Rethinking the Min-max Problem for Adversarial Robustness

Yisen Wang
Peking University

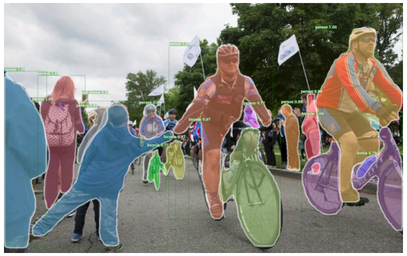
yisen.wang@pku.edu.cn
https://yisenwang.github.io/

Guest Lecture for CS498@UIUC
Mar 17, 2021

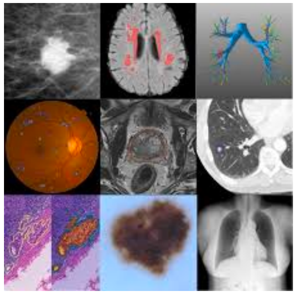
ML is Everywhere

Image classification

Speech recognition

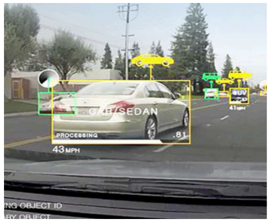


Object detection



Medical diagnosis

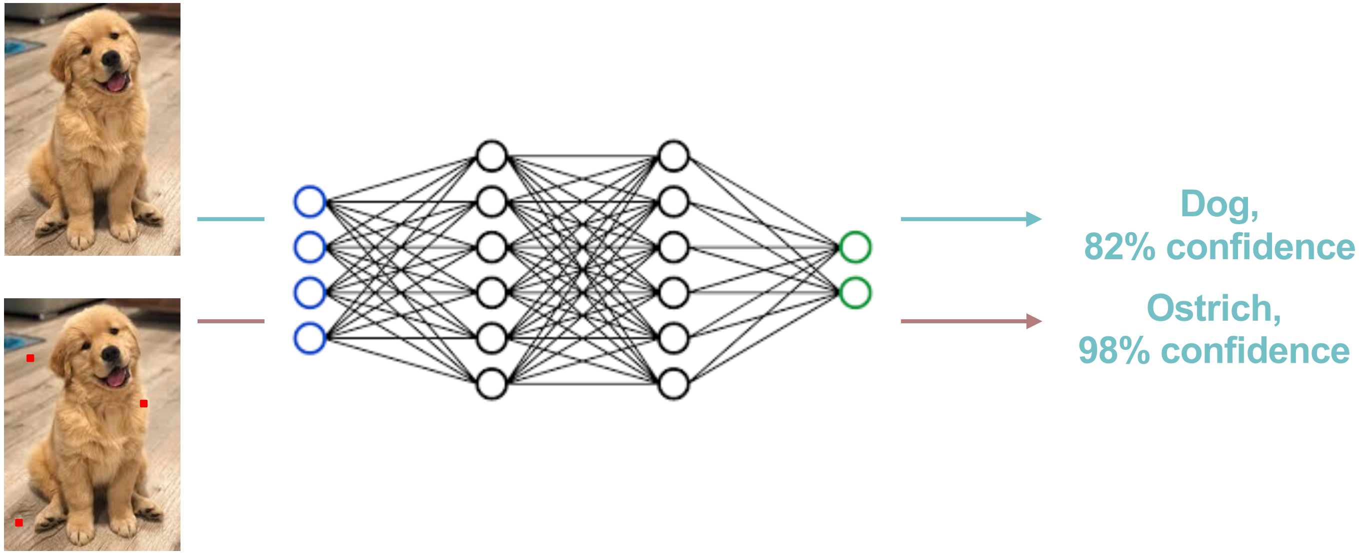
Machine Learning

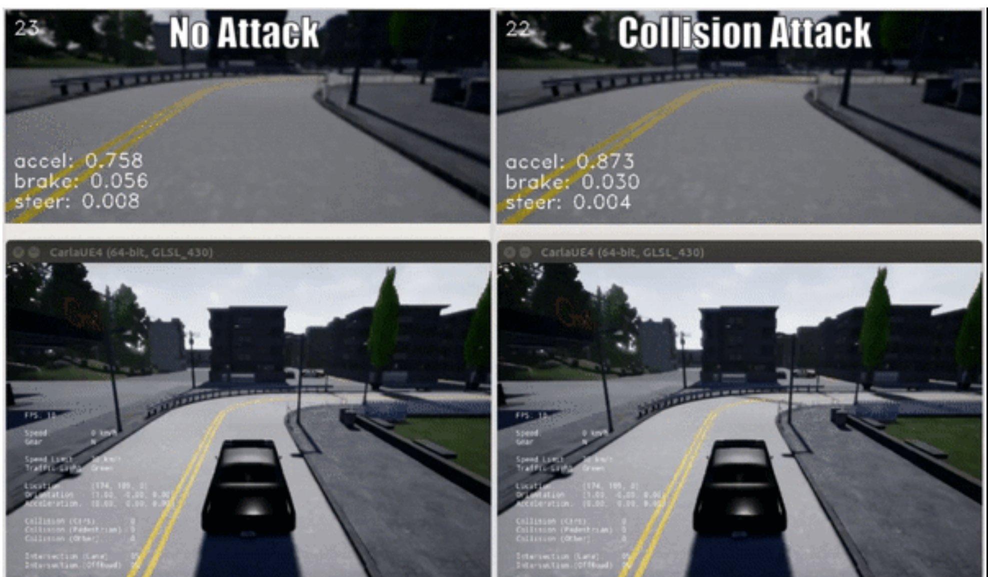


Autonomous driving

Playing games

However


Dog,
82% confidence
Ostrich,
98% confidence


No Attack
accel: 0.758
brake: 0.056
steer: 0.008
Collision Attack
accel: 0.873
brake: 0.030
steer: 0.004

Are we doomed?
(Is ML inherently not reliable?)

