Robustness and Accuracy Could Be Reconcilable by (Proper) Definition

Tianyu Pang[1 2], Min Lin[2], Xiao Yang[1], Jun Zhu[1], Shuicheng Yan[2]

[1] Tsinghua University [2] Sea AI Lab

清華大學 Tsinghua University

sea connecting the dots | AI Lab

ICML | 2022

Trade-off between robustness and accuracy

Empirically:

Standard training
clean accuracy 95%
robust accuracy 0%

Adversarial training
clean accuracy 85%
robust accuracy 50%

Where the trade-off stems from?

Theoretically:

Exists in some simple cases

[Zhang et al. ICML 2019; Tsipras et al. ICLR 2019]

What is an **accurate** model?

An **accurate** model refers to the one with **low standard error**:

$$\mathbf{R}_{\text{Standard}} = \mathbb{E}_{p_d(x)}\left[\text{KL}\left(p_d(y|x) \middle\| p_\theta(y|x)\right)\right]$$

data distribution (p_d)   **model distribution** (p_θ)

Optimal solution: $p_{\theta^*}(y|x) = p_d(y|x)$

What is a **robust** model?

A **robust** model refers to the one with **low robust error**:

$$\mathbf{R}_{\text{Madry}} = \mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \text{KL}\left(p_d(y|x) \| p_\theta(y|x')\right)\right]$$

Optimal solution: $p_{\theta^*}(y|x) \neq p_d(y|x)$

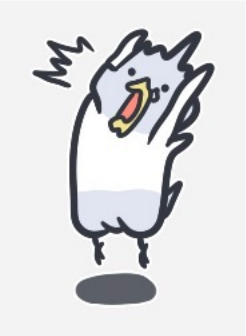
[Madry et al. ICLR 2018]

Trade-off naturally comes!

An optimally **accurate** model is **NOT** an optimally **robust** model

⬇ paradox

$p_d(y|x)$ is not an optimally **robust** model w.r.t. itself???!!!



Self-COnsistent Robust Error (SCORE)

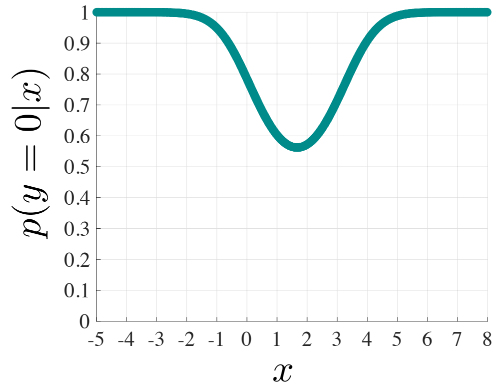
$$\mathbf{R}_{\text{SCORE}}(\theta) = \mathbb{E}_{p_d(x)} \left[\max_{x' \in B(x)} \text{KL}\left(p_d(y|x') \| p_\theta(y|x')\right) \right]$$

- Optimal solution: $p_{\theta^*}(y|x) = p_d(y|x)$

  (**self-consistency**, i.e., $p_d(y|x)$ is the optimally robust model w.r.t. itself under supervised learning framework)

- Keep the paradigm of robust optimization

Toy demo (self-consistency)

