# Robustness and Accuracy Could be Reconcilable

Tianyu Pang

Tsinghua University & Sea AI Lab


# Joint work with

Tsinghua University

sea | AI Lab
connecting the dots

Min Lin

Xiao Yang

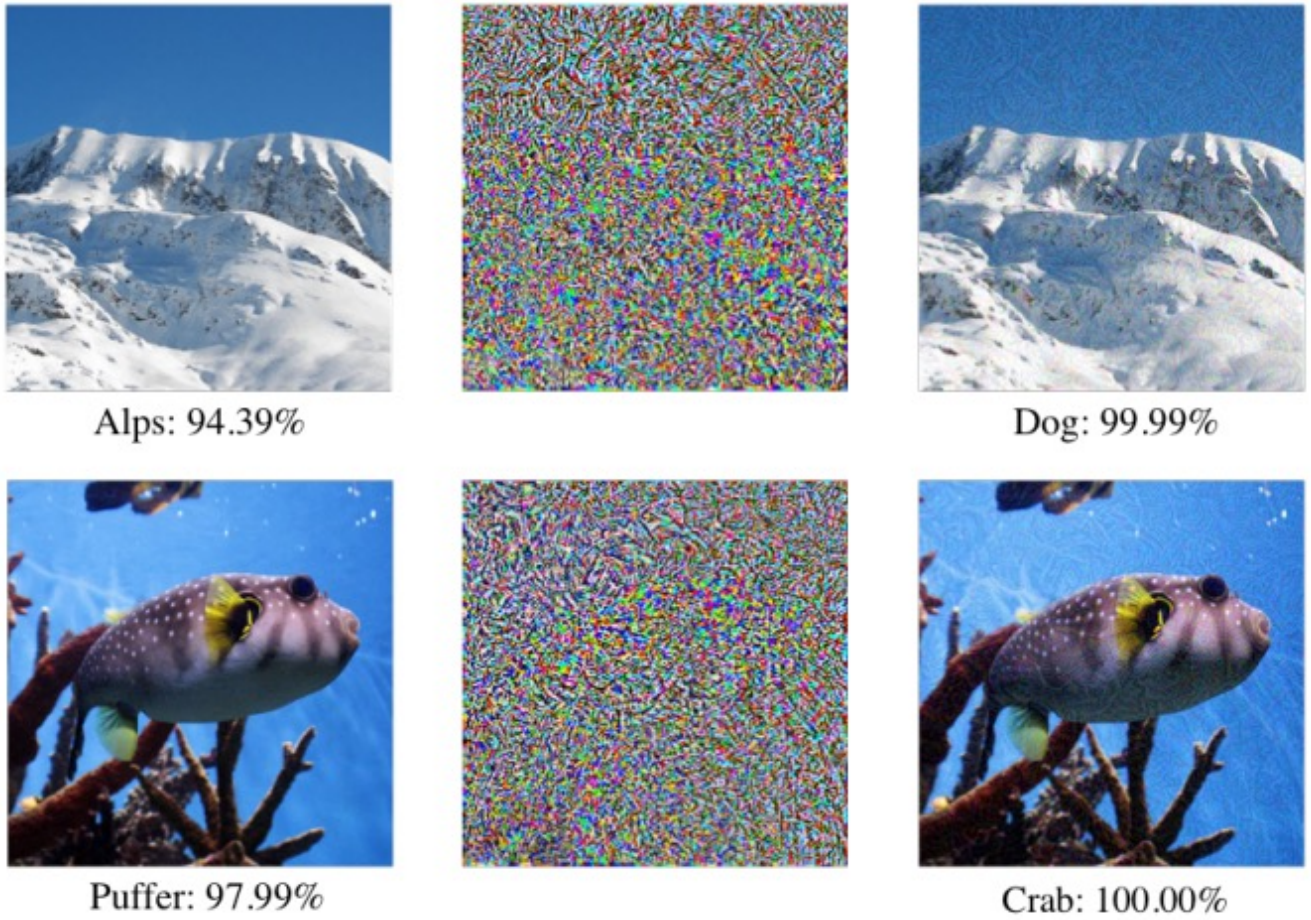
Yinpeng Dong

Hang Su

Jun Zhu

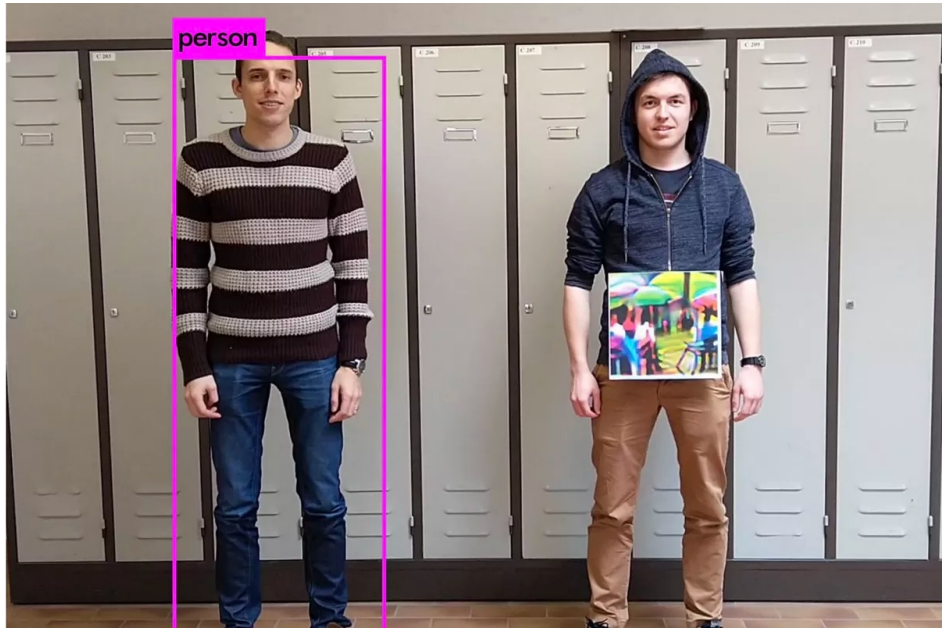

Shuicheng Yan

# Adversarial vulnerability




[Dong et al. CVPR 2018]