NO! But we need to re-think how we do ML

(adversarial aspects = stress-testing our solutions)

Adversarial Example

D_{train}: training data
x_i: training sample
y_i: class label
L: loss function
f_θ: model

Model training: $\min_{\theta} \sum_{(x_i, y_i) \in D_{train}} L(f_\theta(x_i), y_i)$

Adversarial attack: $\underbrace{\max_{x'} L(f_\theta(x'), y)}_{\text{increase error}} \;\; st. \underbrace{\|x' - x\|_p \leq \epsilon}_{\text{small change}} \text{ for } \underbrace{\mathrm{x} \in D_{test}}_{\text{test time attack}}$

$$\|x' - x\|_\infty \leq \epsilon = \frac{8}{255} \approx 0.031$$

- Fast Gradient Sign Method (FGSM) (*Goodfellow et al., 2014*):

$$x' = x + \varepsilon \cdot \operatorname{sign} \nabla_x L(f_\theta(x), y)$$

x': adv examples

- Projected Gradient Descent (PGD) is a iterative version of FGSM (*Madry et al., 2018*)

$$x'^{(k+1)} = \Pi_\epsilon \left(x'^{(k)} + \alpha \cdot \operatorname{sign} \nabla_x L\left(f_\theta\left(x'^{(k)}\right), y\right)\right)$$

How to obtain adversarially robust models?

Adversarial Training

Adversarial training is a **min-max optimization** process:

$$\min_{\boldsymbol{\theta}} \frac{1}{n}\sum_{i=1}^{n} \overbrace{\max_{\|x_i'-x_i\|_p \leq \epsilon}}^{\text{attacking}} L(f_\theta(x_i'), y_i)$$

L: loss, f_θ: model, x_i: clean example, y_i: class,
x_i': adversarial example.

1. Inner Maximization:

– This is to generate adversarial examples, by maximizing the loss L.

– It is a constrained optimization problem: $\|x_i' - x_i\|_p \leq \epsilon$.

2. Outer Minimization:

– A typical process to train a model, but on adversarial examples x_i' generated by the inner maximization.

Convergence Score of the Maximization

Question: How well the inner maximization is solved?

Definition (First-Order Stationary Condition (FOSC))

Given a data sample $x^0 \in X$, let x^k be an intermediate example found at the $\mathrm{k^{th}}$ step of the inner maximization. The First-Order Stationary Condition of x^k is

$$c\left(x^k\right) = \max_{x \in \chi} \left\langle x - x^k, \nabla_x f(\boldsymbol{\theta}, x^k) \right\rangle,$$

where $\chi = \{x \,|\, \|x - x^0\|_\infty \leq \epsilon\}$ is the input domain of the ϵ-ball around normal example x^0, $f\left(\boldsymbol{\theta}, x^k\right) = \ell\left(h_{\boldsymbol{\theta}}\left(x^k\right), y\right)$, and $\langle \cdot \rangle$ is the inner product.

FOSC:

- A smaller value of $c\left(x^k\right)$ indicates a better solution of the inner maximization, or equivalently, better convergence quality of the adversarial example x^k.
- To help Danskin's Theorem hold.

Yisen Wang*, *Xingjun Ma**, *et al.*, On the Convergence and Robustness of Adversarial Training. ***ICML 2019***.

Convergence Theorem

Theorem 1

Under certain assumptions, let $\Delta = L_S(\theta^0) - \min_\theta L_S(\theta)$. *If the step size of the outer minimization is set to* $\eta_t = min\left(\frac{1}{L}, \sqrt{\frac{\Delta}{L\sigma^2 T}}\right)$. *Then the output of* ***Adversarial Training*** *satisfies*

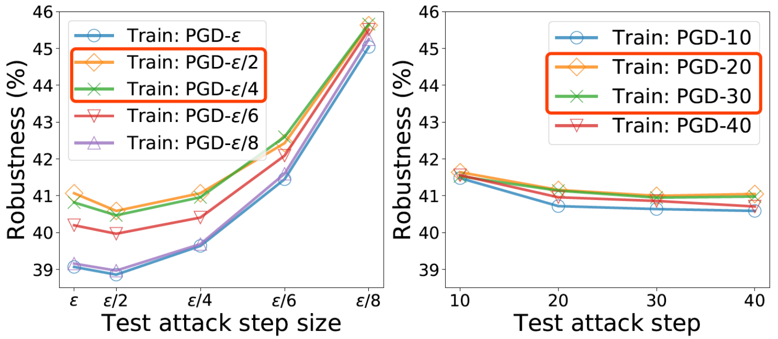
$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla L_S(\theta^t)\|_2^2] \leq 4\sigma\sqrt{\frac{L\Delta}{T}} + \frac{5L_{\theta x}^2\delta}{\mu},$$

where $L = \left(\frac{L_{\theta x}L_{\theta x}}{\mu} + L_{\theta\theta}\right)$.

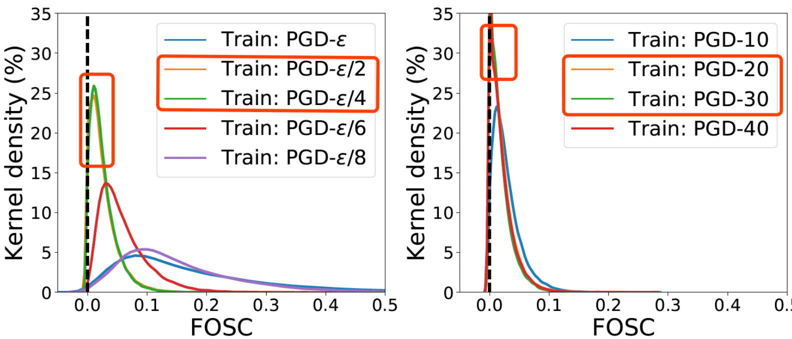
- Inner maximization: FOSC $\leq \delta$, adversarial training can converge to a first-order stationary point up to a precision of $\frac{5L_{\theta x}^2\delta}{\mu}$

- If δ is sufficiently small such that $\frac{5L_{\theta x}^2\delta}{\mu}$ small enough, adversarial training can find a robust model θ^T.