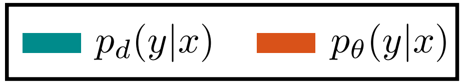


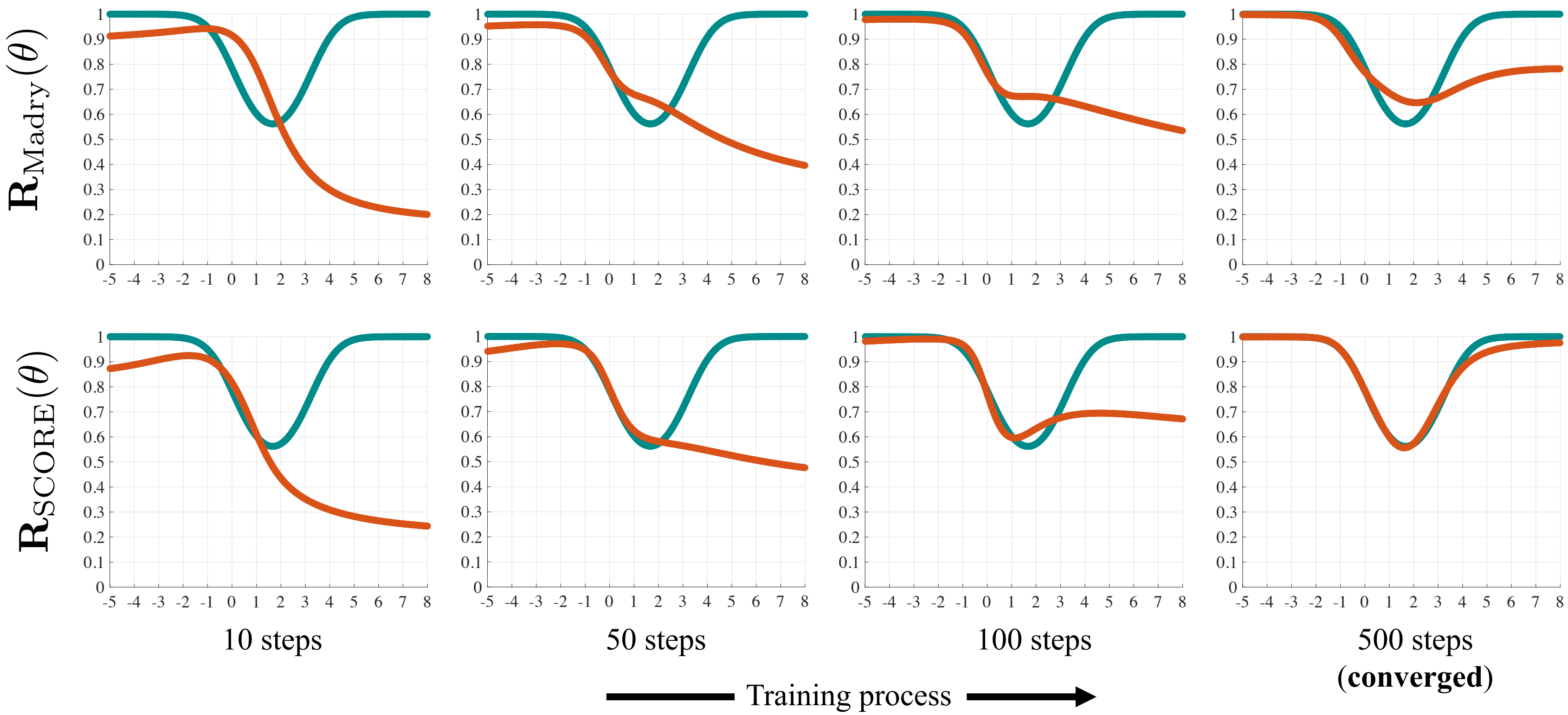
Construction of $p_d(x,y)$**:**

$p_d(y=0) = \frac{5}{6}, p_d(y=1) = \frac{1}{6};$

$p_d(x|y=0) \sim \mathcal{N}(-1,4);$

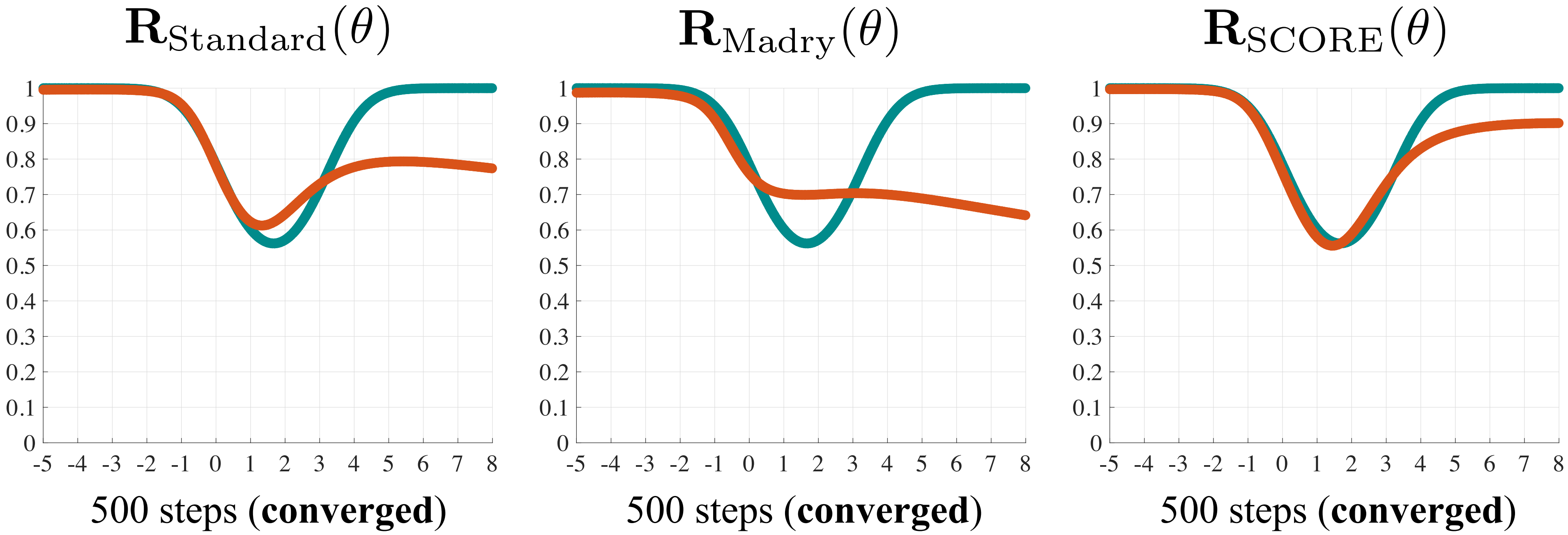
$p_d(x|y=1) \sim \mathcal{N}(1,1).$





60,000 training pairs, mimics the expectation form

Toy demo (robust optimization)



6 training pairs, mimics the finite-sample form

Standard error has the same optimal solution as SCORE, but does not enjoy robust optimization in finite-sample cases

In practice, how to optimize SCORE?

Directly applying first-order optimizers requires:

$$
\begin{aligned}
&\nabla_x \mathrm{KL}\left(p_d(y|x) \| p_\theta(y|x)\right) \\
=&\mathbb{E}_{p_d(y|x)}\Big[-\underbrace{\nabla_x \log p_\theta(y|x)}_{\text{model gradient}} + \left(\log \frac{p_d(y|x)}{p_\theta(y|x)}\right) \cdot \underbrace{\nabla_x \log p_d(y|x)}_{\text{data gradient}} \Big]
\end{aligned}
$$

- Initial experiments using **score matching** are of high variance
- More advanced score matching like **[Chao et al. ICLR 2022]** could be explored

Goodbye KL divergence!

