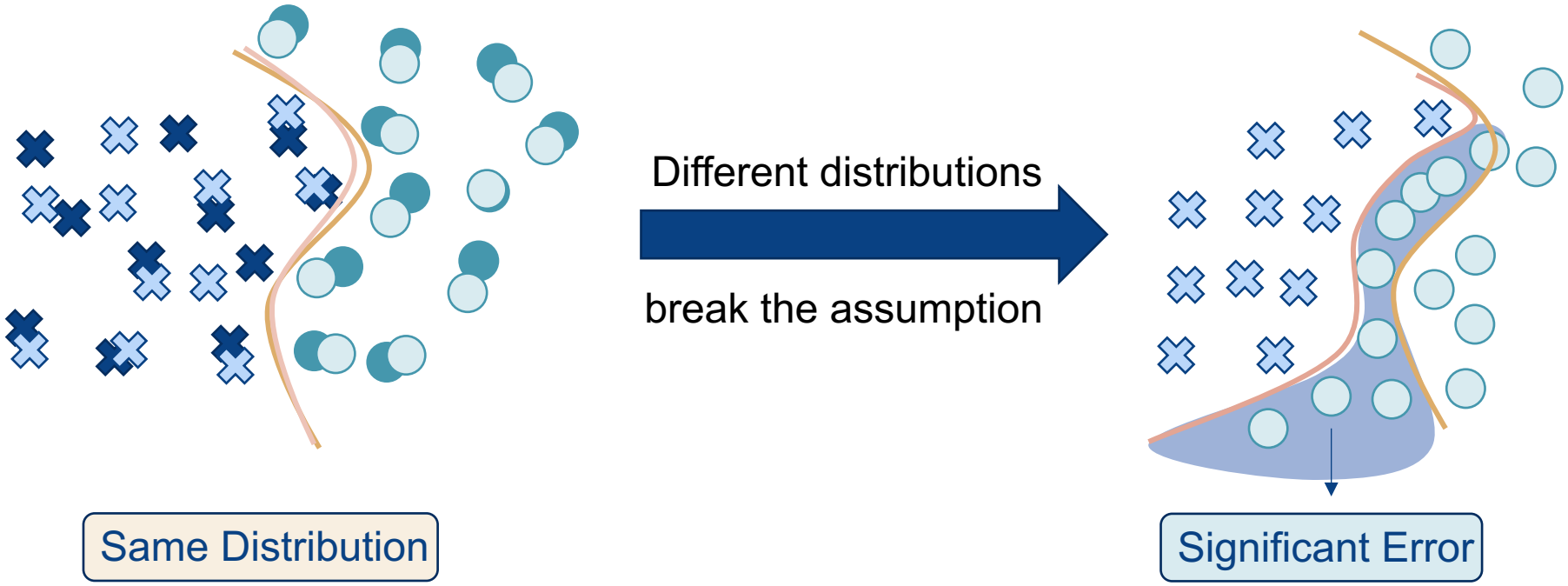
CIFAR10

CIFAR10.1 [ICML'19]



Basic Assumption in Machine Learning

Basic Assumption in Machine Learning

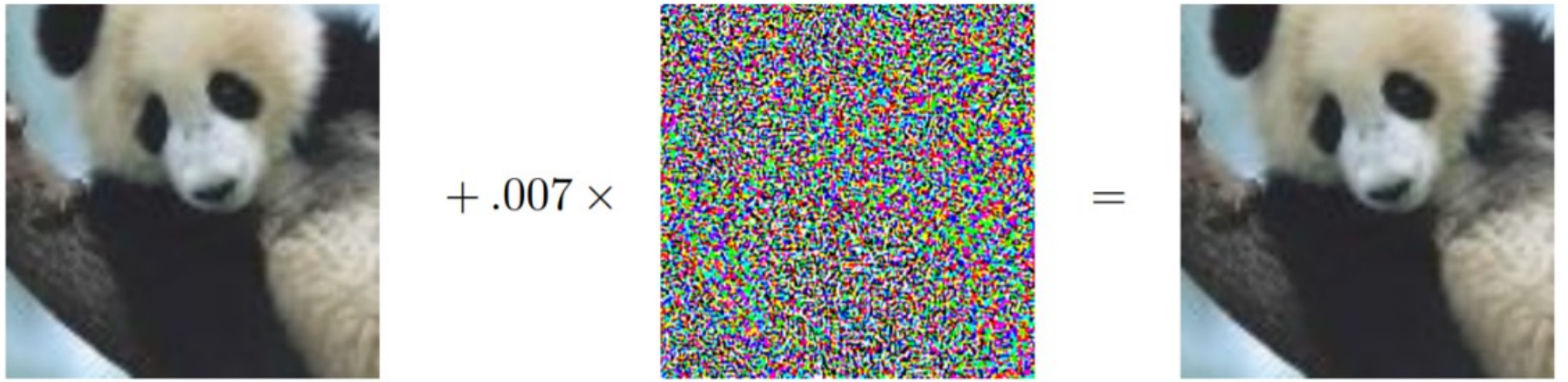


Basic Assumption in Machine Learning

Why adversarial attack can be successful

Adversarial attacks

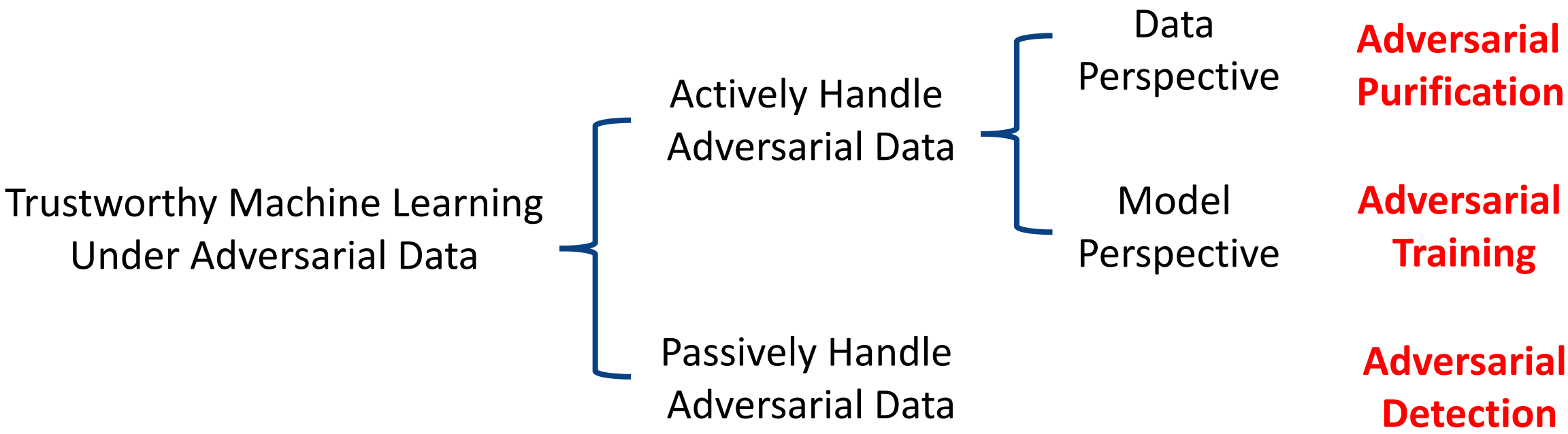
From: Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015.



Different distributions → Significant Error

Gao et al., Maximum Mean Discrepancy Test is Aware of Adversarial Attacks. ICML, 2021.

How to prevent from adversarial attacks

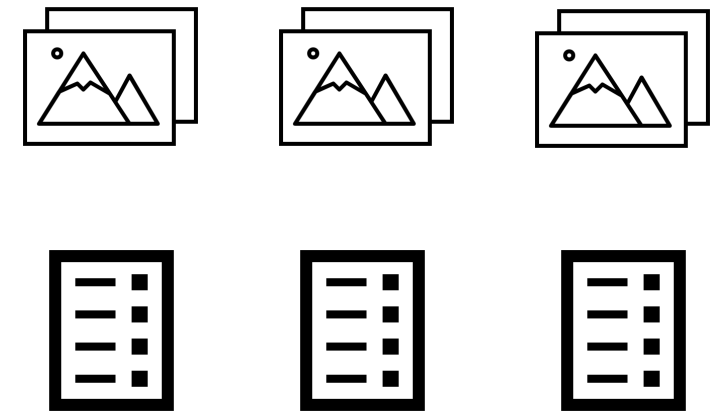


How to actively prevent from adversarial attacks

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial attack happens:



Test Data + Adversarial **Perturbations**

Purify

Input

Train a robust model

Well-trained NN,
Well-trained CNN
Well-trained Transformer
.
.

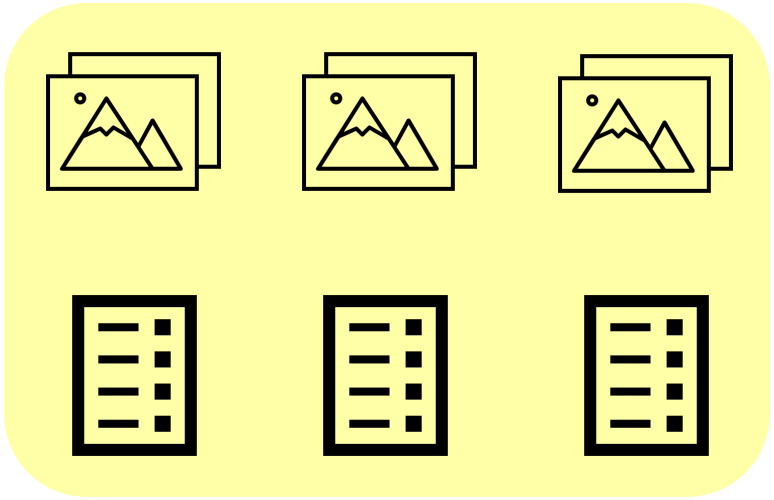
Predicted Labels

How to passively prevent from adversarial attacks

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial attack happens:



Test Data + Adversarial **Perturbations**

Discard the adversarial data

Input

Well-trained NN,
Well-trained CNN
Well-trained Transformer
.
.

Predicted Labels

ACML 2023 Tutorial: Trustworthy Machine Learning on Adversarial Data

Part I: Adversarial Purification

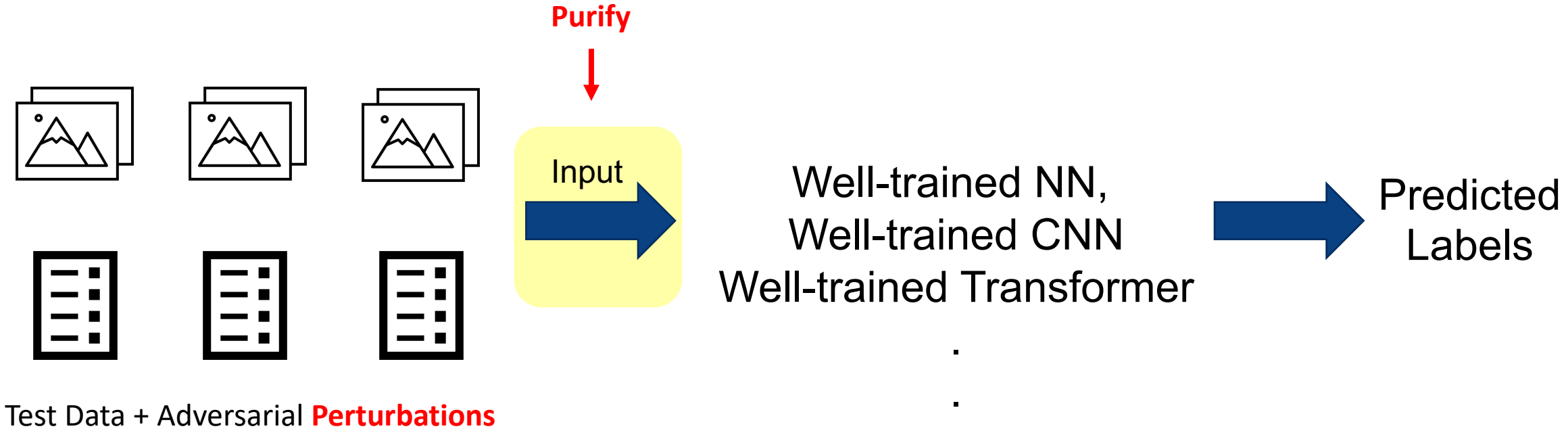
Let's try purify the upcoming test data!!

How to actively prevent from adversarial attacks

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial attack happens:



Adversarial Purification: General Pipeline

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Research trajectory: use the power of ***generative models*** to purify adversarial examples.

High-level idea: use **a *pre-trained generative model on natural examples*** to reconstruct adversarial examples into samples that are close to natural examples.

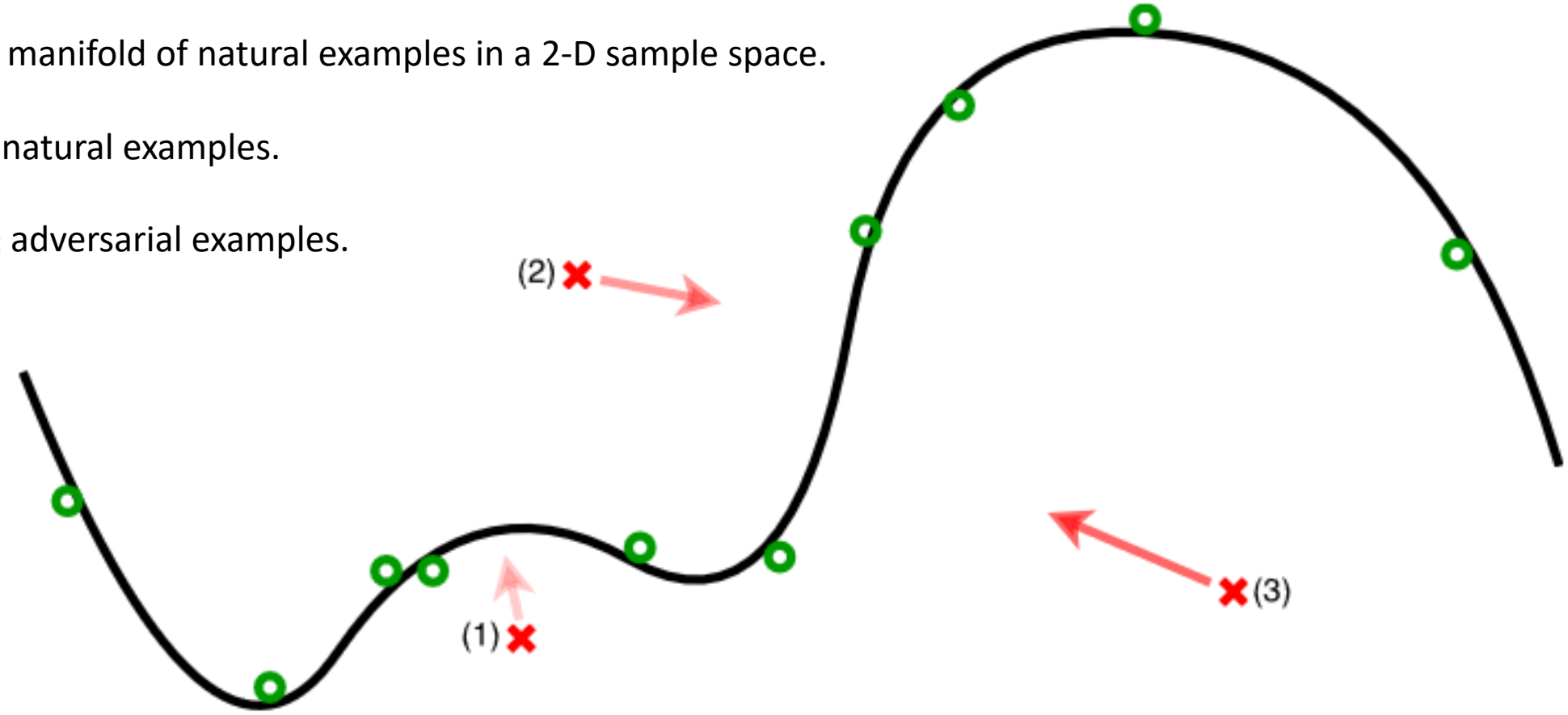
Intuition: a generative model reconstructs samples based on its learned knowledge (e.g., the manifold of natural examples). Thus, **a *pre-trained generative model on natural examples*** will align any samples to the manifold of natural examples during the reconstruction.