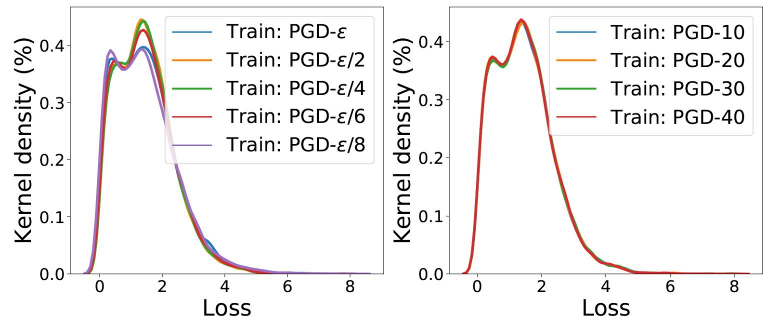
Why do we need FOSC?



(a) Robustness vs. Step size (b) Robustness vs. Step number



(c) FOSC vs. Step size (d) FOSC vs. Step number



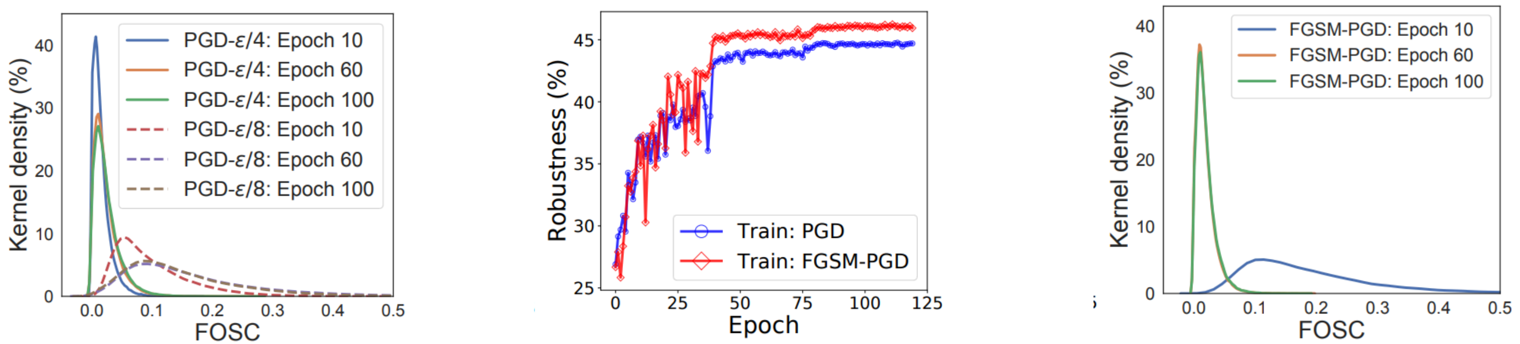
(e) Loss vs. Step size (f) Loss vs. Step number

Adversarial Training with different settings for PGD-based inner maximization.

- **PGD step size**: PGD-$\frac{\epsilon}{2}$ / PGD-$\frac{\epsilon}{4}$ produces the best robustness, their FOSC values are also concentrated around 0.
- **PGD step number**: similar robustness, with PGD-20/30 are slightly better, reflected by the distribution of FOSC.
- **Loss distributions** are similar for different robustness.

FOSC is a good and reliable indicator of the final robustness

FOSC View of Adversarial Training



(a) FOSC (b) Robustness (c) FOSC

- Standard adversarial training **overfits** to strong PGD adversarial examples at the **early stage**.
- Simply use **weak attack FGSM** at the **early stage** can improve robustness.
- Improvement in robustness is also reflected in FOSC distribution.

The principle behind warm-up techniques

Warm-up is a method to solve max better,
is there other options?

Rethinking the Robust Generalization Gap

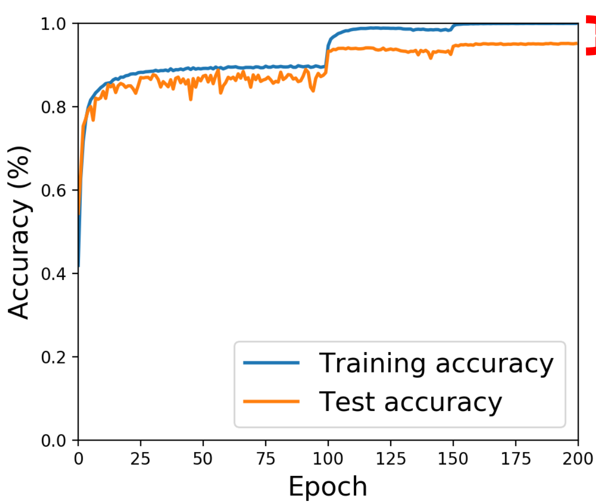
Adversarial training is a **min-max optimization** process:

$$\min_{\boldsymbol{\theta}} \frac{1}{n}\sum_{i=1}^{n} \max_{\|x_i'-x_i\|_p \le \epsilon} L(f_\theta(x_i'), y_i)$$

can be rewritten as

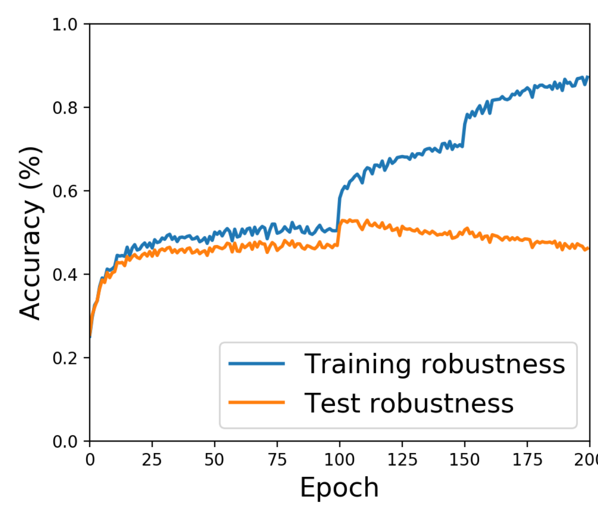
$$\min_{\mathbf{w}} \rho(\mathbf{w}), \quad \text{where} \quad \rho(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i'-\mathbf{x}_i\|_p \le \epsilon} \ell(f_{\mathbf{w}}(\mathbf{x}_i'), y_i),$$