# Adversarial examples in physical world



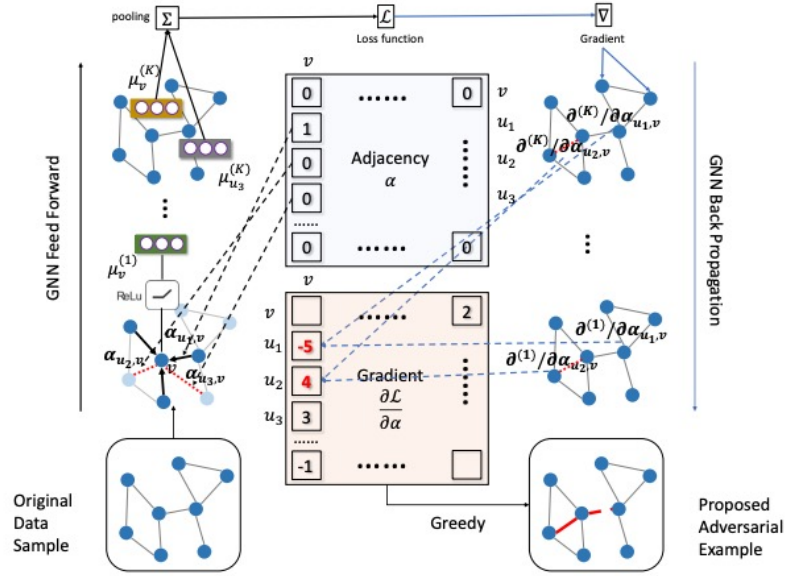




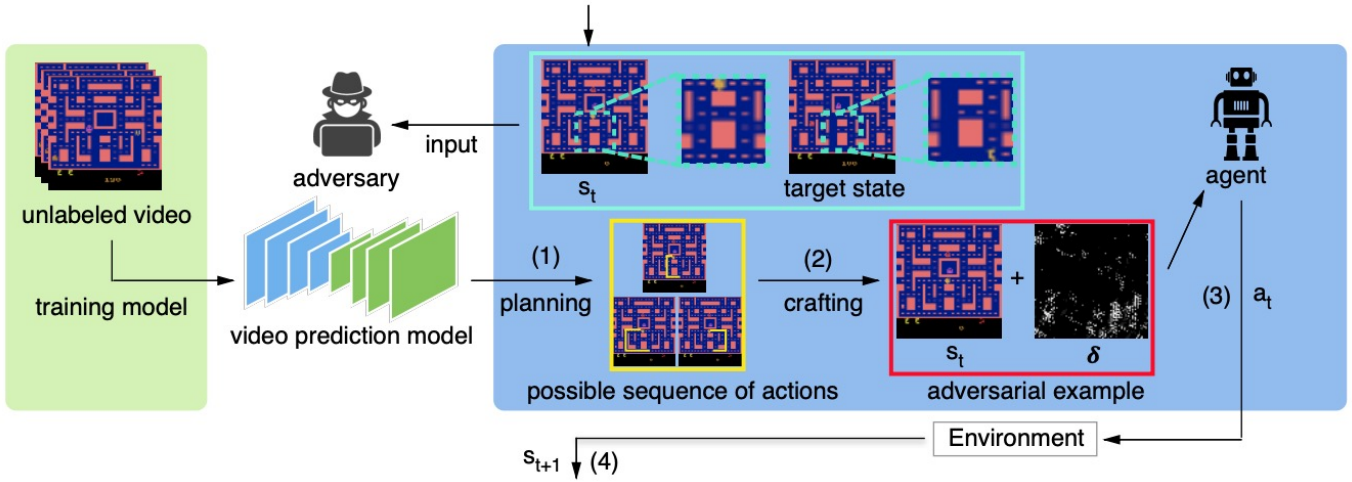
# Not only in computer vision

| | |
|---|---|
| **Movie Review (Positive (POS) ↔ Negative (NEG))** | |
| **Original (Label: NEG)** | The characters, cast in impossibly **contrived situations**, are **totally** estranged from reality. |
| **Attack (Label: POS)** | The characters, cast in impossibly **engineered circumstances**, are **fully** estranged from reality. |
| **Original (Label: POS)** | It cuts to the **knot** of what it actually means to face your **scares**, and to ride the **overwhelming** **metaphorical wave** that life wherever it takes you. |
| **Attack (Label: NEG)** | It cuts to the **core** of what it actually means to face your **fears**, and to ride the **big** **metaphorical wave** that life wherever it takes you. |
| **SNLI (Entailment (ENT), Neutral (NEU), Contradiction (CON))** | |
| **Premise** | Two small boys in blue soccer uniforms use a wooden set of steps to wash their hands. |
| **Original (Label: CON)** | The boys are in band **uniforms**. |
| **Adversary (Label: ENT)** | The boys are in band **garment**. |
| **Premise** | A child with wet hair is holding a butterfly decorated beach ball. |
| **Original (Label: NEU)** | The **child** is at the **beach**. |
| **Adversary (Label: ENT)** | The **youngster** is at the **shore**. |

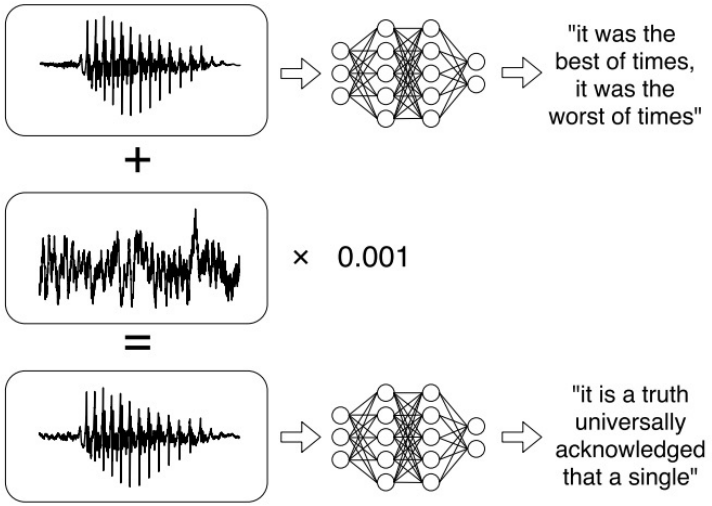
**NLP** (Jin et al. AAAI 2020)




