Robustness

AI602: Recent Advances in Deep Learning

Lecture 11

Slide made by

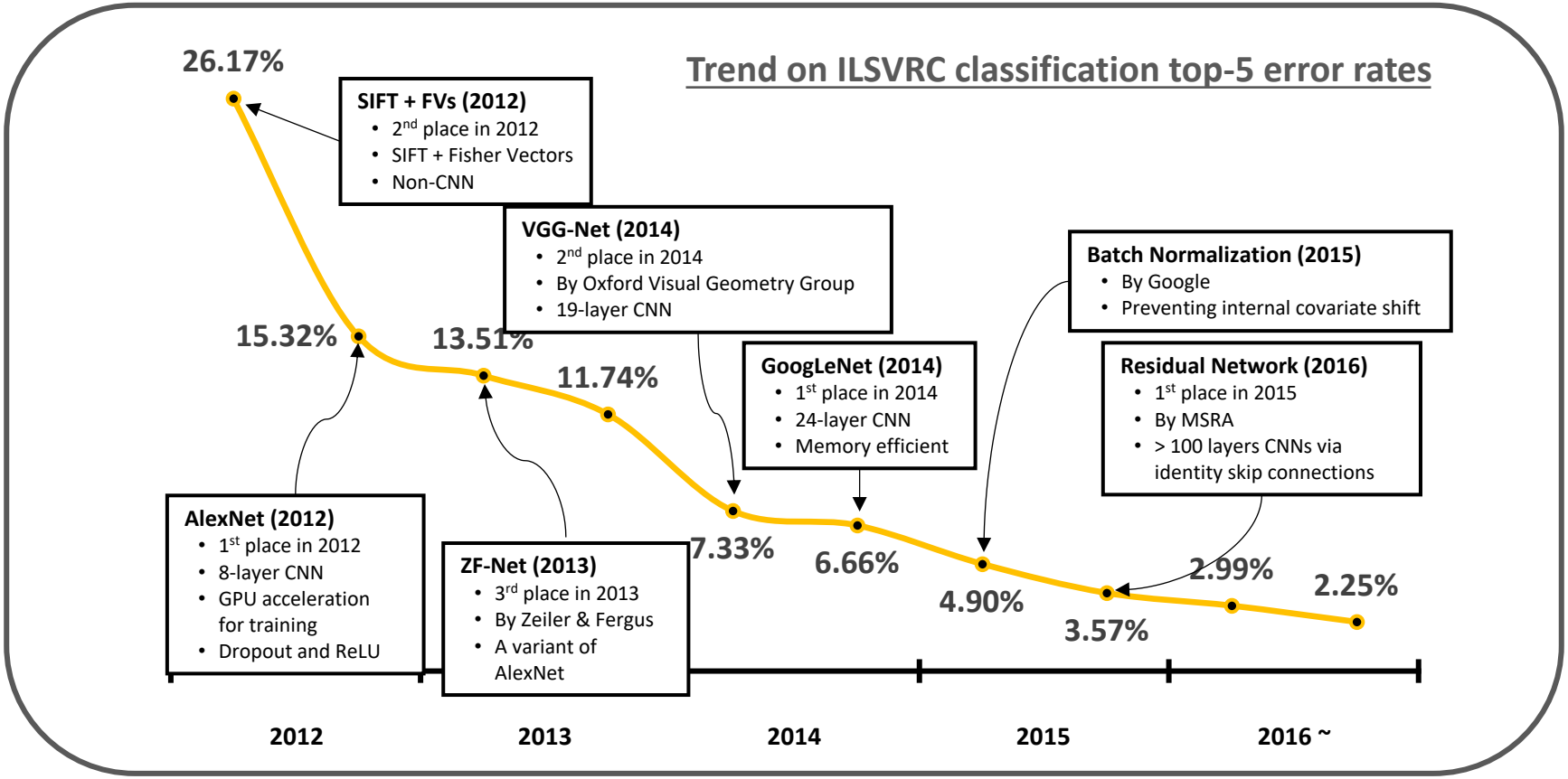
Jongheon Jeong , Jihoon Tack and Seojin Kim

KAIST EE & AI

Table of Contents

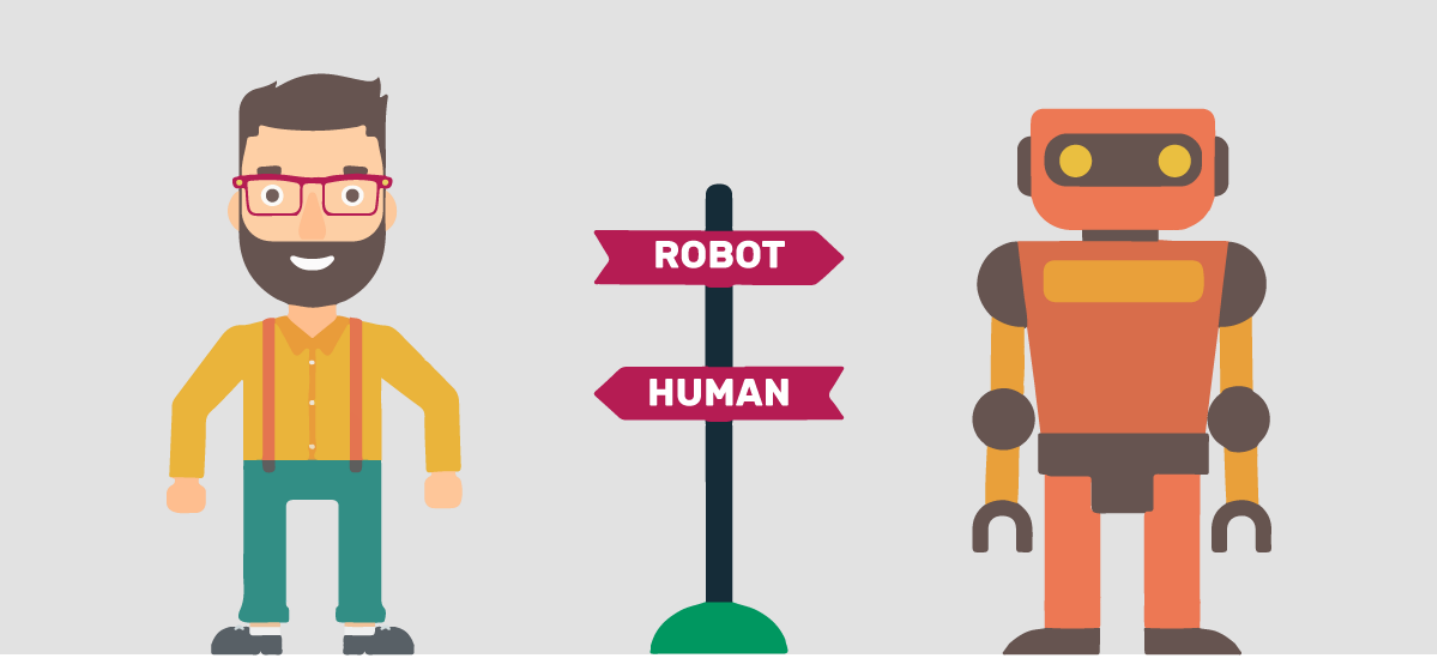
Table of Contents

- **Deep learning system** have achieved state-of-art on various AI-related tasks
  - Super-human performance on image recognition problems



Back Side of The Neural Network Success

- **Deep learning system** have achieved state-of-art on various AI-related tasks
  - Super-human performance on image recognition problems
- **Problem**: ML systems are highly vulnerable
  - (a) to a small noise on input that are specifically designed by an adversary
  - (b) to distributionally shifted inputs, i.e., train and test input's distribution differs
  - In other words, **answer of machine ≠ answer of human**



WORDBERRY
SOFTWARE TRANSLATION & LOCALIZATION

*source: https://wordberry.com/choosing-human-vs-machine-website-translation/

What is The Adversarial Example?

- **Deep learning system** have achieved state-of-art on various AI-related tasks
  - Super-human performance on image recognition problems
- **Problem**: ML systems are highly vulnerable
  - **(a) to a small noise on input that are specifically designed by an adversary**
  - (b) to distributionally shifted inputs, i.e., train and test input's distribution differs
  - In other words, **answer of machine ≠ answer of human**
- Even state-of-the-art-level neural networks make erroneous outputs
  - **Example**: GoogleNet trained on ImageNet dataset

Humans can not distinguish them

*source: Goodfellow et al., Explaining and Harnessing Adversarial Examples, ICLR 2015

What is The Adversarial Example?

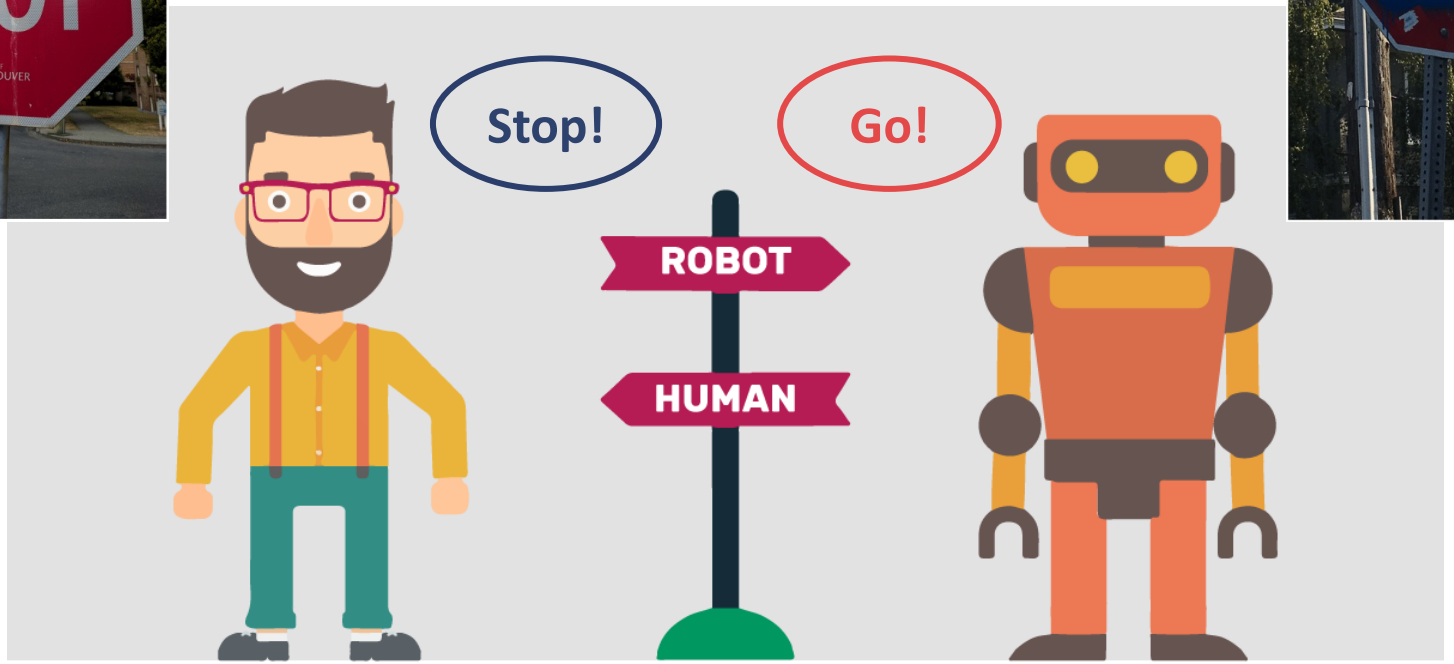
- **Deep learning system** have achieved state-of-art on various AI-related tasks
  - Super-human performance on image recognition problems
- **Problem**: ML systems are highly vulnerable
  - **(a) to a small noise on input that are specifically designed by an adversary**
  - (b) to distributionally shifted inputs, i.e., train and test input's distribution differs
  - In other words, **answer of machine ≠ answer of human**
- Even state-of-the-art-level neural networks make erroneous outputs
  - **Example**: GoogleNet trained on ImageNet dataset

It is called an adversarial example!

*source: Goodfellow et al., Explaining and Harnessing Adversarial Examples, ICLR 2015

Threat of Adversarial Examples

- Adversarial examples raise issues critical to the "**AI safety**" in the real world
  - e.g. Autonomous vehicles may misclassify graffiti stop signs



WORDBERRY
SOFTWARE TRANSLATION & LOCALIZATION

*source: Eykholt et al., Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR 2018

Threat of Adversarial Examples

- Furthermore, adversarial examples exist across various tasks or modalities
  - Adversarial examples for segmentation task [Xie et al., 2017]

  - Adversarial examples for automatic speech recognition [Qin et al., 2019]

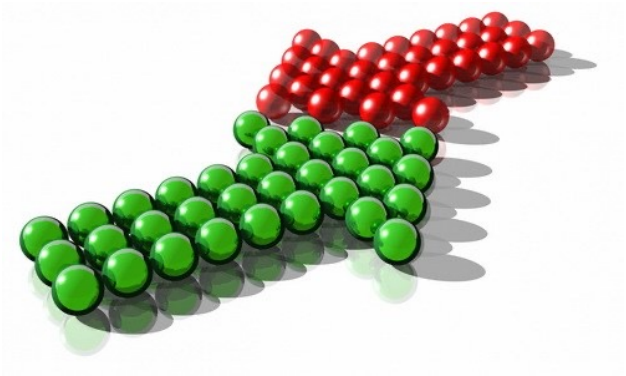
Clean: "The sight of you bartley to see you living and happy and successful can I never make you understand what that means to me"

Adversarial: "Hers happened to be in the same frame too but she evidently didn't care about that"

*source:
Xie et al., Adversarial Examples for Semantic Segmentation and Object Detection, ICCV 2017
Qin et al., Imperceptible, Robust, and Targeted Adversarial Examples for Automatic Speech Recognition, ICML 2019

The Adversarial Game: Attacks and Defenses

- The literature of adversarial example commonly stated in security perspective
  - **Attacks:** Design inputs for a ML system to produce erroneous outputs
  - **Defenses:** Prevent the misclassification by adversarial examples



- In this perspective, specifying a **threat model** of the game is important

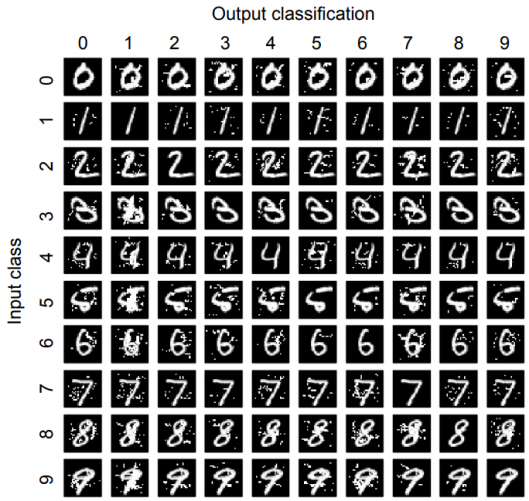
1. **Adversary goals**
2. **Adversarial capabilities**
3. **Adversary knowledge**

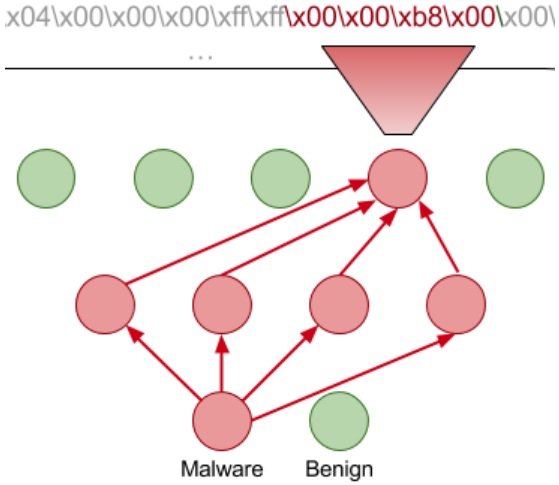
*source: https://gwynteatro.wordpress.com/2011/10/30/ambiguity-and-contradiction-leadership-certainties

The Adversarial Game: Threat model

- The literature of adversarial example commonly stated in security perspective
- In this perspective, specifying a **threat model** of the game is important

1. **Adversary goals**: Simply to cause misclassification, or else?
   - Some adversary may be interested in to attack into a target class of their choice
     - "**Source-target**" [Papernot et al., 2016], or "**targeted**" [Carlini & Wagner, 2017] attack
   - In other setting, only a specific type of misclassification may be interesting
     - e.g. Malware detection: "Benign → malware" is usually out-of-interest





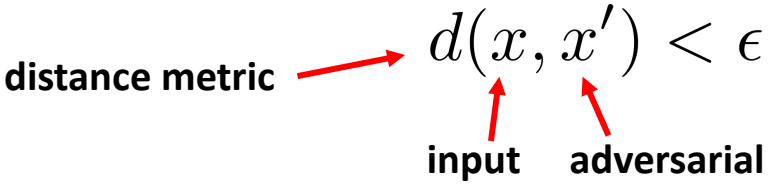
*source:
Carlini & Wagner, Towards Evaluating the Robustness of Neural Networks, IEEE SSP 2017
https://devblogs.nvidia.com/malware-detection-neural-networks/

- The literature of adversarial example commonly stated in security perspective
- In this perspective, specifying a **threat model** of the game is important

2. Adversarial capabilities

- Reasonable constraints to adversary allow us to build more meaningful defenses
  - Too large perturbations to an image may break even the human's decision

- To date, most defenses restrict the adversary to make "small" changes to inputs

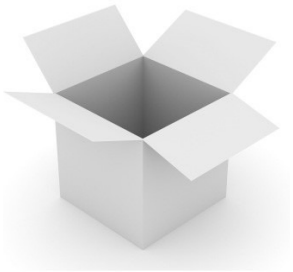


- A common choice for $d(\cdot,\cdot)$ is ℓ_p**-distance** (especially for image classification)
  - ℓ_∞**-norm ball**: the adversary cannot modify each pixel by more than ϵ
  - ℓ_0**-norm ball**: the adversary can arbitrary change at most ϵ pixels

- The literature of adversarial example commonly stated in security perspective
- In this perspective, specifying a **threat model** of the game is important