Standard training



Standard gap

Adversarial training

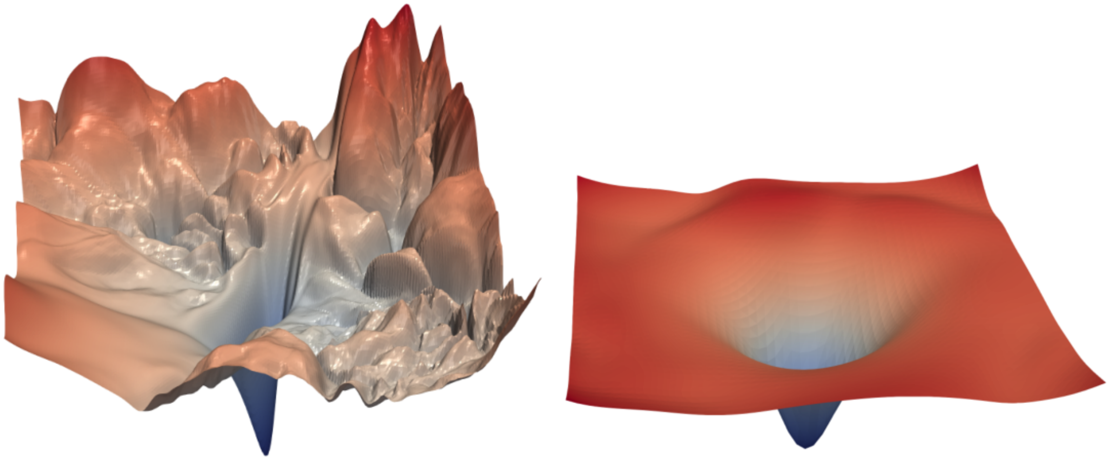


Robust gap

*Dongxian Wu**, Shu-Tao Xia,*** *Yisen Wang*[#]**, Adversarial Weight Perturbation Helps Robust Generalization. ***NeurIPS 2020***.

View from weight loss landscape

- Inspiring from standard Training:
  - flatter weight loss landscape, smaller standard generalization gap



(a) without skip connections (b) with skip connections

Is this conclusion still existing in adversarial training?

Hao Li et al. Visualizing the Loss Landscape of Neural Nets. *NeurIPS 2018.*

Adapted Visualization Method

- Inspiring from standard Training:
  - flatter weight loss landscape, smaller standard generalization gap
- Is this conclusion still existing in adversarial training?

Failed to draw the conclusion

Visualization method in *Hao Li et al.* NeurIPS2018

Standard training → Adversarial training

$g(\alpha) = L(f_{w+\alpha d}(x_i), y_i)$

$g(\alpha) = L(f_{w+\alpha d}(x_i'), y_i)$?

x_i' is from pre-generated adversarial examples[1,2]

The correct way:

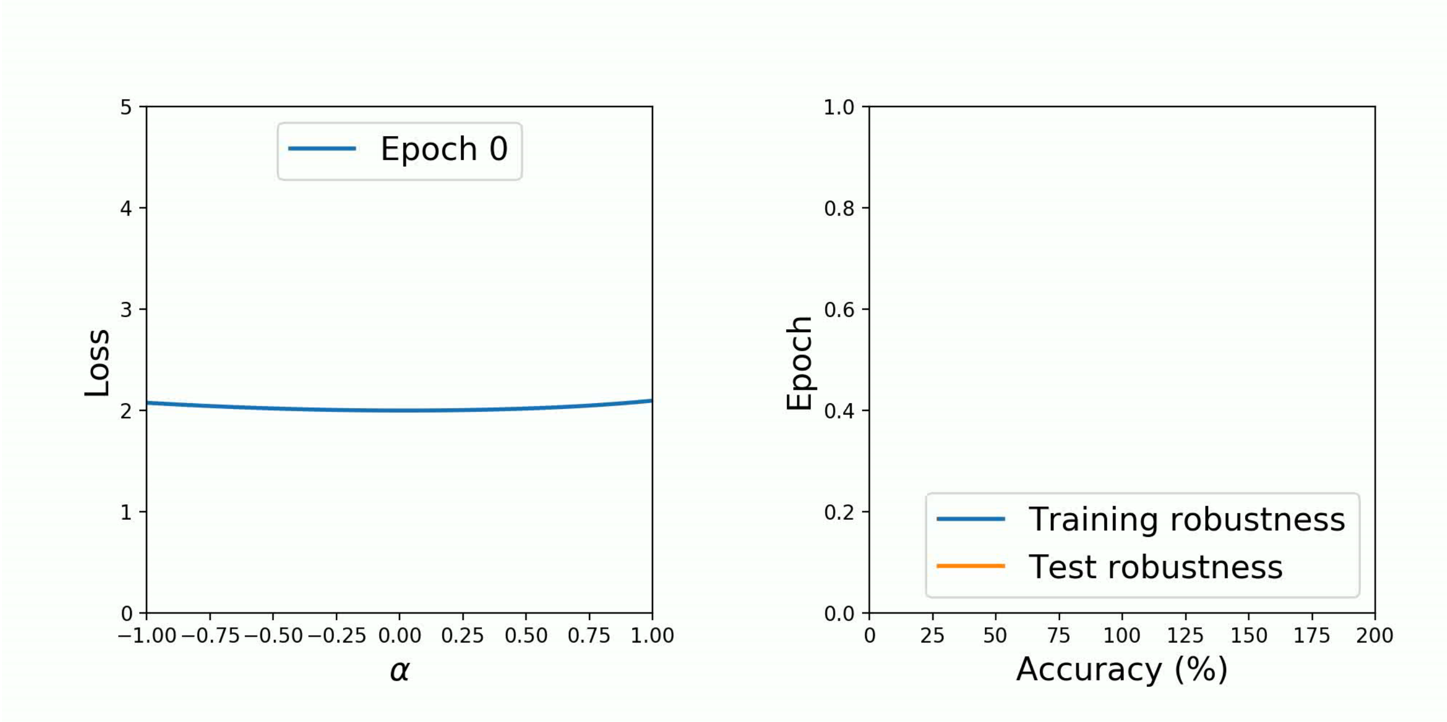
$$g(\alpha) = \rho(\mathbf{w} + \alpha\mathbf{d}) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i' - \mathbf{x}_i\|_p \le \epsilon} \ell(f_{\mathbf{w}+\alpha\mathbf{d}}(\mathbf{x}_i'), y_i),$$