Adversarial Purification: General Pipeline

Black curve: manifold of natural examples in a 2-D sample space.

Green dots: natural examples.

Red crosses: adversarial examples.



Meng & Chen, MagNet: a Two-Pronged Defense against Adversarial Examples. ACM CCS, 2017.

Adversarial Purification: MagNet

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

MagNet: a Two-Pronged Defense against Adversarial Examples

Main idea: use the reconstructive power of ***an autoencoder*** to purify (i.e., reform) adversarial examples.

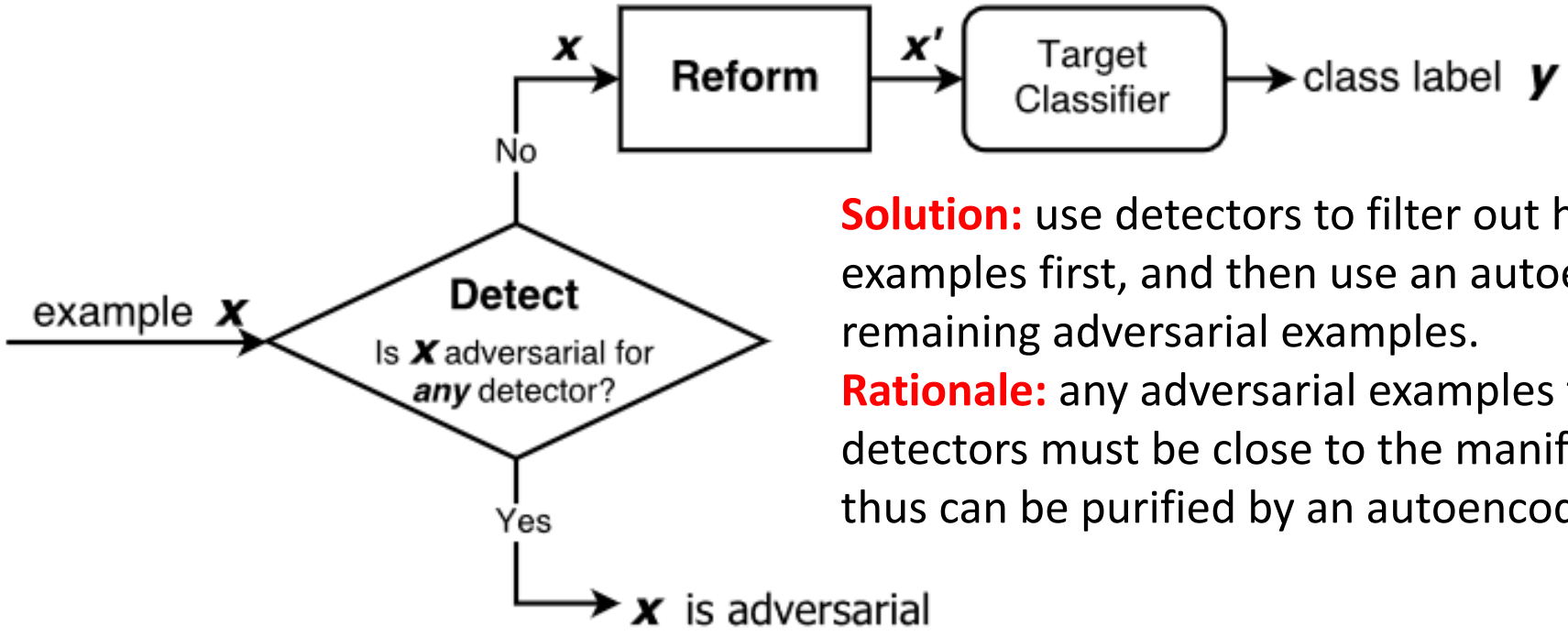
Limitation: the autoencoder is not strong enough, i.e., cannot purify examples that are too far away from the manifold of natural examples.

We can use detectors to filter out hard-to-purify adversarial examples first!

Meng & Chen, MagNet: a Two-Pronged Defense against Adversarial Examples. ACM CCS, 2017.

Adversarial Purification: MagNet

MagNet: a Two-Pronged Defense against Adversarial Examples



Solution: use detectors to filter out hard-to-purify adversarial examples first, and then use an autoencoder to purify the remaining adversarial examples.

Rationale: any adversarial examples that cannot be detected by detectors must be close to the manifold of natural examples, and thus can be purified by an autoencoder.

Meng & Chen, MagNet: a Two-Pronged Defense against Adversarial Examples. ACM CCS, 2017.

Adversarial Purification: MagNet

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

MagNet: a Two-Pronged Defense against Adversarial Examples

(a) MNIST

Attack	Norm	Parameter	No Defense	With Defense
FGSM	L^∞	$\epsilon = 0.005$	96.8%	100.0%
FGSM	L^∞	$\epsilon = 0.010$	91.1%	100.0%
Iterative	L^∞	$\epsilon = 0.005$	95.2%	100.0%
Iterative	L^∞	$\epsilon = 0.010$	72.0%	100.0%
Iterative	L^2	$\epsilon = 0.5$	86.7%	99.2%
Iterative	L^2	$\epsilon = 1.0$	76.6%	100.0%
Deepfool	L^∞		19.1%	99.4%
Carlini	L^2		0.0%	99.5%
Carlini	L^∞		0.0%	99.8%
Carlini	L^0		0.0%	92.0%

(b) CIFAR

Attack	Norm	Parameter	No Defense	With Defense
FGSM	L^∞	$\epsilon = 0.025$	46.0%	99.9%
FGSM	L^∞	$\epsilon = 0.050$	40.5%	100.0%
Iterative	L^∞	$\epsilon = 0.010$	28.6%	96.0%
Iterative	L^∞	$\epsilon = 0.025$	11.1%	99.9%
Iterative	L^2	$\epsilon = 0.25$	18.4%	76.3%
Iterative	L^2	$\epsilon = 0.50$	6.6%	83.3%
Deepfool	L^∞		4.5%	93.4%
Carlini	L^2		0.0%	93.7%
Carlini	L^∞		0.0%	83.0%
Carlini	L^0		0.0%	77.5%

Q: Can we purify adversarial examples without filtering out those hard-to-purify ones?

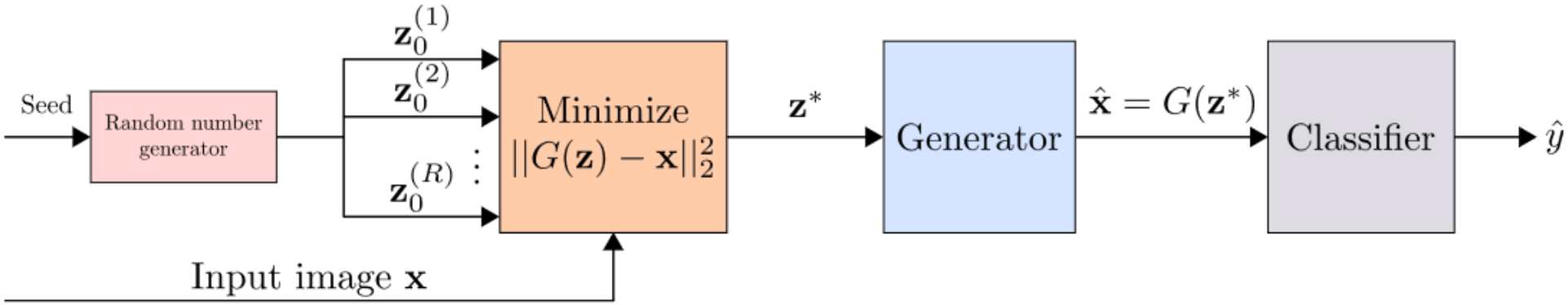
A: Yes! A natural idea is to use a stronger generative model.

Meng & Chen, MagNet: a Two-Pronged Defense against Adversarial Examples. ACM CCS, 2017.

Adversarial Purification: Defense GAN

Protecting Classifiers Against Adversarial Attacks Using Generative Models

Solution: use a pre-trained GAN to reconstruct adversarial examples.

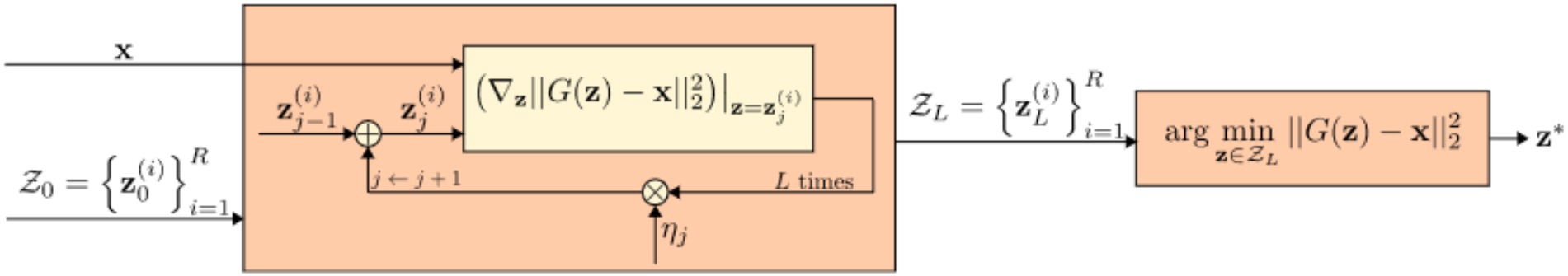


Samangouei et al., *Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. ICLR, 2018.*

Adversarial Purification: Defense GAN

Protecting Classifiers Against Adversarial Attacks Using Generative Models

Minimization: L steps of Gradient Descent are used to estimate the projection of the image onto the range of the generator.



Samangouei et al., *Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. ICLR, 2018.*

Adversarial Purification: Defense GAN

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Protecting Classifiers Against Adversarial Attacks Using Generative Models

On MNIST:

Attack	Classifier Model	No Attack	No Defense	**Defense-GAN-Rec**	MagNet	Adv. Tr. $\epsilon = 0.3$
FGSM $\epsilon = 0.3$	A	0.997	0.217	**0.988**	0.191	0.651
	B	0.962	0.022	**0.956**	0.082	0.060
	C	0.996	0.331	**0.989**	0.163	0.786
	D	0.992	0.038	**0.980**	0.094	0.732
RAND+FGSM $\epsilon = 0.3, \alpha = 0.05$	A	0.997	0.179	**0.988**	0.171	0.774
	B	0.962	0.017	**0.944**	0.091	0.138
	C	0.996	0.103	**0.985**	0.151	0.907
	D	0.992	0.050	**0.980**	0.115	0.539
CW ℓ_2 norm	A	0.997	0.141	**0.989**	0.038	0.077
	B	0.962	0.032	**0.916**	0.034	0.280
	C	0.996	0.126	**0.989**	0.025	0.031
	D	0.992	0.032	**0.983**	0.021	0.010

Samangouei et al., *Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. ICLR, 2018.*

Adversarial Purification: Defense GAN

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Protecting Classifiers Against Adversarial Attacks Using Generative Models

On F-MNIST:

Attack	Classifier Model	No Attack	No Defense	**Defense-GAN-Rec**	MagNet	Adv. Tr. $\epsilon = 0.3$
FGSM $\epsilon = 0.3$	A	0.934	0.102	**0.879**	0.089	0.797
	B	0.747	0.102	**0.629**	0.168	0.136
	C	0.933	0.139	**0.896**	0.110	0.804
	D	0.892	0.082	**0.875**	0.099	0.698
RAND+FGSM $\epsilon = 0.3, \alpha = 0.05$	A	0.934	0.102	**0.888**	0.096	0.447
	B	0.747	0.131	**0.661**	0.161	0.119
	C	0.933	0.105	**0.893**	0.112	0.699
	D	0.892	0.091	**0.862**	0.104	0.626
CW ℓ_2 norm	A	0.934	0.076	**0.896**	0.060	0.157
	B	0.747	0.172	**0.656**	0.131	0.118
	C	0.933	0.063	**0.896**	0.084	0.107
	D	0.892	0.090	**0.875**	0.069	0.149

Samangouei et al., *Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. ICLR, 2018.*

Adversarial Purification: PixelDefend

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Leveraging Generative Models to Understand and Defend against Adversarial Examples

Solution: use a pre-trained PixelCNN to reconstruct adversarial examples.

PixelCNN is a generative model with tractable likelihood especially designed for images. The model defines the joint distribution over all pixels by factorizing it into a product of conditional distributions.

$$p_{\text{CNN}}(\mathbf{X}) = \prod_i p_{\text{CNN}}(x_i | x_{1:(i-1)}).$$

Song et al., *Leveraging Generative Models to Understand and Defend against Adversarial Exmaples. ICLR, 2018.*

Adversarial Purification: PixelDefend

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Leveraging Generative Models to Understand and Defend against Adversarial Examples

Algorithm 1 PixelDefend

Input: Image $\mathbf{X}$, Defense parameter ϵ_{defend}, Pre-trained PixelCNN model p_{CNN}
Output: Purified Image $\mathbf{X}^*$

$\mathbf{X}^* \leftarrow \mathbf{X}$
for each row i **do**
  for each column j **do**
    for each channel k **do**
      $x \leftarrow \mathbf{X}[i, j, k]$
      Set feasible range $R \leftarrow [\max(x - \epsilon_{\text{defend}}, 0), \min(x + \epsilon_{\text{defend}}, 255)]$
      Compute the 256-way softmax $p_{\text{CNN}}(\mathbf{X}^*)$.
      Update $\mathbf{X}^*[i, j, k] \leftarrow \arg\max_{z \in R} p_{\text{CNN}}[i, j, k, z]$
    end for
  end for
end for

Song et al., *Leveraging Generative Models to Understand and Defend against Adversarial Exmaples. ICLR, 2018.*

Adversarial Purification: PixelDefend

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Leveraging Generative Models to Understand and Defend against Adversarial Examples

Table 1: **Fashion MNIST** ($\epsilon_{\text{attack}} = 8/25$, $\epsilon_{\text{defend}} = 32$)

NETWORK	TRAINING TECHNIQUE	CLEAN	RAND	FGSM	BIM	DEEP FOOL	CW	STRONGEST ATTACK
ResNet	Normal	**93**/**93**	89/71	38/24	00/00	06/06	20/01	00/00
VGG	Normal	92/92	91/87	73/58	36/08	49/14	43/23	36/08
ResNet	Adversarial FGSM	**93**/**93**	**92**/89	85/85	51/00	63/07	67/21	51/00
	Adversarial BIM	92/91	**92**/**91**	84/79	76/63	82/72	81/70	76/63
	Label Smoothing	**93**/**93**	91/76	73/45	16/00	29/06	33/14	16/00
	Feature Squeezing	84/84	84/70	70/28	56/25	83/83	83/83	56/25
	Adversarial FGSM + Feature Squeezing	88/88	87/82	80/77	70/46	86/82	84/85	70/46
ResNet	Normal + ***PixelDefend***	88/88	88/89	85/74	83/76	87/87	87/87	83/74
VGG	Normal + ***PixelDefend***	89/89	89/89	87/82	85/83	88/88	88/88	85/82
ResNet	Adversarial FGSM + ***PixelDefend***	90/89	91/90	**88**/82	**85**/76	**90**/88	**89**/**88**	**85**/76
	Adversarial FGSM +***Adaptive PixelDefend***	91/91	91/**91**	**88**/**88**	**85**/**84**	89/**90**	**89**/84	**85**/**84**

Song et al., *Leveraging Generative Models to Understand and Defend against Adversarial Exmaples. ICLR, 2018.*

Adversarial Purification: PixelDefend

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Leveraging Generative Models to Understand and Defend against Adversarial Examples

Table 2: **CIFAR-10** ($\epsilon_{\text{attack}} = 2/8/16$, $\epsilon_{\text{defend}} = 16$)