3. Adversary knowledge

- A threat model must describe what knowledge the adversary is assumed to have
  - **White-box model:** Complete knowledge of the model and its parameter
  - **Black-box model:** No knowledge of the model
- **Gray-box**: Some threat models specify the various degree of access
  - A limited number of queries to the model
  - Access to the predicted probabilities, or just class
  - Access to the training data
- The guiding principle: **Kerckhoffs' principle** [Kerckhoffs, 1883]
  - The adversary is assumed to completely know the inner workings of the defense



white-box

black-box

*source:
https://emperorsgrave.wordpress.com/2016/10/18/black-box/
https://reqtest.com/testing-blog/test-design-techniques-explained-1-black-box-vs-white-box-testing/

- A precise threat model → **well-defined measures** of adversarial robustness
  1. "**Adversarial risk**": The worst-case loss L for a given perturbation budget

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{x':d(x,x')<\epsilon} L(f(x'),y)\right]$$

Data distribution **model**

  2. The average minimum-distance of the adversarial perturbation

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\min_{x'\in A_{x,y}} d(x,x')\right]$$

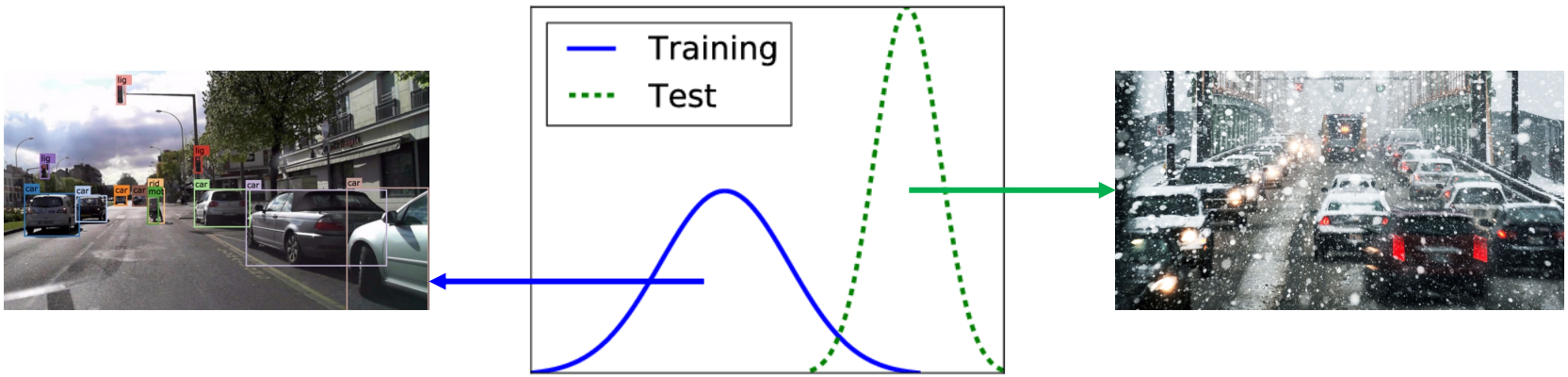
set of adv. examples

- For misclassification, $A_{x,y} = \{x' : f(x') \neq y\}$
- For targeted attack, $A_{x,y} = \{x' : f(x') = t\}$ for some target class t

- **Key challenge:** Computing adversarial risk is usually intractable
  - We have to approximate these quantities
    - Much harder problem than approximating "average-case" robustness
  - The heart reason of why evaluating adversarial robustness is difficult

What is Distributional Shift?

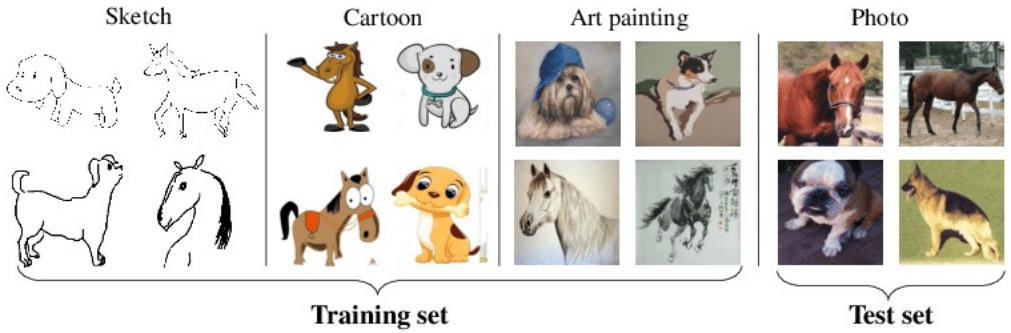
- **Deep learning system** have achieved state-of-art on various AI-related tasks
  - Super-human performance on image recognition problems
- **Problem**: ML systems are highly vulnerable
  - (a) to a small noise on input that are specifically designed by an adversary
  - **(b) to distributionally shifted inputs, i.e., train and test input's distribution differs**
  - In other words, **answer of machine ≠ answer of human**
- Various machine learning method assumes $P_{\text{train}} = P_{\text{test}}$
  - However in real-world scenarios, distributional shift occurs $P_{\text{train}} \neq P_{\text{test}}$
  - E.g., autonomous driving car trained on Korea may not generalize on Canada



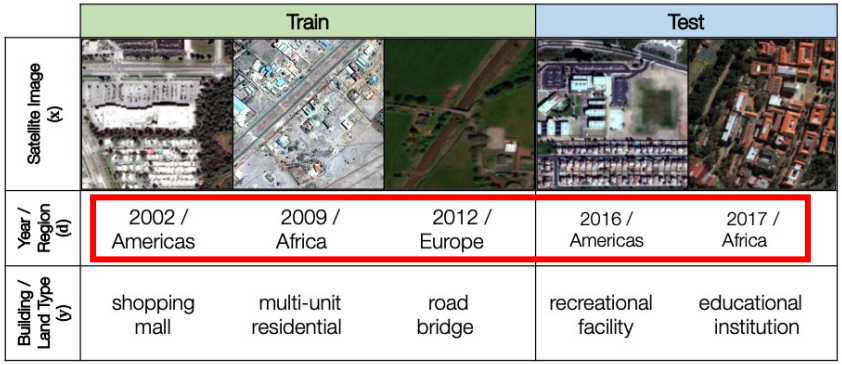
*source: https://www.researchgate.net/figure/Example-of-covariate-shift-training-and-test-data-having-different-distributions_fig1_322568228

What is Distributional Shift?

- **Distributional shift occurs across various domains**
  - Image, e.g., natural corruptions
  - Reinforcement learning (RL), e.g., offline RL
  - Time-series and natural language, e.g., shift between the prior and future data
  - Even on segmentation and chemical classification problems



Domain shift



Distribution shit across time [Koh et al., 2021]

*source:
https://www.researchgate.net/figure/Examples-from-the-dataset-PACS-1-for-domain-generalization-The-training-set-is_fig1_349787277
Koh et al., WILDS: A Benchmark of in-the-Wild Distribution Shifts, ICML 2021

Table of Contents

Table of Contents

Adversarial Attacks: Fast Gradient Sign Method [Goodfellow et al., 2015]

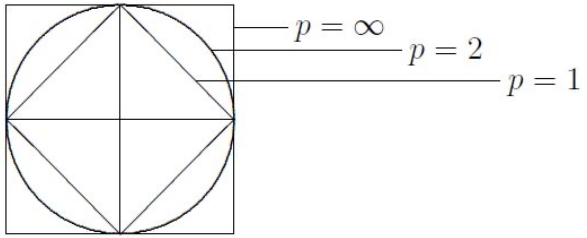
- In **vision ML system**, the following threat model is common:
  1. **Goal -** Untargeted attack: Find $\text{argmax}_{x':d(x,x')<\epsilon} L(f(x'), y)$
  2. **Capabilities -** Pixel-wise restriction: $d(x, x') = ||x - x'||_\infty := \max_i |x_i - x'_i| \leq \epsilon$
  3. **Knowledge -** White-box: Full access to the target network

- **Fast Gradient Sign Method (FGSM)**: A fast approximation of this threat model
  - **Idea:** In white-box setting, one can get the gradients w.r.t input of the network
- FGSM solves the maximization via **linearizing** the loss:

$$\max_{x':||x-x'||_\infty \leq \epsilon} L(f(x'), y) \approx L(f(x), y) + \delta \cdot \nabla_x L(f(x), y)$$

- To meet the max-norm constraint, FGSM takes $\text{sign}(\cdot)$ on the gradient
  - **Quiz.** Why the use of $\text{sign}(\cdot)$ maximizes the loss?

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x L(f(x), y))$$



*source: Goodfellow et al., Explaining and Harnessing Adversarial Examples, ICLR 2015

Adversarial Attacks: Least-likely Class Method [Kurakin et al., 2017b]

- The idea of FGSM can be directly applied to **targeted** attack model:
  1. **Goal -** Targeted attack
  2. **Capabilities -** Pixel-wise restriction: $d(x, x') = ||x - x'||_\infty := \max_i |x_i - x'_i| \leq \epsilon$
  3. **Knowledge -** White-box: Full access to the target network

- Unlike FGSM, **Least-likely Class Method** minimizes the loss for the target class
- Nevertheless, one could also linearize the loss L

$$\min_{x': ||x - x'||_\infty \leq \epsilon} L(f(x'), y_{\text{target}})$$

- This formulation leads to an attack method similar to FGSM:

$$x' = x - \epsilon \cdot \text{sign}(\nabla_x L(f(x), y_{\text{target}}))$$

Now, we perform "gradient descent"

*source: Kurakin et al., Adversarial Machine Learning at Scale, ICLR 2017

Adversarial Attacks: Projected Gradient Descent [Madry et al., 2018]

- FGSM can be **generalized** toward a stronger method
  1. Single-step update → multi-step optimization
  2. $\text{sign}(\cdot)$ → Generalized projection operation

- Essentially, our goal is to solve the following optimization:

$$\max_{x' \in x+\mathcal{B}} L(f(x'), y)$$

set of neighbors

- **Projected Gradient Descent (PGD)** is a direct way to solve this:

$$x^{t+1} = \Pi_{x+\mathcal{B}}(x^t + \alpha \cdot \text{sign}(\nabla_x L(f(x^t), y)))$$

projection

  - **Basic Iterative Method (BIM):** $x^0 := x$
  - Usually, **PGD** refers the case when x^0 is randomly-chosen inside $x + \mathcal{B}$

- In some sense, PGD is regarded as the strongest first-order adversary
  - It is the best way we could try using only gradient information

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

- **Carlini & Wagner (CW)**: Even tighter approximation is possible:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\min_{x'\in A_{x,y}} d(x,x')\right]$$

- **CW** attempts to directly minimize the distance $||\delta||$ in targeted attack

$$\min_{\delta: k(x+\delta)=y_{\text{target}}} \|\delta\|_2$$

  - **Key challenge**: How to incorporate the constraint during optimization
  - CW takes the Lagrangian relaxation to allow the gradient-based optimization:

$$\min_{\delta} \|\delta\|_2 + \alpha \cdot g(x+\delta)$$

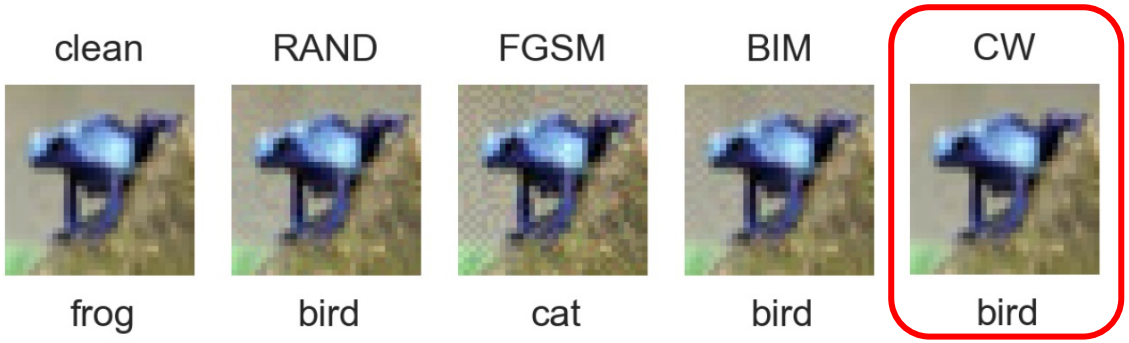
  - $g(x) = \left(\max_{i\neq\text{target}} f_i(x) - f_{y_{\text{target}}}(x)\right)^+$ where $\max(0,x)$
  - $g(x)$ attains the minimum when x is an adversarial example

*source: Carlini & Wagner, Towards Evaluating the Robustness of Neural Networks, IEEE S&P 2017

- **Experimental Results**
  - CW finds **much smaller avg. minimum-distance** than DeepFool

		CIFAR-10		CIFAR-100		SVHN	
		L_∞	Acc.	L_∞	Acc.	L_∞	Acc.
DenseNet	Clean	0	95.19%	0	77.63%	0	96.38%
	FGSM	0.21	20.04%	0.21	4.86%	0.21	56.27%
	BIM	0.22	0.00%	0.22	0.02%	0.22	0.67%
	DeepFool	0.30	0.23%	0.25	0.23%	0.57	0.50%
	CW	0.05	0.10%	0.03	0.16%	0.12	0.54%
ResNet	Clean	0	93.67%	0	78.34%	0	96.68%
	FGSM	0.25	23.98%	0.25	11.67%	0.25	49.33%
	BIM	0.26	0.02%	0.26	0.21%	0.26	2.37%
	DeepFool	0.36	0.33%	0.27	0.37%	0.62	13.20%
	CW	0.08	0.00%	0.08	0.01%	0.15	0.04%

  - Comparison of images generated from several attacks [Y. Song et al., 2018]

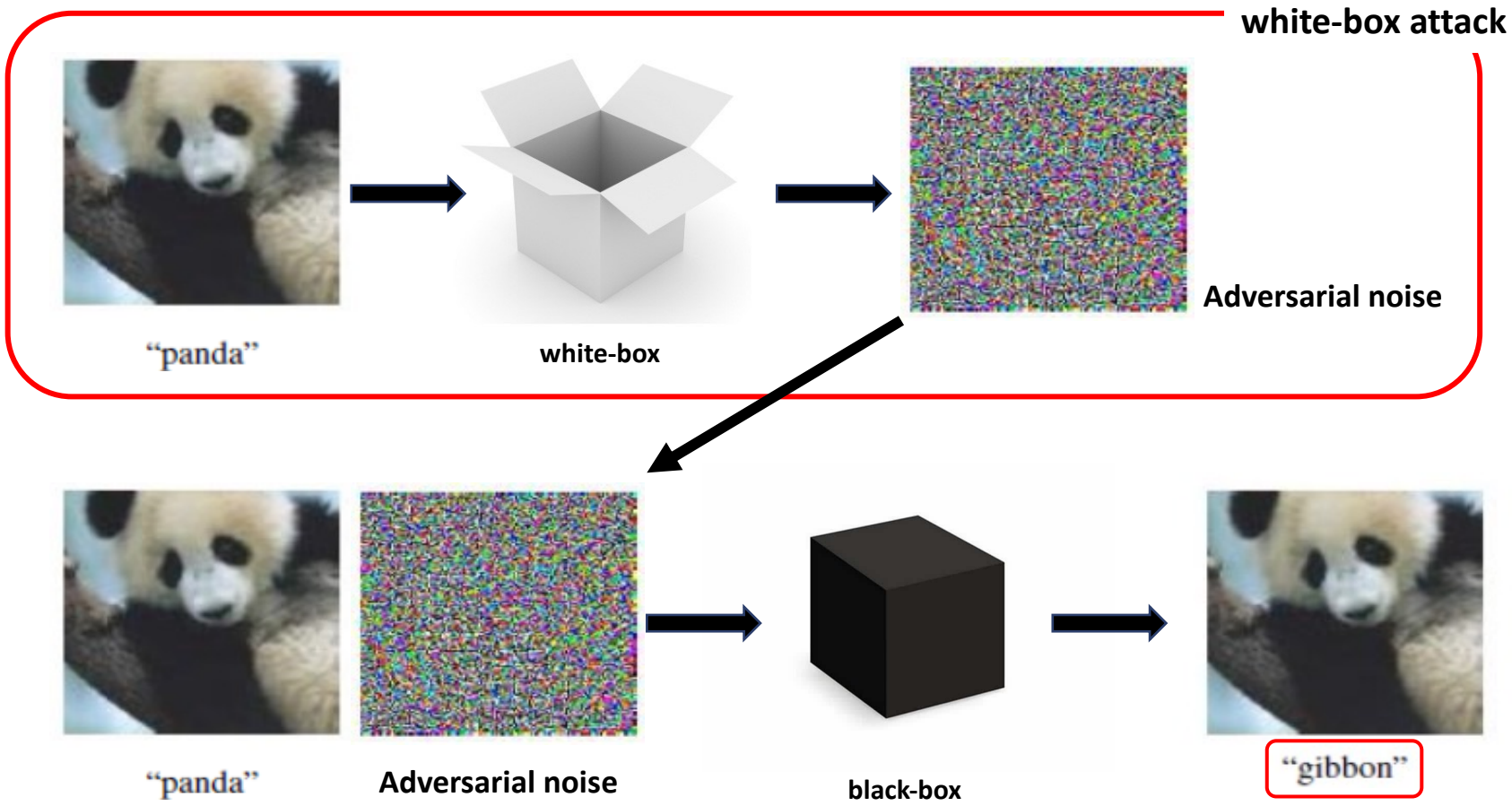


It is the most similar to clean image

*source:
Carlini & Wagner, Towards Evaluating the Robustness of Neural Networks, IEEE S&P 2017
Y. Song et al., PixelDefend, ICLR 2018

Adversarial Attacks: Transferability of Adversarial Example

- Some adversarial examples strongly transfer across **different networks**



*source: Goodfellow et al., Explaining and Harnessing Adversarial Examples, ICLR 2015