Generating adversarial examples on-the-fly

[1] Understanding adversarial robustness through loss landscape geometries, *arxiv 2019.*
[2] Interpreting adversarial robustness: A view from decision surface in input space, arxiv 2018

Weight loss landscape

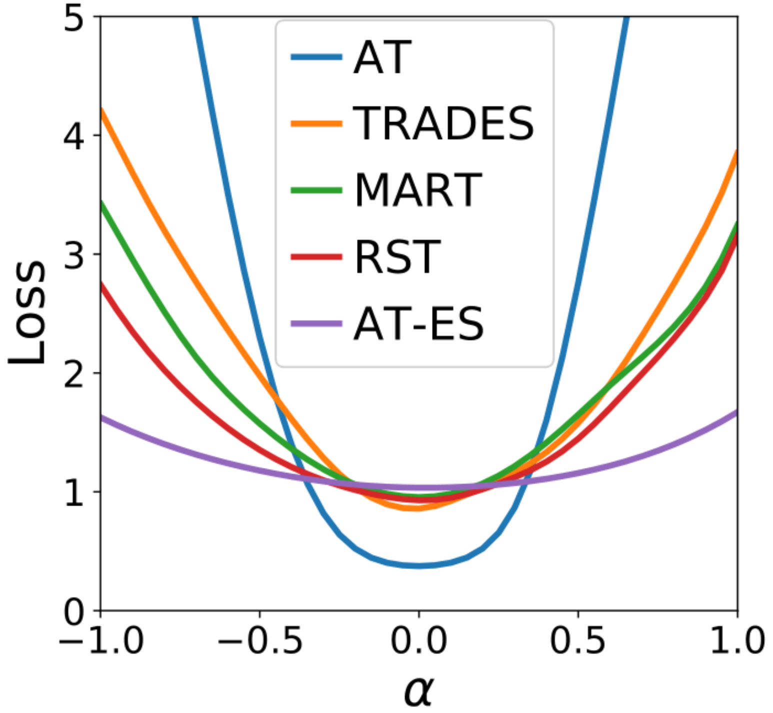
In the learning process of adversarial training

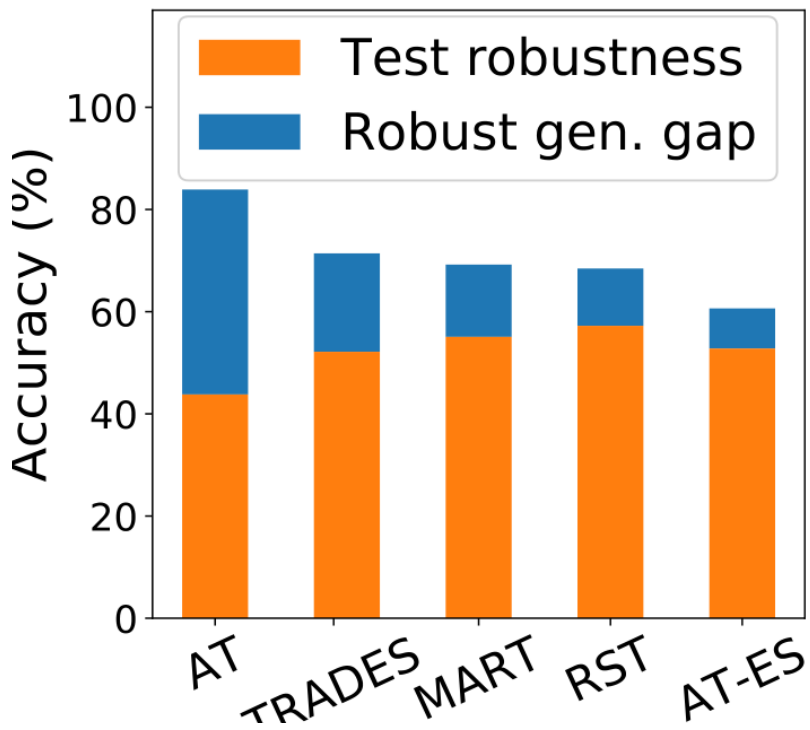


Weight loss landscape has a strong correlation with robust generalization gap

Weight loss landscape

Across different adversarial training methods





Weight loss landscape has a strong correlation with robust generalization gap

Theoretical view

- Informally from PAC-Bayesian bound

$$\mathbb{E}_{\{\mathbf{x}_i,y_i\}_{i=1}^n,\mathbf{u}}[\rho(\mathbf{w}+\mathbf{u})] \leq \rho(\mathbf{w}) + \{\underbrace{\mathbb{E}_{\mathbf{u}}[\rho(\mathbf{w}+\mathbf{u})] - \rho(\mathbf{w})}_{\text{flatness of weight loss landscape}}\} + 4\sqrt{\frac{1}{n}KL(\mathbf{w}+\mathbf{u}\|P) + \ln\frac{2n}{\delta}}.$$