Substitute KL divergence with any **distance metric** $\mathcal{D}$

$$\mathbf{R}^{\mathcal{D}}_{\text{Madry}}(\theta) = \mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \mathcal{D}\left(p_d(y|x) \| p_\theta(y|x')\right)\right];$$

$$\mathbf{R}^{\mathcal{D}}_{\text{SCORE}}(\theta) = \mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \mathcal{D}\left(p_d(y|x') \| p_\theta(y|x')\right)\right]$$

Upper and lower bounds for SCORE

Theorem 1:

$$|\mathbf{R}^{\mathcal{D}}_{\text{Madry}}(\theta) - C^{\mathcal{D}}| \leq \mathbf{R}^{\mathcal{D}}_{\text{SCORE}}(\theta) \leq \mathbf{R}^{\mathcal{D}}_{\text{Madry}}(\theta) + C^{\mathcal{D}},$$

$$\text{where } \underline{C^{\mathcal{D}}} = \mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \mathcal{D}\left(p_d(y|x) \| p_d(y|x')\right)\right]$$

intrinsic property of data distribution, indicates the (Madry) robust error of $p_d(y|x)$ itself

Upper and lower bounds for SCORE

Theorem 1:

$$|\mathbf{R}^{\mathcal{D}}_{\text{Madry}}(\theta) - C^{\mathcal{D}}| \leq \mathbf{R}^{\mathcal{D}}_{\text{SCORE}}(\theta) \leq \mathbf{R}^{\mathcal{D}}_{\text{Madry}}(\theta) + C^{\mathcal{D}},$$

$$\text{where } C^{\mathcal{D}} = \mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \mathcal{D}\left(p_d(y|x) \| p_d(y|x')\right)\right]$$

- **Upper bound:** minimizing SCORE without estimating $\nabla_x \log p_d(y|x)$
- **Lower bound:** indicates the overfitting phenomenon

Upper and lower bounds for SCORE

Theorem 1:

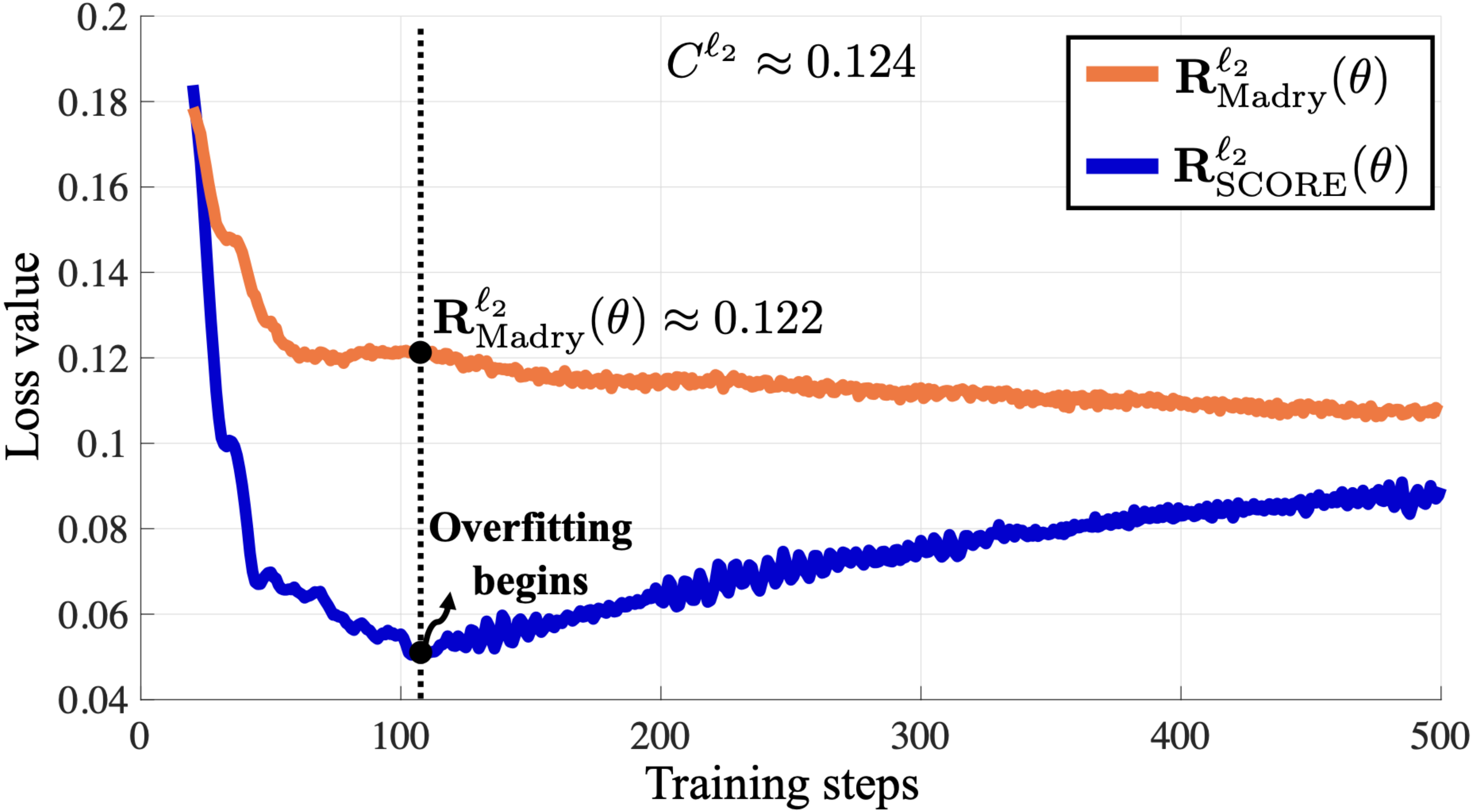
$$|\mathbf{R}^{\mathcal{D}}_{\mathrm{Madry}}(\theta) - C^{\mathcal{D}}| \leq \mathbf{R}^{\mathcal{D}}_{\mathrm{SCORE}}(\theta) \leq \mathbf{R}^{\mathcal{D}}_{\mathrm{Madry}}(\theta) + C^{\mathcal{D}},$$