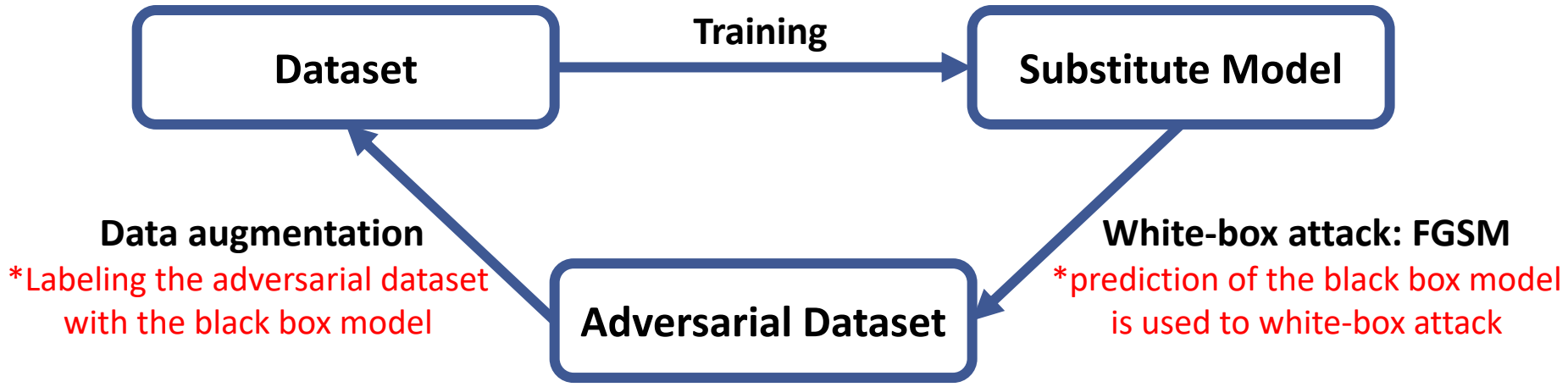
- **Motivation:** The transferability enables us to attack a black-box model
  - **Idea:** Finding an adversarial example via white-box attack on the local substitute model
  - **Goal:** Training a local substitute model via FGSM-based adversarial dataset augmentation
    - FGSM-based adversarial examples are computed to change the prediction of the black-box model

$$x' = x + \epsilon \cdot \mathrm{sign}(\nabla_x L(f(x), y_{\mathrm{pred}}))$$

substitute model

black-box prediction

- **Method:**



- **Experimental Results**
  - Black-box attack to the Amazon and Google Oracle
  - Two types of architecture:
    - **DNN**: Deep Neural Network
    - **LR**: Logistic Regression

Misclassification rates (%)

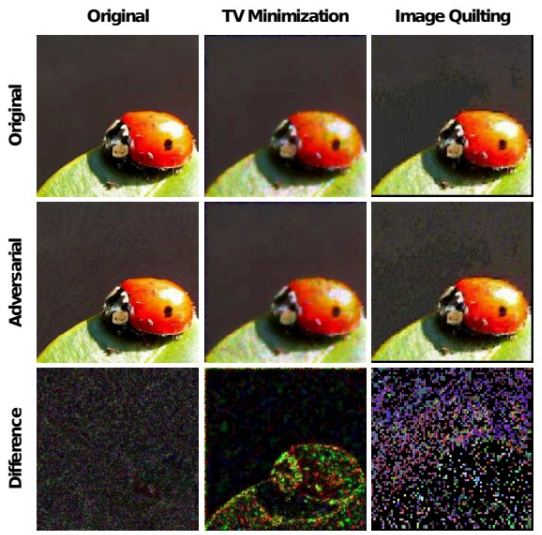
		Amazon		Google	
Epochs	Queries	DNN	LR	DNN	LR
$\rho = 3$	800	87.44	96.19	84.50	88.94
$\rho = 6$	6,400	**96.78**	**96.43**	**97.17**	92.05

Number of queries to train the local substitute model

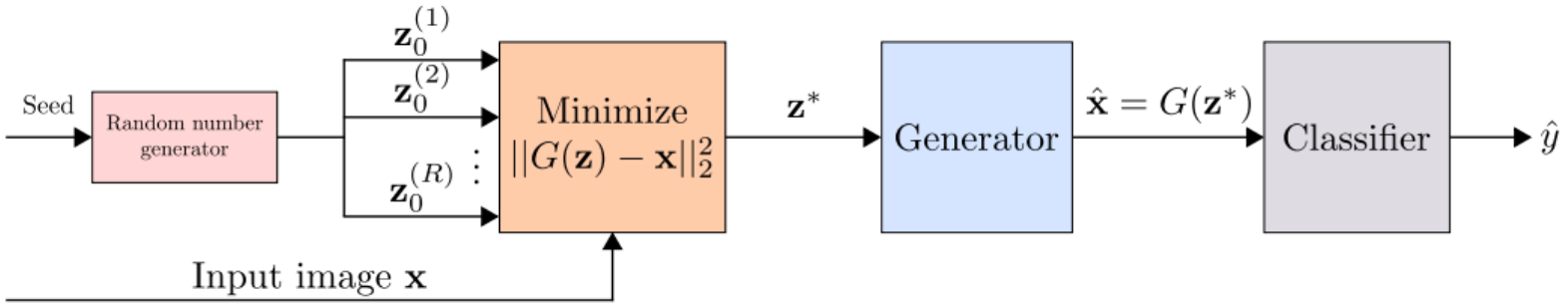
*source: Papernot et al., Practical Black-Box Attacks against Machine Learning, ACM CCS 2017

Obfuscated Gradients: False Sense of Security [Athalye et al., 2018]

- In ICLR 2018, 9 defense papers were published including adversarial training:
  - Adversarial training [Madry et al., 2018]
  - Thermometer Encoding [Buckman et al., 2018]
  - Input Transformations [Guo et al., 2018]
  - Local Intrinsic Dimensionality [Ma et al., 2018]
  - Stochastic Activation Pruning [Dhillon et al., 2018]
  - Defense-GAN [Samangouei et al., 2018]
  - PixelDefend [Song et al., 2018]
  - …



Input transformation [Guo et al., 2018]

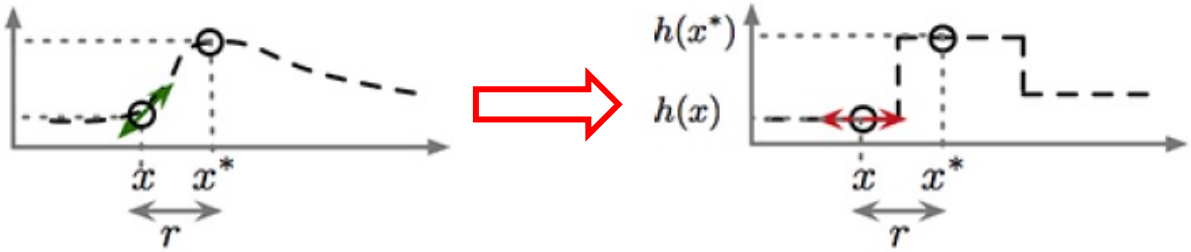


Defense-GAN [Samangouei et al., 2018]

*source: Athalye et al., Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples, ICML 2018

Obfuscated Gradients: False Sense of Security [Athalye et al., 2018]

- In ICLR 2018, 9 defense papers were published including adversarial training:
  - Adversarial training [Madry et al., 2018]
  - Thermometer Encoding [Buckman et al., 2018]
  - Input Transformations [Guo et al., 2018]
  - Local Intrinsic Dimensionality [Ma et al., 2018]
  - Stochastic Activation Pruning [Dhillon et al., 2018]
  - Defense-GAN [Samangouei et al., 2018]
  - PixelDefend [Song et al., 2018]
  - …
- **Athalye et al. (ICML 2019)**: In fact, most of them are "fake" defenses
  - **"Fake" defense?**: They don't aim the non-existence of adversarial example
    - Rather, they aim to obfuscate the gradient information
    - Obfuscated gradient makes gradient-based attacks (FGSM, PGD, …) harder



*source: Athalye et al., Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples, ICML 2018

Obfuscated Gradients: False Sense of Security [Athalye et al., 2018]

- **Athalye et al. (ICML 2019)**: In fact, most of them are "fake" defenses
  - **"Fake" defense?**: They don't aim the non-existence of adversarial example
    - Rather, they aim to obfuscate the gradient information
    - Obfuscated gradient makes gradient-based attacks (FGSM, PGD, …) harder

  - They identified **three obfuscation techniques** used in the defenses

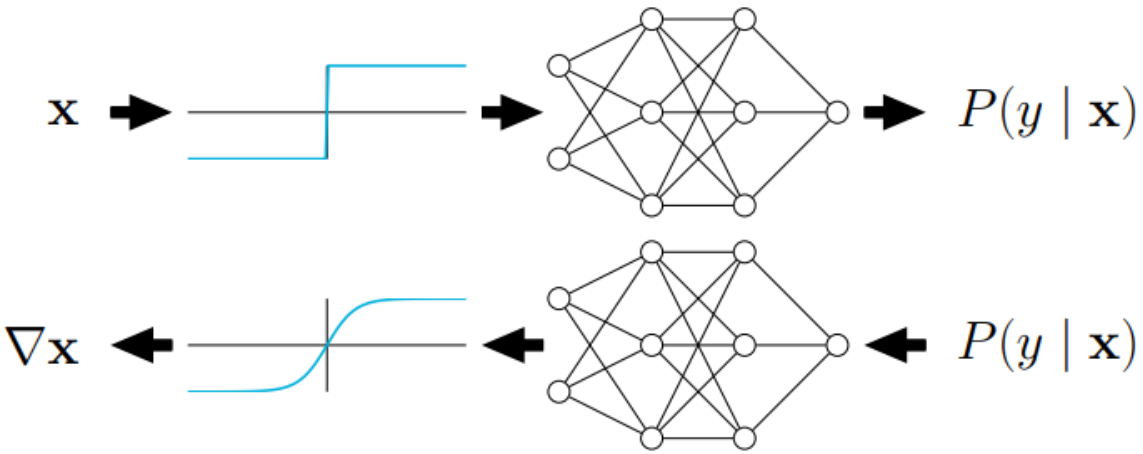
Obfuscation	Defenses
Shattered Gradients	**Existence of a non-differentiable layer**
	• Thermometer Encoding [Buckman et al., 2018] • Input Transformation [Guo et al., 2018] • Local Intrinsic Dimensionality (LID) [Ma et al., 2018]
Stochastic Gradients	**Artificial randomness on computing gradient**
	• Stochastic Activation Pruning (SAP) [Dhillon et al., 2018] • Mitigating Through Randomization [Xie et al., 2018]
Exploding & Vanishing Gradients	**Multiple iterations, or extremely deep DNN**
	• Pixel Defend [Song et al., 2018] • Defense-GAN [Samangouei et al., 2018]

*source: Athalye et al., Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples, ICML 2018

Obfuscated Gradients: False Sense of Security [Athalye et al., 2018]

- **Athalye et al. (ICML 2019)**: In fact, most of them are "fake" defenses
  - **"Fake" defense?**: They don't aim the non-existence of adversarial example
    - Rather, they aim to obfuscate the gradient information
    - Obfuscated gradient makes gradient-based attacks (FGSM, PGD, …) harder

- Those kinds of defenses can be easily bypassed by **3 simple tricks**
  1. **Backward Pass Differentiable Approximation (BPDA)**
     - Replace the non-differentiable parts only at backward pass
     - Use some differentiable approximative function



*source: Athalye et al., Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples, ICML 2018

Obfuscated Gradients: False Sense of Security [Athalye et al., 2018]

- **Athalye et al. (ICML 2019)**: In fact, most of them are "fake" defenses
  - **"Fake" defense?**: They don't aim the non-existence of adversarial example
    - Rather, they aim to obfuscate the gradient information
    - Obfuscated gradient makes gradient-based attacks (FGSM, PGD, …) harder

- Those kinds of defenses can be easily bypassed by **3 simple tricks**

2. Expectation Over Transformation (EOT)

- Take the expectation of attacks to mitigate stochastic defenses

$$\max_{x':d(x,x')<\epsilon} \mathbb{E}_{t\sim T}[L(f(t(x')),y)]$$

Random transformation

3. Reparameterization

- Replace deep or recurrent parts by simpler differentiable function

*source: Athalye et al., Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples, ICML 2018

Obfuscated Gradients: False Sense of Security [Athalye et al., 2018]

- **Athalye et al. (ICML 2019)**: In fact, most of them are "fake" defenses
  - **"Fake" defense?**: They don't aim the non-existence of adversarial example
    - Rather, they aim to obfuscate the gradient information
    - Obfuscated gradient makes gradient-based attacks (FGSM, PGD, …) harder

- Those kinds of defenses can be easily bypassed by **3 simple tricks**
  - **6 of the 9** defense papers were completely broken using those tricks
  - 1 of the 9 was partially broken (Defense-GAN)
  - Adversarial training [Madry et al. 2018; Na et al., 2018] were the only survivals

Defense	Type	Behavior	Attack technique
Thermometer Encoding	Shattered	Black-box is better	BPDA
Local Intrinsic Dimensionality (LID)	Shattered	Unbounded attack do not reach 100%	BPDA
Input Transformation	Shattered	Black-box is better	BPDA, EOT
Stochastic Activation Pruning (SAP)	Stochastic, Exploding	·	modified EOT
Mitigating Through Randomization	Stochastic	·	EOT
Pixel Defend	Vanishing	·	BPDA
Defense-GAN	Vanishing	Unbounded attack do not reach 100%	BPDA

*source: Athalye et al., Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples, ICML 2018

Obfuscated Gradients: False Sense of Security [Athalye et al., 2018]

- **Athalye et al. (ICML 2019)**: In fact, most of them are "fake" defenses
  - **"Fake" defense?**: They don't aim the non-existence of adversarial example
    - Rather, they aim to obfuscate the gradient information
    - Obfuscated gradient makes gradient-based attacks (FGSM, PGD, …) harder

- Then… what should we do?
  - At least, we have to do sanity checks on evaluating defenses
  - Do your best to show that the proposed defense is a "real" defense

  - **Some "red-flags" indicating obfuscated gradients**

> ① One-step attacks perform better than iterative attacks
>
> ② Black-box attacks are better than white-box attacks
>
> ③ Unbounded attacks do not reach 100% success
>
> ④ Random sampling finds adversarial examples better

*source: Athalye et al., Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples, ICML 2018

- **AutoAttack is the state-of-the art attack method**
  - AutoAttack uses four advanced attacks, and check whether any attack succeeds
  - Two white-box attacks: APGD-untargeted, APGD-targeted [Croce et al., 2020]
  - Two black-box attacks: FAB [Croce et al., 2020], Square [Andriushchenko et al., 2020]

- **Experimental Results**
  - AutoAttack largely reduced the obfuscated gradient issues in prior evaluations

#	paper	clean	APGD_{CE}	$\text{APGD}^{\text{T}}_{\text{DLR}}$	FAB^{T}	Square	AA	reported	reduct.
CIFAR-10 - l_∞ - $\epsilon = 8/255$									
1	(Carmon et al., 2019)	89.69	61.74	**59.54**	60.12	66.63	59.53	62.5	-2.97
2	(Alayrac et al., 2019)	86.46	60.17	**56.27**	56.81	66.37	56.03	56.30	-0.27
3	(Hendrycks et al., 2019)	87.11	57.23	**54.94**	55.27	61.99	54.92	57.4	-2.48
4	(Rice et al., 2020)	85.34	57.00	**53.43**	53.83	61.37	53.42	58	-4.58
5	(Qin et al., 2019)	86.28	55.70	52.85	53.28	60.01	52.84	52.81	0.03
6	(Engstrom et al., 2019)	87.03	51.72	**49.32**	49.81	58.12	49.25	53.29	-4.04
7	(Kumari et al., 2019)	87.80	51.80	**49.15**	49.54	58.20	49.12	53.04	-3.92
8	(Mao et al., 2019)	86.21	49.65	**47.44**	47.91	56.98	47.41	50.03	-2.62
9	(Zhang et al., 2019a)	87.20	46.15	**44.85**	45.39	55.08	44.83	47.98	-3.15
10	(Madry et al., 2018)	87.14	44.75	**44.28**	44.75	53.10	44.04	47.04	-3.00
11	(Pang et al., 2020)	80.89	57.07	**43.50**	44.06	49.73	43.48	55.0	-11.52
12	(Wong et al., 2020)	83.34	45.90	**43.22**	43.74	53.32	43.21	46.06	-2.85
13	(Shafahi et al., 2019)	86.11	43.66	**41.64**	43.44	51.95	41.47	46.19	-4.72
14	(Ding et al., 2020)	84.36	50.12	**41.74**	42.47	55.53	41.44	47.18	-5.74
15	(Moosavi-Dezfooli et al., 2019)	83.11	41.72	**38.50**	38.97	47.69	38.50	41.4	-2.90
16	(Zhang & Wang, 2019)	89.98	64.42	**37.29**	38.48	59.12	36.64	60.6	-23.96
17	(Zhang & Xu, 2020)	90.25	71.40	**37.54**	38.99	66.88	36.45	68.7	-32.25

*source: Croce et al., Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-free Attacks, ICML 2020

Table of Contents

Distributional Shifts

- Various distributional shift scenarios have been proposed
- We will introduce some scenarios that are widely considered in recent works

- Various distributional shift scenarios have been proposed
- We will introduce some scenarios that are widely considered in recent works

- **Shape and texture bias** [Geirhos et al., 2019]
  - Suggest benchmarks to measure whether the model is biased to textures or shapes
  - Moreover, found that the ImageNet-trained models are biased to textures

Example of **Stylized-ImageNet (SIN)**: only change the style (i.e., the texture) of the given input

* source: Geirhos et al., ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness, ICLR 2019

- Various distributional shift scenarios have been proposed
- We will introduce some scenarios that are widely considered in recent works

- **Shape and texture bias** [Geirhos et al., 2019]
  - Suggest benchmarks to measure whether the model is biased to textures or shapes
  - Moreover, found that the ImageNet-trained models are biased to textures