**Graph** (Dai et al. ICML 2018)

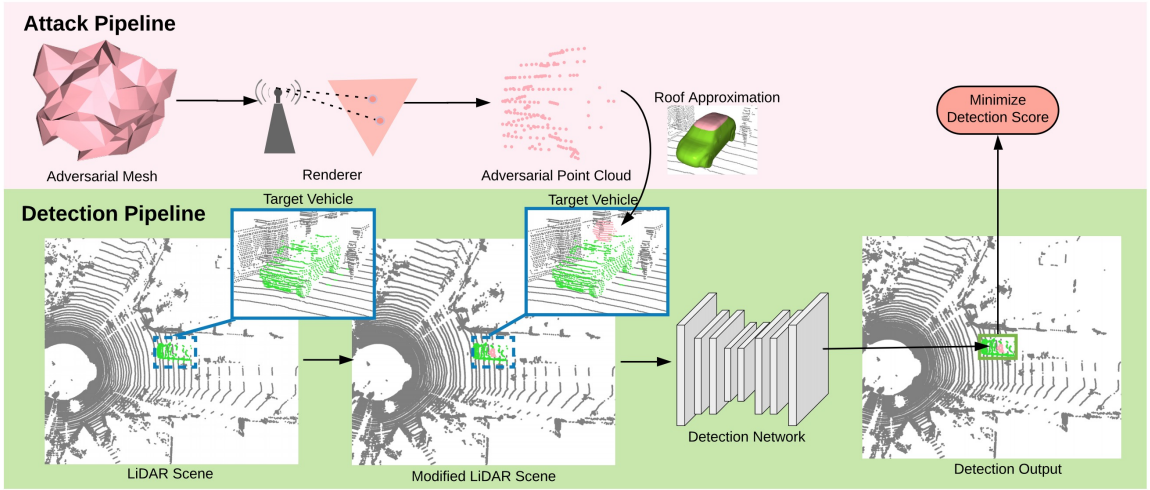



**Reinforcement Learning** (Lin et al. IJCAI 2017)

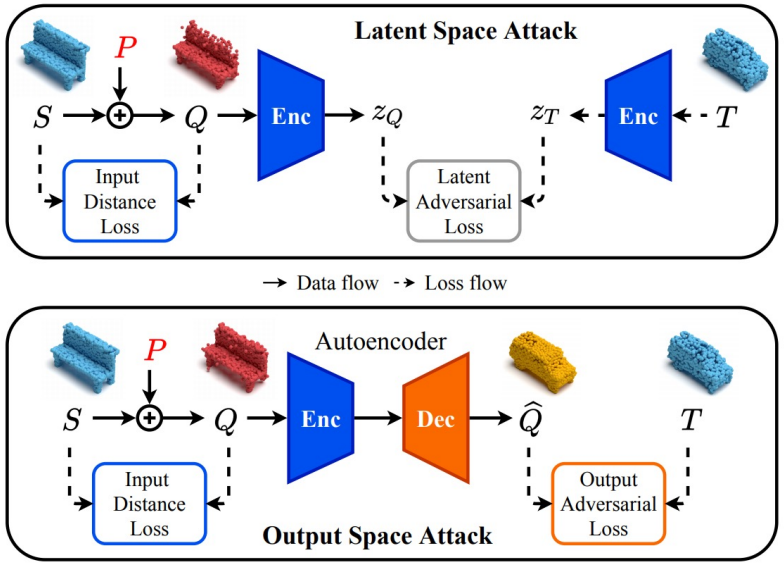



**Audio** (Carlini and Wagner. S&P 2018)

# Not only in computer vision



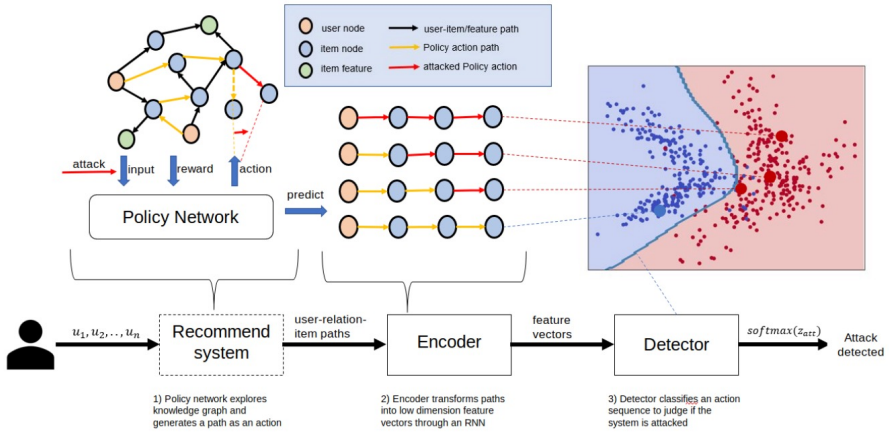

**LiDAR** (Tu et al. CVPR 2020)




**3D Point Cloud** (Lang et al. 2020)

| Class | Representative Example |
|---|---|
| VC | **OD:** Given a string a, what is the length of a.<br>**OO:** `(strlen a)`<br>**AD:** Given a string b, what is the length of b.<br>**AO:** `(strlen a)` |
| RR | **OD:** Given a number a, compute the product of **all** the numbers from 1 to a.<br>**OO:** `(invoke1 (lambda1 (if ( ≤ arg1 1 )1(*( self( -arg1 1 )) arg1 ))) a)`<br>**AD:** Given a number a, compute the product of the numbers from 1 to a.<br>**AO:** `( * a 1 )` |
| SR | **OD:** consider an array of numbers, what is reverse of elements in the given array that are odd<br>**OO:** `(reverse ( filter a ( lambda1 ( == ( % arg1 2 )1))))`<br>**AD:** consider an array of numbers, what equals reverse of elements in the given array that are odd<br>**AO:** `(reduce ( filter a ( lambda1 ( == ( % arg1 2 )1))))` |

**Code Generation** (Anand et al. 2021)




**Recommender System** (Cao et al. SIGIR 2020)

# Trade-off between robustness and accuracy

## Empirically:

**Standard training**
clean accuracy 95%
robust accuracy 0%

**Adversarial training**
clean accuracy 85%
robust accuracy 50%

## Theoretically:

**Exists in some simple cases**

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

# Trade-off between robustness and accuracy

**Empirically:**

**Standard training**
clean accuracy 95%
robust accuracy 0%

**Adversarial training**
clean accuracy 85%
robust accuracy 50%

**Theoretically:**

**Exists in some simple cases**

**Where the trade-off stems from?**

[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** **model distribution**

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**:

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

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

[Madry et al. ICLR 2018]

# Trade-off naturally comes!

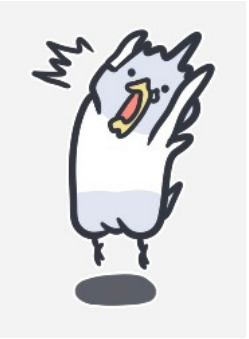
An optimally **accurate** model is **NOT** an optimally **robust** model