NETWORK	TRAINING TECHNIQUE	CLEAN	RAND	FGSM	BIM	DEEP FOOL	CW	STRONGEST ATTACK
ResNet	Normal	**92/92/92**	**92**/87/76	33/15/11	10/00/00	12/06/06	07/00/00	07/00/00
VGG	Normal	89/89/89	89/88/80	60/46/30	44/02/00	57/25/11	37/00/00	37/00/00
ResNet	Adversarial FGSM	91/91/91	90/**88**/84	**88/91/91**	24/07/00	45/00/00	20/00/07	20/00/00
	Adversarial BIM	87/87/87	87/87/86	80/52/34	74/32/06	79/48/25	76/42/08	74/32/06
	Label Smoothing	**92/92/92**	91/**88**/77	73/54/28	59/08/01	56/20/10	30/02/02	30/02/01
	Feature Squeezing	84/84/84	83/82/76	31/20/18	13/00/00	75/75/75	78/78/78	13/00/00
	Adversarial FGSM + Feature Squeezing	86/86/86	85/84/81	73/67/55	55/02/00	**85/85/85**	83/83/83	55/02/00
ResNet	Normal + ***PixelDefend***	85/85/88	82/83/84	73/46/24	71/46/25	80/80/80	78/78/78	71/46/24
VGG	Normal + ***PixelDefend***	82/82/82	82/82/84	80/62/52	80/61/48	81/76/76	81/79/79	80/61/48
ResNet	Adversarial FGSM + ***PixelDefend***	88/88/86	86/86/**87**	81/68/67	**81**/69/**56**	**85/85/85**	**84/84/84**	**81**/69/**56**
	Adversarial FGSM + ***Adaptive PixelDefend***	90/90/90	86/87/**87**	81/70/67	**81/70/56**	82/81/82	81/80/81	**81/70/56**

Song et al., *Leveraging Generative Models to Understand and Defend against Adversarial Exmaples. ICLR, 2018.*

Adversarial Purification based on Score

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial Purification with Score-based Generative Models

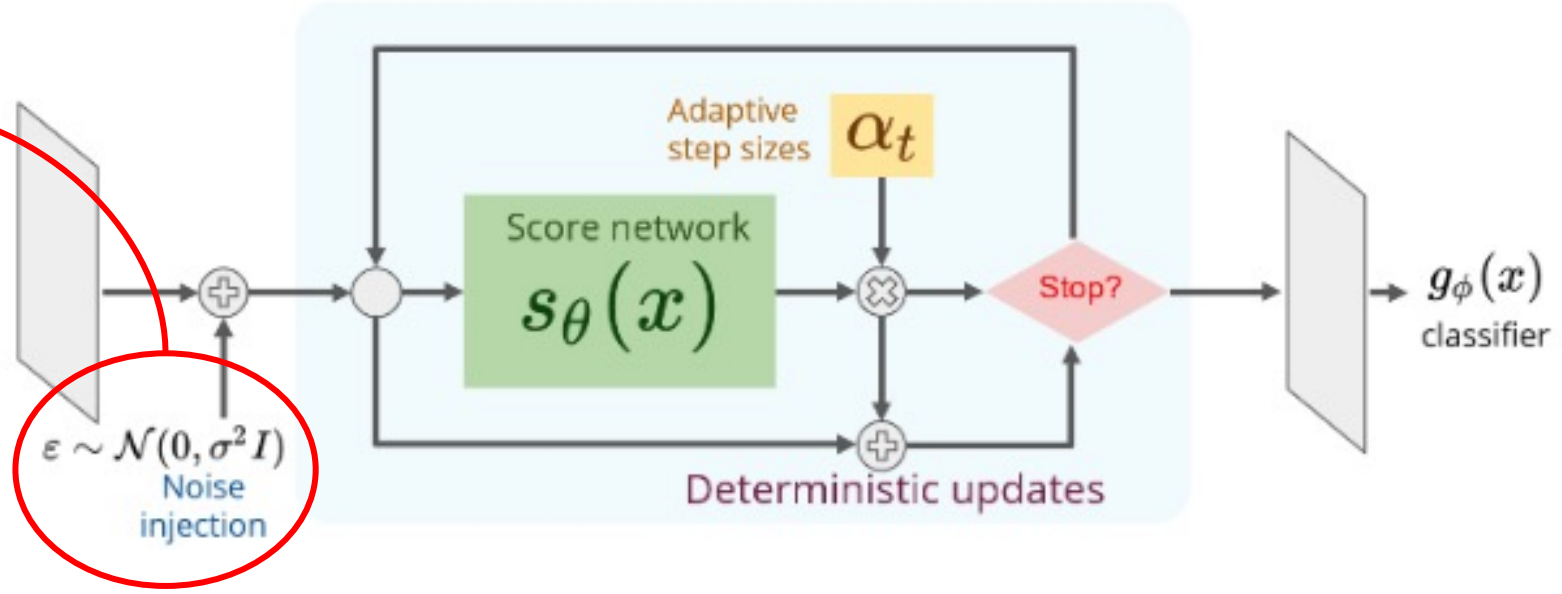
Motivation: if we treat adversarial purification as a process of denoising adversarial attacks, why not try a generative model that is designed to denoise the perturbed samples?

Solution: use a pre-trained generative model with Denoising Score Matching (DSM), i.e., the score network, which is particularly good at denoising perturbed images with noise.

Yoon et al., Adversarial Purification with Score-based Generative Models. ICML, 2021.

Adversarial Purification based on Score

A breakthrough: injecting noise (e.g., Gaussian noise) into adversarial examples before feeding into the score network.



Benefits of injecting noise to adversarial examples:

1. by injecting relatively larger noise, we can make the adversarial perturbations negligible.
2. convert adversarial images to noisy images that are similar to the ones seen by the network.

Yoon et al., Adversarial Purification with Score-based Generative Models. *ICML, 2021.*

Adversarial Purification based on Score

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

on CIFAR-10:

Models	Accuracy (%) Standard	Robust	Architecture
Raw WideResNet	95.80	0.00	WRN-28-10
ADP ($\sigma = 0.1$)	93.09	85.45	WRN-28-10
ADP ($\sigma = 0.25$)	86.14	80.24	WRN-28-10
Adversarial purification methods			
(Hill et al., 2021)	84.12	78.91	WRN-28-10
(Shi et al., 2021)*	96.93	63.10	WRN-28-10
(Du & Mordatch, 2019)*	48.7	37.5	WRN-28-10
(Grathwohl et al., 2020)*	75.5	23.8	WRN-28-10
(Yang et al., 2019)*			
$p = 0.8 \rightarrow 1.0$	94.9	82.5	ResNet-18
$p = 0.6 \rightarrow 0.8$	92.1	80.3	ResNet-18
$p = 0.4 \rightarrow 0.6$	89.2	77.4	ResNet-18
(Song et al., 2018)*			
Natural + PixelCNN	82	61	ResNet-62
AT + PixelCNN	90	70	ResNet-62
Adversarial training methods, transfer-based			
(Madry et al., 2018)*	87.3	70.2	ResNet-56
(Zhang et al., 2019)*	84.9	72.2	ResNet-56

on CIFAR-10-C:

	Accuracy	Noise	Blur	Weather	Digital
Raw WideResNet	71.89	42.37	72.74	86.23	78.83
ADP ($\sigma = 0.25$)	77.45	84.68	75.52	77.98	73.42
ADP ($\sigma = 0.25$)+Detection	78.96	84.57	71.76	84.34	76.60
ADP ($\sigma = 0.1$)	76.25	86.88	68.55	78.88	73.36
ADP, ($\sigma = 0.0$)	80.49	84.58	73.12	86.39	78.89
ADP, ($\sigma = 0.0$ + DCT)	80.74	84.96	73.11	87.26	78.69
ADP, ($\sigma = 0.0$ + AugMix)	82.63	87.52	75.20	88.82	80.20
ADP, ($\sigma = 0.0$ + DCT+AugMix)	82.40	85.05	75.80	88.11	81.30
Adversarial training methods					
(Zhang et al., 2019)	75.63	77.83	78.37	74.98	71.88
(Carmon et al., 2019)	80.40	81.20	83.44	80.19	76.98
(Cohen et al., 2019)	73.70	81.48	72.60	72.19	70.46
Training classifiers with augmentations					
(Hendrycks et al., 2020)*	88.78	84.66	89.68	90.93	88.47
(Wang et al., 2021)*	89.52	85.61	89.68	91.88	89.95
(Hossain et al., 2020)*	89.17	86.80	88.75	91.25	89.28

A natural idea arises again! We can use stronger denoising generative models: diffusion models.

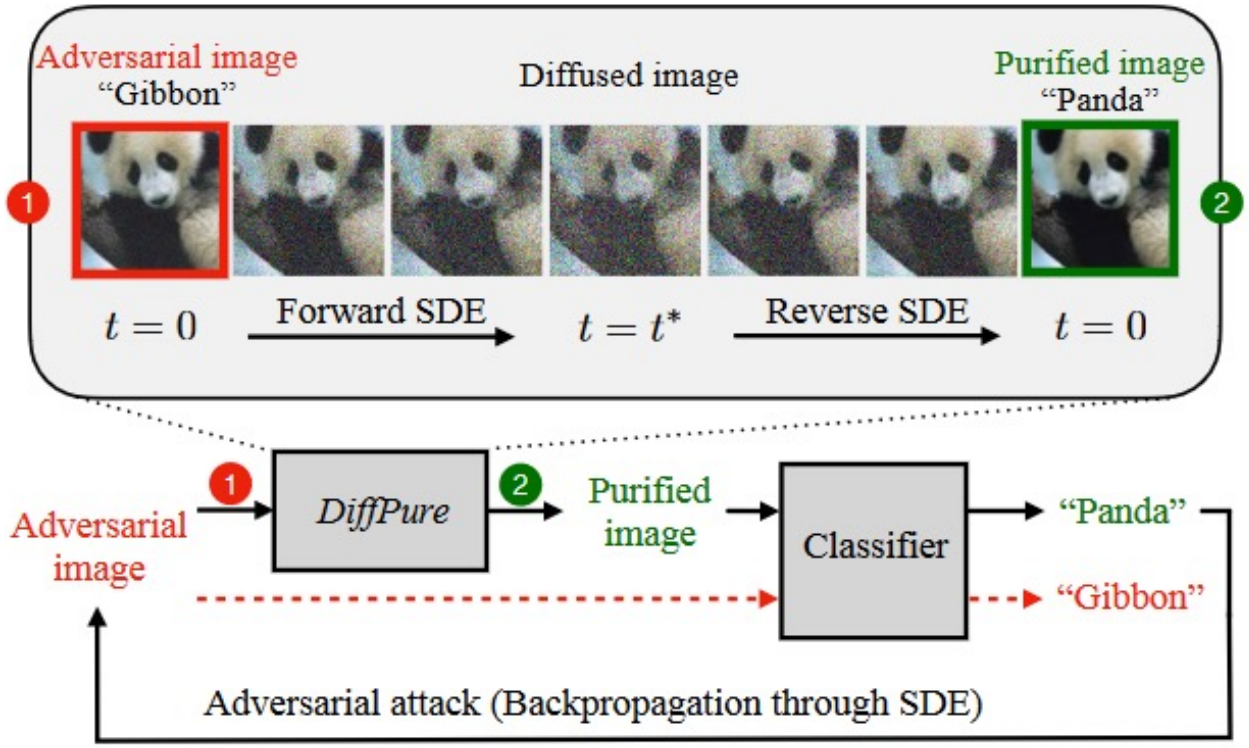
Yoon et al., Adversarial Purification with Score-based Generative Models. ICML, 2021.

Adversarial Purification based on Diffusion Models

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Diffusion Models for Adversarial Purification



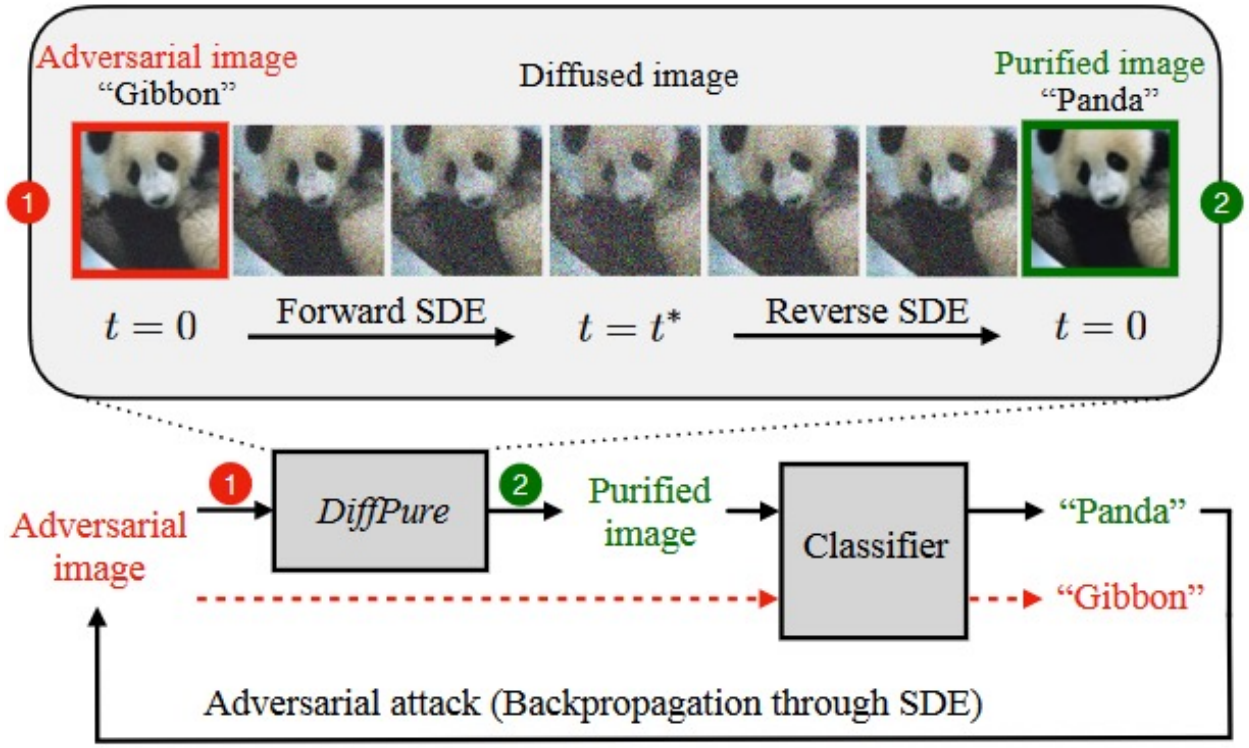
Solution: use a pre-trained diffusion model to denoise adversarial examples.

Rationale: injecting Gaussian noise to adversarial examples makes the distribution of adversarial examples approach a Gaussian distribution.

Nie et al., Diffusion Models for Adversarial Purification . ICML, 2022.

Adversarial Purification based on Diffusion Models

Diffusion Models for Adversarial Purification



Rationale: ideally, the distribution of the noise-injected adversarial examples should approach the distribution of noise-injected natural examples, as they all approach a Gaussian distribution. Then denoising noise-injected adversarial examples is equivalent to denoising noise-injected natural examples.

Nie et al., Diffusion Models for Adversarial Purification . ICML, 2022.

Adversarial Purification based on Diffusion Models

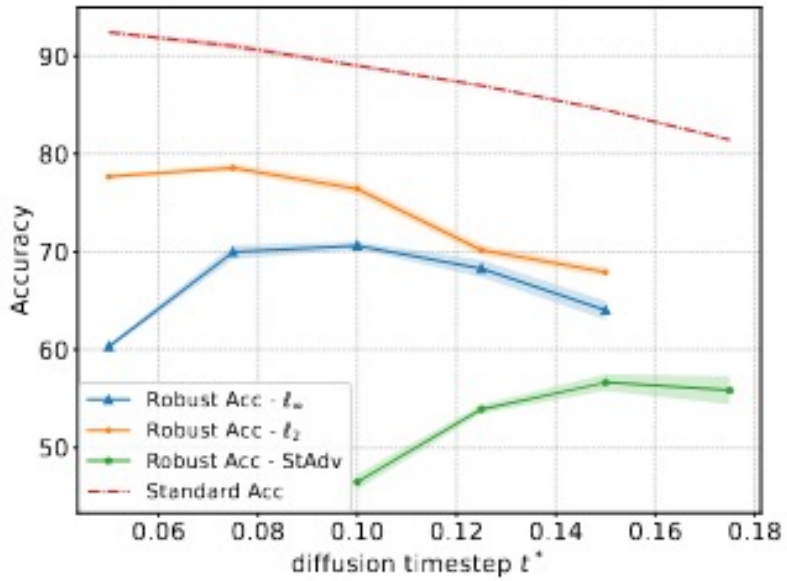


Figure 3. Impact of diffusion time t^* in our method on standard accuracy and robust accuracies against AutoAttack ℓ_∞ ($\epsilon = 8/255$), ℓ_2 ($\epsilon = 0.5$) and StAdv ($\epsilon = 0.05$) threat models, respectively, where we evaluate on WideResNet-28-10 for CIFAR-10.