(a) Texture image

81.4%	**Indian elephant**
10.3%	indri
8.2%	black swan

(b) Content image

71.1%	**tabby cat**
17.3%	grey fox
3.3%	Siamese cat

(c) Texture-shape cue conflict

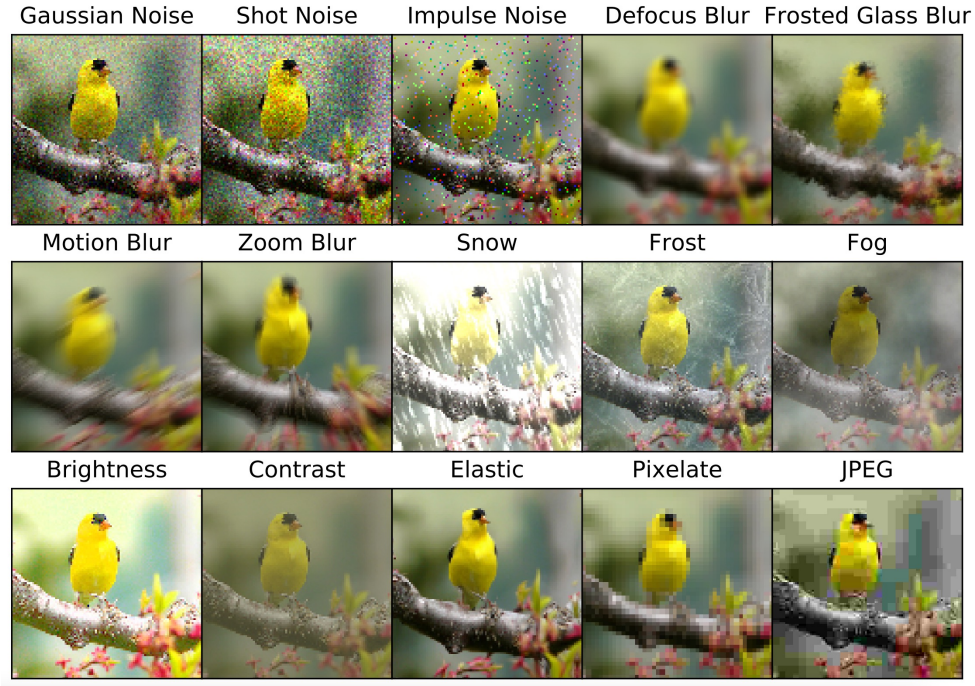
63.9%	**Indian elephant**
26.4%	indri
9.6%	black swan

* source: Geirhos et al., ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness, ICLR 2019

Distributional Shifts

- Various distributional shift scenarios have been proposed
- We will introduce some scenarios that are widely considered in recent works

- **Common corruption** [Hendrycks et al., 2019]
  - Suggest 15 types of corruptions that highly degrade the classifier's performance

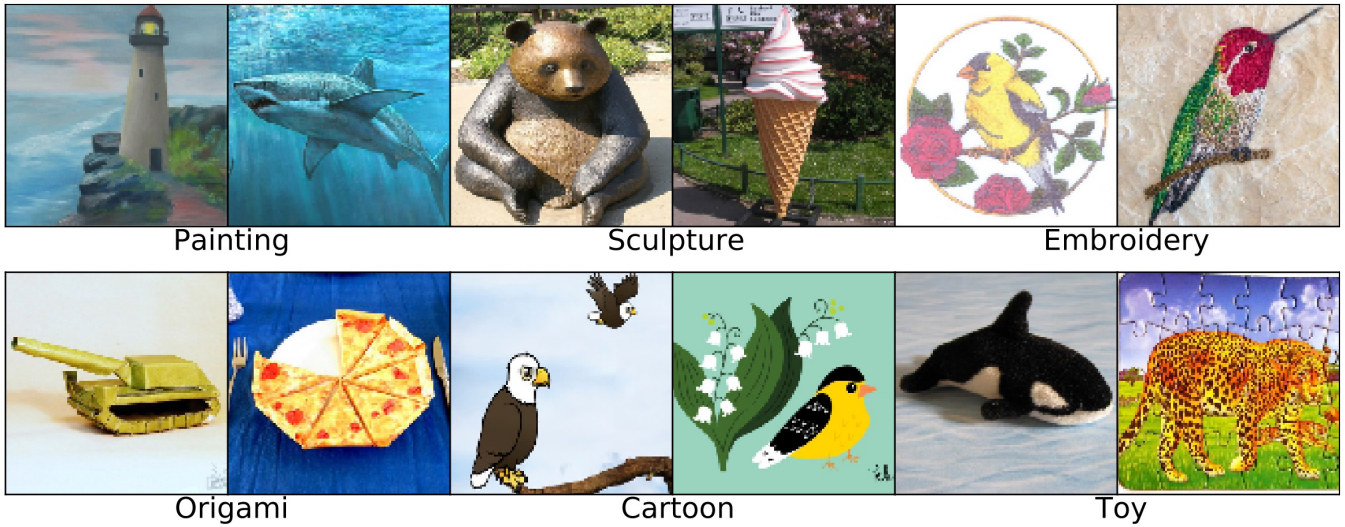


*source: Hendrcyks et al., Benchmarking Neural Network Robustness to Common Corruptions and Perturbations, ICLR 2019

Distributional Shifts

- Various distributional shift scenarios have been proposed
- We will introduce some scenarios that are widely considered in recent works

- **Domain shifts** [Hendrycks et al., 2019]
  - Suggest 16 types of renditions of ImageNet classes, i.e., domain shift

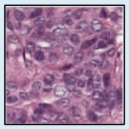
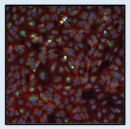
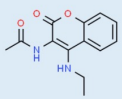
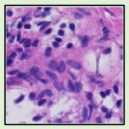
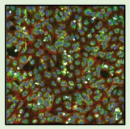
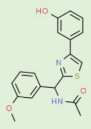


*source: Hendrcyks et al., The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization, ICCV 2021

Distributional Shifts

- Various distributional shift scenarios have been proposed
- We will introduce some scenarios that are widely considered in recent works

- **WILDS** [Koh et al., 2019]
  - WILDS consists of various real-world distributional shift scenarios
  - E.g., distributional shifts on the medical imaging and natural language processing

	Domain generalization					Subpopulation shift	Domain generalization + subpopulation shift			
Dataset	iWildCam	Camelyon17	RxRx1	OGB-MolPCBA	GlobalWheat	CivilComments	FMoW	PovertyMap	Amazon	Py150
Input (x)	camera trap photo	tissue slide	cell image	molecular graph	wheat image	online comment	satellite image	satellite image	product review	code
Prediction (y)	animal species	tumor	perturbed gene	bioassays	wheat head bbox	toxicity	land use	asset wealth	sentiment	autocomplete
Domain (d)	camera	hospital	batch	scaffold	location, time	demographic	time, region	country, rural-urban	user	git repository
# domains	323	5	51	120,084	47	16	16 x 5	23 x 2	2,586	8,421
# examples	203,029	455,954	125,510	437,929	6,515	448,000	523,846	19,669	539,502	150,000
Train example						What do Black and LGBT people have to do with bicycle licensing?			Overall a solid package that has a good quality of construction for the price.	`import numpy as np … norm=np.___`
Test example						As a Christian, I will not be patronizing any of those businesses.			I *loved* my French press, it's so perfect and came with all this fun stuff!	`import subprocess as sp p=sp.Popen() stdout=p.___`
Adapted from	Beery et al. 2020	Bandi et al. 2018	Taylor et al. 2019	Hu et al. 2020	David et al. 2021	Borkan et al. 2019	Christie et al. 2018	Yeh et al. 2020	Ni et al. 2019	Raychev et al. 2016

*source: Koh et al., WILDS: A Benchmark of in-the-Wild Distribution Shifts, ICML 2021

Table of Contents

Table of Contents

Adversarial Training [Madry et al., 2018]

- **Motivation:** An optimization view on attacks and defenses
  - **Recall:** Adversarial attacks aim to find inputs so that:

$$\max_{x':d(x,x')<\epsilon} L(f(x'), y)$$

  - In the viewpoint of defense, our goal is to minimize the **adversarial risk:**

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{x':d(x,x')<\epsilon} L(f(x'), y)\right]$$

- Adversarial training (AT) aims to minimize adversarial risk in training

$$\min_{\theta}\left(\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{x':d(x,x')<\epsilon} L(f(x'), y; \theta)\right]\right)$$

Training parameters

FGSM, PGD, …

  - **Challenge**: Computing the inner-maximization is difficult
  - **Idea**: Use strong attack methods to approximate the inner-maximization
    - e.g. FGSM, PGD, DeepFool, …

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

- **Experimental results**
  - MNIST results

Method	Steps	Restarts	Source	Accuracy
Natural	-	-	-	98.8%
FGSM	-	-	A	95.6%
PGD	40	1	A	93.2%
PGD	100	1	A	91.8%
PGD	40	20	A	90.4%
PGD	100	20	A	**89.3%**
Targeted	40	1	A	92.7%
CW	40	1	A	94.0%
CW+	40	1	A	93.9%

White-box

Method	Steps	Restarts	Source	Accuracy
FGSM	-	-	A'	96.8%
PGD	40	1	A'	96.0%
PGD	100	20	A'	**95.7%**
CW	40	1	A'	97.0%
CW+	40	1	A'	96.4%
FGSM	-	-	B	**95.4%**
PGD	40	1	B	96.4%
CW+	-	-	B	95.7%

Black-box

  - CIFAR10 results

Method	Steps	Source	Accuracy
Natural	-	-	87.3%
FGSM	-	A	56.1%
PGD	7	A	50.0%
PGD	20	A	**45.8%**
CW	30	A	46.8%

White-box

Method	Steps	Source	Accuracy
FGSM	-	A'	67.0%
PGD	7	A'	**64.2%**
CW	30	A'	78.7%
FGSM	-	A_{nat}	85.6%
PGD	7	A_{nat}	86.0%

Black-box

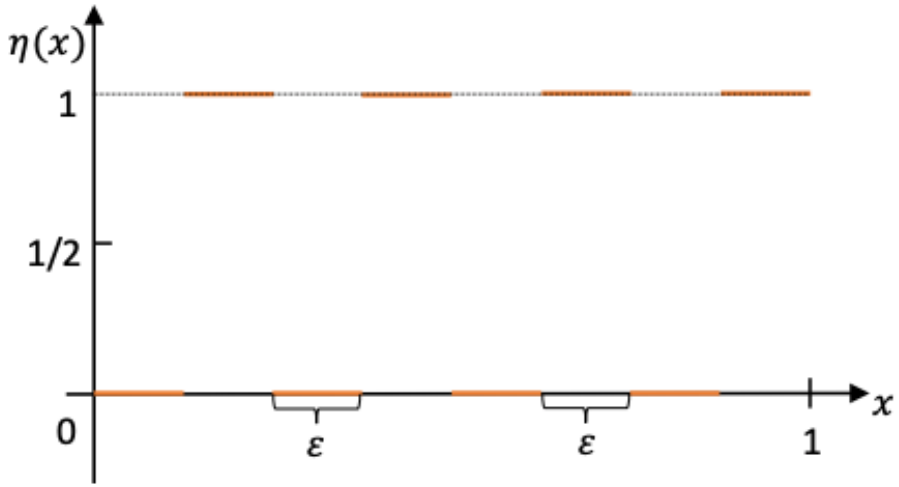
*source: Madry et al., Towards Deep Learning Models Resistant to Adversarial Attacks, ICLR 2018

TRADES: Tradeoff Between Accuracy and Robustness [Zhang et al., 2019]

- **Motivation**: Robust model → accuracy reduction? [Tsipras et al. ,2019]

- Consider (X, Y) modeled by $\eta(x)$
  - Bayes optimal classifier: $\text{sign}(2\eta(x) - 1)$
- We are using an "accuracy-biased" loss function

- Can we exploit this **trade-off** for better robustness?



$$\eta(x) := \Pr(Y = 1 | X = x)$$
$$= \begin{cases} 0, & x \in [2k\epsilon, (2k+1)\epsilon), \\ 1, & x \in ((2k+1)\epsilon, (2k+2)\epsilon]. \end{cases}$$

	Bayes Optimal Classifier	All-One Classifier
$\mathcal{R}_{\text{nat}}$	0 (optimal)	1/2
$\mathcal{R}_{\text{bdy}}$	1	0
$\mathcal{R}_{\text{rob}}$	1	1/2 (optimal)

*source: Zhang et al., Theoretically Principled Trade-off between Robustness and Accuracy, ICML 2018

TRADES: Tradeoff Between Accuracy and Robustness [Zhang et al., 2019]

- We re-write the relationship between **robust error** and **natural error**
- Consider a binary classification with $Y \in \{-1, 1\}$
  - **Natural error**: $\mathcal{R}_{\text{nat}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}[f(\boldsymbol{X})Y \leq 0]$
  - **Robust error** under ϵ-perturbation:
    - [Schmidt et al., 2018; Cullina et al., 2018; Bubeck et al., 2018]

$$\mathcal{R}_{\text{rob}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}[\exists \boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon) \text{ s.t. } f(\boldsymbol{X}')Y \leq 0]$$

*source: Zhang et al., Theoretically Principled Trade-off between Robustness and Accuracy, ICML 2018

TRADES: Tradeoff Between Accuracy and Robustness [Zhang et al., 2019]

- We re-write the relationship between **robust error** and **natural error**
- Consider a binary classification with $Y \in \{-1, 1\}$
  - **Natural error**: $\mathcal{R}_{\text{nat}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}[f(\boldsymbol{X})Y \leq 0]$
  - **Robust error** under ϵ-perturbation:
    - [Schmidt et al., 2018; Cullina et al., 2018; Bubeck et al., 2018]

$$\mathcal{R}_{\text{rob}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}[\exists \boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon) \text{ s.t. } f(\boldsymbol{X}')Y \leq 0]$$

- Zhang et al. (2019) also defines the **boundary error:**

$$\mathcal{R}_{\text{bdy}}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\mathbf{1}[\boldsymbol{X} \in \mathbb{B}(\text{DB}(f), \epsilon), f(\boldsymbol{X})Y > 0]$$

Decision boundary

- $\mathbb{B}(\text{DB}(f), \epsilon) := \{\boldsymbol{x} : \exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } f(\boldsymbol{x})f(\boldsymbol{x}') \leq 0\}$

- **Boundary error** identifies the gap between $\mathcal{R}_{\text{rob}}(f)$ and $\mathcal{R}_{\text{nat}}(f)$

$$\mathcal{R}_{\text{rob}}(f) = \mathcal{R}_{\text{nat}}(f) + \mathcal{R}_{\text{bdy}}(f)$$

*source: Zhang et al., Theoretically Principled Trade-off between Robustness and Accuracy, ICML 2018

TRADES: Tradeoff Between Accuracy and Robustness [Zhang et al., 2019]

- **Goal**: Find $\hat{f}$ such that $\mathcal{R}_{\text{rob}}(\hat{f}) - \mathcal{R}^*_{\text{nat}}$ is small

$$\mathcal{R}_{\text{rob}}(\hat{f}) - \mathcal{R}^*_{\text{nat}} = \underbrace{(\mathcal{R}_{\text{nat}}(\hat{f}) - \mathcal{R}^*_{\text{nat}})}_{\text{Natural error gap}} + \mathcal{R}_{\text{bdy}}(\hat{f}) \leq \delta$$

- **Theorem 1 (upper bound, informal).** Let ϕ be a usual surrogate loss. We have:

$$\begin{aligned}
\mathcal{R}_{\text{rob}}(f) - \mathcal{R}^*_{\text{nat}} &= (\mathcal{R}_{\text{nat}}(f) - \mathcal{R}^*_{\text{nat}}) + \mathcal{R}_{\text{bdy}}(f) \\
&\leq (\mathcal{R}_\phi(f) - \mathcal{R}^*_\phi) + \mathcal{R}_{\text{bdy}(f)} \quad \textbf{(Bartlett et al., 2006)} \\
&\leq (\mathcal{R}_\phi(f) - \mathcal{R}^*_\phi) + \mathbb{E}\left[\max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon)} \phi(f(\boldsymbol{X}')f(\boldsymbol{X})/\underline{\lambda})\right]
\end{aligned}$$

- **Theorem 2 (lower bound, informal).** for any $\xi > 0$, there exist $\mathcal{D}$, f, and $\lambda > 0$ such that:

$$\mathcal{R}_{\text{rob}}(f) - \mathcal{R}^*_{\text{nat}} \geq (\mathcal{R}_\phi(f) - \mathcal{R}^*_\phi) + \mathbb{E}\left[\max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon)} \phi(f(\boldsymbol{X}')f(\boldsymbol{X})/\lambda)\right] - \xi$$

  - The upper bound is tight if there is no assumption on $\mathcal{D}$

*source: Zhang et al., Theoretically Principled Trade-off between Robustness and Accuracy, ICML 2018

TRADES: Tradeoff Between Accuracy and Robustness [Zhang et al., 2019]

- **Goal**: Find $\hat{f}$ such that $\mathcal{R}_{\text{rob}}(\hat{f}) - \mathcal{R}^*_{\text{nat}}$ is small

$$\mathcal{R}_{\text{rob}}(\hat{f}) - \mathcal{R}^*_{\text{nat}} = (\mathcal{R}_{\text{nat}}(\hat{f}) - \mathcal{R}^*_{\text{nat}}) + \mathcal{R}_{\text{bdy}}(\hat{f}) \leq \delta$$

- The theorems naturally suggests **a new surrogate loss**:

$$\min_f \mathbb{E}\left[\underbrace{\mathcal{L}(f(\boldsymbol{X}), \boldsymbol{Y})}_{\text{accuracy}} + \underbrace{\max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon)} \mathcal{L}(f(\boldsymbol{X}), f(\boldsymbol{X}'))/\lambda}_{\text{robustness}}\right]$$

- **TRADES**: **TR**adeoff-inspired **A**dv. **DE**fense via **S**urrogate-loss minimization
  - λ: The **balancing** hyper-parameter
    - We can boost the robust accuracy with little loss of natural accuracy
- **Key difference**: TRADES finds $\boldsymbol{X}'$ by solving $\max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon)} L(f(\boldsymbol{X}), f(\boldsymbol{X}'))/\lambda$
  - Adversarial training [Madry et al., 2018]: $\max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon)} L(f(\boldsymbol{X}'), Y)$

- Up to now, **TRADES** is regarded as the **state-of-the-art** defense method

*source: Zhang et al., Theoretically Principled Trade-off between Robustness and Accuracy, ICML 2018

- **Experimental results**
  - White-box attack results (CIFAR-10 & MNIST)

Table 5: Comparisons of TRADES with prior defense models under white-box attacks.