# 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???!!!



# Did we properly define robustness?

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

differentiable surrogate

**0-1 robust error:** $\mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathbf{1} \left( \mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x) \right) \right]$

$\mathcal{Y}_\theta(x) = \text{argmax}_y \; p_\theta(y|x)$

**hard label of model distribution (i.e., predicted label)**

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

**hard label of data distribution (i.e., true label)**

# Did we properly define robustness?

**0-1 robust error:** $\mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \mathbf{1}\left(\mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x)\right)\right]$

true label is invariant in $B(x)$ 

**Self-consistent 0-1 robust error:** $\mathbb{E}_{p_d(x)}\left[\max_{x' \in B(x)} \mathbf{1}\left(\mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x')\right)\right]$

- no assumption on $p_d(y|x)$
- allows for flexible $B(x)$

# Did we properly define robustness?

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

differentiable surrogate
($p_d(y|x)$ is invariant in $B(x)$)

$$\mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathbf{1} \left( \mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x) \right) \right]$$

true label is invariant in $B(x)$

$$\mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathbf{1} \left( \mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x') \right) \right]$$

# Did we properly define robustness?

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

differentiable surrogate
($p_d(y|x)$ is invariant in $B(x)$)

**Unreasonable**
(overcorrection towards smoothness)

$$\mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathbf{1} \left( \mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x) \right) \right]$$

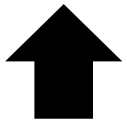
true label is invariant in $B(x)$

**Reasonable**

$$\mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathbf{1} \left( \mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x') \right) \right]$$

# Self-COnsistent Robust Error (SCORE)

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

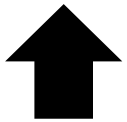


differentiable surrogate

$$\mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathbf{1} \left( \mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x') \right) \right]$$

# Self-COnsistent Robust Error (SCORE)

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



differentiable surrogate

$$\mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathbf{1} \left( \mathcal{Y}_\theta(x') \neq \mathcal{Y}_d(x') \right) \right]$$

$\mathbf{R}_{\mathrm{Madry}}(\theta)$ **invariance** $\Longrightarrow$ $\mathbf{R}_{\mathrm{SCORE}}(\theta)$ **equivariance**

# Self-COnsistent Robust Error (SCORE)

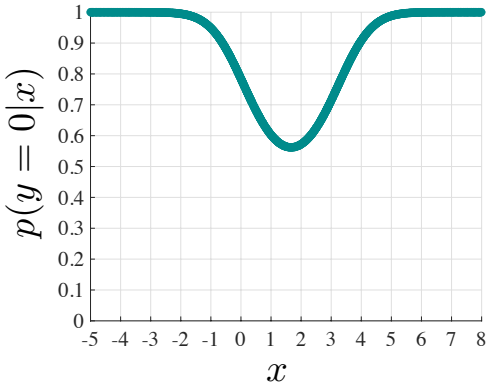
$$\mathbf{R}_{\mathrm{SCORE}}(\theta) = \mathbb{E}_{p_d(x)} \left[ \max_{x' \in B(x)} \mathrm{KL} \left( p_d(y|x') \middle|\middle| p_\theta(y|x') \right) \right]$$

- 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)