$$\text{where } C^{\mathcal{D}} = \mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \mathcal{D}\left(p_d(y|x) \| p_d(y|x')\right)\right]$$

- **Upper bound:** minimizing SCORE without estimating $\nabla_x \log p_d(y|x)$
- **Lower bound:** indicates the overfitting phenomenon

Upper and lower bounds for SCORE

$\mathcal{D}$ is ℓ_2-distance : $\|A - B\|_2$



Back to KL divergence with new insights

Corollary 1:

$$|\mathbf{R}^{\ell_1}_{\mathrm{SCORE}}(\theta) - C^{\ell_1}| \leq \sqrt{2 \cdot \underline{\mathbf{R}_{\mathrm{Madry}}(\theta)}}$$

original **KL-based** robust error

Explaining overfitting and early-stopping

$$|\mathbf{R}^{\ell_1}_{\text{SCORE}}(\theta) - C^{\ell_1}| \leq \sqrt{2 \cdot \mathbf{R}_{\text{Madry}}(\theta)}$$

$$\Downarrow \quad \mathbf{R}^{\ell_1}_{\text{SCORE}}(\theta) = 0$$

indicates early-stopping

$$C^{\ell_1} \leq \sqrt{2 \cdot \mathbf{R}_{\text{Madry}}(\theta)} \Longrightarrow \boxed{\mathbf{R}_{\text{Madry}}(\theta) \geq \frac{\left(C^{\ell_1}\right)^2}{2}}$$

Explaining semantic gradients (for adversarial training)

Theorem 4: (under mild condition)

$$\mathbf{R}^{\ell_1}_{\text{SCORE}}(\theta) = \mathbf{R}^{\ell_1}_{\text{Standard}}(\theta) +$$

$$2\epsilon \cdot \mathbb{E}_{p_d(x)} \left[\left\| \nabla_x p_d(\mathcal{Y}_d(x)|x) - \nabla_x p_\theta(\mathcal{Y}_d(x)|x) \right\|_q \right] + o(\epsilon)$$

alignment between **model gradient** and **data gradient**

where $\mathcal{Y}_d(x) = \text{argmax}_y p_d(y|x)$

Explaining semantic gradients (for randomized smoothing)