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%
[SKC18]	gradient mask	[ACW18]	MNIST	0.005 (ℓ_2)	-	55%
[MMS+18]	robust opt.	FGSM^{40} (PGD)	MNIST	0.3 (ℓ_∞)	99.36%	96.01%
TRADES ($1/\lambda = 6$)	regularization	FGSM^{1,000} (PGD)	MNIST	0.3 (ℓ_∞)	99.48%	95.60%
TRADES ($1/\lambda = 6$)	regularization	FGSM^{40} (PGD)	MNIST	0.3 (ℓ_∞)	99.48%	96.07%
TRADES ($1/\lambda = 6$)	regularization	C&W	MNIST	0.005 (ℓ_2)	99.48%	99.46%

*source: Zhang et al., Theoretically Principled Trade-off between Robustness and Accuracy, ICML 2018

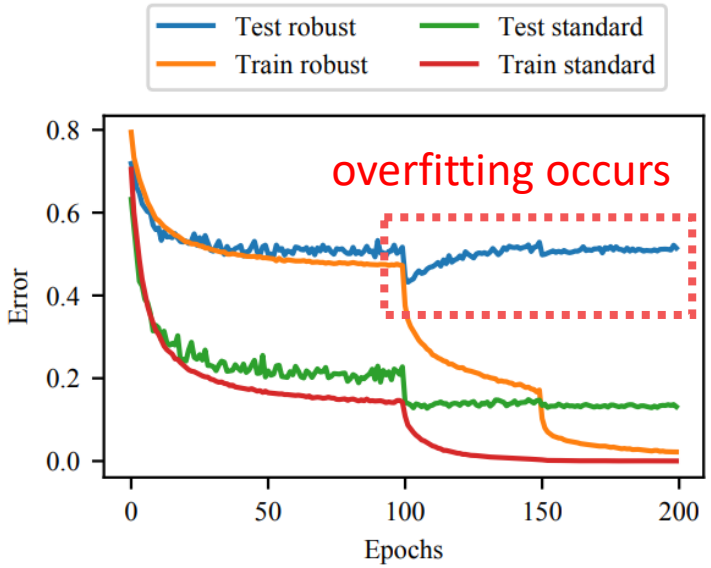
Table of Contents

Overfitting of Adversarial Training [Rice et al., 2020]

- Problem of AT: **Robust overfitting**
  - The robust error of test set, gradually increases from the middle of the training
  - This phenomenon occurs across dataset, architectures and objectives (e.g., TRADES)

- **Early stopping** is an effective way to prevent the overfitting

- Most of recent advanced AT methods aims to resolve this issue



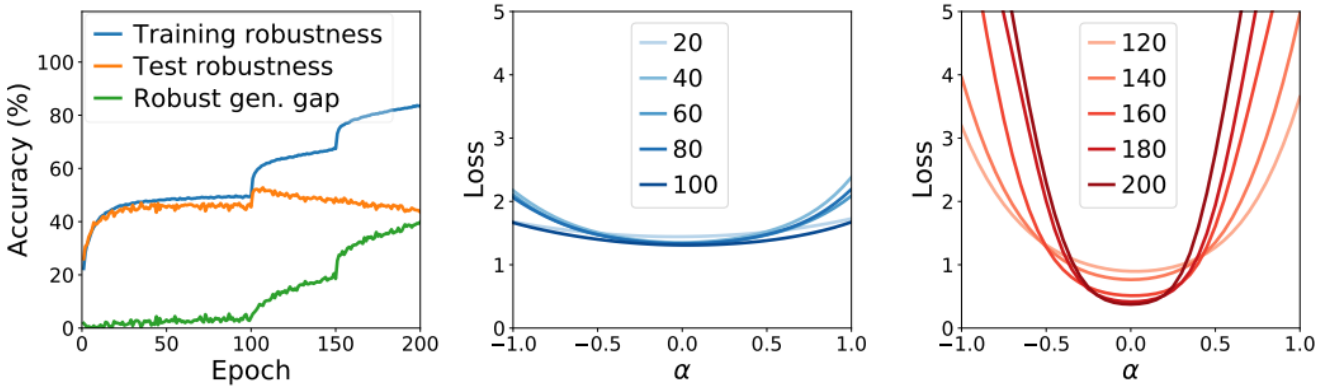
Comparison of methods for preventing overfitting

Reg Method	Robust Test Error (%) Final	Best	Diff
Early stopping w/ val	**46.9**	46.7	0.2
ℓ_1 regularization	53.0 ± 0.39	48.6	4.4
ℓ_2 regularization	55.2 ± 0.4	46.4	55.2
Cutout	48.8 ± 0.79	46.7	2.1
Mixup	49.1 ± 1.32	46.3	2.8
Semi-supervised	47.1 ± 4.32	40.2	6.9

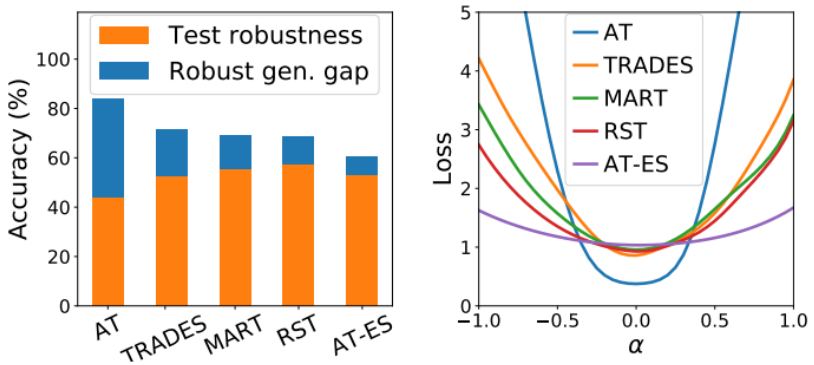
*source: Rice et al., Overfitting in adversarially robust deep learning, ICML 2020

Adversarial Weight Perturbation [Wu et al., 2020]

- **Observation**: model with the high robustness have a smooth loss landscape
- We measure the loss landscape of the adversarial risk
  - (1) During AT, the best model (at 100 epoch) has the most smooth landscape



  - (2) The AT objectives with strong robustness tend to have a smoother landscape



*source: Wu et al, Adversarial Weight Perturbation Helps Robust Generalization, NeurIPS 2020

- **Adversarial Weight Perturbation (AWP)**
  - Optimize the loss on the worst case weight parameter to force the smoothness

$$\min_{\mathbf{w}} \max_{\mathbf{v}\in\mathcal{V}} \rho(\mathbf{w}+\mathbf{v}) \rightarrow \min_{\mathbf{w}} \max_{\mathbf{v}\in\mathcal{V}} \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i'-\mathbf{x}_i\|_p\leq\epsilon} \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i),$$

Maximize the **weight perturbation**

Maximize the input perturbation, i.e., adversarial training

  - In detail, AWP use a projected gradient decent to attack the weight parameters

$$\mathbf{v} \leftarrow \Pi_\gamma\left(\mathbf{v} + \eta_2 \frac{\nabla_{\mathbf{v}} \frac{1}{m}\sum_{i=1}^{m} \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)}{\|\nabla_{\mathbf{v}} \frac{1}{m}\sum_{i=1}^{m} \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)\|}\|\mathbf{w}\|\right),$$

*source: Wu et al, Adversarial Weight Perturbation Helps Robust Generalization, NeurIPS 2020

Adversarial Weight Perturbation [Wu et al., 2020]

- Experimental results
  - AWP effectively **prevents the overfitting** issues of AT

Threat Model	Method	SVHN		CIFAR-10		CIFAR-100	
		Best	Last	Best	Last	Best	Last
L_∞	AT	53.36	44.49	52.79	44.44	27.22	20.82
	AT-AWP	**59.12**	**55.87**	**55.39**	**54.73**	**30.71**	**30.28**
L_2	AT	66.87	65.03	69.15	65.93	41.33	35.27
	AT-AWP	**72.57**	**67.73**	**72.69**	**72.08**	**45.60**	**44.66**

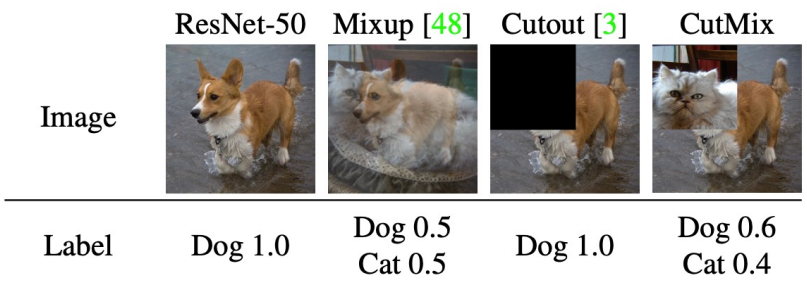
  - Moreover, AWP achieves the state-of-the-art robustness on various benchmarks

Defense	Natural	FGSM	PGD-20	PGD-100	CW_∞	SPSA	AA
AT	**86.07**	61.76	56.10	55.79	54.19	61.40	52.60 ¶
AT-AWP	85.57	**62.90**	**58.14**	**57.94**	**55.96**	**62.65**	**54.04**
TRADES	84.65	61.32	56.33	56.07	54.20	61.10	53.08
TRADES-AWP	**85.36**	**63.49**	**59.27**	**59.12**	**57.07**	**63.85**	**56.17**
MART	84.17	61.61	58.56	57.88	54.58	58.90	51.10
MART-AWP	**84.43**	**63.98**	**60.68**	**59.32**	**56.37**	**62.75**	**54.23**
Pre-training	87.89	63.27	57.37	56.80	55.95	62.55	54.92
Pre-training-AWP	**88.33**	**66.34**	**61.40**	**61.21**	**59.28**	**65.55**	**57.39**
RST	**89.69**	**69.60**	62.60	62.22	60.47	67.60	59.53
RST-AWP	88.25	67.94	**63.73**	**63.58**	**61.62**	**68.72**	**60.05**

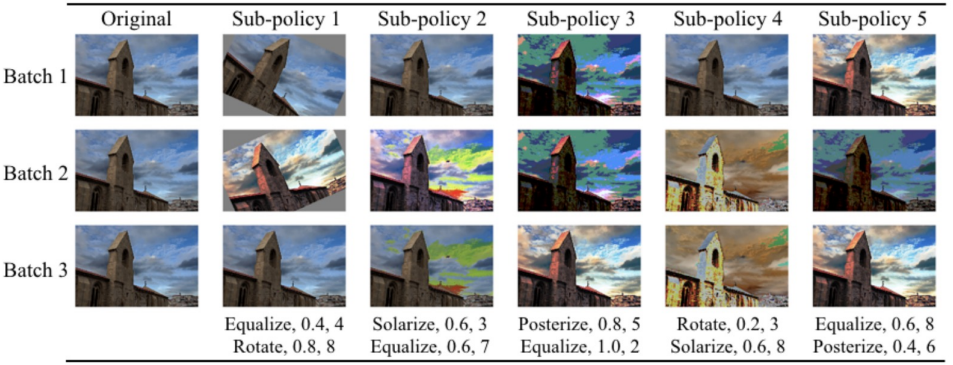
*source: Wu et al, Adversarial Weight Perturbation Helps Robust Generalization, NeurIPS 2020

Data Augmentation Can Improve Robustness [Rebuffit et al., 2021]

- Recently, **data augmentations** are used in AT to improve the robustness
  - Several works research the effect of **conventional augmentations** in AT

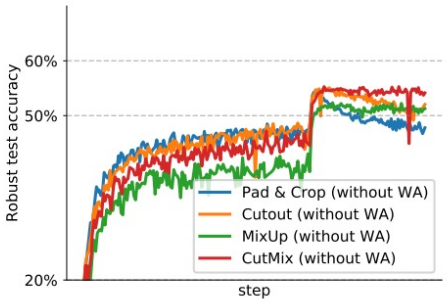


CutMix [Yun et al., 2019]

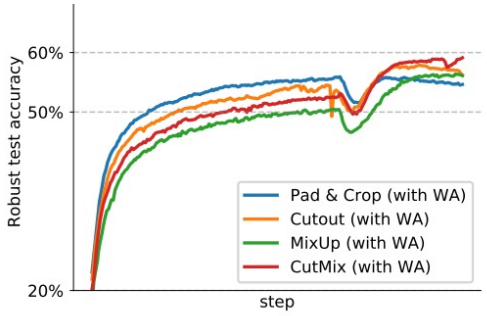


AutoAugment [Cubuk et al., 2019]

- For instance, **CutMix** can prevent the robust overfitting
  - Additionally using **weight averaging (WA)** can further improve the robustness



(a) Without WA



(b) With WA

*source:

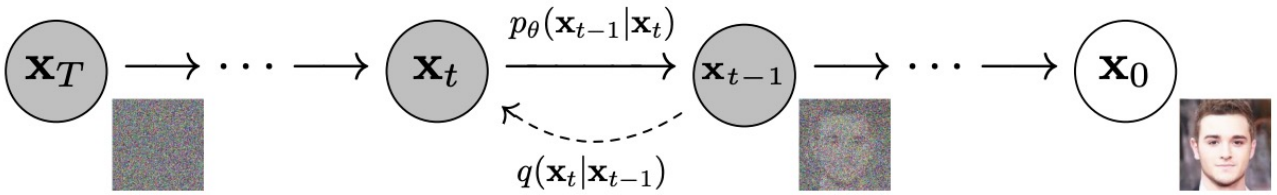
Yun et al., CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features, ICCV 2019

Cubuk et al., AutoAugment: Learning Augmentation Strategies from Data, CVPR 2020

Rebuffi et al, Data Augmentation Can Improve Robustness, NeurIPS 2021

Improving Robustness using Generated Data [Gowal et al., 2021]

- Recently, **data augmentations** are used in AT to improve the robustness
  - Other works find that using generating more training data, is highly effective in AT



DDPM [Ho et al., 2019]

- Using the generated data (from **DDPM**) for AT, improves the robustness
  - As DDPM is unconditional generative model, one should use pseudo-labels from the pre-trained classifier (not adversarially trained classifier)
  - DDPM augmentation achieves **the state-of-the art performance**

MODEL	DATASET	NORM	CLEAN	ROBUST
Wu et al. [76] (WRN-34-10)	CIFAR-10	ℓ_∞	85.36%	56.17%
Gowal et al. [30] (WRN-70-16)			85.29%	57.14%
Ours (DDPM) (WRN-28-10)			85.97%	60.73%
Ours (DDPM) (WRN-70-16)			86.94%	63.58%
Ours (100M DDPM)* (ResNet-18)			87.35%	58.50%
Ours (100M DDPM)* (WRN-28-10)			87.50%	63.38%
Ours (100M DDPM)* (WRN-70-16)			**88.74%**	**66.10%**

*source:
Ho et al, Denoising Diffusion Probabilistic Models, NeurIPS 2020
Gowal et al., Improving Robustness using Generated Data, NeurIPS 2021

Table of Contents

Back Side of Adversarial Training

- **Adversarial training** has achieved reasonable adversarial robustness
  - Adversarially-trained networks are resistant for the corresponding **threat model**

Back Side of Adversarial Training

- **Adversarial training** has achieved reasonable adversarial robustness
  - Adversarially-trained networks are resistant for the corresponding **threat model**
- **Problem**: The conjectured robustness may be broken under a stronger "adaptive" attack mechanism
  - In other words, **we can not guarantee the robustness under unseen threat model**

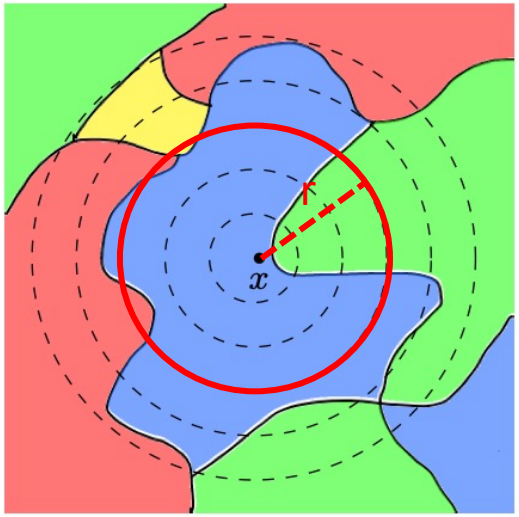
Back Side of Adversarial Training

- **Adversarial training** has achieved reasonable adversarial robustness
  - Adversarially-trained networks are resistant for the corresponding **threat model**
- **Problem**: The conjectured robustness may be broken under a stronger "adaptive" attack mechanism
  - In other words, **we can not guarantee the robustness under unseen threat model**
- The most that can be said is that **a specific attack was unable to find**

Typical approaches for certified robustness

- For a classifier f, we want to **'certify' the robustness** for the given input x
  - For a radius r, is there a perturbation δ with $||\delta|| \leq r$ where $f(x) \neq f(x+\delta)$?