A key challenge: the choice of t.

1. if t is too small, then the distribution of noise-injected adversarial examples cannot align well with the distribution of noise-injected natural examples, which means the adversarial noise cannot be fully removed.

2. if t is too large, then the image will become too noisy, which loses the semantic information. Without the guidance of the original semantic information, the purified image may have a different semantic meaning.

Thus, how to retain the original semantic information after purification becomes a problem.

Nie et al., Diffusion Models for Adversarial Purification . ICML, 2022.

Adversarial Purification based on Diffusion Models

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Table 1. Standard accuracy and robust accuracy against AutoAttack ℓ_∞ ($\epsilon = 8/255$) on CIFAR-10, obtained by different classifier architectures. In our method, the diffusion timestep is $t^* = 0.1$.

Method	Extra Data	Standard Acc	Robust Acc
WideResNet-28-10			
(Zhang et al., 2020)	✓	89.36	59.96
(Wu et al., 2020)	✓	88.25	62.11
(Gowal et al., 2020)	✓	89.48	62.70
(Wu et al., 2020)	✗	85.36	59.18
(Rebuffi et al., 2021)	✗	87.33	61.72
(Gowal et al., 2021)	✗	87.50	65.24
Ours	✗	**89.02±0.21**	**70.64±0.39**
WideResNet-70-16			
(Gowal et al., 2020)	✓	91.10	66.02
(Rebuffi et al., 2021)	✓	92.23	68.56
(Gowal et al., 2020)	✗	85.29	59.57
(Rebuffi et al., 2021)	✗	88.54	64.46
(Gowal et al., 2021)	✗	88.74	66.60
Ours	✗	**90.07±0.97**	**71.29±0.55**

Table 3. Standard accuracy and robust accuracy against AutoAttack ℓ_∞ ($\epsilon = 4/255$) on ImageNet, obtained by different classifier architectures. In our method, the diffusion timestep is $t^* = 0.15$. (†Robust accuracy is directly reported from the respective paper.)

Method	Extra Data	Standard Acc	Robust Acc
ResNet-50			
(Engstrom et al., 2019)	✗	62.56	31.06
(Wong et al., 2020)	✗	55.62	26.95
(Salman et al., 2020)	✗	64.02	37.89
(Bai et al., 2021)†	✗	67.38	35.51
Ours	✗	**67.79±0.43**	**40.93±1.96**
WideResNet-50-2			
(Salman et al., 2020)	✗	68.46	39.25
Ours	✗	**71.16±0.75**	**44.39±0.95**
DeiT-S			
(Bai et al., 2021)†	✗	66.50	35.50
Ours	✗	**73.63±0.62**	**43.18±1.27**

Nie et al., Diffusion Models for Adversarial Purification . ICML, 2022.

Adversarial Purification based on Diffusion Models

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

DensePure: Understanding Diffusion Models for Adversarial Robustness

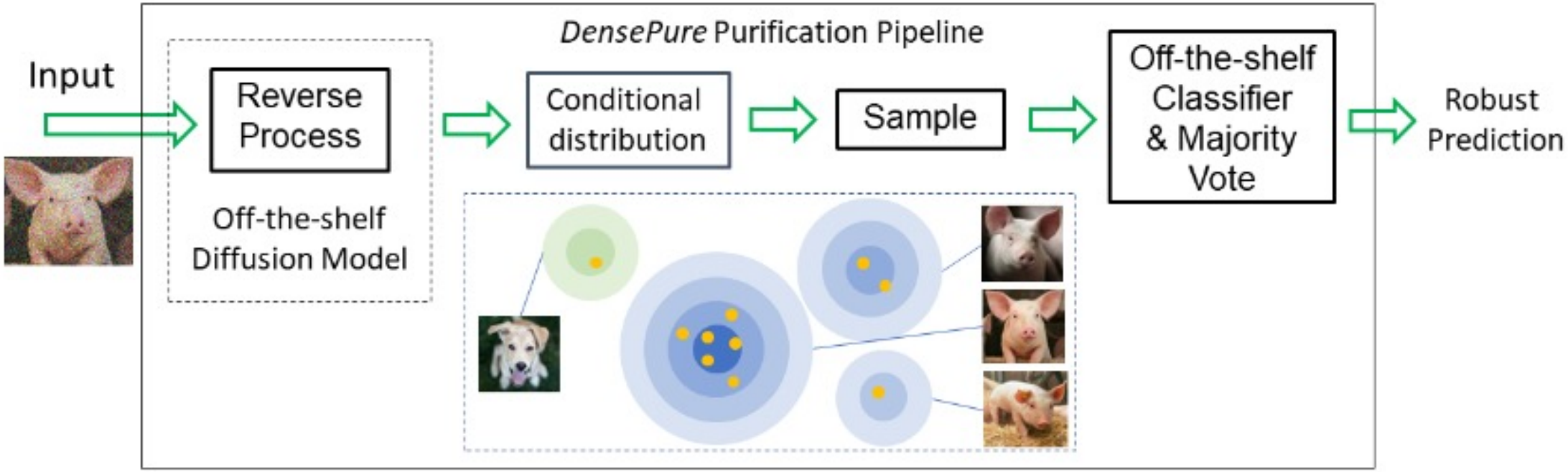
Theoretical proof: when the data density of natural samples is sufficiently high, the conditional density of the purified samples will also be high. This means that samples drawn from this conditional distribution have a high probability of recovering the original semantic meaning.

Solution: purify an input multiple times to get multiple purified samples, then feed them into the classifier and use majority voting to select the final prediction.

Xiao et al., DensePure: Understanding Diffusion Models for Adversarial Robustness. ICLR, 2023.

Adversarial Purification based on Diffusion Models

DensePure: Understanding Diffusion Models for Adversarial Robustness



Xiao et al., DensePure: Understanding Diffusion Models for Adversarial Robustness. ICLR, 2023.

Adversarial Purification based on Diffusion Models

TMLR TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

DensePure: Understanding Diffusion Models for Adversarial Robustness

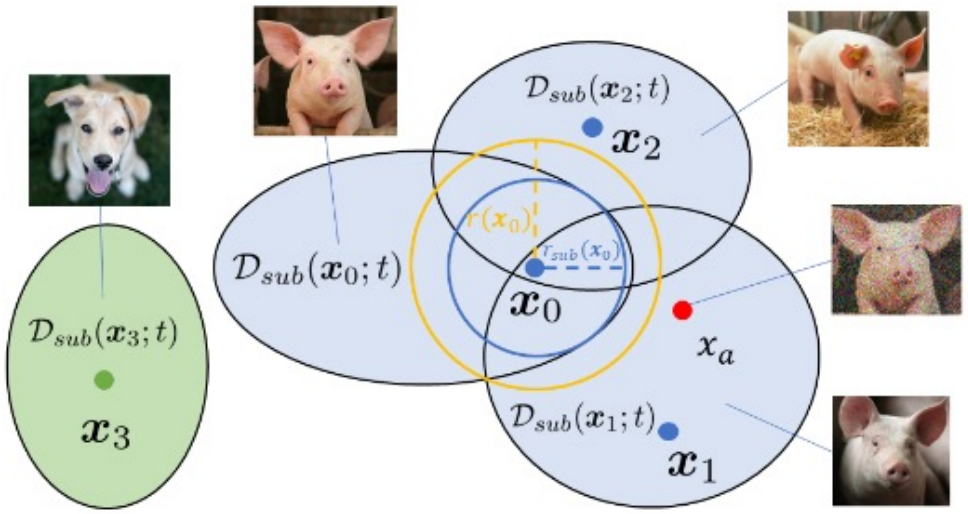


Figure 2: An illustration of the robust region $\mathcal{D}(\boldsymbol{x}_0;t) = \bigcup_{i=1}^{3} \mathcal{D}_{sub}(\boldsymbol{x}_i;t)$, where $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2$ are samples with ground-truth label and $\boldsymbol{x}_3$ is a sample with another label. $\boldsymbol{x}_a = \boldsymbol{x}_0 + \boldsymbol{\epsilon}_a$ is an adversarial sample such that $\mathcal{P}(\boldsymbol{x}_a;t) = \boldsymbol{x}_1 \neq \boldsymbol{x}_0$ and thus the classification is correct but $\boldsymbol{x}_a$ is not reversed back to $\boldsymbol{x}_0$. $r_{sub}(\boldsymbol{x}_0) < r(\boldsymbol{x}_0)$ shows our claim that the union leads to a larger robust radius.

Xiao et al., DensePure: Understanding Diffusion Models for Adversarial Robustness. ICLR, 2023.

Adversarial Purification based on Diffusion Models

TMLR TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

DensePure: Understanding Diffusion Models for Adversarial Robustness

Method	Off-the-shelf	Certified Accuracy at ϵ(%) CIFAR-10 0.25	0.5	0.75	1.0	ImageNet 0.5	1.0	1.5	2.0	3.0
PixelDP (Lecuyer et al., 2019)	✗	(71.0) 22.0	(44.0) 2.0	-	-	(33.0) 16.0	-	-	-	-
RS (Cohen et al., 2019)	✗	(75.0) 61.0	(75.0) 43.0	(65.0) 32.0	(65.0) 23.0	(67.0) 49.0	(57.0) 37.0	(57.0) 29.0	(44.0) 19.0	(44.0) 12.0
SmoothAdv (Salman et al., 2019a)	✗	(82.0) 68.0	(76.0) 54.0	(68.0) 41.0	(64.0) 32.0	(63.0) 54.0	(56.0) 42.0	(56.0) 34.0	(41.0) 26.0	(41.0) 18.0
Consistency (Jeong & Shin, 2020)	✗	(77.8) 68.8	(75.8) 58.1	(72.9) 48.5	(52.3) 37.8	(55.0) 50.0	(55.0) 44.0	(55.0) 34.0	(41.0) 24.0	(41.0) 17.0
MACER (Zhai et al., 2020)	✗	(81.0) 71.0	(81.0) 59.0	(66.0) 46.0	(66.0) 38.0	(68.0) 57.0	(64.0) 43.0	(64.0) 31.0	(48.0) 25.0	(48.0) 14.0
Boosting (Horváth et al., 2021)	✗	(83.4) 70.6	(76.8) 60.4	(71.6) **52.4**	(73.0) **38.8**	(65.6) 57.0	(57.0) 44.6	(57.0) 38.4	(44.6) 28.6	(38.6) 21.2
SmoothMix (Jeong et al., 2021)	✓	(77.1) 67.9	(77.1) 57.9	(74.2) 47.7	(61.8) 37.2	(55.0) 50.0	(55.0) 43.0	(55.0) 38.0	(40.0) 26.0	(40.0) 17.0
Denoised (Salman et al., 2020)	✓	(72.0) 56.0	(62.0) 41.0	(62.0) 28.0	(44.0) 19.0	(60.0) 33.0	(38.0) 14.0	(38.0) 6.0	-	-
Lee (Lee, 2021)	✓	60.0	42.0	28.0	19.0	41.0	24.0	11.0	-	-
Carlini (Carlini et al., 2022)	✓	(88.0) 73.8	(88.0) 56.2	(88.0) 41.6	(74.2) 31.0	(82.0) 74.0	(77.2.0) 59.8	(77.2) 47.0	(64.6) 31.0	(64.6) 19.0
Ours	✓	(87.6) **76.6**	(87.6) **64.6**	(87.6) 50.4	(73.6) 37.4	(84.0) **77.8**	(80.2) **67.0**	(80.2) **54.6**	(67.8) **42.2**	(67.8) **25.8**

Xiao et al., DensePure: Understanding Diffusion Models for Adversarial Robustness. ICLR, 2023.

Conclusion

Use weak generative model + weak detector:

1. Meng & Chen, MagNet: a Two-Pronged Defense against Adversarial Examples. ACM CCS, 2017.

Use stronger generative models:

2. Samangouei et al., Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. ICLR, 2018.
3. Song et al., Leveraging Generative Models to Understand and Defend against Adversarial Exmaples. ICLR, 2018.

Use denoising generative models:

4. Yoon et al., Adversarial Purification with Score-based Generative Models. ICML, 2021.

Use stronger denoising generative models:

5. Nie et al., Diffusion Models for Adversarial Purification. ICML, 2022
6. Xiao et al., DensePure: Understanding Diffusion Models for Adversarial Robustness. ICLR, 2023.

ACML 2023 Tutorial: Trustworthy Machine Learning on Adversarial Data

Part II: Adversarial Training

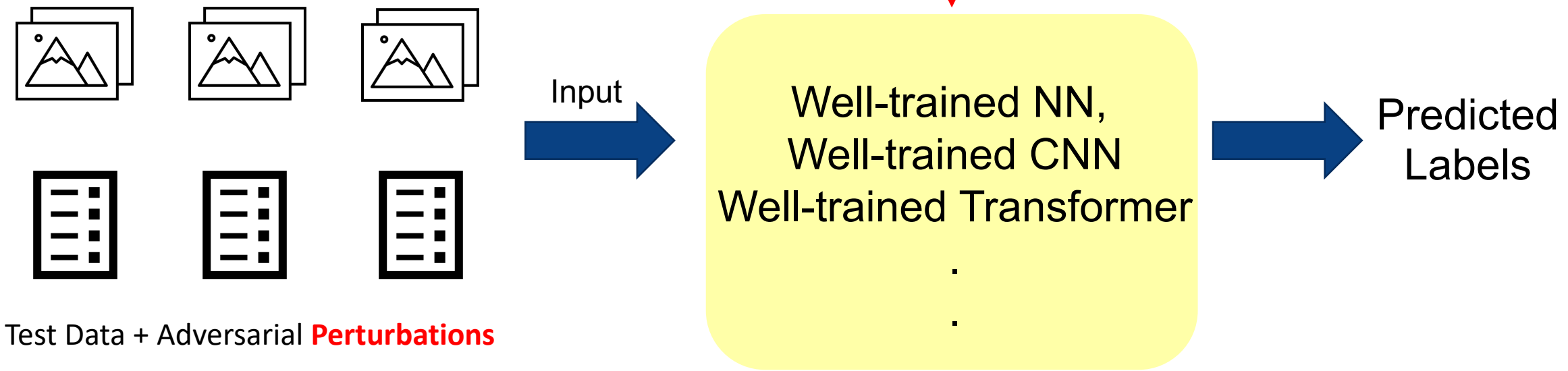
Let's try train a robust model!!

