Adversarial Robustification via Text-to-Image Diffusion Models

Daewon Choi[A*] Jongheon Jeong[B*] Huiwon Jang[A] Jinwoo Shin[A]

[A]KAIST [B]Korea University

Contact: daeone0920@kaist.ac.kr

Paper Code

TL;DR: Your vision classifier can obtain adversarial robustness **without any training data**;
A strong **text-to-image diffusion model** is all you need.

Introduction

Adversarial robustness has been conventionally a challenging property to obtain, requiring plenty of training data.

$$\text{Goal: } f(\mathbf{x}) = f(\mathbf{x}+\boldsymbol{\delta}),\ \boxed{\forall \boldsymbol{\delta}}: \|\boldsymbol{\delta}\|_2 \le \varepsilon$$

(Classifier; Challenge)

Adversarial Training [1] generate adversarial examples (ex. via PGD) and add them to training set.

(–) Empirical robustness, Need target dataset for training classifier

Denoised smoothing [2,3] is recent framework of RS[4] using "denoise-and-classify" pipeline.

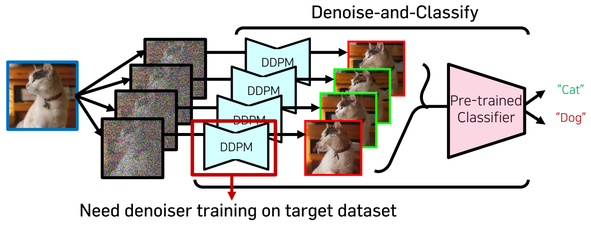
(+) Provable robustness, Not need to training classifier

(–) Need target dataset for separate training denoiser

An implementation of randomized smoothing (RS)

$$\hat{f}(\mathbf{x}) := \arg\max_{k\in\mathcal{Y}} \{\mathbb{P}_{\boldsymbol{\delta}\sim\mathcal{N}(0,\sigma^2 I)}(f(\mathbf{x}+\boldsymbol{\delta})=k)\}$$

(Gaussian noise)



Need denoiser training on target dataset

Recently, **[Mao et al., 2023]** has attempted to transfer adversarial robustness with zero-shot manner.

(+) Not need target dataset

(–) Still need external dataset for obtaining robustness

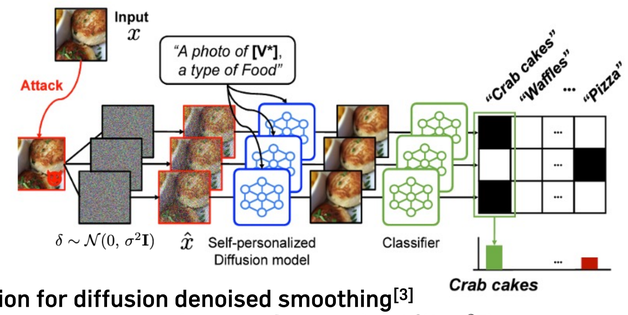
Research Question : Can we robustify a classifier without using external data ?

Key Idea: Text-to-image diffusion models (T2I) for robustification

- Incorporate T2I into the denoised smoothing pipeline with careful design
- Generate a few reference samples re-utilizing T2I, and leveraging them to adapt for target tasks

[1] [Madry et al., 2018] Towards Deep Learning Models Resistant to Adversarial Attacks, ICML 2018
[2] [Salman et al., 2020] Denoising Smoothing: A Provable Defense for Pretrained Classifiers, NeurIPS 2020
[3] [Carlini et al., 2023] (Certified!!) Adversarial Robustness for Free!, ICLR 2023
[4] [Cohen et al., 2019] Certified Adversarial Robustness via Randomized Smoothing, ICML 2019
[5] [Mao et al., 2023] Understanding Zero-shot Adversarial Robustness for Large-Scale Models, ICLR 2023
[6] [Saharia et al., 2022] Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding, NeurIPS 2022
[7] https://stability.ai/news/deepfloyd-if-text-to-image-model
[8] [Ruiz et al., 2023] Dream-Booth: Fine-tuning Text-to-Image Diffusion Models for Subject-Driven Generation, CVPR 2023

Denoised Smoothing from T2I



Notation for diffusion denoised smoothing[3]

- Gaussian noisy input: $\hat{x} := x + \delta$ * $\delta \sim \mathcal{N}(0, \sigma^2\mathbf{I})$
- $x_t := \sqrt{\alpha_t}\cdot x + \sqrt{1-\alpha_t}\cdot\varepsilon$ * α_t : diffusion schedule factor, $\varepsilon \sim \mathcal{N}(0,\mathbf{I})$