- *Exact* certification
  - Report **whether or not there exists such perturbation δ**
  - Satisfiability modulo theories [Katz et al., 2017]
  - Mixed integer linear programming [Cheng et al., 2017]
  - Can't be scaled **beyond moderate-sized networks**

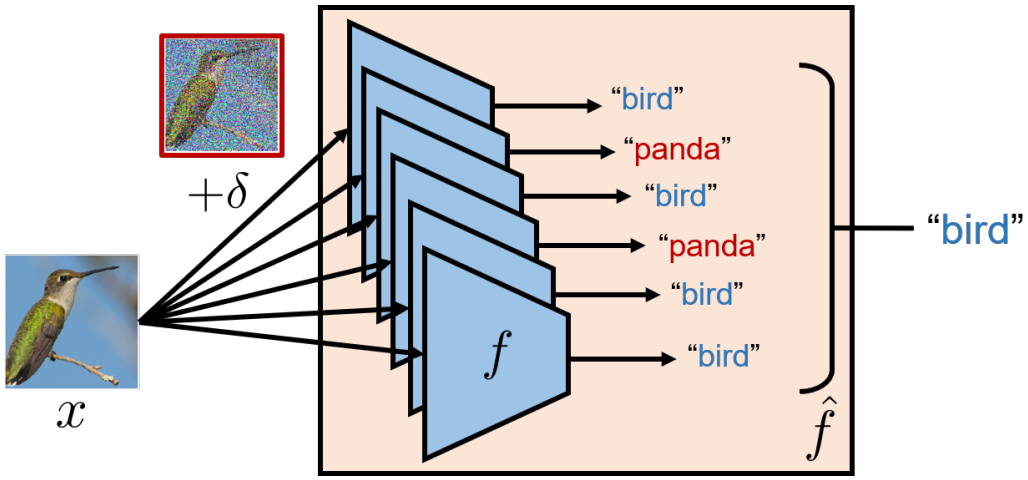


- *Conservative* certification
  - Certify that there is no such perturbation $\boldsymbol{\delta}$ or abstain
  - Bound the global Lipschitz constants [Gouk et al., 2018]
  - Measure the local smoothness [Hein et al., 2017]
  - Assume **specific network architectures** (e.g., ReLU activations)

- Both approaches have some **limits to be applied to modern architectures**

- Problem: Directly training **a robust DNN f is challenging**
  - **Randomized smoothing** instead constructs **another classifier g from f**
  - The **l_2-norm robust radius** of g is lower-bounded in terms of f
  - Note: f does not have to be perfectly smooth for adversarial robustness

$$g(x) := \text{argmax}_{c \in \mathcal{Y}} \left\{ \mathbb{P}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}(f(x+\epsilon) = c) \right\}$$



- Problem: Directly training **a robust DNN f is challenging**
  - **Randomized smoothing** instead constructs **another classifier g from f**
  - The **l_2-norm robust radius** of g is lower-bounded in terms of f
  - Note: f does not have to be perfectly smooth for adversarial robustness

$$g(x) := \text{argmax}_{c \in \mathcal{Y}} \left\{ \mathbb{P}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}(f(x+\epsilon) = c) \right\}$$

Theorem 1. *Let $f : \mathbb{R}^d \to \mathcal{Y}$ be any deterministic or random function, and let $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. Let g be defined as in (1). Suppose $c_A \in \mathcal{Y}$ and $\underline{p_A}, \overline{p_B} \in [0,1]$ satisfy:*

$$\mathbb{P}(f(x+\varepsilon) = c_A) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c \neq c_A} \mathbb{P}(f(x+\varepsilon) = c) \quad (2)$$

Then $g(x+\delta) = c_A$ for all $\|\delta\|_2 < R$, where

$$R = \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) \quad (3)$$

The certified radius R is large when:

- The noise level σ is high
- The probability of the top class c_A is high
- The probabiliy of each other class is low

- Problem: Directly training **a robust DNN f is challenging**
  - **Randomized smoothing** instead constructs **another classifier g from f**
  - The **l_2-norm robust radius** of g is lower-bounded in terms of f
  - Note: f does not have to be perfectly smooth for adversarial robustness

$$g(x) := \text{argmax}_{c \in \mathcal{Y}} \left\{ \mathbb{P}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}(f(x+\epsilon) = c) \right\}$$

Theorem 1. *Let $f : \mathbb{R}^d \to \mathcal{Y}$ be any deterministic or random function, and let $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. Let g be defined as in (1). Suppose $c_A \in \mathcal{Y}$ and $\underline{p_A}, \overline{p_B} \in [0,1]$ satisfy:*

$$\mathbb{P}(f(x+\varepsilon) = c_A) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c \neq c_A} \mathbb{P}(f(x+\varepsilon) = c) \quad (2)$$

Then $g(x+\delta) = c_A$ for all $\|\delta\|_2 < R$, where

$$R = \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) \quad (3)$$

Theorem 2. *Assume $\underline{p_A} + \overline{p_B} \leq 1$. For any perturbation δ with $\|\delta\|_2 > R$, there exists a base classifier f consistent with the class probabilities (2) for which $g(x+\delta) \neq c_A$.*

- If (2) is all that is known about f, then the **robustness guarantee is *tight***
- In other words, it is **impossible to certify an l_2 ball with radius larger than R**

- Problem: Directly training **a robust DNN f is challenging**
  - **Randomized smoothing** instead constructs **another classifier g from f**
  - The **l_2-norm robust radius** of g is lower-bounded in terms of f
  - Note: f does not have to be perfectly smooth for adversarial robustness

$$g(x) := \text{argmax}_{c \in \mathcal{Y}} \left\{ \mathbb{P}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}(f(x+\epsilon) = c) \right\}$$

  - Intractability of "averaging over Gaussian noise" -> Monte Carlo & hypothesis test

Theorem 1. *Let $f : \mathbb{R}^d \to \mathcal{Y}$ be any deterministic or random function, and let $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. Let g be defined as in (1). Suppose $c_A \in \mathcal{Y}$ and $\underline{p_A}, \overline{p_B} \in [0,1]$ satisfy:*

$$\mathbb{P}(f(x+\varepsilon) = c_A) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c \neq c_A} \mathbb{P}(f(x+\varepsilon) = c) \quad (2)$$

Then $g(x+\delta) = c_A$ for all $\|\delta\|_2 < R$, where

$$R = \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) \quad (3)$$

- Problem: Directly training **a robust DNN f is challenging**
  - **Randomized smoothing** instead constructs **another classifier g from f**
  - The **l_2-norm robust radius** of g is lower-bounded in terms of f
  - Note: f does not have to be perfectly smooth for adversarial robustness

$$g(x) := \text{argmax}_{c \in \mathcal{Y}} \left\{ \mathbb{P}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}(f(x+\epsilon) = c) \right\}$$

  - Intractability of "averaging over Gaussian noise" -> Monte Carlo & hypothesis test

 # certify the robustness of g around x
function CERTIFY($f, \sigma, x, n_0, n, \alpha$)
`counts0` $\leftarrow$ SAMPLEUNDERNOISE(f, x, n_0, σ)
$\hat{c}_A \leftarrow$ top index in `counts0`
`counts` $\leftarrow$ SAMPLEUNDERNOISE(f, x, n, σ)
$\underline{p_A} \leftarrow$ LOWERCONFBOUND(`counts`$[\hat{c}_A], n, 1-\alpha$)
if $\underline{p_A} > \frac{1}{2}$ **return** prediction $\hat{c}_A$ and radius $\sigma\, \Phi^{-1}(\underline{p_A})$
else return ABSTAIN

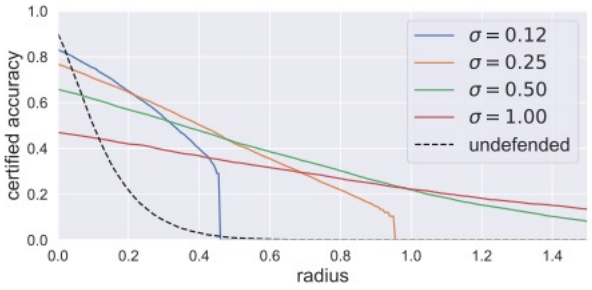
Proposition 2. *With probability at least $1-\alpha$ over the randomness in* CERTIFY, *if* CERTIFY *returns a class $\hat{c}_A$ and a radius R (i.e. does not abstain), then g predicts $\hat{c}_A$ within radius R around x: $g(x+\delta) = \hat{c}_A \;\; \forall \, \|\delta\|_2 < R$.*

- Experimental results:
  - The base classifier f is trained with **cross-entropy loss of Gaussian augmentations**

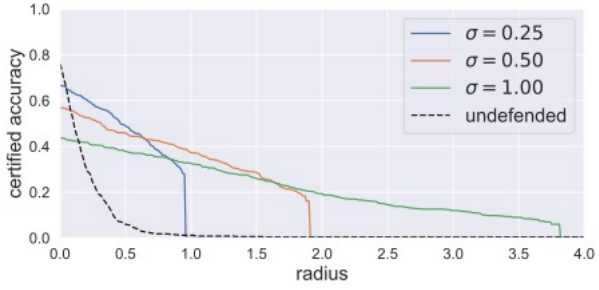
$$\mathcal{L}^{nat} := \mathbb{E}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}[\mathcal{L}(F(x+\epsilon), y)]$$

Softmax outputs of f

  - The choice of σ determines **the trade-off between the accuracy and robustness**



Results on CIFAR-10



Results on ImageNet

  - Q. Can we train f more effectively to yield **more robust g**?

- Idea: **Adversarial training of the smoothed classifier g** (approx.)
  - How to **attack the smoothed classifier g** through the base classifier f?
  - Recall: g is the most provable output of f under Gaussian augmentation

$$g(x) \coloneqq \text{argmax}_{c \in \mathcal{Y}} \left\{ \mathbb{P}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}(f(x+\epsilon) = c) \right\}$$

  - Define **"soft" smoothed classifier G** (corresponding to g) as follows:

$$G(x) \coloneqq \big(F * \mathcal{N}(0,\sigma^2 I)\big)(x) = \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)}[F(x+\delta)]$$

Softmax outputs of f

  - **Adversarial attack** is performed on **the soft smoothed classifier G**
  - Note: Attack on g does not yield a valid gradient because of argmax function

- Idea: **Adversarial training of the smoothed classifier g** (approx.)
  - **Adversarial attack** is performed on **the soft smoothed classifier G**

$$G(x) \coloneqq \big(F * \mathcal{N}(0, \sigma^2 I)\big)(x) = \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)}[F(x+\delta)]$$

Softmax outputs of f

  - One can find **an adversarial example $\hat{x}$** of a clean sample x w.r.t. G as follows:

$$\hat{x} \coloneqq \max_{||x'-x||_2 \leq \epsilon} (\mathcal{L}(G(x'), y)$$

$$= \max_{||x'-x||_2 \leq \epsilon} (-\log \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)} \left[\big(F(x'+\delta)\big)_y \right]$$

- Idea: Adversarial training of the smoothed classifier g (approx.)
  - One can find **an adversarial example $\hat{x}$** of a clean sample x as follows:

$$\hat{x} := \max_{||x'-x||_2 \leq \epsilon} (\mathcal{L}(G(x'), y)$$

$$= \max_{||x'-x||_2 \leq \epsilon} \left(-\log \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)} \left[\left(F(x'+\delta) \right)_y \right] \right)$$

  - The gradient ascent step is performed on x' via Monte-Carlo approximation

$$\nabla_{x'} (\mathcal{L}(G(x'), y) = \nabla_{x'} \left(-\log \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)} \left[\left(F(x'+\delta) \right)_y \right] \right)$$

$$\approx \nabla_{x'} \left(-\log \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)} \left(\frac{1}{m} \sum_{i=1}^{m} F(x'+\delta_i)_y \right) \right)$$

  - Then, **the adversarial example $\hat{x}$ w.r.t. G is used to train the base classifier f**

- Experiments
  - Comparison of [Cohen et al., 2019]
  - Improves **the certified accuracy** of the smoothed classifier g

Table 1: Certified top-1 accuracy of our best ImageNet classifiers at various ℓ_2 radii.

ℓ_2 RADIUS (IMAGENET)	0.5	1.0	1.5	2.0	2.5	3.0	3.5
COHEN ET AL. [6] (%)	49	37	29	19	15	12	9
OURS (%)	**56**	**45**	**38**	**28**	**26**	**20**	**17**

Table 2: Certified top-1 accuracy of our best CIFAR-10 classifiers at various ℓ_2 radii.

ℓ_2 RADIUS (CIFAR-10)	0.25	0.5	0.75	1.0	1.25	1.5	1.75	2.0	2.25
COHEN ET AL. [6] (%)	61	43	32	22	17	13	10	7	4
OURS (%)	73	58	48	38	33	29	24	18	16

  - However, it requires:
    - **High computational cost** due to the adversarial attack
    - **Many hyperparameters** such as
      - (1) ϵ: maximum allowed l_2 perturbation
      - (2) T: number of steps of the attack
      - (3) m: the number of noise samples for Monte Carlo approximation

- Idea: Maintain **the prediction consistency** over Gaussian noise
  - 0-1 loss of smoothed classifier [Zhai et al., 2020]

$$\mathbb{E}_{(x,y)\in\mathbb{D}}\left[1-\mathbb{1}_{R(g;x,y)\geq\epsilon}\right]=\mathbb{E}\left[\mathbb{1}_{g(x)\neq y}\right]+\mathbb{E}\left[\mathbb{1}_{g(x)=y,R(g;x,y)<\epsilon}\right]$$

Indicator function

Natural error

robust error

  - **"Consistent" prediction** in logit space over Gaussian noise (i.e., $F(x+\delta)$ is constant) implies $\mathbb{P}_\delta\big(f(x+\delta)=g(x)\big)$ **to become 1**

  - Then, **the robust term of 0-1 loss is upper-bounded** by the following:

$$\mathbb{E}\left[\mathbb{1}_{g(x)=y,R(g;x,y)<\epsilon}\right]=\mathbb{E}\left[\mathbb{1}_{g(x)=y,R(g;x,g(x))<\epsilon}\right]$$

robust error

$$=\mathbb{E}\left[\mathbb{1}_{R(g;x,g(x))<\epsilon}\right]\leq\mathbb{E}\left[\mathbb{1}_{\mathbb{P}_\delta(f(x+\delta)=g(x))<\Phi(\epsilon/\sigma)]}\right]$$

  - In other words, consistent prediction minimizes **upper-bound of the robust 0-1 loss**
  - Note: As $\mathbb{P}_\delta\big(f(x+\delta)=g(x)\big)$ goes to 1, $\mathbb{E}\left[\mathbb{1}_{\mathbb{P}_\delta(f(x+\delta)=g(x))<\Phi(\epsilon/\sigma)]}\right]$ goes to 0

- Idea: Maintain the prediction consistency over Gaussian noise
  - Objective: **"Consistent" prediction** in logit space over Gaussian noise

$$L^{con} \coloneqq \lambda \cdot \mathbb{E}_{\delta}[KL(\hat{F}(x)||F(x+\delta)] + \eta \cdot H(\hat{F}(x))$$

where $\hat{F}(x) \coloneqq \mathbb{E}[F(x+\delta)]$, and $H(\cdot)$ denotes the entropy

  - Entropy term is to prevent $\hat{F}(x)$ to be too close the the uniform
  - Empirically, the expectation over Gaussian is approximated by Monte-Carlo

$$L \coloneqq L^{nat} + L^{con} \coloneqq \mathbb{E}[\mathcal{L}(F(x+\delta), y) + \lambda \cdot KL(\hat{F}(x)||F(x+\delta) + \eta \cdot H(\hat{F}(x))]$$

$$\approx \frac{1}{m}\sum_{i}(\mathcal{L}(F(x+\delta_i), y) + \lambda \cdot KL(\hat{F}(x)||F(x+\delta_i))) + \lambda \cdot H(\hat{F}(x))$$

  - **No attack, fewer hyperparameter** compared to [Salman et al., 2020]

- Experimental results
  - ACR (Average Certified Radius) is defined by:

$$\frac{1}{|\mathcal{D}_{test}|} \sum_{(x,y)\in\mathbb{D}_{test}} CR(f,\sigma,x) \cdot \mathbb{1}_{g(x)=y}$$

Certified radius

  - ACR reflects the trade-off between robustness and accuracy of smoothed classifier

- Experimental results
  - **Effectively trade-off the robustness and accuracy**
  - The robustness of smoothed classifier dramatically increases

Table 1: Comparison of approximate certified test accuracy (%) on CIFAR-10. Every model is certified with σ used for its training. We set our result bold-faced whenever the value improves the baseline. For ACRs, we underline the best model per σ. For the results in "+ Hyperparameter search", we evaluate the best model among those released by Salman et al. [32] for each σ.

σ	Models (CIFAR-10)	ACR	0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25
0.25	Gaussian [10]	0.424	76.6	61.2	42.2	25.1	0.0	0.0	0.0	0.0	0.0	0.0
	+ Consistency ($\lambda = 10$)	**0.544**	**77.8**	**68.8**	**57.4**	**43.8**	0.0	0.0	0.0	0.0	0.0	0.0
	+ Consistency ($\lambda = 20$)	**<u>0.552</u>**	75.8	**67.6**	**58.1**	**46.7**	0.0	0.0	0.0	0.0	0.0	0.0
	SmoothAdv [32]	0.544	73.4	65.6	57.0	47.5	0.0	0.0	0.0	0.0	0.0	0.0
	+ Consistency ($\lambda = 2$)	**0.548**	72.9	65.6	**57.5**	**48.5**	0.0	0.0	0.0	0.0	0.0	0.0
	Stability training [23]	0.421	72.3	58.0	43.3	27.3	0.0	0.0	0.0	0.0	0.0	0.0
	MACER [44]	0.531	79.5	69.0	55.8	40.6	0.0	0.0	0.0	0.0	0.0	0.0
0.50	Gaussian [10]	0.525	65.7	54.9	42.8	32.5	22.0	14.1	8.3	3.9	0.0	0.0
	+ Consistency ($\lambda = 10$)	**0.720**	64.3	**57.5**	**50.6**	**43.2**	**36.2**	**29.5**	**22.8**	**16.1**	0.0	0.0
	SmoothAdv [32]	0.689	64.4	57.2	49.0	40.6	33.6	27.4	21.8	14.0	0.0	0.0
	+ Hyperparameter search	0.717	53.1	49.2	44.9	41.0	37.2	33.2	29.1	24.0	0.0	0.0
	+ Consistency ($\lambda = 1$)	**<u>0.726</u>**	52.3	48.9	**45.1**	**41.3**	**37.8**	**33.9**	**29.9**	**25.2**	0.0	0.0
	Stability training [23]	0.521	60.6	51.5	41.4	32.5	23.9	15.3	9.6	5.0	0.0	0.0
	MACER [44]	0.691	64.2	57.5	49.9	42.3	34.8	27.6	20.2	12.6	0.0	0.0
1.00	Gaussian [10]	0.542	47.2	39.2	34.0	27.8	21.6	17.4	14.0	11.8	10.0	7.6
	+ Consistency ($\lambda = 5$)	**0.734**	**48.1**	**43.9**	**39.3**	**34.7**	**29.9**	**26.1**	**22.1**	**18.8**	**15.4**	**12.2**
	+ Consistency ($\lambda = 10$)	**0.756**	46.3	**42.2**	**38.1**	**34.3**	**30.0**	**26.3**	**22.9**	**19.7**	**16.6**	**13.8**
	SmoothAdv [32]	0.682	50.2	44.0	37.6	33.8	28.8	24.0	20.2	15.8	13.2	10.2
	+ Hyperparameter search	0.785	45.6	41.9	38.0	34.2	30.9	27.4	24.1	20.7	17.7	14.9
	+ Consistency ($\lambda = 1$)	**<u>0.816</u>**	41.7	39.0	36.2	33.5	30.7	**27.6**	**24.7**	**22.0**	**19.5**	**17.3**
	Stability training [23]	0.526	43.5	38.9	32.8	27.0	23.1	19.1	15.4	11.3	7.8	5.7
	MACER [44]	0.744	41.4	38.5	35.2	32.3	29.3	26.4	23.4	20.2	17.4	14.5

- Experimental results
  - **Fewer hyperparameter/training time** compared to baselines
  - At least achieves comparable results

Table 2: Comparison of approximate certified test accuracy (%) on ImageNet. We set our result bold-faced whenever the value improves the baseline. We use $\eta = 0.1$ instead of 0.5 when $\sigma = 1.0$.