How to actively prevent from adversarial attacks

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial attack happens:



Adversarial Training: Two Main Directions

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial Training

(From: Madry, et al., 2018)

- **generate adversarial examples**

$$\delta^{(t)} \leftarrow \text{Proj}\left[\delta^{(t-1)} + \alpha \,\text{sign}\left(\nabla_\theta \ell\left(x_i + \delta_i^{(t-1)}, y_i; \theta\right)\right)\right] (1)$$

- **train with adversarial examples**

$$\text{argmin}_\theta \sum_i \ell\left(x_i + \delta_i^{(T)}, y_i; \theta\right) (2)$$

Instance-reweighted Adversarial Training

$$\text{argmin}_\theta \sum_i \omega_i \ell\left(x_i + \delta_i^{(T)}, y_i; \theta\right) (3)$$

$$s.t.\ \omega_i \geq 0 \text{ and } \sum_i \omega_i = 1$$

Adversarial Training: Two Main Directions

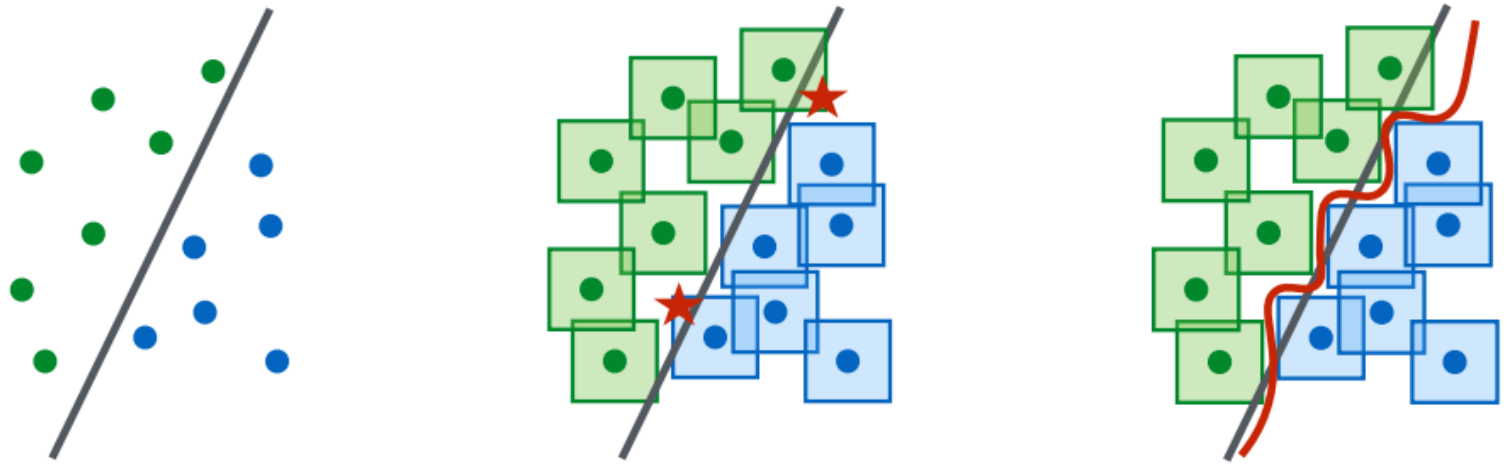


Figure 3: A conceptual illustration of standard vs. adversarial decision boundaries. Left: A set of points that can be easily separated with a simple (in this case, linear) decision boundary. Middle: The simple decision boundary does not separate the ℓ_∞-balls (here, squares) around the data points. Hence there are adversarial examples (the red stars) that will be misclassified. Right: Separating the ℓ_∞-balls requires a significantly more complicated decision boundary. The resulting classifier is robust to adversarial examples with bounded ℓ_∞-norm perturbations.

Madry, et al., Towards Deep Learning Models Resistant to Adversarial Attacks. ICLR 2018.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Trade-off between Adversarial Robustness and Accuracy (TRADES)

$$\text{argmin}_\theta \sum_i \underbrace{\text{CE}(\boldsymbol{p}(\boldsymbol{x}_i;\boldsymbol{\theta}), y)}_{natural\ error} + \lambda \underbrace{\text{KL}\left(\boldsymbol{p}(\boldsymbol{x}_i;\boldsymbol{\theta}) || \boldsymbol{p}\left(\boldsymbol{x}_i + \boldsymbol{\delta}_i^{(T)};\boldsymbol{\theta}\right)\right)}_{robust\ error}$$

Natural Error encourages the natural error to be optimized by minimizing the risk w.r.t. the ground truth.

Robust Error encourages the output to be smooth in combating adversarial attack.

H. Zhang, et al., Theoretically Principled Trade-off between Robustness and Accuracy. ICML 2019.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Trade-off between Adversarial Robustness and Accuracy (TRADES)

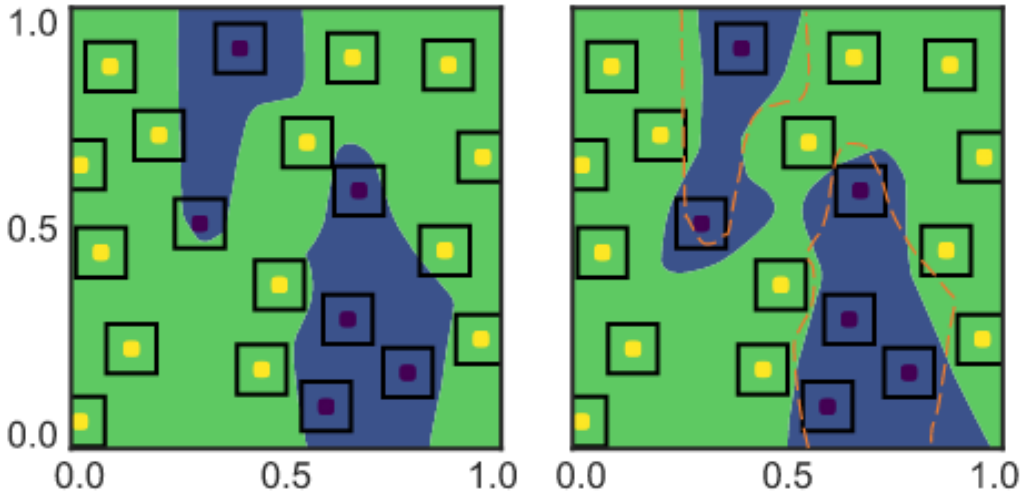


Figure 1: **Left figure:** decision boundary learned by natural training method. **Right figure:** decision boundary learned by our adversarial training method, where the orange dotted line represents the decision boundary in the left figure. It shows that both methods achieve zero natural training error, while our adversarial training method achieves better robust training error than the natural training method.

H. Zhang, et al., Theoretically Principled Trade-off between Robustness and Accuracy. ICML 2019.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Trade-off between Adversarial Robustness and Accuracy (TRADES)

Algorithm 1 Adversarial training by TRADES

Input: Step sizes η_1 and η_2, batch size m, number of iterations K in inner optimization, netwo architecture parametrized by θ
Output: Robust network f_θ
Randomly initialize network f_θ, or initialize network with pre-trained configuration
repeat
  Read mini-batch $B = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_m\}$ from training set
  for $i = 1, ..., m$ (in parallel) **do**
    $\boldsymbol{x}_i' \leftarrow \boldsymbol{x}_i + 0.001 \cdot \mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\mathcal{N}(\mathbf{0}, \mathbf{I})$ is the Gaussian distribution with zero mean and ident variance
    for $k = 1, ..., K$ **do**
      $\boldsymbol{x}_i' \leftarrow \Pi_{\mathbb{B}(\boldsymbol{x}_i, \epsilon)}(\eta_1 \mathsf{sign}(\nabla_{\boldsymbol{x}_i'} \mathcal{L}(f_\theta(\boldsymbol{x}_i), f_\theta(\boldsymbol{x}_i'))) + \boldsymbol{x}_i')$, where Π is the projection operator
    end for
  end for
  $\theta \leftarrow \theta - \eta_2 \sum_{i=1}^{m} \nabla_\theta [\mathcal{L}(f_\theta(\boldsymbol{x}_i), \boldsymbol{y}_i) + \mathcal{L}(f_\theta(\boldsymbol{x}_i), f_\theta(\boldsymbol{x}_i'))/\lambda]/m$
until training converged

H. Zhang, et al., Theoretically Principled Trade-off between Robustness and Accuracy. ICML 2019.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Defense	Defense type	Under which attack	Dataset	Distance	$\mathcal{A}_{\text{nat}}(f)$	$\mathcal{A}_{\text{rob}}(f)$
[BRRG18]	gradient mask	[ACW18]	CIFAR10	0.031 (ℓ_∞)	-	0%
[MLW+18]	gradient mask	[ACW18]	CIFAR10	0.031 (ℓ_∞)	-	5%
[DAL+18]	gradient mask	[ACW18]	CIFAR10	0.031 (ℓ_∞)	-	0%
[SKN+18]	gradient mask	[ACW18]	CIFAR10	0.031 (ℓ_∞)	-	9%
[NKM17]	gradient mask	[ACW18]	CIFAR10	0.015 (ℓ_∞)	-	15%
[WSMK18]	robust opt.	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	27.07%	23.54%
[MMS+18]	robust opt.	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	87.30%	**47.04%**
[ZSLG16]	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	94.64%	0.15%
[KGB17]	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	85.25%	45.89%
[RDV17]	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	95.34%	0%
TRADES ($1/\lambda = 1$)	regularization	FGSM^{1,000} (PGD)	CIFAR10	0.031 (ℓ_∞)	88.64%	48.90%
TRADES ($1/\lambda = 6$)	regularization	FGSM^{1,000} (PGD)	CIFAR10	0.031 (ℓ_∞)	84.92%	**56.43%**
TRADES ($1/\lambda = 1$)	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	88.64%	49.14%
TRADES ($1/\lambda = 6$)	regularization	FGSM^{20} (PGD)	CIFAR10	0.031 (ℓ_∞)	84.92%	**56.61%**
TRADES ($1/\lambda = 1$)	regularization	DeepFool (ℓ_∞)	CIFAR10	0.031 (ℓ_∞)	88.64%	59.10%
TRADES ($1/\lambda = 6$)	regularization	DeepFool (ℓ_∞)	CIFAR10	0.031 (ℓ_∞)	84.92%	61.38%
TRADES ($1/\lambda = 1$)	regularization	LBFGSAttack	CIFAR10	0.031 (ℓ_∞)	88.64%	84.41%
TRADES ($1/\lambda = 6$)	regularization	LBFGSAttack	CIFAR10	0.031 (ℓ_∞)	84.92%	81.58%
TRADES ($1/\lambda = 1$)	regularization	MI-FGSM	CIFAR10	0.031 (ℓ_∞)	88.64%	51.26%
TRADES ($1/\lambda = 6$)	regularization	MI-FGSM	CIFAR10	0.031 (ℓ_∞)	84.92%	57.95%
TRADES ($1/\lambda = 1$)	regularization	C&W	CIFAR10	0.031 (ℓ_∞)	88.64%	84.03%
TRADES ($1/\lambda = 6$)	regularization	C&W	CIFAR10	0.031 (ℓ_∞)	84.92%	81.24%

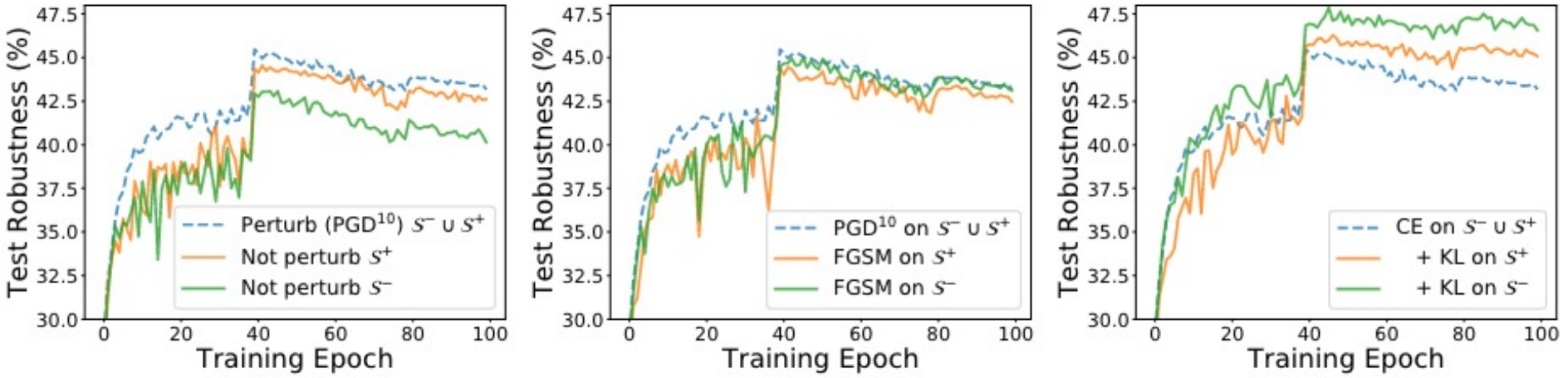
H. Zhang, et al., Theoretically Principled Trade-off between Robustness and Accuracy. ICML 2019.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Misclassification Aware Adversarial Training (MART)



(a) Not perturb (b) Inner maximization (c) Outer minimization

Figure 1: The distinctive influence of misclassified examples ($\mathcal{S}^-$) versus correctly classified ones ($\mathcal{S}^+$) on the robustness of adversarial training. We test the white-box robustness of different strategies on either subset of examples: (a) using them directly for training ("not perturb"); (b) using weak attack (FGSM) in the inner maximization; and (c) using "regularized CE" loss in the outer minimization.

Wang, et al., Improving Adversarial Robustness Requires Revising Misclassified Examples. ICLR 2020.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Misclassification Aware Adversarial Training (MART)

$$\text{argmin}_\theta \sum_i \underbrace{\text{BCE}\left(\boldsymbol{p}\left(\boldsymbol{x}_i + \boldsymbol{\delta}_i^{(T)}\right), y_i\right)}_{\textbf{Term 1}} + \lambda \underbrace{\text{KL}\left(\boldsymbol{p}\left(\boldsymbol{x}_i + \boldsymbol{\delta}_i^{(T)}; \boldsymbol{\theta}\right) || \boldsymbol{p}(\boldsymbol{x}_i; \boldsymbol{\theta})\right)}_{\textbf{Term 2}} \cdot \underbrace{\left(1 - \boldsymbol{p}_{y_i}(\boldsymbol{x}_i; \boldsymbol{\theta})\right)}_{\textbf{Term 3}}$$

Term 1. Commonly used CE loss with the margin term to improve the decision margin.

Term 2. Measuring if model predictions are consistent before and after adversarial perturbing.

Term 3. Emphasizing learning on misclassified examples.

Wang, et al., Improving Adversarial Robustness Requires Revising Misclassified Examples. ICLR 2020.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Misclassification Aware Adversarial Training (MART)

Algorithm 1 Misclassification Aware adveRsarial Training (MART)

Input: Training data $\{\mathbf{x}_i, y_i\}_{i=1,\dots,n}$, outer iteration number T_O, inner iteration number T_I, maximum perturbation ϵ, step size for inner optimization η_I, step size for outer optimization η_O
Initialization: Standard random initialization of $h_{\boldsymbol{\theta}}$
for $t = 1, \dots, T_O$ **do**
  Uniformly sample a minibatch of training data $B^{(t)}$
  for $\mathbf{x}_i \in B^{(t)}$ **do**
    $\mathbf{x}_i' = \mathbf{x}_i + \epsilon \cdot \xi$, with $\xi \sim \mathcal{U}(-1, 1)$ # $\mathcal{U}$ is a uniform distribution
    for $s = 1, \dots, T_I$ **do**
      $\mathbf{x}_i' \leftarrow \Pi_{\mathcal{B}_\epsilon(\mathbf{x}_i)}\left(\mathbf{x}_i' + \eta_I \cdot \text{sign}(\nabla_{\mathbf{x}_i'} \text{CE}(\mathbf{p}(\mathbf{x}_i', \boldsymbol{\theta}), y_i))\right)$ # $\Pi(\cdot)$ is the projection operator
    end for
    $\hat{\mathbf{x}}_i' \leftarrow \mathbf{x}_i'$
  end for
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_O \sum_{\mathbf{x}_i \in B^{(t)}} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathbf{x}_i, y_i, \hat{\mathbf{x}}_i'; \boldsymbol{\theta})$
end for
Output: Robust classifier $h_{\boldsymbol{\theta}}$

Wang, et al., Improving Adversarial Robustness Requires Revising Misclassified Examples. ICLR 2020.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Table 2: White-box robustness (accuracy (%) on white-box test attacks) on MNIST and CIFAR-10.

Defense	MNIST				CIFAR-10			
	Natural	FGSM	PGD^{20}	CW_∞	Natural	FGSM	PGD^{20}	CW_∞
Standard	99.11	97.17	94.62	94.25	84.44	61.89	47.55	45.98
MMA	98.92	97.25	95.25	94.77	**84.76**	62.08	48.33	45.77
Dynamic	98.96	97.34	95.27	94.85	83.33	62.47	49.40	46.94
TRADES	**99.25**	96.67	94.58	94.03	82.90	62.82	50.25	48.29
MART	98.74	**97.87**	**96.48**	**96.10**	83.07	**65.65**	**55.57**	**54.87**

Table 3: Black-box robustness (accuracy (%) on black-box test attacks) on MNIST and CIFAR-10.