- Explicitly flattening the weight loss landscape via replacing expectation by maximization

$$\min_{\boldsymbol{\theta}} \frac{1}{n}\sum_{i=1}^{n} \max_{\|x_i'-x_i\|_p \leq \epsilon} L(f_\theta(x_i'), y_i) \Rightarrow \min_{\boldsymbol{\theta}} \max_{\|v\|_p \leq \gamma\|\theta\|_p} \frac{1}{n}\sum_{i=1}^{n} \max_{\|x_i'-x_i\|_p \leq \epsilon} L(f_{\theta+v}(x_i'), y_i)$$

- Two max makes the maximization (min-max) solve better
- How to intuitively understand these two perturbations?
  - Input perturbation is local worst for each example
  - Weight perturbation is global worst for multiple examples

Implementation

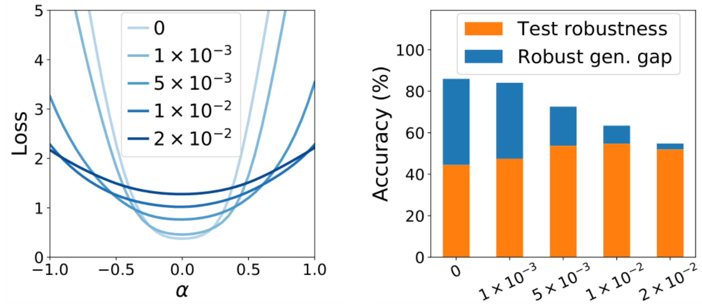
AWP-based Adversarial training (AT-AWP)

$$\min_{\boldsymbol{\theta}} \max_{\|v\|_p \leq \gamma \|\theta\|_p} \frac{1}{n} \sum_{i=1}^{n} \max_{\|x_i' - x_i\|_p \leq \epsilon} L(f_{\theta+v}(x_i'), y_i)$$

- An empirical implementation:
  1. craft adversarial examples $\boldsymbol{x}_i'$;
  2. calculate AWP based on $\boldsymbol{x}_i'$ using one extra forward and backward propagation;
  3. update the parameter using the gradient based on $\boldsymbol{x}_i'$ and AWP.
- Only ~8% time overhead in our implementation of AT-AWP.
- AWP is easily extended to other methods, such as TRADES, MART and RST.

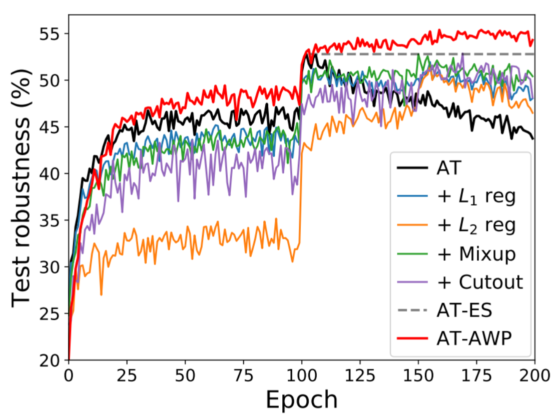
Real robustness improvement

- AWP indeed flattens weight loss landscape, and reduces the robust generalization gap.



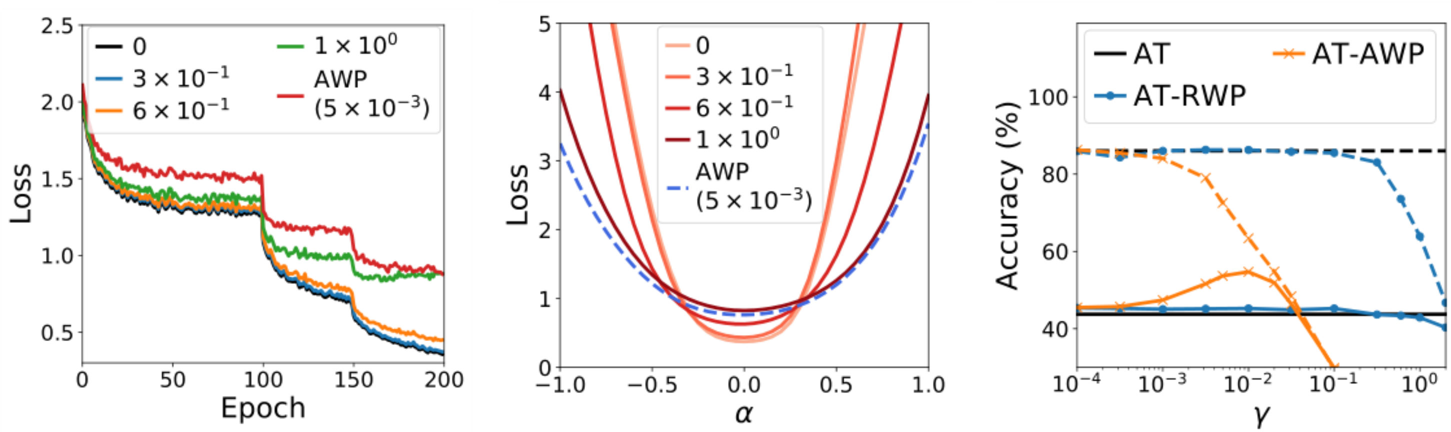
(c) Weight loss landscape (d) Generalization gap

- AWP really improves both the last and best robustness during training.



AWP vs. Random WP

- AWP easily finds the worst-case perturbation, while RWP needs a relatively large perturbation;
- AWP obtain a flatter weight loss landscape using smaller perturbations;
- AWP balances the training robustness and robust gap well.



(a) Loss curve (b) Weight loss landscapes (c) Robustness

Universal robustness improvement

Table 2: Test robustness (%) on CIFAR-10 using WideResNet under L_∞ threat model. We omit the standard deviations of 5 runs as they are very small ($< 0.40\%$), which hardly effect the results.

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 [4]
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	53.18
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	54.99
Pre-training-AWP	**88.33**	**66.34**	**61.40**	**61.21**	**59.28**	**65.55**	**57.39**
RST	**89.69**	67.94	62.60	62.22	60.47	67.60	59.65
RST-AWP	88.25	**69.60**	**63.73**	**63.58**	**61.62**	**68.72**	**61.10**

Table 3: Test robustness (%) on CIFAR-10 using WideResNet under L_∞ threat model. In brackets, $+$ indicates improvements over Pre-training.