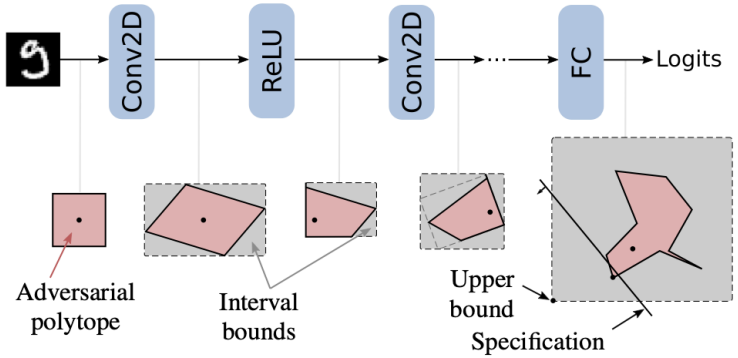
σ	Models (ImageNet)	ACR	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5
0.50	Gaussian [10]	0.733	57	46	37	29	0	0	0	0
	+ Consistency ($\lambda = 5$)	**0.822**	55	**50**	**44**	**34**	0	0	0	0
	SmoothAdv [32]	0.825	54	49	43	37	0	0	0	0
1.00	Gaussian [10]	0.875	44	38	33	26	19	15	12	9
	+ Consistency ($\lambda = 5$)	**0.982**	41	37	32	**28**	**24**	**21**	**17**	**14**
	SmoothAdv [32]	1.040	40	37	34	30	27	25	20	15

Table 3: Comparison of training time statistics on CIFAR-10 with $\sigma = 0.50$. All the baselines are trained on their official implementations separately.

Models	# HP	ACR	Mem.	Time (h)
Gaussian		0.525	2.9G	4.6
+ Consistency	**2**	**0.720**	2.9G	**8.7**
SmoothAdv	4	0.717	3.0G	23.1
MACER	4	0.691	9.4G	14.1

- **Randomized smoothing** cares about the l_2**-norm certified robustness**
  - A few approaches consider l_∞**-norm certified robustness**
  - So far, not scaled up to full-resolution ImageNet

  - On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models [Gowal et al., 2018]
    - Interval Bound Propagation (IBP)



    - Maximize the lower bound of the logit value of the true label
    - Minimize the upper bound of the logit values of other labels

- **Randomized smoothing** cares about the l_2**-norm certified robustness**
  - A few approaches consider l_∞**-norm certified robustness**
  - So far, not scaled up to full-resolution ImageNet

  - Boosting the Certified Robustness of L-infinity Distance Nets [Zhang et al., 2022]
    - Introduce l_∞-distance layer

$$x_i^{(l)} = u(\boldsymbol{x}^{(l-1)}, \{\boldsymbol{w}^{(l,i)}, b_i^{(l)}\}) = \|\boldsymbol{x}^{(l-1)} - \boldsymbol{w}^{(l,i)}\|_\infty + b_i^{(l)}, \quad l \in [L], i \in [n_l]$$

    - l_∞-distance net satisfies **1-Lipschitz continuity in l_∞-norm**

Proposition 2.2. *The mapping of an ℓ_∞-distance layer is 1-Lipschitz with respect to ℓ_∞-norm. Thus by composition, any ℓ_∞-distance net $\boldsymbol{g}(\cdot)$ is 1-Lipschitz with respect to ℓ_∞-norm.*

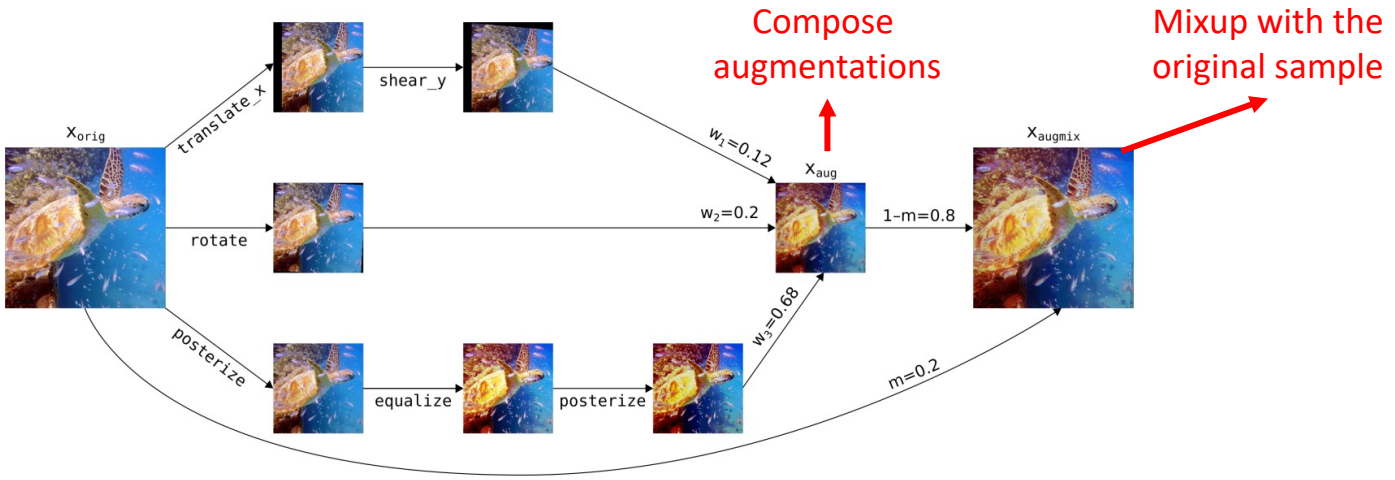
  - One can guarantee $g(x) = g(x + \delta)$ when $||\delta||_\infty < \text{margin}(\boldsymbol{x}, y; \boldsymbol{g})/2$

$$\text{margin}(\boldsymbol{x}, y; \boldsymbol{g}) = [\boldsymbol{g}(\boldsymbol{x})]_y - \max_{j \neq y} [\boldsymbol{g}(\boldsymbol{x})]_j$$

Table of Contents

Robust training schemes: AugMix [Hendrycks et al., 2020]

- Motivation: Data augmentation largely improve the generalization performance
- **AugMix**: Mixup the original image with the composed augmentations
  - Intuitively, it generates diverse image without veering too far from the original



  - Then, regularize the predictive distribution to be consistency across augmentations
    - This injects an inductive bias to the classifier

$$\mathcal{L}(p_{\text{orig}}, y) + \lambda \, \text{JS}(p_{\text{orig}}; p_{\text{augmix1}}; p_{\text{augmix2}})$$

JS: Jensen-Shannon divergence
p_{orig}: original sample's output
$p_{\text{augmix}-i}$: AugMix sample's output

*source: Hendrycks et al, AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty, ICLR 2020

- **Experimental results**
  - AugMix significantly outperforms the baseline augmentation schemes

		Standard	Cutout	Mixup	CutMix	AutoAugment*	Adv Training	AUGMIX
CIFAR-10-C	AllConvNet	30.8	32.9	24.6	31.3	29.2	28.1	**15.0**
	DenseNet	30.7	32.1	24.6	33.5	26.6	27.6	**12.7**
	WideResNet	26.9	26.8	22.3	27.1	23.9	26.2	**11.2**
	ResNeXt	27.5	28.9	22.6	29.5	24.2	27.0	**10.9**
Mean		29.0	30.2	23.5	30.3	26.0	27.2	**12.5**
CIFAR-100-C	AllConvNet	56.4	56.8	53.4	56.0	55.1	56.0	**42.7**
	DenseNet	59.3	59.6	55.4	59.2	53.9	55.2	**39.6**
	WideResNet	53.3	53.5	50.4	52.9	49.6	55.1	**35.9**
	ResNeXt	53.4	54.6	51.4	54.1	51.3	54.4	**34.9**
Mean		55.6	56.1	52.6	55.5	52.5	55.2	**38.3**

CIFAR-10 and CIFAR-100 results

		Noise			Blur				Weather				Digital				
Network	Clean	Gauss.	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	**mCE**
Standard	23.9	79	80	82	82	90	84	80	86	81	75	65	79	91	77	80	80.6
Patch Uniform	24.5	67	68	70	74	83	81	77	80	74	75	62	77	84	71	71	74.3
AutoAugment* (AA)	22.8	69	68	72	77	83	80	81	79	75	64	**56**	70	88	**57**	71	72.7
Random AA*	23.6	70	71	72	80	86	82	81	81	77	72	61	75	88	73	72	76.1
MaxBlur pool	23.0	73	74	76	74	86	78	77	77	72	**63**	**56**	68	86	71	71	73.4
SIN	27.2	69	70	70	77	84	76	82	74	75	69	65	69	80	64	77	73.3
AUGMIX	**22.4**	65	66	67	70	80	66	**66**	75	72	67	58	58	79	69	69	68.4
AUGMIX+SIN	25.2	**61**	**62**	**61**	**69**	**77**	**63**	72	**66**	**68**	**63**	59	**52**	**74**	60	**67**	**64.9**

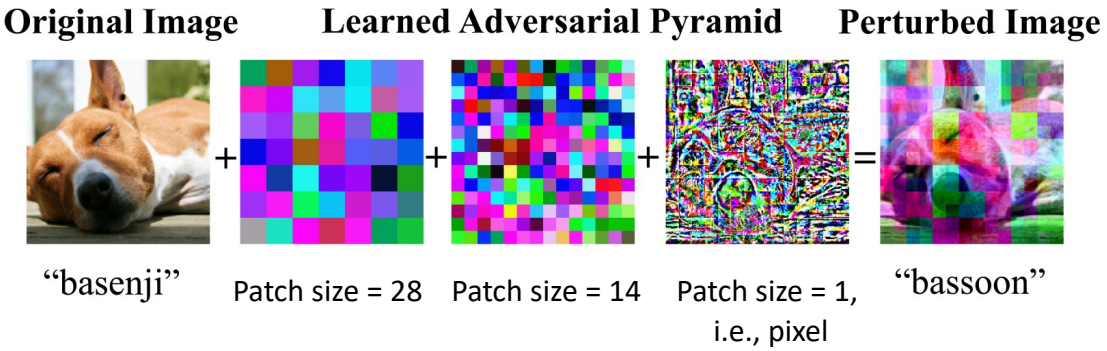
ImageNet result

*source: Hendrycks et al, AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty, ICLR 2020

Pyramid Adversarial Training [Herrmann et al., 2022]

- Pyramid AT: utilize adversarial examples as data augmentations
  - This method is typically designed for **patch-based models**, e.g., ViT or MLP-Mixer

- Pyramid AT use a **patch-wise adversarial attack**
  - Constraint the patch to have the same noise scale
  - Add the adversarial noise across various patch sizes



- + one should remove the **randomness** of the model when using adversaries
  - Note that ViT consist of dropout (and stochastic depth)
  - Such randomness may induce gradient obfuscations

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[L(\mathcal{M}(\theta),\tilde{x},y)+\lambda\max_{\delta\in\mathcal{P}}L(\theta,x^a,y)+f(\theta)\right]$$

Randomness (dropout mask, $\mathcal{M}$) for clean data **Fixed parameter for adversaries**

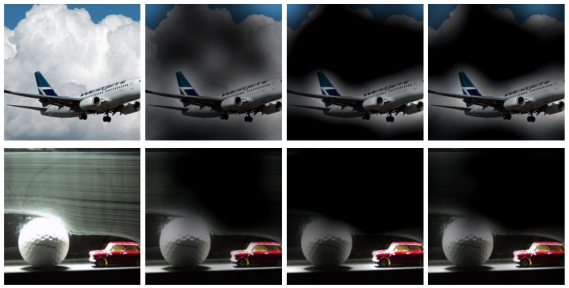
*source: Herrmann et al, Pyramid Adversarial Training Improves ViT Performance, CVPR 2022

Pyramid Adversarial Training [Herrmann et al., 2022]

- **Experimental results**
  - Pyramid AT significantly improves the distributional shift robustness
  - More intriguingly, the clean accuracy also improves

Method	ImageNet	Real	Out of Distribution Robustness Test						
			A	C↓	ObjectNet	V2	Rendition	Sketch	Stylized
ViT [13]	72.82	78.28	8.03	74.08	17.36	58.73	27.07	17.28	6.41
ViT+CutMix [60]	75.49	80.53	14.75	64.07	21.61	62.37	28.47	17.15	7.19
ViT+Mixup [61]	77.75	82.93	12.15	61.76	25.65	64.76	34.90	25.97	9.84
RegViT (RandAug) [48]	79.92	85.14	17.48	52.46	29.30	67.49	38.24	29.08	11.02
+Random Pixel	79.72	84.72	17.81	52.83	28.72	67.17	39.01	29.26	12.11
+Random Pyramid	80.06	85.02	19.15	52.49	29.41	67.81	39.78	30.30	11.64
+Adv Pixel	80.42	85.78	19.15	47.68	30.11	68.78	45.39	34.40	18.28
+Adv Pyramid (ours)	**81.71**	**86.82**	**22.99**	**44.99**	**32.92**	**70.82**	**47.66**	**36.77**	**19.14**
RegViT [48] on 384x384	81.44	86.38	26.20	58.19	35.59	70.09	38.15	28.13	8.36
+Random Pixel	81.32	86.18	25.95	58.69	34.12	69.50	37.66	28.79	9.77
+Random Pyramid	81.42	86.30	27.55	57.31	34.83	70.53	38.12	29.16	9.61
+Adv Pixel	82.24	87.35	31.23	48.56	37.41	71.67	44.07	33.68	13.52
+Adv Pyramid	**83.26**	**88.14**	**36.41**	**47.76**	**39.79**	**73.14**	**46.68**	**36.73**	**15.00**

- Moreover, the attention and saliency map well aligns with the object



Attention map

Saliency map

*source: Herrmann et al, Pyramid Adversarial Training Improves ViT Performance, CVPR 2022

Test-time Adaptation

- Another direction is to **adapt the model to the unseen distribution**
  - Use the test input (from unseen distribution) for the adaptation

- This direction have some benefits compare to the robust training schemes
  - (i) Modifying the training may not be feasible due to computation (of re-training)
  - (ii) Can utilize the information of unseen distribution with the test inputs
  - (iii) **does not require any assumptions about the training procedure**
    - E.g., domain adaptation requires domain labels during training

setting	source data	target data	train loss	test loss
fine-tuning	-	x^t, y^t	$L(x^t, y^t)$	-
domain adaptation	x^s, y^s	x^t	$L(x^s, y^s) + L(x^s, x^t)$	-
test-time training	x^s, y^s	x^t	$L(x^s, y^s) + L(x^s)$	$L(x^t)$
fully test-time adaptation	-	x^t	-	$L(x^t)$

*source: Wang et al., Tent: Fully Test-time Adaptation by Entropy Minimization, ICLR 2021

Tent: Fully Test-time Adaptation by Entropy Minimization [Wang et al., 2021]

- Prior work: **Batch Normalization (BN) adaptation** [Schnider et al., 2020]
  - Adapting the batch statistic of the BN significantly improves the robustness
  - One can obtain the test (target) mean and variance statistic with single forward

μ_s: source mean, μ_t: target mean, σ_s: source mean, σ_t: target mean

$$\bar{\mu} = \frac{N}{N+n}\mu_s + \frac{n}{N+n}\mu_t, \quad \bar{\sigma}^2 = \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\sigma_t^2.$$

New batch statistics

  - BN adaptation can be applied to any models with BN

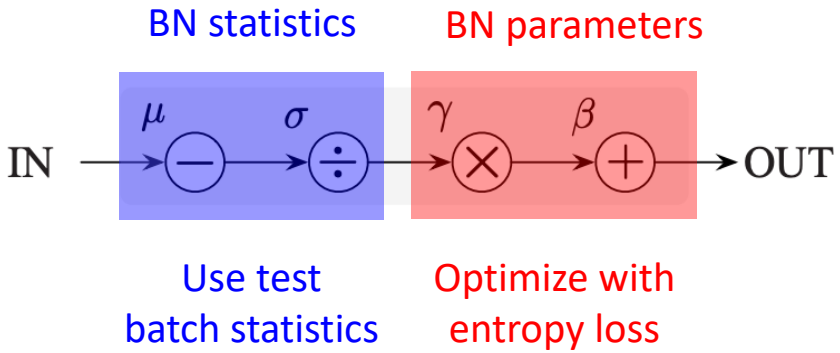
	IN-C mCE (↘)				Top1 accuracy (↗)			
Model	w/o adapt	partial adapt	full adapt	Δ	w/o adapt	partial adapt	full adapt	Δ
Vanilla ResNet-50	76.7	65.0	62.2	(−14.5)	39.2	48.6	50.7	(+11.5)
SIN [28]	69.3	61.5	59.5	(−9.8)	45.2	51.6	53.1	(+7.9)
ANT [29]	63.4	56.1	53.6	(−9.8)	50.4	56.1	58.0	(+7.6)
ANT+SIN [29]	60.7	55.3	53.6	(−7.0)	52.6	56.8	58.0	(+5.4)
AugMix [AM; 30]	65.3	55.4	51.0	(−14.3)	48.3	56.3	59.8	(+11.4)
Assemble Net [32]	52.3	–	50.1	(−1.2)	59.2	–	60.8	(+1.5)
DeepAug [36]	60.4	52.3	49.4	(−10.9)	52.6	59.0	61.2	(+8.6)
DeepAug+AM [36]	53.6	48.4	45.4	(−8.2)	58.1	62.2	64.5	(+6.4)
DeepAug+AM+RNXt101 [36]	**44.5**	40.7	**38.0**	(−6.6)	**65.2**	68.2	**70.3**	(+5.1)