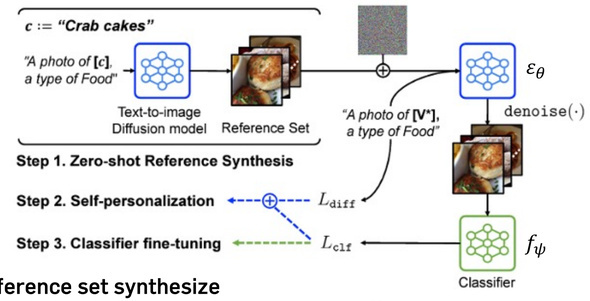
Super-resolution Diffusion Model as a Denoiser

- Super-resolution module (SR) in cascaded model[6,7] $\varepsilon_\theta(x_t, t, \tau_\theta(c)|\bar{x}_{t'}, t')$
  * τ_θ : text encoder, $(\bar{x}_{t'}, t')$: low-resolution module's input

Overall Pipeline

- Given a noisy input $\hat{\boldsymbol{x}}$, we define a denoiser function using SR:
  $\hat{x} - \sigma\cdot\varepsilon_\theta(\sqrt{\alpha_{\hat{t}}}\hat{x}, \hat{t}, \tau_\theta(\mathtt{C}(\text{“ ”}))|\sqrt{\alpha_{\hat{t}}}\hat{x}, k\hat{t})$
  * $\mathtt{C}(c)$: textual template, k is correction factor $t' := k\cdot\hat{t}$

Self-adaptation Schemes



Step 1. Reference set synthesize

Generating a few reference images $D^g = \{(x_i^g, c_i)\}_{i=1}^K$ from textual label c

Step 2&3. Classifier-Guided Self-personalization, Classifier Fine-tuning

$L_{\texttt{diff}}(\theta) := \mathbb{E}_{x^g,\varepsilon,t}\left[||\varepsilon - \varepsilon_\theta(x_t^g, t, \tau_\theta(\mathtt{C}(\text{“}sks\text{”}))|x_t^g, kt)||_2^2\right]$: Dream-Booth[7]

$L_{\texttt{clf}}(\theta,\psi) := \mathbb{E}_{(x^g,c)\sim D^g,t}\left[\mathbb{CE}(f_\psi(\tilde{x}^g), c)\right]$: Classifier-guided regularization

- Classifier-Guided Self-personalization
  $\theta^* = \arg\min_\theta \{L_{\texttt{diff}}(\theta) + \lambda\cdot L_{\texttt{clf}}(\theta,\psi)\}$
- Classifier Fine-tuning
  $\psi^* = \arg\min_\psi L_{\texttt{clf}}(\theta^*, \psi)$

Experiments

Our framework applied to the pre-trained CLIP could improve the (provable) adversarial robustness on diverse benchmarks while maintaining accuracy.

	Method	STL	SUN	Cars	Food	Pets	Flower	DTD	Caltech	Average
$\varepsilon = 0.5$	CLIP	10.8	1.2	0.0	1.8	2.7	0.8	2.7	12.0	4.0
	CLIP-Smooth	42.6	23.7	14.3	8.9	36.4	16.6	10.2	44.8	24.7
	Mao et al. [38]	59.4	29.9	12.5	32.9	51.2	33.5	18.8	56.2	36.8
	Ours	**80.4**	**41.8**	**33.2**	**59.0**	**68.6**	**45.2**	**29.7**	**71.3**	**53.7**
	(Certified)	(66.0)	(32.1)	(28.4)	(45.7)	(60.8)	(34.9)	(23.0)	(65.1)	(44.5)
$\varepsilon = 1.0$	CLIP	2.4	0.0	0.0	0.2	0.2	0.2	1.0	7.8	1.5
	CLIP-Smooth	16.2	5.8	1.6	0.4	6.7	4.5	5.4	18.7	7.4
	Mao et al. [38]	21.2	11.0	2.8	10.3	23.2	14.4	12.0	33.9	16.1
	Ours	**66.0**	**38.3**	**27.0**	**47.3**	**59.6**	**33.1**	**24.9**	**64.1**	**45.0**
	(Certified)	(41.2)	(22.5)	(18.9)	(28.9)	(46.5)	(18.9)	(18.2)	(55.8)	(31.4)