Theorem 5:

$$\frac{d}{d\sigma}\mathbf{R}_G(\theta;\sigma) = \frac{1}{2}\mathbb{E}_{p_d^\sigma(x,y)}\left[\nabla_x \log p_\theta(y|x)^\top \nabla_x \log p_d^\sigma(x|y)\right]$$

$$\mathbf{R}_G(\theta;\sigma) = \mathbf{R}_G(\theta;0) + \sigma \cdot \frac{d}{d\sigma}\mathbf{R}_G(\theta;\sigma)\Big|_{\sigma=0} + o(\sigma)$$

cross-entropy (without augmentation)

gradient alignment

Table 4. Classification accuracy (%) on clean images and under AutoAttack. The results of our methods are in **bold**, and no clipping loss is executed. Here ‡ means *no CutMix applied*, following Rade and Moosavi-Dezfooli (2021). We use a batch size of 512 and train for 400 epochs due to limited resources, while a larger batch size of 1024 and training for 800 epochs are expected to achieve better performance.

Dataset	Method	Architecture	DDPM	Batch	Epoch	Clean	AutoAttack
CIFAR-10 ($\ell_\infty, \epsilon = 8/255$)	Rice et al. (2020)	WRN-34-20	✗	128	200	85.34	53.42
	Zhang et al. (2020)	WRN-34-10	✗	128	120	84.52	53.51
	Pang et al. (2021)	WRN-34-20	✗	128	110	86.43	54.39
	Wu et al. (2020)	WRN-34-10	✗	128	200	85.36	56.17
	Gowal et al. (2020)	WRN-70-16	✗	512	200	85.29	57.14
	Rebuffi et al. (2021)‡	WRN-28-10	1M	1024	800	85.97	60.73
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 3$)	WRN-28-10	1M	512	400	**88.61**	**61.04**
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 4$)	WRN-28-10	1M	512	400	**88.10**	**61.51**
	Rebuffi et al. (2021)‡	WRN-70-16	1M	1024	800	86.94	63.58
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 3$)	WRN-70-16	1M	512	400	**89.01**	**63.35**
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 4$)	WRN-70-16	1M	512	400	**88.57**	**63.74**
	Gowal et al. (2021)	WRN-70-16	100M	1024	2000	88.74	66.10
CIFAR-10 ($\ell_2, \epsilon = 128/255$)	Wu et al. (2020)	WRN-34-10	✗	128	200	88.51	73.66
	Gowal et al. (2020)	WRN-70-16	✗	512	200	90.90	74.50
	Rebuffi et al. (2021)‡	WRN-28-10	1M	1024	800	90.24	77.37
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 3$)	WRN-28-10	1M	512	400	**91.52**	**77.89**
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 4$)	WRN-28-10	1M	512	400	**90.83**	**78.10**
CIFAR-100 ($\ell_\infty, \epsilon = 8/255$)	Wu et al. (2020)	WRN-34-10	✗	128	200	60.38	28.86
	Gowal et al. (2020)	WRN-70-16	✗	512	200	60.86	30.03
	Rebuffi et al. (2021)‡	WRN-28-10	1M	1024	800	59.18	30.81
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 3$)	WRN-28-10	1M	512	400	**63.66**	**31.08**
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 4$)	WRN-28-10	1M	512	400	**62.08**	**31.40**
	Rebuffi et al. (2021)‡	WRN-70-16	1M	1024	800	60.46	33.49
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 3$)	WRN-70-16	1M	512	400	**65.56**	**33.05**
	+ **Ours** (KL $\rightarrow$ SE, $\beta = 4$)	WRN-70-16	1M	512	400	**63.99**	**33.65**

Thank you!

Paper: **https://arxiv.org/abs/2202.10103**

Code: **https://p2333.github.io/**

Tsinghua University

sea connecting the dots | AI Lab

ICML | 2022