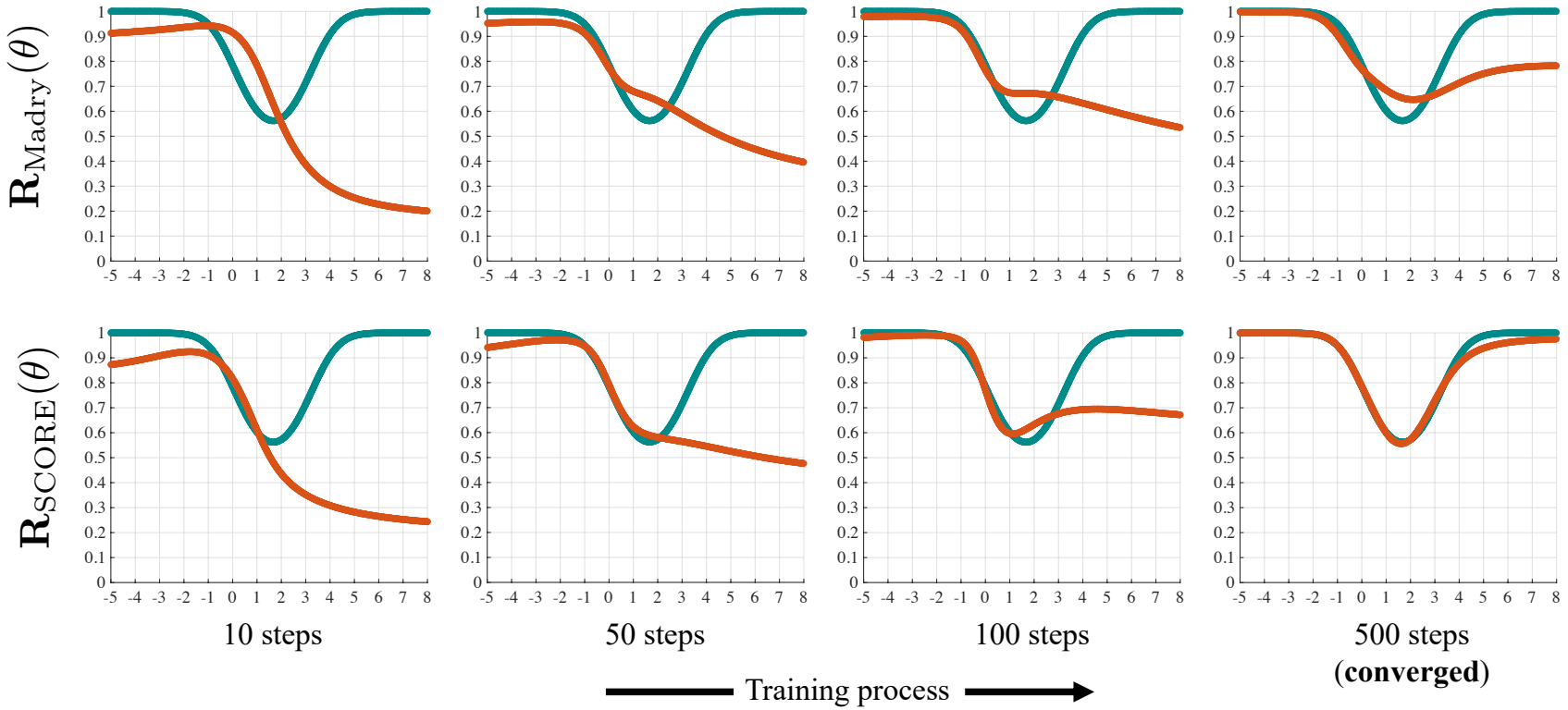



**Construction of $p_d(x, y)$:**

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

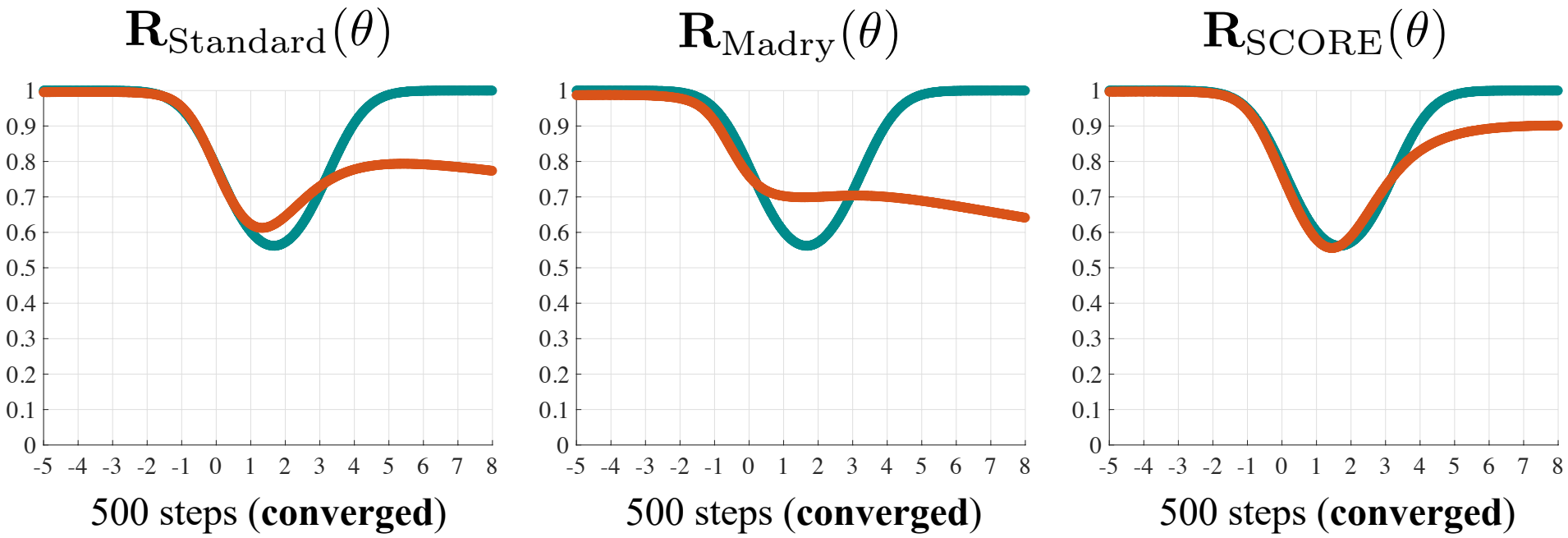
$p_d(x|y=0) \sim \mathcal{N}(-1, 4);$

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


**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:

$$\nabla_x \mathrm{KL}\left(p_d(y|x)\middle\|p_\theta(y|x)\right)$$

$$=\mathbb{E}_{p_d(y|x)}\Big[\underbrace{-\nabla_x \log p_\theta(y|x)}_{\textbf{model gradient}} + \left(\log \frac{p_d(y|x)}{p_\theta(y|x)}\right)\cdot \underbrace{\nabla_x \log p_d(y|x)}_{\textbf{data gradient}}\Big]$$

- 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** $\mathcal{D}$

- Symmetry: $\mathcal{D}(A\|B) = \mathcal{D}(B\|A)$
- Triangle inequality: $\mathcal{D}(A\|C) \leq \mathcal{D}(A\|B) + \mathcal{D}(B\|C)$

Typical distance metrics include $\|A - B\|_p$

# Goodbye KL divergence!

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

**does not satisfy**

- Symmetry: $\mathcal{D}(A\|B) = \mathcal{D}(B\|A)$
- Triangle inequality: $\mathcal{D}(A\|C) \leq \mathcal{D}(A\|B) + \mathcal{D}(B\|C)$

Typical distance metrics include $\|A - B\|_p$

# Goodbye KL divergence!

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

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

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

# Upper and lower bounds for SCORE

## Theorem 1:

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

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

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

# Upper and lower bounds for SCORE

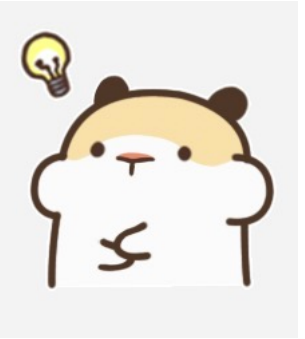
## Theorem 1:

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

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

- **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:

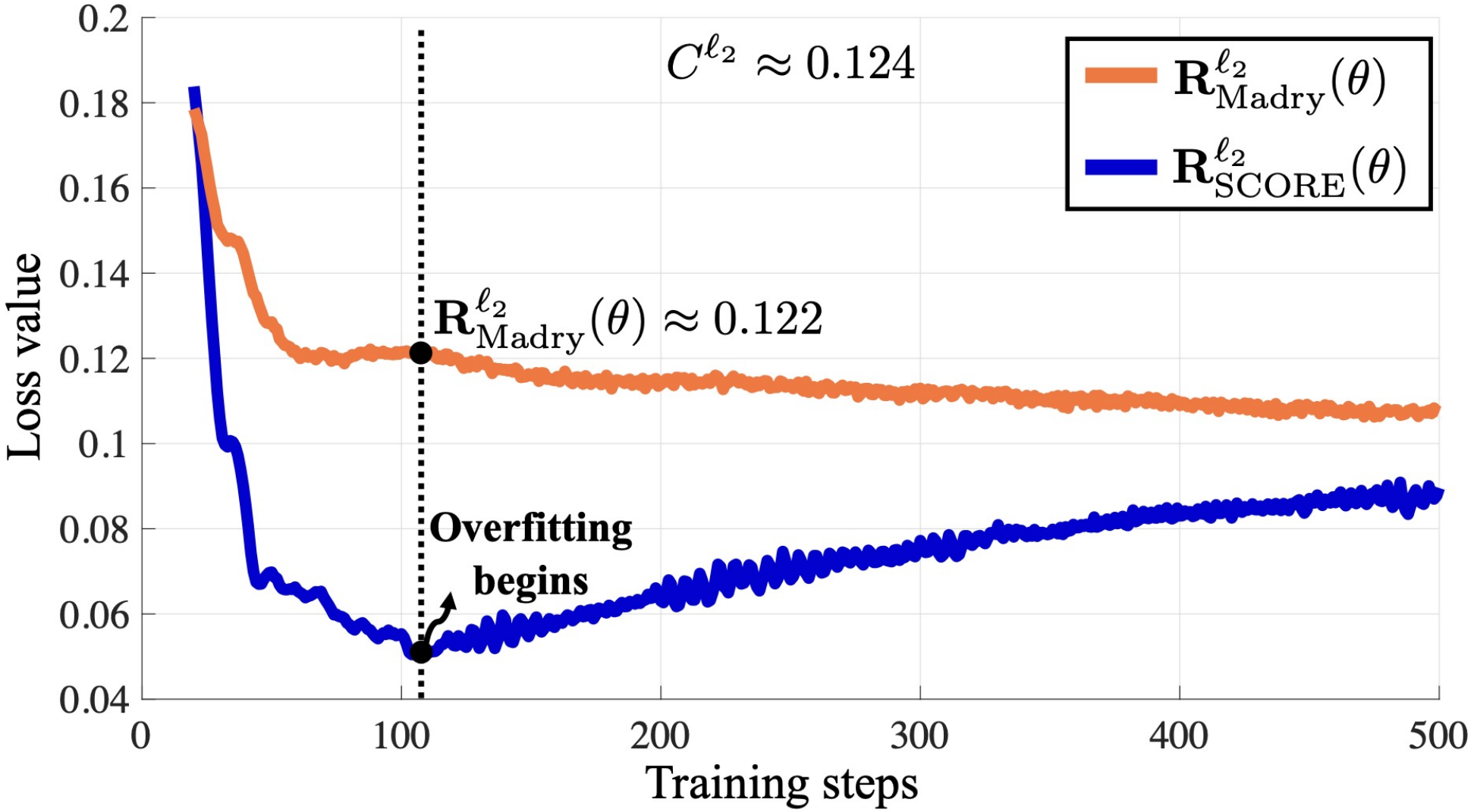
$$|\mathbf{R}^{\mathcal{D}}_{\text{Madry}}(\theta) - C^{\mathcal{D}}| \leq \mathbf{R}^{\mathcal{D}}_{\text{SCORE}}(\theta) \leq \mathbf{R}^{\mathcal{D}}_{\text{Madry}}(\theta) + C^{\mathcal{D}},$$

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

- **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 $\ell_2$-distance : $\|A - B\|_2$



# Extending to composite function of distance

## Theorem 2:

$$|\mathbf{R}^{\mathcal{D}}_{\mathrm{SCORE}}(\theta) - C^{\mathcal{D}}| \leq \phi^{-1}\left(\mathbf{R}^{\phi\circ\mathcal{D}}_{\mathrm{Madry}}(\theta)\right)$$

$\phi(\cdot)$ is a **monotonically increasing convex** function, e.g., square function

Examples of $\phi \circ \mathcal{D}$ include **squared error (SE)** and **JS-divergence**

# Composite function of distance empirically works better

| Loss | Alias | $l.r. = 0.1$ | | $l.r. = 0.05$ | | $l.r. = 0.01$ | |
|---|---|---|---|---|---|---|---|
| | | Clean | PGD | Clean | PGD | Clean | PGD |
| $\|P - Q\|_2$ | $\ell_2$-dis. | 75.91 | 52.16 | 77.98 | 52.74 | 78.45 | 51.13 |
| $\|P - Q\|_1$ | $\ell_1$-dis. | 58.51 | 43.87 | 64.88 | 46.77 | 70.02 | 47.76 |
| $\|P - Q\|_\infty$ | $\ell_\infty$-dis. | 58.34 | 43.71 | 59.75 | 45.02 | 65.65 | 46.36 |
| $\sqrt{\text{JS}(P\|Q)}$ | JS-dis. | 53.06 | 40.08 | 55.27 | 41.86 | 68.50 | 46.49 |
| $\text{JS}(P\|Q)$ | JS-div. | 79.41 | 51.75 | 81.27 | 51.85 | 80.12 | 49.10 |
| $\text{KL}(P\|Q)$ | KL-div. | 82.74 | 53.02 | 83.21 | 51.52 | 82.65 | 47.45 |
| $\|P - Q\|_1^2$ | - | 79.87 | 50.96 | 81.49 | 52.00 | 81.26 | 47.51 |
| $\|P - Q\|_2^2$ | SE | 80.59 | 54.63 | 83.38 | 54.01 | 81.43 | 51.13 |

# PGD-AT and TRADES are equivalent (under $\mathcal{D}$)

Theorem 3: For $\beta \geq 1$

$$\mathbf{R}^{\mathcal{D}}_{\mathrm{Madry}}(\theta) \leq \mathbf{R}^{\mathcal{D}}_{\mathrm{TRADES}}(\theta; \beta) \leq (1 + 2\beta) \cdot \mathbf{R}^{\mathcal{D}}_{\mathrm{Madry}}(\theta)$$

- Similar as the equivalence among $\ell_p$-norms
- Induce the same topology of loss landscapes in parameter space [Conrad 2018]

# Back to KL divergence with new insights

A bridge between **KL divergence** and **distance metrics**:
**Pinsker's inequality**

$$\frac{1}{2}\|P - Q\|_1^2 \leq \mathrm{KL}(P||Q)$$

[Csiszar and Korner 2011]

# Back to KL divergence with new insights

## Corollary 1:

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

original **KL-based** robust error

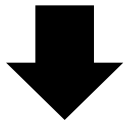
# Explaining overfitting and early-stopping

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

minimized in previous work

# Explaining overfitting and early-stopping

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



$$\mathbf{R}^{\ell_1}_{\mathrm{SCORE}}(\theta) = 0$$

condition for optimal solution

# Explaining overfitting and early-stopping

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

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

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

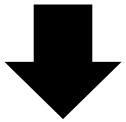
indicates early-stopping

# Explaining overfitting and early-stopping

$$\mathbf{R}^{\ell_1}_{\mathrm{SCORE}}(\theta) = 0 \Rightarrow p_\theta(y|x) = p_d(y|x)$$