	Method	STL	SUN	Cars	Food	Pets	Flower	DTD	Caltech	Average
-	CLIP	97.8	56.8	52.7	83.0	85.7	66.3	37.8	81.9	70.3
$\varepsilon = 0.5$	CLIP-Smooth	75.0	46.8	42.1	52.3	66.7	43.5	17.2	68.3	51.5
	Mao et al. [38]	**94.8**	**60.0**	48.7	69.7	80.8	57.7	34.0	79.7	65.7
	Ours	**94.8**	58.6	**54.1**	**80.2**	**83.6**	**61.4**	**42.7**	**81.7**	**69.6**
	(Certified)	(90.4)	(55.4)	(49.7)	(74.5)	(81.9)	(58.7)	(38.9)	(79.1)	(66.1)
$\varepsilon = 1.0$	CLIP-Smooth	32.4	27.7	40.8	31.3	43.8	36.8	7.0	54.0	34.2
	Mao et al. [38]	93.4	58.2	42.9	61.2	77.0	53.6	30.8	78.5	62.0
	Ours	**93.8**	**59.4**	**52.9**	**78.8**	**83.1**	**58.9**	**39.1**	**81.7**	**68.5**
	(Certified)	(80.2)	(53.6)	(45.5)	(64.8)	(77.7)	(48.3)	(32.4)	(75.7)	(59.8)

8 standard zero-shot benchmarks (upper: robust / lower: clean)

Method	Data-free?	Robust accuracy (%) $\varepsilon=0.5$	Robust accuracy (%) $\varepsilon=1.0$	Clean accuracy (%) $\varepsilon=0.5$	Clean accuracy (%) $\varepsilon=1.0$
CLIP	✓	1.4	0.2	**58.2**	**58.2**
CLIP-Smooth	✓	16.8 (9.8)	2.2 (1.2)	45.2 (25.0)	35.2 (3.8)
Ours (w/o adapt)	✓	40.0 (29.6)	31.0 (17.6)	56.2 (50.8)	55.2 (42.0)
Ours	✓	**42.6** (**34.2**)	31.4 (**20.6**)	57.6 (**53.4**)	56.2 (**46.0**)
Mao et al. [38]	✗	26.0	12.3	51.2	47.2
Carlini et al. [7]	✗	38.6 (30.2)	**32.4** (19.8)	54.4 (49.8)	53.6 (44.2)

ImageNet results

ε	Adapt. T2I	Adapt. CLIP	Certified accuracy at ε (%) ImageNet	STL	SUN	Food
0.5	✗	✗	29.6	55.2	28.3	43.6
	✓	✗	31.8	**66.0**	30.7	43.8
	✓	✓	**34.2**	**66.0**	**32.1**	**45.7**
1.0	✗	✗	17.6	27.0	17.3	21.8
	✓	✗	19.4	40.8	19.3	27.3
	✓	✓	**20.6**	**41.2**	**22.5**	**28.9**

Self-adaptation is crucial

Method	Data-free?	Robust accuracy (%) $\varepsilon=0.5$	Robust accuracy (%) $\varepsilon=1.0$	Clean accuracy (%) $\varepsilon=0.5$	Clean accuracy (%) $\varepsilon=1.0$
Standard Training	✗	5.2	1.0	**74.4**	**74.4**
+ Ours (w/o adapt)	✓	56.2 (47.0)	44.2 (27.4)	73.0 (67.0)	68.8 (57.2)
+ Ours	✓	**57.0** (**50.4**)	**47.8** (34.0)	70.4 (**68.2**)	71.8 (**60.8**)
Adversarial Training [37]	✗	51.0	46.8	55.0	55.0
Randomized Smoothing [13]	✗	55.2 (48.6)	43.8 (**37.0**)	65.4 (66.8)	55.4 (57.0)
Carlini et al. [7]	✗	56.2 (49.2)	45.2 (33.2)	72.6 (67.4)	70.0 (57.8)

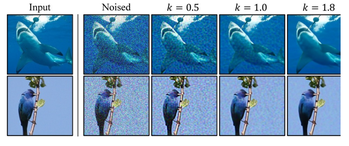
Robustifying other vision classifiers e.g., ResNet-50

σ	λ	ACR	Certified accuracy at ε (%) 0.0	0.25	0.5	0.75	1.0	1.25
0.25	0.0	0.270	49.6	39.0	30.0	19.6		
	0.001	0.280	50.6	40.2	30.8	20.2		
	0.01	**0.292**	**52.2**	**43.0**	**31.8**	**21.4**		
	0.1	0.290	51.8	42.8	31.2	20.6		
0.50	0.0	0.358	38.0	32.6	27.8	22.2	18.0	14.4
	0.001	0.379	40.4	35.0	30.0	24.6	**20.0**	14.6
	0.01	**0.394**	**44.0**	**37.0**	**30.8**	**25.2**	19.4	15.2
	0.1	0.390	43.4	37.0	30.4	24.2	**20.0**	**15.4**

Effect of λ

	Sample size n 25	50	100	200	400
Clean accuracy (%)	**58.0**	57.0	56.2	55.6	54.2
Robust accuracy (%)	26.0	29.4	31.4	33.0	**35.2**
Inference time (sec)	**0.64** ±0.09	0.92 ±0.10	1.39 ±0.08	2.56 ±0.13	5.14 ±0.12

Inference time



Correction factor is crucial