Defense	MNIST				CIFAR-10			
	FGSM	PGD^{10}	PGD^{20}	CW_∞	FGSM	PGD^{10}	PGD^{20}	CW_∞
Standard	96.12	95.73	95.47	96.34	79.98	80.27	80.01	80.85
MMA	96.11	95.94	95.81	96.87	80.28	80.52	80.48	81.32
Dynamic	97.60	96.25	95.82	97.03	81.37	81.71	81.38	82.05
TRADES	97.49	96.03	95.73	97.20	81.52	81.73	81.53	82.11
MART	**97.77**	**96.96**	**96.97**	**98.36**	**82.75**	**82.93**	**82.70**	**82.95**

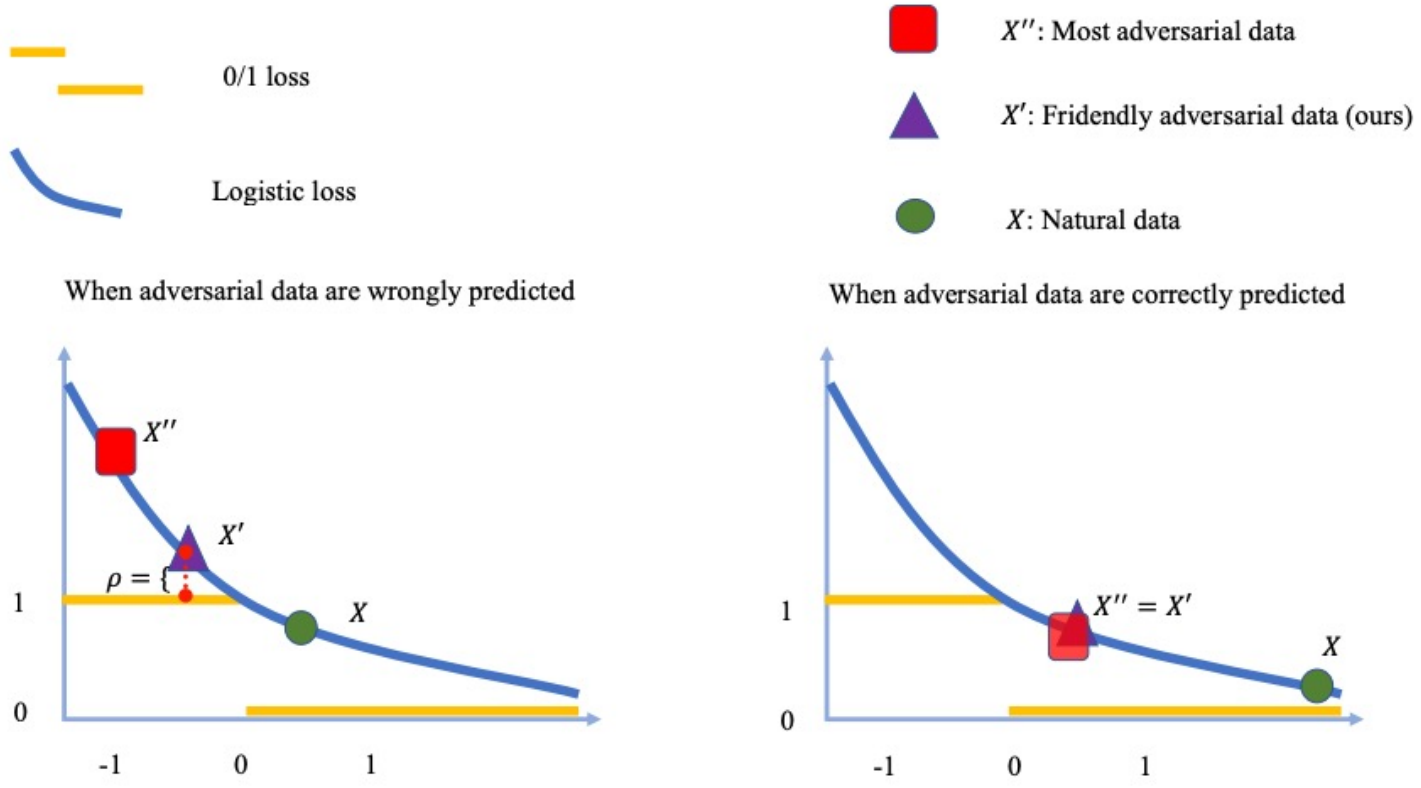
Wang, et al., Improving Adversarial Robustness Requires Revising Misclassified Examples. ICLR 2020.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Friendly Adversarial Training (FAT)



Zhang, et al., Attacks Which Do Not Kill Training Make Adversarial Training Stronger. ICML 2020.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Friendly Adversarial Training (FAT)

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(\widetilde{x_i}), y_i),$$

$$\widetilde{x_i} = \underbrace{\arg\min_{\tilde{x} \in \mathcal{B}_\epsilon[x_i]} \ell(f(\tilde{x}), y_i)}_{Objective} \text{ s.t. } \underbrace{\ell(f(\tilde{x}), y_i) - \min_{y \in Y} \ell(f(\tilde{x}), y) \geq \rho}_{Constraint}$$

Constraint. Ensures $y_i \neq \arg\min_{y \in Y} \ell(f(\tilde{x}), y)$ or $\tilde{x}$ is misclassified.

Constraint. Ensures for $\tilde{x}$, the wrong prediction is better than the desired prediction y_i by at least ρ in terms of the loss value.

Objective. Minimize the adversarial loss given that some confident adversarial data has been found. This $\widetilde{x_i}$ could be regarded as a 'friend' among the adversaries.

Zhang, et al., Attacks Which Do Not Kill Training Make Adversarial Training Stronger. ICML 2020.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Friendly Adversarial Training (FAT)

Algorithm 1 PGD-K-τ

Input: data $x \in \mathcal{X}$, label $y \in \mathcal{Y}$, model f, loss function ℓ, maximum PGD step K, step τ, perturbation bound ϵ, step size α
Output: $\tilde{x}$
$\tilde{x} \leftarrow x$
while $K > 0$ **do**
  if $\arg\max_i f(\tilde{x}) \neq y$ and $\tau = 0$ **then**
    break
  else if $\arg\max_i f(\tilde{x}) \neq y$ **then**
    $\tau \leftarrow \tau - 1$
  end if
  $\tilde{x} \leftarrow \Pi_{\mathcal{B}[x,\epsilon]}\left(\alpha \operatorname{sign}(\nabla_{\tilde{x}} \ell(f(\tilde{x}), y)) + \tilde{x}\right)$
  $K \leftarrow K - 1$
end while

Algorithm 2 Friendly Adversarial Training (FAT)

Input: network f_θ, training dataset $S = \{(x_i, y_i)\}_{i=1}^n$, learning rate η, number of epochs T, batch size m, number of batches M
Output: adversarially robust network f_θ
for epoch $= 1, \ldots, T$ **do**
  for mini-batch $= 1, \ldots, M$ **do**
    Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^m$ from S
    for $i = 1, \ldots, m$ (in parallel) **do**
      Obtain adversarial data $\tilde{x}_i$of x_i by Algorithm 1
    end for
    $\theta \leftarrow \theta - \eta \frac{1}{m} \sum_{i-1}^m \nabla_\theta \ell(f_\theta(\tilde{x}_i), y_i)$
  end for
end for

Zhang, et al., Attacks Which Do Not Kill Training Make Adversarial Training Stronger. ICML 2020.

Adversarial Training: Instance Equivalence

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Friendly Adversarial Training (FAT)

Table 1. Evaluations (test accuracy) of deep models (WRN-32-10) on CIFAR-10 dataset

Defense	Natural	FGSM	PGD-20	C&W$_\infty$	PGD-100
Madry	87.30	56.10	45.80	46.80	-
CAT	77.43	57.17	46.06	42.28	-
DAT	85.03	63.53	48.70	47.27	-
FAT ($\epsilon_{train} = 8/255$)	**89.34** $\pm$ 0.221	65.52 $\pm$ 0.355	46.13 $\pm$ 0.409	46.82 $\pm$ 0.517	45.31 $\pm$ 0.531
FAT ($\epsilon_{train} = 16/255$)	87.00 $\pm$ 0.203	**65.94** $\pm$ 0.244	**49.86** $\pm$ 0.328	**48.65** $\pm$ 0.176	**49.56** $\pm$ 0.255

Results of Madry, CAT and DAT are reported in (Wang et al., 2019). FAT has the same evaluations.

Table 2. Evaluations (test accuracy) of deep models (WRN-34-10) on CIFAR-10 dataset

Defense	Natural	FGSM	PGD-20	C&W$_\infty$	PGD-100
TRADES ($\beta = 1.0$)	88.64	56.38	49.14	-	-
FAT for TRADES ($\epsilon_{train} = 8/255$)	**89.94** $\pm$ 0.303	**61.00** $\pm$ 0.418	**49.70** $\pm$ 0.653	49.35 $\pm$ 0.363	48.35 $\pm$ 0.240
TRADES ($\beta = 6.0$)	84.92	61.06	56.61	**54.47**	55.47
FAT for TRADES ($\epsilon_{train} = 8/255$)	**86.60** $\pm$ 0.548	**61.97** $\pm$ 0.570	55.98 $\pm$ 0.209	54.29 $\pm$ 0.173	55.34 $\pm$ 0.291
FAT for TRADES ($\epsilon_{train} = 16/255$)	84.39 $\pm$ 0.030	61.73 $\pm$ 0.131	**57.12** $\pm$ 0.233	54.36 $\pm$ 0.177	**56.07** $\pm$ 0.155

Results of TRADES ($\beta = 1.0$ and 6.0) are reported in (Zhang et al., 2019b). FAT for TRADES has the same evaluations.

Zhang, et al., Attacks Which Do Not Kill Training Make Adversarial Training Stronger. ICML 2020.

Adversarial Training: Instance Reweighted

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial Training

(From: Madry, et al., 2018)

- generate adversarial examples

$$\delta^{(t)} \leftarrow \text{Proj}\left[\delta^{(t-1)} + \alpha \,\text{sign}\left(\nabla_\theta \ell\left(x_i + \delta_i^{(t-1)}, y_i; \theta\right)\right)\right] (1)$$

- train with adversarial examples

$$\text{argmin}_\theta \sum_i \ell\left(x_i + \delta_i^{(T)}, y_i; \theta\right) (2)$$

Instance-reweighted Adversarial Training

$$\text{argmin}_\theta \sum_i \omega_i \ell\left(x_i + \delta_i^{(T)}, y_i; \theta\right) (3)$$

$$s.t.\ \omega_i \geq 0 \text{ and } \sum_i \omega_i = 1$$

Adversarial Training: Instance Reweighted

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

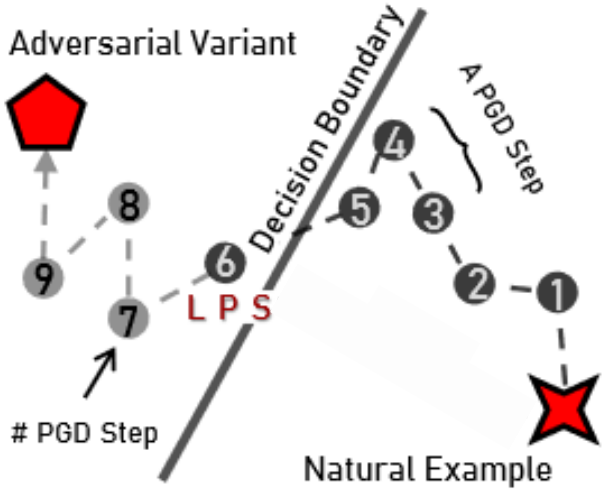
Geometric Aware Instance-Reweighted Adversarial Training (GAIRAT)

(From: Zhang, et al., 2021)

data point closer to the class boundary is less robust, requiring larger weight in training.

The Least PGD Steps (LPS) $\kappa(x, y)$

$$\omega(x, y) = \frac{1 + \tanh(\lambda + 5\times(1 - 2\times\kappa(x, y)\ /K))}{2} \quad (4)$$



Adversarial Training: Instance Reweighted

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

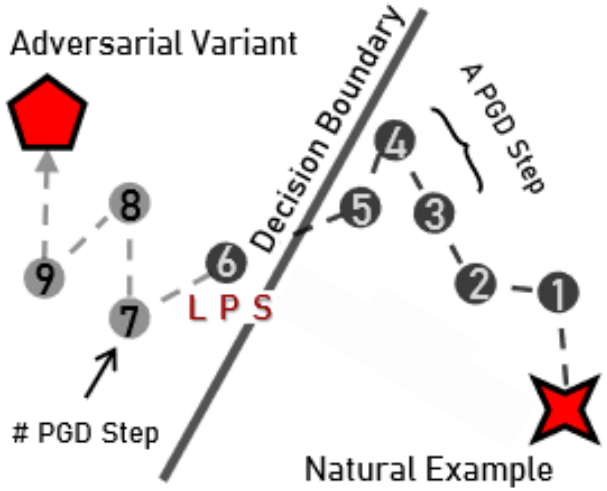
The least PGD steps (LPS)

Path-Dependency

- May change given the same start and end point
- Make the computation unstable

Discreteness

- Take only a few values
- Make the measurement ambiguous



Adversarial Training: Instance Reweighted

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

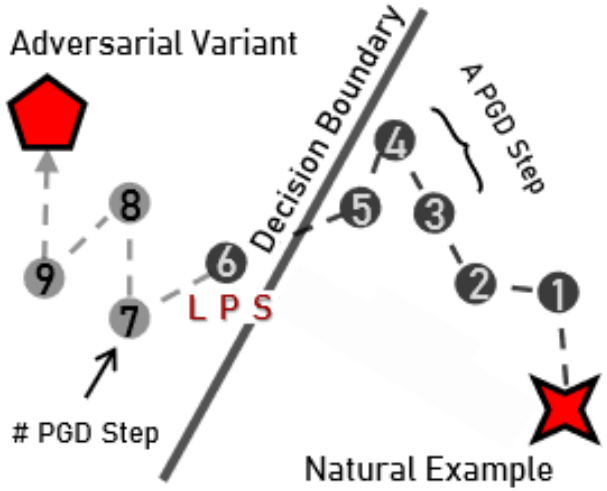
The least PGD steps (LPS)

Path-Dependency

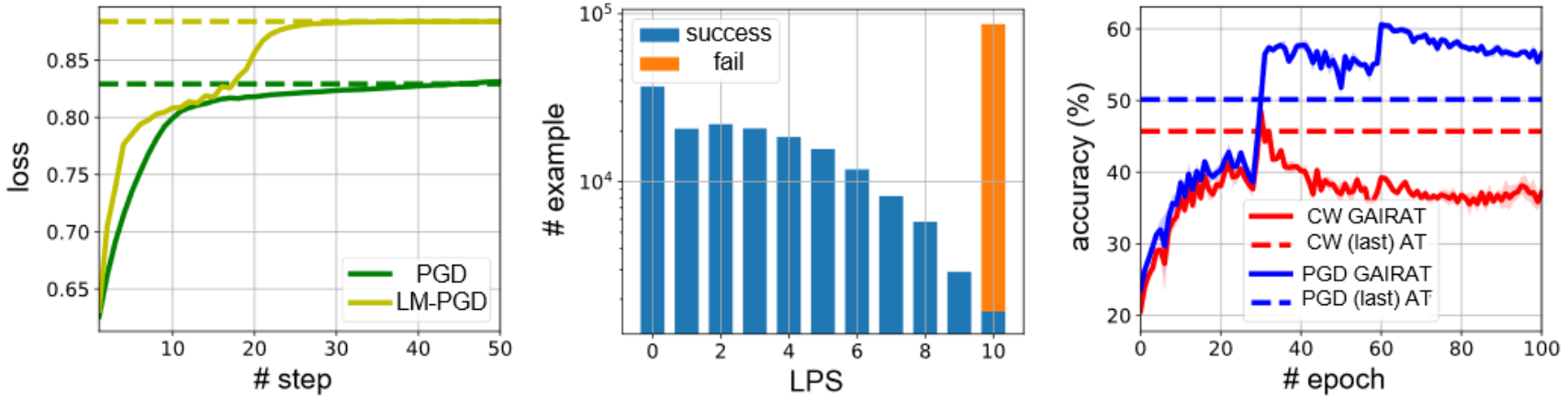
- May change given the same start and end point
- Make the computation unstable

Discreteness

- Take only a few values
- Make the measurement ambiguous



Adversarial Training: Instance Reweighted



(a) Loss Curve (b) Histogram (c) Accuracy Curve

Path-Dependency **Discreteness** **Consequence**

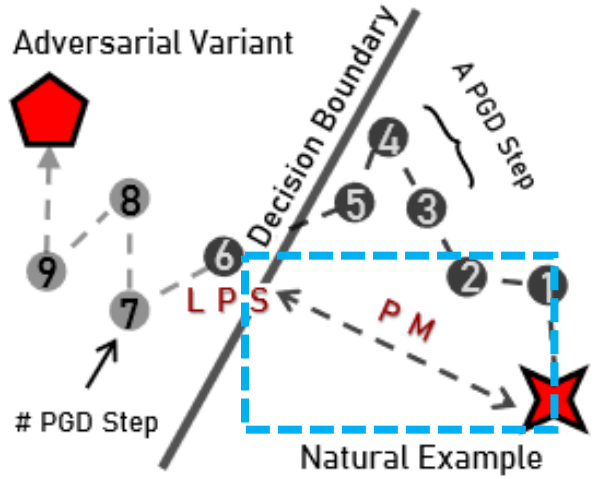
Existing methods measuring the closeness are not very reliable: they are **discrete** and **path-dependent**. Here, we adopt the least PGD steps (LPS) as an example.