# Explaining overfitting and early-stopping

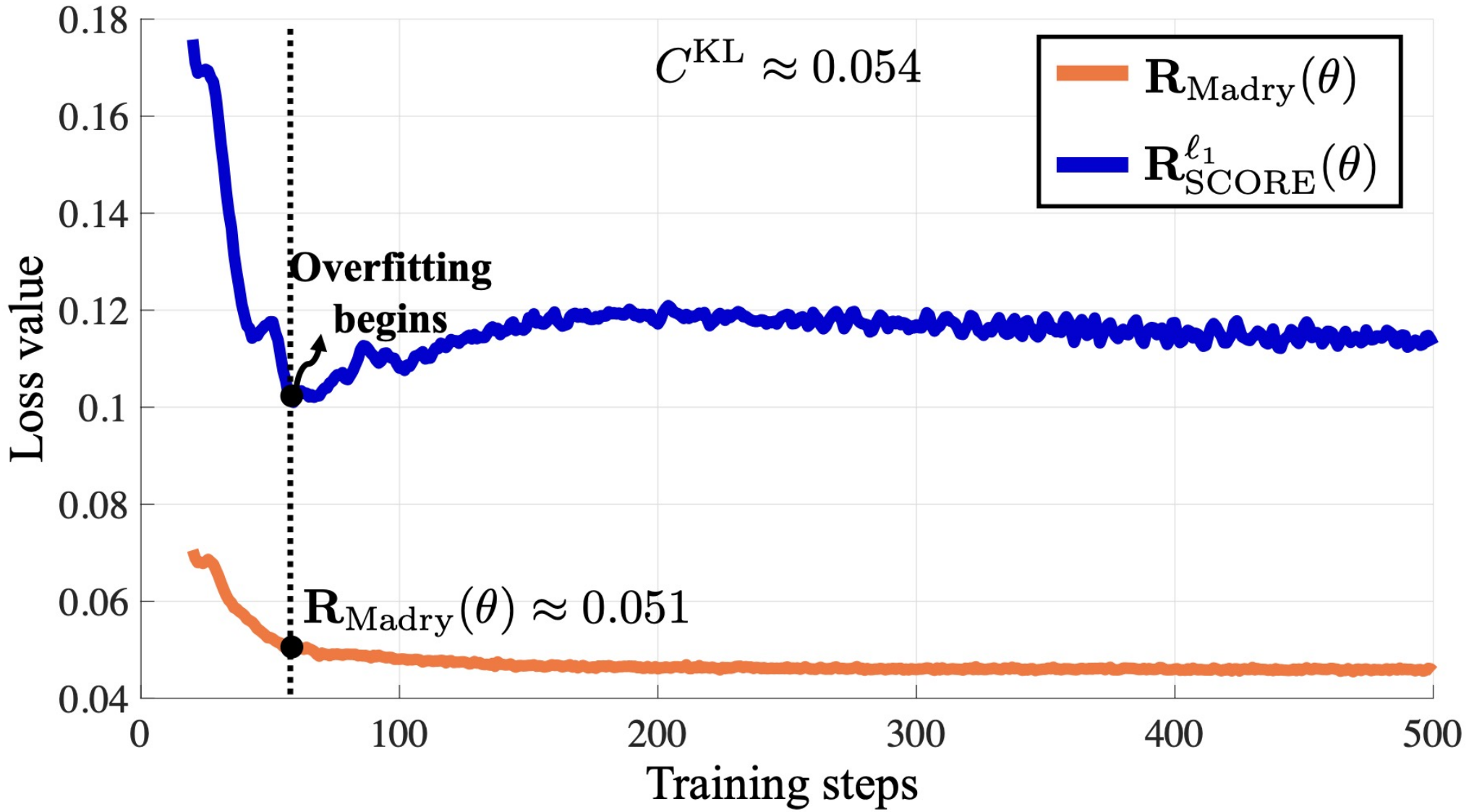
$$\mathbf{R}^{\ell_1}_{\mathrm{SCORE}}(\theta) = 0 \Rightarrow p_\theta(y|x) = p_d(y|x)$$



$$\mathbf{R}_{\mathrm{Madry}}(\theta) = C^{\mathrm{KL}} \geq \frac{(C^{\ell_1})^2}{2}$$

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

# Explaining overfitting and early-stopping



# Explaining semantic gradients (for adversarial training)

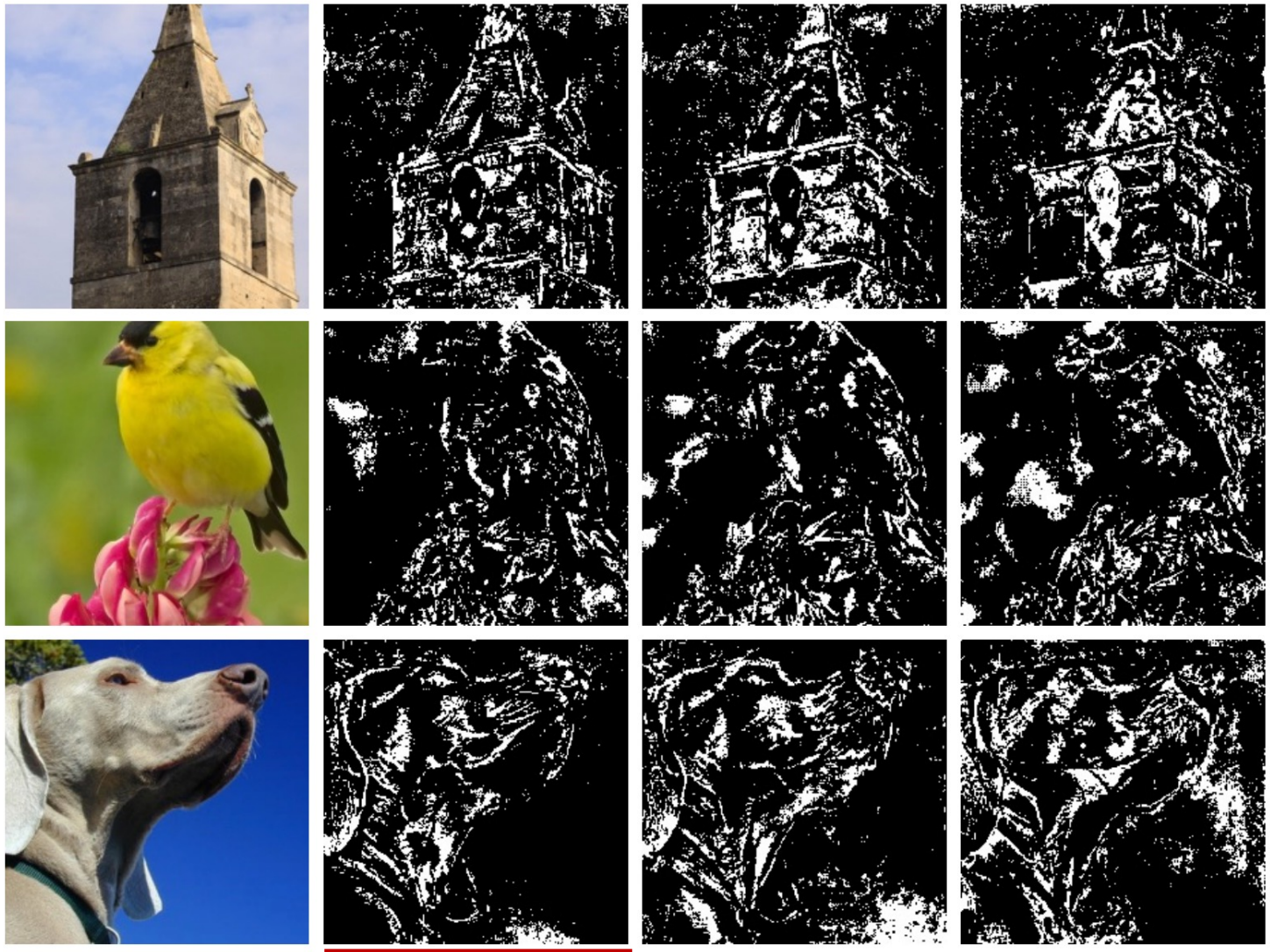
**Theorem 4:** (under mild condition)