Partial: small batch
Full: full batch

*source: Schneider et al., Improving robustness against common corruptions by covariate shift adaptation, NeurIPS 2020

Tent: Fully Test-time Adaptation by Entropy Minimization [Wang et al., 2021]

- Tent adapt the BN parameters by minimizing the **test entropy H**
  - $H(\hat{y}) = -\sum_c p(\hat{y}_c)\log p(\hat{y}_c)$ of model predictions $\hat{y} = f_\theta(\hat{x})$.
  - Also, Tent use the test batch statistics for BN (i.e., fully adapt the batch statistics)



normalization $\mu \leftarrow \mathbb{E}[x_t], \sigma^2 \leftarrow \mathbb{E}[(\mu - x_t)^2]$

transformation $\gamma \leftarrow \gamma + \partial H/\partial\gamma, \beta \leftarrow \beta + \partial H/\partial\beta$

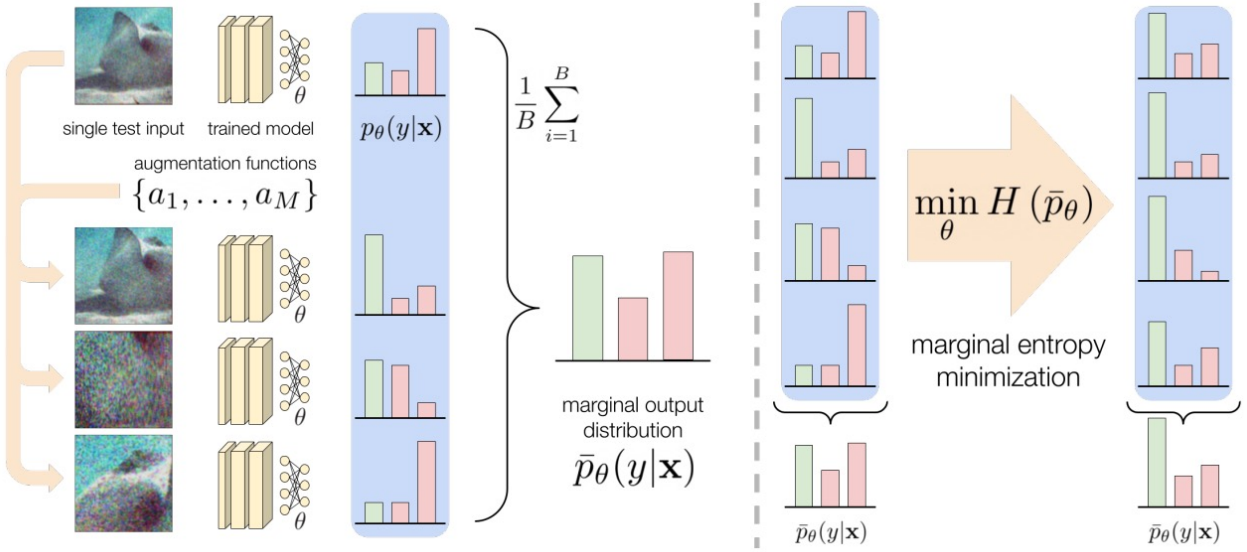
- Tent significantly outperforms the baseline robustification methods

Method	Source	Target	Error (%) C10-C	Error (%) C100-C
Source	train		40.8	67.2
RG	train	train	18.3	38.9
UDA-SS	train	train	16.7	47.0
TTT	train	test	17.5	45.0
BN		test	17.3	42.6
PL		test	15.7	41.2
Tent (ours)		test	**14.3**	**37.3**

*source: Wang et al., Tent: Fully Test-time Adaptation by Entropy Minimization, ICLR 2021

MEMO: Test Time Robustness via Adaptation and Augmentation [Zhang et al., 2021]

- Limitation of prior adaptation works: **require batches or entire test dataset**
- For single sample adaptation, MEMO suggest to augment the test data
  - In this regard, **one can generate a batch** with a single sample



  - Then, MEMO minimize the entropy of average prediction of the batch

$$\bar{p}_\theta(y|\mathbf{x}) \triangleq \mathbb{E}_{\mathcal{U}(\mathcal{A})}\left[p_\theta(y|a(\mathbf{x}))\right] \approx \frac{1}{B}\sum_{i=1}^{B} p_\theta(y|\tilde{\mathbf{x}}_i)$$

*source: Zhang et al., Test Time Robustification of Deep Models via Adaptation and Augmentation, arXiv 2021

MEMO: Test Time Robustness via Adaptation and Augmentation [Zhang et al., 2021]

- **Experimental results**
  - MEMO significantly improve the baselines (i.e., single sample adaptation methods)

	ImageNet-C mCE ↓	ImageNet-R Error (%)	ImageNet-A Error (%)
Baseline ResNet-50 (He et al., 2016)	76.7	63.9	100.0
+ TTA	77.9 (+1.2)	61.3 (−2.6)	98.4 (−1.6)
+ Single point BN	71.4 (−5.3)	61.1 (−2.8)	99.4 (−0.6)
+ MEMO (ours)	69.9 (−6.8)	58.8 (−5.1)	99.1 (−0.9)
+ BN ($N = 256, n = 256$)	61.6 (−15.1)	59.7 (−4.2)	99.8 (−0.2)
+ Tent (online) (Wang et al., 2021)	54.4 (−22.3)	57.7 (−6.2)	99.8 (−0.2)
+ Tent (episodic)	64.7 (−12.0)	61.0 (−2.9)	99.7 (−0.3)
+ DeepAugment+AugMix (Hendrycks et al., 2021a)	53.6	53.2	96.1
+ TTA	55.2 (+1.6)	51.0 (−2.2)	93.5 (−2.6)
+ Single point BN	51.3 (−2.3)	51.2 (−2.0)	95.4 (−0.7)
+ MEMO (ours)	**49.8** (−3.8)	**49.2** (−4.0)	94.8 (−1.3)
+ BN ($N = 256, n = 256$)	45.4 (−8.2)	48.8 (−4.4)	96.8 (+0.7)
+ Tent (online)	**43.5** (−10.1)	**46.9** (−6.3)	96.7 (+0.6)
+ Tent (episodic)	47.1 (−6.5)	50.1 (−3.1)	96.6 (+0.5)
+ MoEx+CutMix (Li et al., 2021)	74.8	64.5	91.9
+ TTA	75.7 (+0.9)	62.7 (−1.8)	89.5 (−2.4)
+ Single point BN	71.0 (−3.8)	62.6 (−1.9)	91.1 (−0.8)
+ MEMO (ours)	69.1 (−5.7)	59.4 (−3.3)	**89.0** (−2.9)
+ BN ($N = 256, n = 256$)	60.9 (−13.9)	61.6 (−2.9)	93.9 (+2.0)
+ Tent (online)	54.0 (−20.8)	58.7 (−5.8)	94.4 (+2.5)
+ Tent (episodic)	66.2 (−8.6)	63.9 (−0.6)	94.7 (+2.8)
RVT*-small (Mao et al., 2021)	49.4	52.3	73.9
+ TTA	53.0 (+3.6)	49.0 (−3.3)	**68.9** (−5.0)
+ Single point BN	48.0 (−1.4)	51.1 (−1.2)	74.4 (+0.5)
+ MEMO (ours)	**40.6** (−8.8)	**43.8** (−8.5)	69.8 (−4.1)
+ BN ($N = 256, n = 256$)	44.3 (−5.1)	51.0 (−1.3)	78.3 (+4.4)
+ Tent (online)	46.8 (−2.6)	50.7 (−1.6)	82.1 (+8.2)
+ Tent (adapt all)	44.7 (−4.7)	74.1 (+21.8)	81.1 (+7.2)

*source: Zhang et al., Test Time Robustification of Deep Models via Adaptation and Augmentation, arXiv 2021

Summary

- **Robustness is critical** for many real-world applications with trustworthy usage
  - E.g., autonomous driving cars

- We discussed **two types** of venerability in modern machine learning
  - Adversarial examples: a perturbation that confuses the network prediction
  - Distributional shifts: where the train and test distribution differs

- To train robust classifier under adversarial examples, we mainly discussed
  - **Adversarial training**, i.e., **t**raining the network with adversarial examples
  - Test-time robustness certification via **randomized smoothing**

- To train robust classifier under distributional shifts, we mainly discussed
  - Robust training schemes with **new data augmentation schemes**
  - **Test-time adaptation** methods to new distributions

Summary: RobustBench

- To see more recent works on robustness, refer to **RobustBench**
  - This site shows the robustness performance of various defense methods
  - URL: https://robustbench.github.io/

Leaderboard: CIFAR-10, $\ell_\infty = 8/255$, untargeted attack

Show 15 entries Search: Papers, architectures, ve

Rank	Method	Standard accuracy	AutoAttack robust accuracy	Best known robust accuracy	AA eval. potentially unreliable	Extra data	Architecture	Venue
1	Fixing Data Augmentation to Improve Adversarial Robustness *66.56% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	92.23%	66.58%	66.56%	×	☑	WideResNet-70-16	arXiv, Mar 2021
2	Improving Robustness using Generated Data *It uses additional 100M synthetic images in training. 66.10% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	88.74%	66.11%	66.10%	×	×	WideResNet-70-16	NeurIPS 2021
3	Uncovering the Limits of Adversarial Training against Norm-Bounded Adversarial Examples *65.87% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	91.10%	65.88%	65.87%	×	☑	WideResNet-70-16	arXiv, Oct 2020
4	Fixing Data Augmentation to Improve Adversarial Robustness *It uses additional 1M synthetic images in training. 64.58% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	88.50%	64.64%	64.58%	×	×	WideResNet-106-16	arXiv, Mar 2021

*source: Croce et al., RobustBench: a standardized adversarial robustness benchmark, NeurIPS 2021

References

[Athalye et al., 2018a] Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples, ICML 2018
https://arxiv.org/abs/1802.00420

[Athalye et al., 2018b] Synthesizing robust adversarial examples, ICML 2018
https://arxiv.org/abs/1707.07397

[Blanchet et al., 2016] Quantifying Distributional Model Risk via Optimal Transport, arXiv 2016
https://arxiv.org/abs/1604.01446

[Buckman et al., 2018] Thermometer Encoding: One Hot Way To Resist Adversarial Examples, ICLR 2018
https://openreview.net/forum?id=S18Su--CW

[Carlini & Wagner, 2017a] Towards Evaluating the Robustness of Neural Networks, IEEE S&P 2017
https://arxiv.org/abs/1608.04644

[Carlini et al., 2019] On Evaluating Adversarial Robustness, arXiv 2019
https://arxiv.org/abs/1902.06705

[Dhillon et al., 2018] Stochastic Activation Pruning for Robust Adversarial Defense, ICLR 2018
https://arxiv.org/abs/1803.01442

[Elsayed et al., 2018] Large Margin Deep Networks for Classification, NIPS 2018
https://arxiv.org/abs/1803.05598

[Eykholt et al., 2017] Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR 2018
https://arxiv.org/abs/1707.08945

[Goodfellow et al., 2015] Explaining and Harnessing Adversarial Examples, ICLR 2015
https://arxiv.org/abs/1412.6572

References

[Karmon et al., 2018] LaVAN: Localized and Visible Adversarial Noise, ICML 2018
https://arxiv.org/abs/1801.02608

[Kurakin et al., 2017a] Adversarial Examples in the Physical World, ICLR Workshop 2017
https://arxiv.org/abs/1607.02533

[Kurakin et al., 2017b] Adversarial Machine Learning at Scale, ICLR 2017
https://arxiv.org/abs/1611.01236

[Lee et al., 2018] A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks, NIPS 2018
https://arxiv.org/abs/1807.03888

[Liu et al., 2017] Delving into Transferable Adversarial Examples and Black-box Attacks, ICLR 2017
https://arxiv.org/abs/1611.02770

[Madry et al., 2018] Towards Deep Learning Models Resistant to Adversarial Attacks, ICLR 2018
https://arxiv.org/abs/1706.06083

[Moosavi-Dezfooli et al., 2016] DeepFool: a simple and accurate method to fool deep neural networks, CVPR 2016
https://arxiv.org/abs/1511.04599

[Moosavi-Dezfooli et al., 2017] Universal adversarial perturbations, CVPR 2017
https://arxiv.org/abs/1610.08401

[Papernot et al., 2017] Practical Black-Box Attacks against Machine Learning, ACM CCS 2017
https://arxiv.org/abs/1602.02697

[Qin et al., 2019] Imperceptible, Robust, and Targeted Adversarial Examples for Automatic Speech Recognition, ICML 2019
http://proceedings.mlr.press/v97/qin19a.html

References

[Katz et al., 2017] Provably Minimally-Distorted Adversarial Examples, arXiv 2017
https://arxiv.org/abs/1709.10207

[Cheng et al., 2017] Maximum Resilience of Artificial Neural Networks, ATVA 2017
https://arxiv.org/abs/1705.01040

[Gouk et al., 2018] Regularization of Neural Networks by Enforcing Lipschitz Continuity, arXiv 2018
https://arxiv.org/abs/1804.04368

[Hein et al., 2018] Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation, NIPS 2017
https://arxiv.org/abs/1804.04368

[Cohen et al., 2019] Certified Adversarial Robustness via Randomized Smoothing, ICML 2019
https://arxiv.org/abs/1902.02918

[Salman et al., 2019] Provably Robust Deep Learning via Adversarially Trained Smoothed Classifiers, NeurIPS 2019
https://arxiv.org/abs/1906.04584

[Jeong and Shin, 2020] Consistency Regularization for Certified Robustness of Smoothed Classifiers, NeurIPS 2020
https://arxiv.org/abs/2006.04062

[Zhai et al., 2020] MACER: Attack-free and Scalable Robust Training via Maximizing Certified Radius, ICLR 2020
https://arxiv.org/abs/2001.02378

[Gowal et al., 2018] On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models, arXiv 2018
https://arxiv.org/abs/1810.12715

[Zhang et al., 2022] Boosting the Certified Robustness of L-infinity Distance Nets, ICLR 2018
https://arxiv.org/abs/2110.06850

References

[Samangouei et al., 2018] Defense-GAN: Protecting Classifiers Against Adversarial Attacks using Generative Models, ICLR 2018
https://arxiv.org/abs/1805.06605

[Sinha et al., 2018] Certifying Some Distributional Robustness with Principled Adversarial Training, ICLR 2018
https://arxiv.org/abs/1710.10571

[Su et al., 2017] One pixel attack for fooling deep neural networks, arXiv 2017
https://arxiv.org/abs/1710.08864

[Wong et al., 2018] Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope, ICML 2018
https://arxiv.org/abs/1711.00851

[Xiao et al., 2018] Spatially Transformed Adversarial Examples, ICLR 2018
https://arxiv.org/abs/1801.02612

[Xie et al., 2017] Adversarial Examples for Semantic Segmentation and Object Detection, ICCV 2017
https://arxiv.org/abs/1703.08603

[Xie et al., 2018] Improving Transferability of Adversarial Examples with Input Diversity, arXiv 2018
https://arxiv.org/abs/1803.06978

[Zhang et al., 2019] Theoretically Principled Trade-off between Robustness and Accuracy, ICML 2019
https://arxiv.org/abs/1901.08573

[Geirhos et al., 2019] ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness, ICLR 2019
https://arxiv.org/abs/1811.12231

References

[Koh et al., 2021] WILDS: A Benchmark of in-the-Wild Distribution Shifts, ICML 2021
https://arxiv.org/abs/2012.07421

[Croce et al., 2020] Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-free Attacks, I CML 2020
https://arxiv.org/abs/2003.01690

[Hendrcyks et al., 2019] Benchmarking Neural Network Robustness to Common Corruptions and Perturbations, ICLR 20 19
https://arxiv.org/abs/1903.12261

[Hendrcyks et al., 2021] The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization, ICCV 20 21
https://arxiv.org/abs/2006.16241

[Rice et al., 2020] Overfitting in adversarially robust deep learning, ICML 2020
https://arxiv.org/abs/2002.11569

[Wu et al., 2020] Adversarial Weight Perturbation Helps Robust Generalization, NeurIPS 2020
https://arxiv.org/abs/2004.05884

[Yun et al., 2019] CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features, ICCV 2019
https://arxiv.org/abs/1905.04899

[Cubuk et al., 2020] AutoAugment: Learning Augmentation Strategies from Data, CVPR 2020
https://arxiv.org/abs/1805.09501

[Rebuffi et al., 2021] Data Augmentation Can Improve Robustness, NeurIPS 2021
https://arxiv.org/abs/2111.05328

References

[Ho et al., 2020] Denoising Diffusion Probabilistic Models, NeurIPS 2020
https://arxiv.org/abs/2006.11239

[Gowal et al., 2021] Improving Robustness using Generated Data, NeurIPS 2021
https://arxiv.org/abs/2110.09468

[Hendrycks et al., 2020] AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty, ICLR 2020
https://arxiv.org/abs/1912.02781

[Herrmann et al., 2022] Pyramid Adversarial Training Improves ViT Performance, CVPR 2022
https://arxiv.org/abs/2111.15121

[Schneider et al., 2020] Improving robustness against common corruptions by covariate shift adaptation, NeurIPS 2020
https://arxiv.org/abs/2006.16971

[Wang et al., 2021] Tent: Fully Test-time Adaptation by Entropy Minimization, ICLR 2021
https://arxiv.org/abs/2006.10726

[Zhang et al., 2021] Test Time Robustification of Deep Models via Adaptation and Augmentation, arXiv 2021
https://arxiv.org/abs/2110.09506

[Croce et al., 2021] RobustBench: a standardized adversarial robustness benchmark, NeurIPS 2021
https://arxiv.org/abs/2010.09670