Wang, et al., Probabilistic Margin for Instance Reweighting in Adversarial Training. NeurIPS 2021.

Adversarial Training: Instance Reweighted

Aiming at giving a geometric metric which are continuous and path-dependent.

- **Bring the idea for the multiclass margin in the margin theory.**
- **Compute in a low dimensional embedding space with normalization.**



Wang, et al., Probabilistic Margin for Instance Reweighting in Adversarial Training. NeurIPS 2021.

Adversarial Training: Instance Reweighted

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

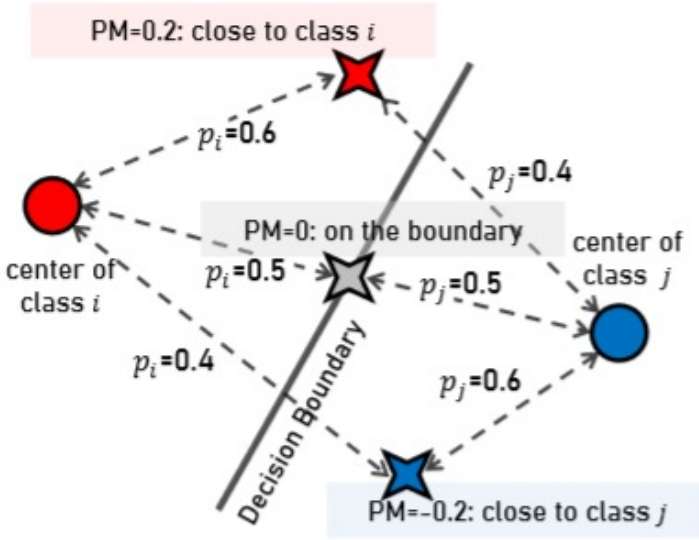
Multi-class Margin

$$h_y(x;\theta) - \max_{j,j\neq y} h_j(x;\theta) \quad (4)$$

Probabilistic Margin

$$p_y(x;\theta) - \max_{j,j\neq y} p_j(x;\theta) \quad (5)$$

- **Positive Value:** safe data with correct prediction.
- **Zero Value:** safe data on the decision boundary
- **Negative Value:** critical data with wrong prediction.



i for the target label y and j for the nearest class except y

Wang, et al., Probabilistic Margin for Instance Reweighting in Adversarial Training. NeurIPS 2021.

Adversarial Training: Instance Reweighted

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Multi-class Margin

$$h_y(x;\theta) - \max_{j,j\neq y} h_j(x;\theta) \quad (4)$$

Probabilistic Margin

$$p_y(x;\theta) - \max_{j,j\neq y} p_j(x;\theta) \quad (5)$$

- **Positive Value:** safe data with correct prediction.
- **Zero Value:** safe data on the decision boundary.
- **Negative Value:** critical data with wrong prediction.

Three Specifications

- **PM_adv**

$$\mathrm{PM}^{\mathrm{adv}} = p_y\left(x+\delta^{(T)};\theta\right) - \max_{j,j\neq y} p_j\left(x+\delta^{(T)};\theta\right) \quad (4)$$

- **PM_nat**

$$\mathrm{PM}^{\mathrm{nat}} = p_y(x;\theta) - \max_{j,j\neq y} p_j(x;\theta) \quad (5)$$

- **PM_dif**

$$\mathrm{PM}^{\mathrm{dif}} = p_y(x;\theta) - \max_{j,j\neq y} p_j\left(x+\delta^{(T)};\theta\right) \quad (6)$$

Wang, et al., Probabilistic Margin for Instance Reweighting in Adversarial Training. NeurIPS 2021.

Adversarial Training: Instance Reweighted

Algorithm 1 MAIL: The Overall Algorithm.

Input: a network model with the parameters θ; and a training dataset S of size n.
Output: a robust model with parameters θ^*.

for $e = 1$ **to** `num_epoch` **do**
  for $b = 1$ **to** `num_batch` **do**
    sample a mini-batch $\{(x_i, y_i)\}_{i=1}^{m}$ from S; ▷ mini-batch of size m.
    for $i = 1$ **to** `batch_size` **do**
      $\delta_i^{(0)} = \xi$, with $\xi \sim \mathcal{U}(0, 1)$;
      for $t = 1$ **to** T **do**
        $\delta_i^{(t)} \leftarrow \texttt{Proj}\left[\delta_i^{(t-1)} + \alpha \texttt{sign}\left(\nabla_\theta \ell(x_i + \delta_i^{(t-1)}, y_i; \theta)\right)\right]$;
      end for
      $w_i^{\text{unn}} = \texttt{sigmoid}(-\gamma(\texttt{PM}_i - \beta))$;
    end for
    $\omega_i = M \times w_i^{\text{unn}} / \sum_j w_j^{\text{unn}}, \forall i \in [m]$; ▷ $\omega_i = 1$ during burn-in period.
    $\theta \leftarrow \theta - \eta \nabla_\theta \sum_{i=1}^{m} \omega_i \ell(x_i + \delta_i, y_i; \theta) + \mathcal{R}(x_i, y_i; \theta)$;
  end for
end for

- Either PM_nat, PM_adv, or PM_dif can be adopted in measuring the geometric distances.
- The sigmoid function is adopted in weight assignment.
- The learning objective of the form $\sum_i \omega_i \log P_{y_i}(x_i + \delta_i^{(T)}; \theta) + \mathcal{R}(x_i, y_i; \theta)$ is adopted.

Wang, et al., Probabilistic Margin for Instance Reweighting in Adversarial Training. NeurIPS 2021.

Adversarial Training: Instance Reweighted

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Table 2: Average accuracy (%) and standard deviation on CIFAR-10 dataset with ResNet-18.

	NAT	PGD	APGD	CW	AA
AT [29]	84.86 ± 0.17	48.91 ± 0.14	47.70 ± 0.06	51.61 ± 0.15	44.90 ± 0.53
TRADES [45]	84.00 ± 0.23	52.66 ± 0.16	52.37 ± 0.24	52.30 ± 0.06	48.10 ± 0.26
MART [40]	82.28 ± 0.14	53.50 ± 0.46	52.73 ± 0.57	51.59 ± 0.16	48.40 ± 0.14
FAT [46]	**87.97** ± **0.15**	46.78 ± 0.12	46.68 ± 0.16	49.92 ± 0.26	43.90 ± 0.82
AWP [42]	85.17 ± 0.40	52.63 ± 0.17	50.40 ± 0.26	51.39 ± 0.18	47.00 ± 0.25
GAIRAT [47]	83.22 ± 0.06	54.81 ± 0.15	50.95 ± 0.49	39.86 ± 0.08	33.35 ± 0.57
MAIL-AT	84.52 ± 0.46	**55.25** ± **0.23**	**53.20** ± **0.38**	48.88 ± 0.11	44.22 ± 0.21
MAIL-TRADES	81.84 ± 0.18	53.68 ± 0.14	52.92 ± 0.62	**52.89** ± **0.31**	**50.60** ± **0.22**

Wang, et al., Probabilistic Margin for Instance Reweighting in Adversarial Training. NeurIPS 2021.

Conclusion

Use instance equivalent adversarial training:

1. Madry, et al., Towards Deep Learning Models Resistant to Adversarial Attacks. ICLR 2018.
2. Zhang, et al., Theoretically Principled Trade-off between Robustness and Accuracy. ICML 2019.
3. Wang, et al., Improving Adversarial Robustness Requires Revising Misclassified Examples. ICLR 2020.
4. Zhang, et al., Attacks Which Do Not Kill Training Make Adversarial Training Stronger. ICML 2020.

Use instance-reweighted adversarial training:

5. Zhang, et al., Geometry-aware Instance-reweighted Adversarial Training. ICLR 2021.
6. Wang, et al., Probabilistic Margin for Instance Reweighting in Adversarial Training. NeurIPS 2021.

Xiao et al., DensePure: Understanding Diffusion Models for Adversarial Robustness. ICLR, 2023.

ACML 2023 Tutorial: Trustworthy Machine Learning on Adversarial Data

Part III: Adversarial Detection

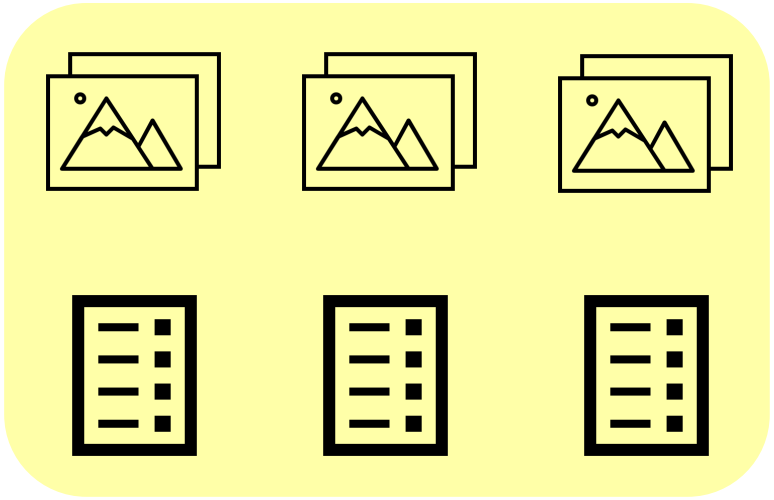
Let's try detect adversarial data!!

How to passively prevent from adversarial attacks

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Adversarial attack happens:



Test Data + Adversarial **Perturbations**

Discard the adversarial data

Input

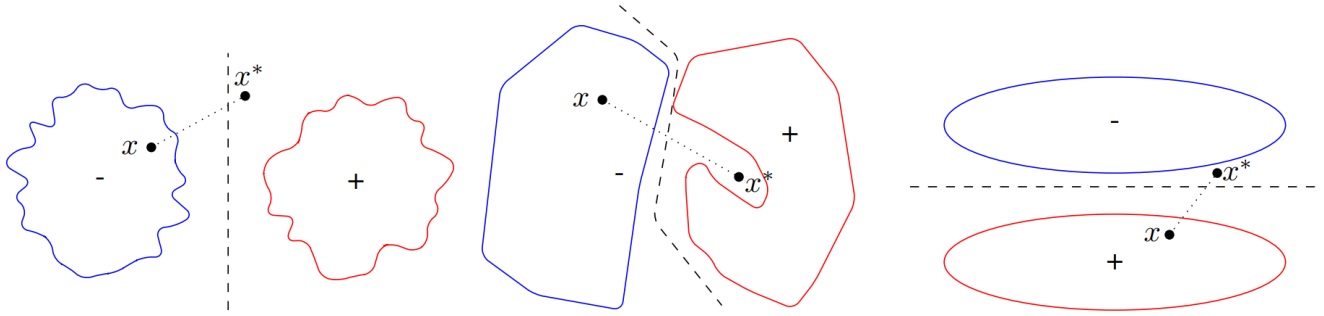
Well-trained NN,
Well-trained CNN
Well-trained Transformer
.
.

Predicted Labels

Adversarial Detection

Detecting Adversarial Samples from Artifacts

- Three possible situations of the adverisial sample x* lies in the data manifold:



(a) Two simple 2D submanifolds. (b) One submanifold has a 'pocket'. (c) Nearby 2D submanifolds.

Given the intuition that adversarial samples lie off the true data manifold, Feinman et al. devise two measures that can be used to detect adversarial samples:

- **Kernel Density:**

$$\hat{K}(x, \boldsymbol{X}_t) = \frac{1}{|\boldsymbol{X}_t|} \sum_{x_i \in \boldsymbol{X}_t} k_\sigma\big(\phi(x), \phi(x_i)\big),$$

where $k_\sigma(x, y) \sim exp(-\|x - y\|^2/\sigma^2)$.

- **Bayesian Uncertainty:**

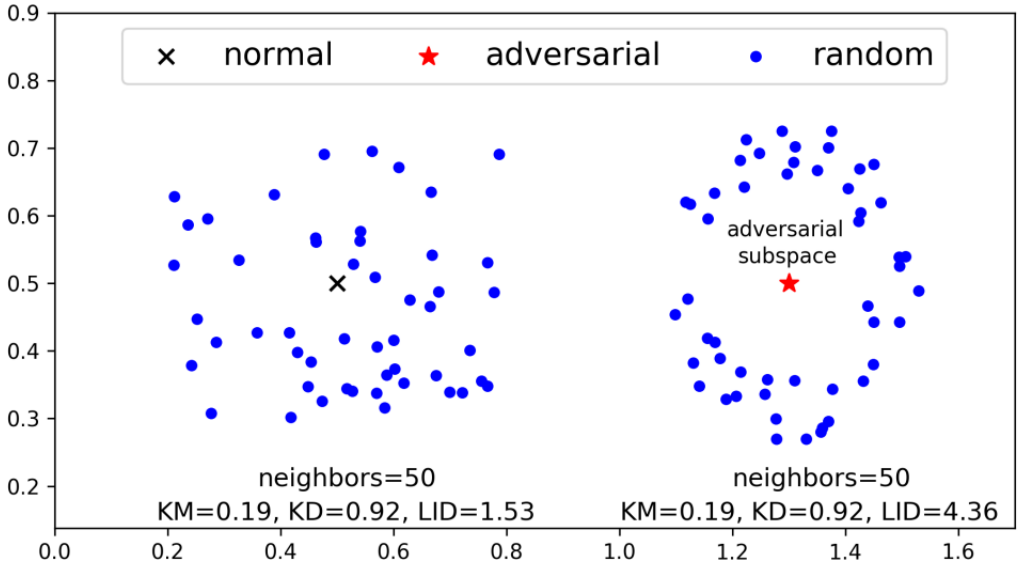
$$U(x^*) = \frac{1}{T}\sum_{i=1}^{T} \widehat{y_i^*}^T \widehat{y_i^*} - \left(\frac{1}{T}\sum_{i=1}^{T} \widehat{y_i^*}\right)^T \left(\frac{1}{T}\sum_{i=1}^{T} \widehat{y_i^*}\right)$$

Feinman et al., Detecting Adversarial Samples from Artifacts. Arxiv, 2017.

Adversarial Detection

Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality

- Ma et al. consider expansion-based measures of intrinsic dimensionality as an alternative density measure.



Kernel density is not effective for the detection when attack falls in local adversarial regions !

- **Local Intrinsic Dimensionality (LID):**

$$LID_F(r) = \lim_{\epsilon \to 0} \frac{\ln\left(F((1+\epsilon)\cdot r)/F(r)\right)}{\ln(1+\epsilon)} = \frac{r \cdot F'(r)}{F(r)},$$

$$LID_F = \lim_{r \to 0} LID_F(r)$$

where r ∈ R denotes the distance from x to other data samples.

- **Maximum Likelihood Estimation:**

$$\widehat{LID}(x) = -\left(\frac{1}{k}\sum_{i=1}^{k} \log \frac{r_i(x)}{r_k(x)}\right)^{-1}$$

Ma et al., Characterizing Adversarial Subspaces Using Local Intrisic Dimensionality. ICLR, 2018.