$$\mathbf{R}^{\ell_1}_{\mathrm{SCORE}}(\theta) = \mathbf{R}^{\ell_1}_{\mathrm{Standard}}(\theta) + 2\epsilon \cdot \mathbb{E}_{p_d(x)}\left[\|\nabla_x p_d(\mathcal{Y}_d(x)|x) - \nabla_x p_\theta(\mathcal{Y}_d(x)|x)\|_q\right] + o(\epsilon)$$

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

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

# Explaining semantic gradients (for adversarial training)



Clean images | $\nabla_x \log p_\theta(\mathcal{Y}_d(x)|x)$ | $\nabla_x \log p_\theta(x, \mathcal{Y}_d(x))$ | $-\nabla_x \log p_\theta(x)$

# Explaining semantic gradients (for randomized smoothing)

**Gaussian augmented** cross-entropy loss:

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

$$\text{where } p_d^\sigma(x,y) = \mathbb{E}_{\mathcal{N}(\omega;0,I)}\left[p_d(x-\sqrt{\sigma}\omega, y)\right]$$


[Kaur et al. NeurIPS 2019 Workshop]

# Explaining semantic gradients (for randomized smoothing)

## Theorem 5:

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

# 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**

# Empirical performance

*Table 2.* Classification accuracy (%) on clean images and under AutoAttack ($\ell_\infty$, $\epsilon = 8/255$). Here we use ResNet-18 trained by PGD-AT or TRADES on CIFAR-10, using KL divergence or squared error (SE) as the loss function. Clipping loss is executed at every training step, compatible with early-stopping. We average the results over five runs and report the mean ± standard deviation.

| Method | Loss | Clip | Clean | AutoAttack |
|---|---|---|---|---|
| PGD-AT | KL div. | - | 82.46 ± 0.41 | 48.39 ± 0.14 |
| | SE | ✗ | 82.13 ± 0.14 | 49.41 ± 0.27 |
| | SE | ✓ | **82.80 ± 0.16** | **49.63 ± 0.17** |
| TRADES | KL div. | - | 81.47 ± 0.12 | 49.14 ± 0.16 |
| | SE | ✗ | 83.50 ± 0.05 | 49.44 ± 0.35 |
| | SE | ✓ | **83.75 ± 0.14** | **49.57 ± 0.28** |

*Table 3.* Classification accuracy (%) on clean images and under AutoAttack ($\ell_\infty$, $\epsilon = 8/255$). The model is WRN-28-10 (SiLU), following the training pipeline in Rebuffi et al. (2021) and using 1M DDPM generated data. KL divergence is substituted with the SE function in TRADES, and no clipping loss is executed.

| Dataset | $\beta$ | Clean | AutoAttack |
|---|---|---|---|
| **CIFAR-10** | 6 | 86.64 ± 0.13 | 60.78 ± 0.16 |
| | 5 | 87.19 ± 0.20 | 61.05 ± 0.11 |
| | 4 | 87.89 ± 0.19 | 61.11 ± 0.27 |
| | 3 | 88.60 ± 0.13 | 60.89 ± 0.09 |
| | 2 | 89.28 ± 0.15 | 60.13 ± 0.21 |
| **CIFAR-100** | 4 | 61.94 ± 0.13 | 31.21 ± 0.12 |
| | 3 | 63.12 ± 0.37 | 31.01 ± 0.09 |

*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 → SE, $\beta = 3$) | WRN-28-10 | 1M | 512 | 400 | **88.61** | **61.04** |
| | + **Ours** (KL → 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 → SE, $\beta = 3$) | WRN-70-16 | 1M | 512 | 400 | **89.01** | **63.35** |
| | + **Ours** (KL → 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 → SE, $\beta = 3$) | WRN-28-10 | 1M | 512 | 400 | **91.52** | **77.89** |
| | + **Ours** (KL → 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 → SE, $\beta = 3$) | WRN-28-10 | 1M | 512 | 400 | **63.66** | **31.08** |
| | + **Ours** (KL → 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 → SE, $\beta = 3$) | WRN-70-16 | 1M | 512 | 400 | **65.56** | **33.05** |
| | + **Ours** (KL → SE, $\beta = 4$) | WRN-70-16 | 1M | 512 | 400 | **63.99** | **33.65** |

# Wait! Why does empirical trade-off still exit?

SCORE makes sure that there is no trade-off for the **optimal solution**, so the remain challenge leaves to **more efficient learning processes**.

- Beyond MLE (KL divergence), resorting to more advanced score matching methods (Fisher divergence) to train SCORE
- Extra data; robust architectures; training tricks

# Thank you!

**Robustness and Accuracy Could Be Reconcilable by (Proper) Definition**
https://arxiv.org/abs/2202.10103

**Bag of Tricks for Adversarial Training**
https://arxiv.org/abs/2010.00467 (ICLR 2021)

**Two Coupled Rejection Metrics Can Tell Adversarial Examples Apart**
https://arxiv.org/abs/2105.14785 (CVPR 2022)

https://p2333.github.io/