Adversarial Detection

A Simple Unified Framework for Detecting OOD Samples and Adversarial Attacks

- Lee et al. propose to detect abnormal test samples including OOD and adversarial ones using a pre-trained softmax neural classifier without re-training
  - Obtain the Mahalanobis distance-based score based on the classifier

$$M(\boldsymbol{x}) = \max_c -(f(\boldsymbol{x}) - \widehat{\mu_c})^T \widehat{\Sigma^{-1}} (f(\boldsymbol{x}) - \widehat{\mu_c})$$

  - Additional techniques to improve detection performance
    - **Input pre-processing:** Add a small controlled noise to a test sample to make in- and out-of-distribution samples more separable

$$\hat{\boldsymbol{x}} = \boldsymbol{x} + \varepsilon sign\big(\nabla_{\boldsymbol{x}} M(\boldsymbol{x})\big) = \boldsymbol{x} - \varepsilon sign\left(\nabla_{\boldsymbol{x}} (f(\boldsymbol{x}) - \widehat{\mu_c})^T \widehat{\Sigma^{-1}} (f(\boldsymbol{x}) - \widehat{\mu_c})\right)$$

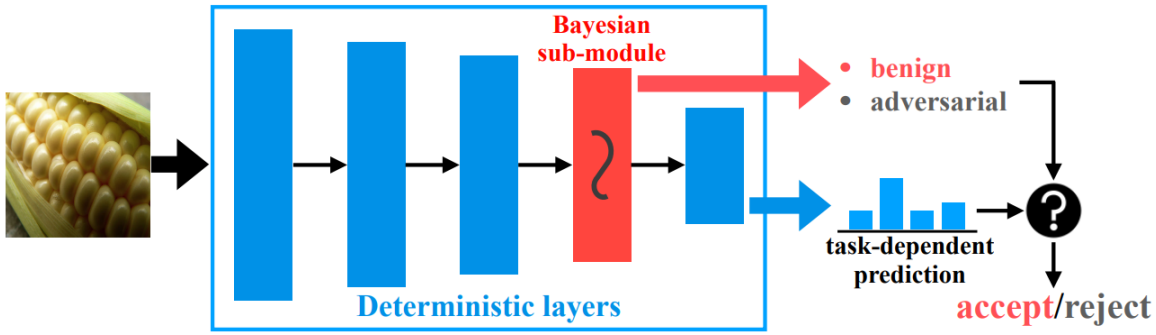
    - **Feature ensemble:** Measure and combine the confidence scores from not only the final features but also the other low-level features in DNNs to further improve the performance.

Lee et al., A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. NIPS, 2018.

Adversarial Detection

LiBRe: A Practical Bayesian Approach to Adversarial Detection

- Deng et al. leverage Bayesian neural networks (BNNs) and propose **Lightweight Bayesian Refinement (LiBRe)** for detecting adversarial test samples



- Given a pre-trained DNN, LiBRe converts its last few layers to be Bayesian and uses the uncertainty as a metric:

$$U(\boldsymbol{x}) = \frac{1}{T-1}\left[\sum_{t=1}^{T}\left\|z^{(t)}\right\|_2^2 - T\left(\left\|\frac{1}{T}\sum_{t=1}^{T}\left\|z^{(t)}\right\|_2^2\right\|_2^2\right)\right]$$

where $z^{(t)}$ is the t-th logits of the BNN.

Deng et al., LiBRe: A Practical Bayesian Approach to Adversarial Detection. CVPR, 2021.

Adversarial Detection: Distributional Difference?

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

The consensus in related works: Two-sample test is unaware of adversarial attacks.

A natural question: are natural and adversarial data really from different distributions? Answer: affirmative.

Our investigation: the previous use of MMD test on the purpose missed three key factors.

Factor 1. Limited representation power of the Gaussian kernel.

Factor 2. The overlook of the optimization for the used kernel parameters.

Factor 3. Non-IID adversarial data break a basic assumption of the MMD test.

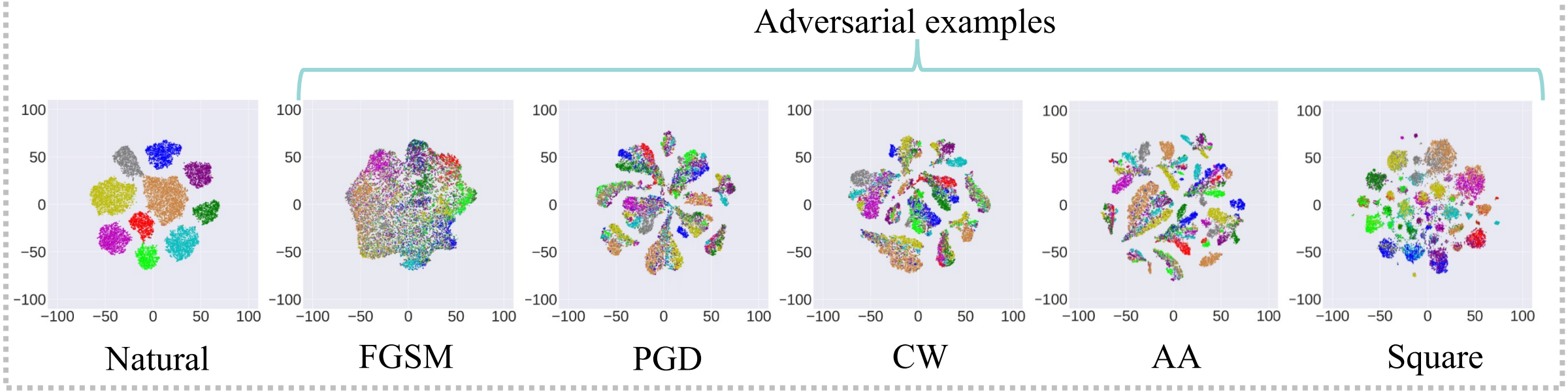
Gao et al., Maximum Mean Discrepancy Test is Aware of Adversarial Attacks. ICML, 2021.

Adversarial Detection: Distributional Difference?

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Different semantic meanings between natural and adversarial examples



Visualization of the difference using t-SNE.

Gao et al., Maximum Mean Discrepancy Test is Aware of Adversarial Attacks. ICML, 2021.

Adversarial Detection: Distributional Difference?

TMLR
TRUSTWORTHY MACHINE LEARNING AND REASONING

THE UNIVERSITY OF MELBOURNE

Kernel Architecture (designed for testing):

$$k_\omega(\boldsymbol{x}, \boldsymbol{y}) = \left[(1-\varepsilon_0)s_{\hat{f}}(\boldsymbol{x}, \boldsymbol{y}) + \varepsilon_0\right]q(\boldsymbol{x}, \boldsymbol{y})$$

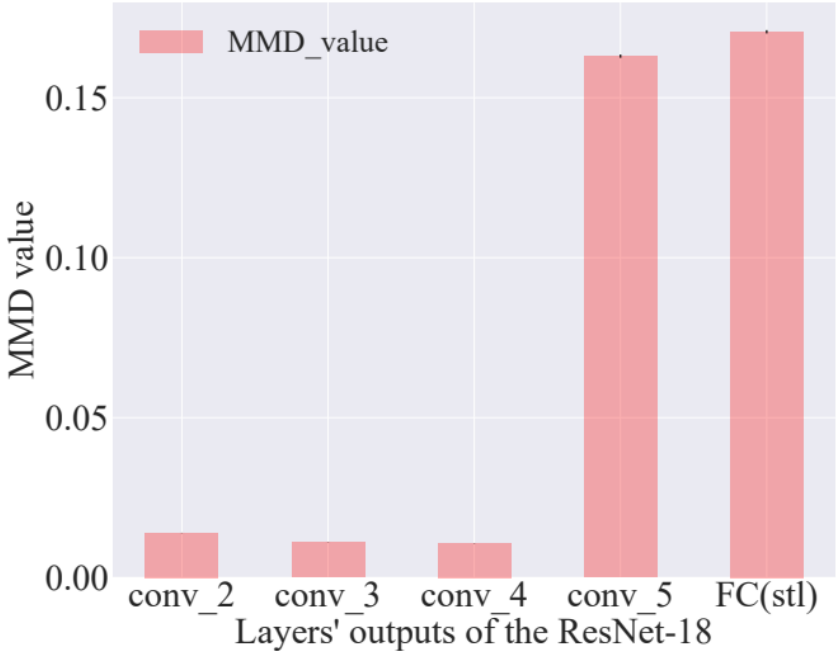
- $s_{\hat{f}}(\boldsymbol{x}, \boldsymbol{y}) = \kappa(\phi_p(\boldsymbol{x}), \phi_p(\boldsymbol{y}))$ is a deep kernel function;
- ϕ_p is the second to the last fully connected layer in $\hat{f}$;

(we use ϕ_p to extract semantic features.)

- κ is the Gaussian kernel with the optimized bandwidth σ_{ϕ_p};
- $q(\boldsymbol{x}, \boldsymbol{y})$ is the Gaussian kernel with bandwidth σ_q;
- $\varepsilon_0 \in (0,1)$;

($q(\boldsymbol{x}, \boldsymbol{y})$ and ε_0 ensure that $k_\omega(\boldsymbol{x}, \boldsymbol{y})$ is a characteristic kernel.)

- The set of parameters of k_ω is $\omega = \left\{\varepsilon_0, \sigma_{\phi_p}, \sigma_q\right\}$.



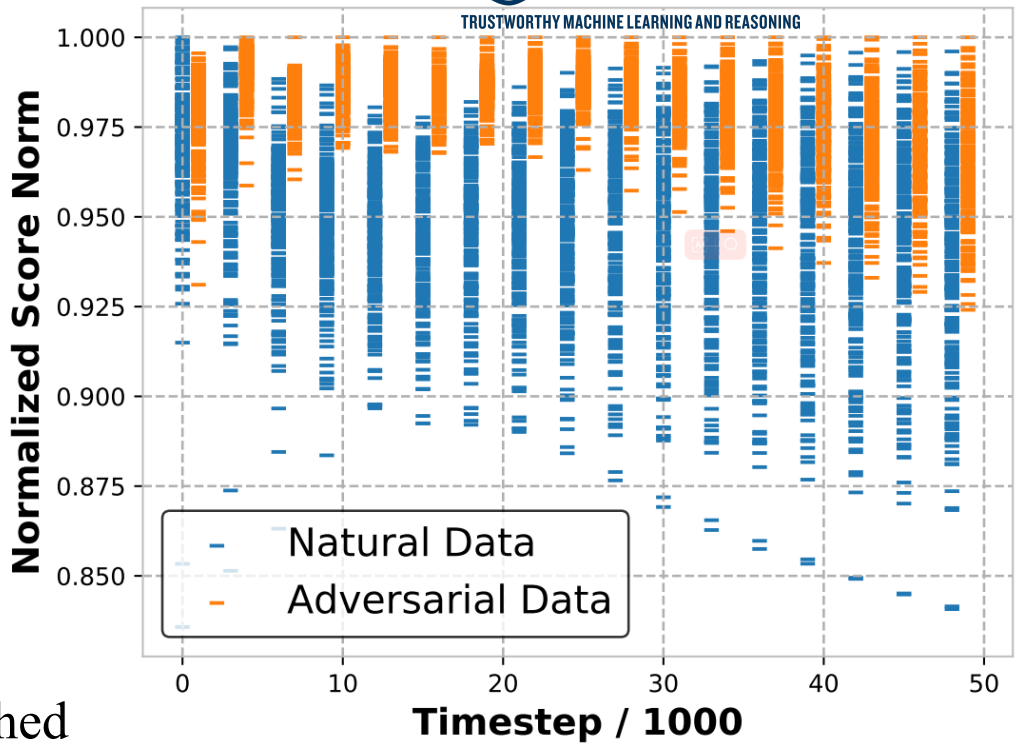
Discrepancy of MMD value between different layers' outputs in $\hat{f}$.

Gao et al., Maximum Mean Discrepancy Test is Aware of Adversarial Attacks. ICML, 2021.

Adversarial Detection from the score

Adversarial Detection with Expected Perturbation Score

- Intuition from score function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$
  - Score $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ represents the momentum of the sample towards high density areas of natural data (Song et al., 2019)
  - A lower score norm indicates the sample is closer to the high-density areas of natural data
- Consider the score of data perturbed by a diffusion process due to the score of original natural/adversarial data is not distinguished



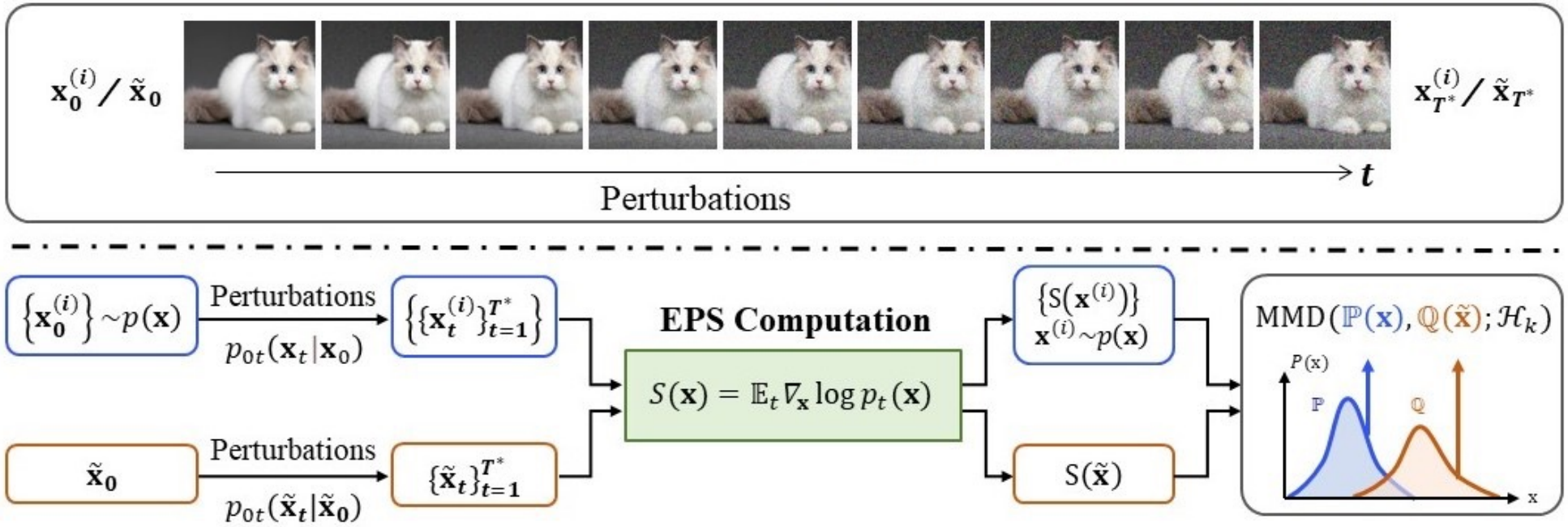
Most natural samples have lower score norms than adversarial samples, but they are very sensitive to the timesteps due to the significant overlap across all timesteps

One score is useful but not effective enough!

Zhang et al., Detecting Adversarial Data by Probing Multiple Perturbations Using Expected Perturbation Score. ICML, 2023.

Adversarial Detection from the score

Adversarial Detection with Expected Perturbation Score



- **Computing expected perturbation score (EPS)** using a pre-trained score model
  - Add perturbations to a set of **natural images** and **a test image** following a diffusion process with time step T^* and obtain their EPSs via the score model

Zhang et al., Detecting Adversarial Data by Probing Multiple Perturbations Using Expected Perturbation Score. ICML, 2023.

Can we detect the single-instance-based difference?

In the second work, we detect adversarial data using batches instead of instances (like many papers did). However, we have a question:

Can we do the single-instance-based adversarial data detection perfectly?

In our recent work [1], we find that the answer is **NO.** Because adversarial data and natural data are too close to be perfectly distinguished. We provide an **impossibility theorem** in [1] to support this answer.

[1] Z. Fang, Y. Li, J. Lu, J. Dong, B. Han, **F. Liu***, Is Out-of-distribution Detection Learnable?. In NeurIPS 2022. **Outstanding Paper Award.** (* represents corresponding author.)

THE UNIVERSITY OF MELBOURNE

Thank you

fengliu.ml@gmail.com

https://fengliu90.github.io/