Defense	PGD-20	CW_∞	AA
Pre-training	57.37	55.95	54.92
TRADES-AWP	59.27 (+1.90)	57.07 (+1.12)	56.17 (+1.25)
Pre-training-AWP	61.40 (+4.03)	59.28 (+3.33)	57.39 (+2.47)

Except max process, how about min process?

Revisiting the Input Examples

Adversarial training is a **min-max optimization** process:

$$\min_{\boldsymbol{\theta}} \frac{1}{n}\sum_{i=1}^{n} \max_{\|x_i' - x_i\|_p \leq \epsilon} L(f_\theta(x_i'), y_i)$$

L: loss, f_θ: model, $\mathrm{x_i}$: clean example, $\mathrm{y_i}$: class, $\mathrm{x_i'}$: adversarial example.

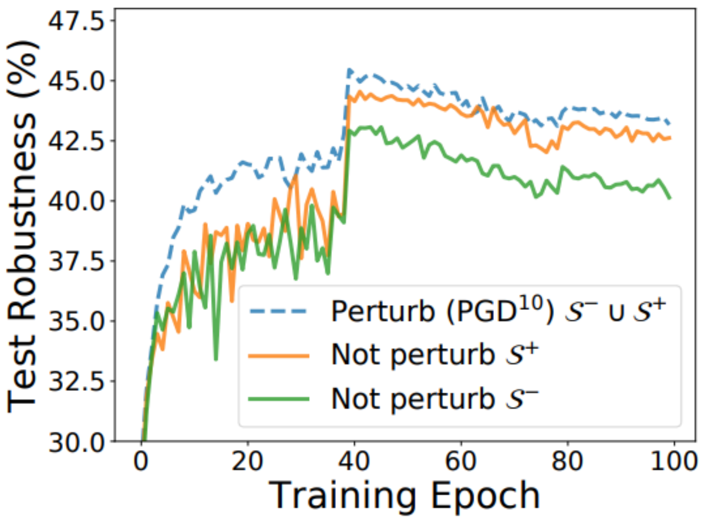
Adversarial examples are only defined on correctly classified examples

How about misclassified examples?

Yisen Wang*, *Difan Zou**, *et al.*, Improving Adversarial Robustness Requires Revisiting Misclassified Examples. ***ICLR 2020***.

Misclassified vs. correctly classified examples

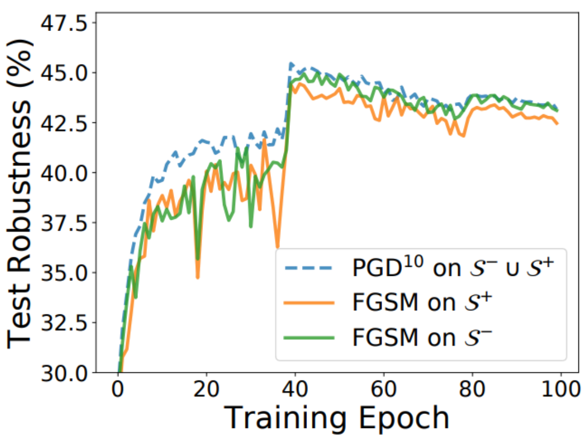
- A pre-trained network to select the **same size (13%)**
  - Subset of misclassified examples S^-
  - Subset of correctly classified examples S^+



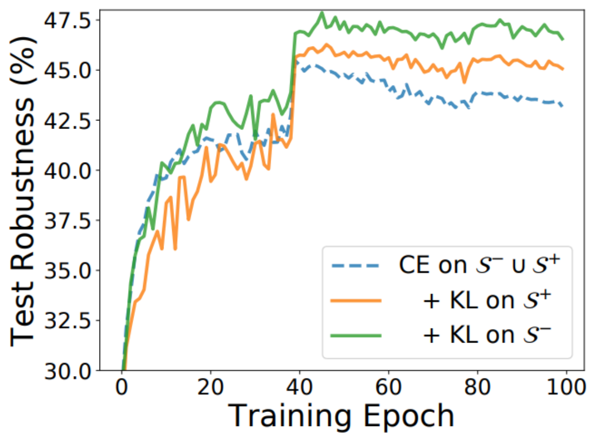
Misclassified examples have a significant impact on the final robustness

Delving into the max and min processes

- For inner maximization process:
  - Weak attack on misclassified examples $\mathcal{S}^-$
  - Weak attack on correctly classified examples $\mathcal{S}^+$
- For outer minimization process:
  - Regularization on misclassified examples $\mathcal{S}^-$
  - Regularization on correctly classified examples $\mathcal{S}^+$



(b) Inner maximization



(c) Outer minimization

different maximization techniques have negligible effect

different minimization techniques have significant effect

Misclassification aware adversarial risk

- Adversarial risk:

$$\mathcal{R}(h_{\boldsymbol{\theta}}) = \frac{1}{n}\sum_{i=1}^{n} \max_{\mathbf{x}_i' \in \mathcal{B}_\epsilon(\mathbf{x}_i)} \mathbb{1}\big(h_{\boldsymbol{\theta}}(\mathbf{x}_i') \neq y_i\big),$$

- Correctly classified and misclassified examples:

$$\mathcal{S}_{h_{\boldsymbol{\theta}}}^{+} = \{i : i \in [n], h_{\boldsymbol{\theta}}(\mathbf{x}_i) = y_i\} \quad \text{and} \quad \mathcal{S}_{h_{\boldsymbol{\theta}}}^{-} = \{i : i \in [n], h_{\boldsymbol{\theta}}(\mathbf{x}_i) \neq y_i\}$$

- Misclassification aware adversarial risk:

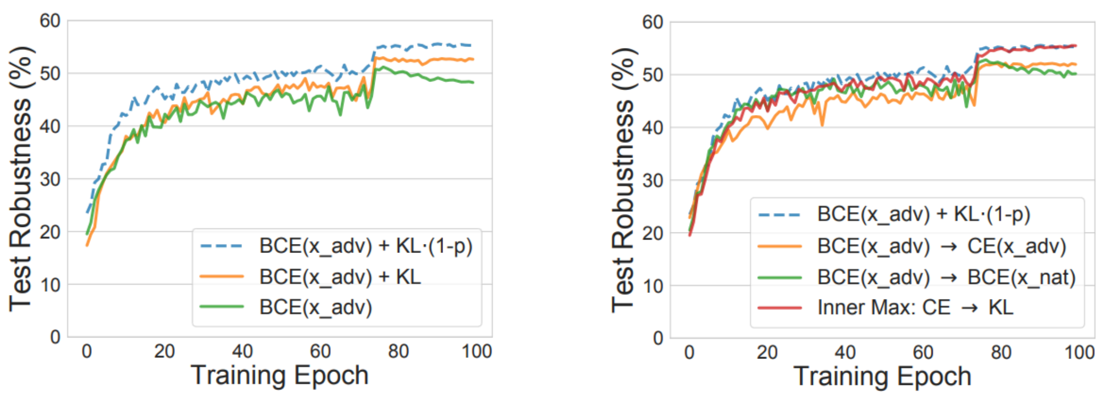
$$\begin{aligned}\min_{\boldsymbol{\theta}} \mathcal{R}_{\text{misc}}(h_{\boldsymbol{\theta}}) :&= \tfrac{1}{n}\Big(\textstyle\sum_{i\in\mathcal{S}_{h_{\boldsymbol{\theta}}}^{+}} \mathcal{R}^{+}(h_{\boldsymbol{\theta}}, \mathbf{x}_i) + \sum_{i\in\mathcal{S}_{h_{\boldsymbol{\theta}}}^{-}} \mathcal{R}^{-}(h_{\boldsymbol{\theta}}, \mathbf{x}_i)\Big) \\ &= \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \big\{\mathbb{1}(h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_i') \neq y_i) + \mathbb{1}(h_{\boldsymbol{\theta}}(\mathbf{x}_i) \neq h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_i')) \cdot \mathbb{1}(h_{\boldsymbol{\theta}}(\mathbf{x}_i) \neq y_i)\big\}\end{aligned}$$

Misclassification Aware adveRsarial Training (MART)

- Surrogate loss functions (existing methods and MART):

Defense Method	Loss Function
Standard	$\mathrm{CE}(\mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta}), y)$
ALP	$\mathrm{CE}(\mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta}), y) + \lambda \cdot \lVert \mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta}) - \mathbf{p}(\mathbf{x}, \boldsymbol{\theta}) \rVert_2^2$
CLP	$\mathrm{CE}(\mathbf{p}(\mathbf{x}, \boldsymbol{\theta}), y) + \lambda \cdot \lVert \mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta}) - \mathbf{p}(\mathbf{x}, \boldsymbol{\theta}) \rVert_2^2$
TRADES	$\mathrm{CE}(\mathbf{p}(\mathbf{x}, \boldsymbol{\theta}), y) + \lambda \cdot \mathrm{KL}\big(\mathbf{p}(\mathbf{x}, \boldsymbol{\theta}) \Vert \mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta})\big)$
MMA	$\mathrm{CE}(\mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta}), y) \cdot \mathbb{1}(h_{\boldsymbol{\theta}}(\mathbf{x}) = y) + \mathrm{CE}(\mathbf{p}(\mathbf{x}, \boldsymbol{\theta}), y) \cdot \mathbb{1}(h_{\boldsymbol{\theta}}(\mathbf{x}) \neq y)$
MART	$\mathrm{BCE}(\mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta}), y) + \lambda \cdot \mathrm{KL}(\mathbf{p}(\mathbf{x}, \boldsymbol{\theta}) \Vert \mathbf{p}(\hat{\mathbf{x}}', \boldsymbol{\theta})) \cdot (1 - \mathbf{p}_y(\mathbf{x}, \boldsymbol{\theta}))$

$$\mathrm{BCE}\big(\mathbf{p}(\hat{\mathbf{x}}'_i, \boldsymbol{\theta}), y_i\big) = -\log\left(\mathbf{p}_{y_i}(\hat{\mathbf{x}}'_i, \boldsymbol{\theta})\right) - \log\left(1 - \max_{k \neq y_i} \mathbf{p}_k(\hat{\mathbf{x}}'_i, \boldsymbol{\theta})\right)$$



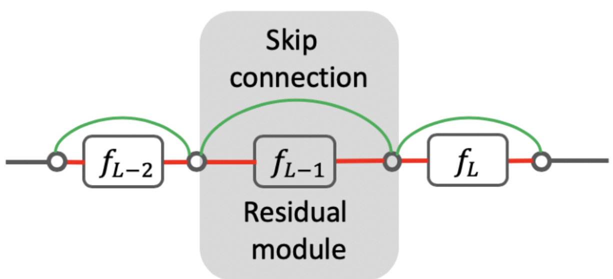
(a) Removing (b) Replacing

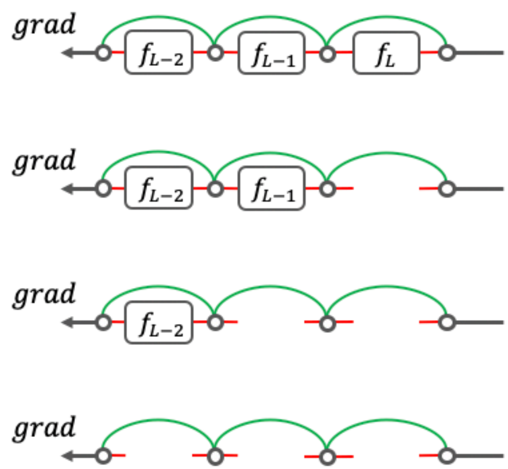
Beyond training objective,

is model architecture related to robustness?

Skip connection matters

- Neural network architectures:
  - Skip connection, activation, batch normalization, …
- Skip connection



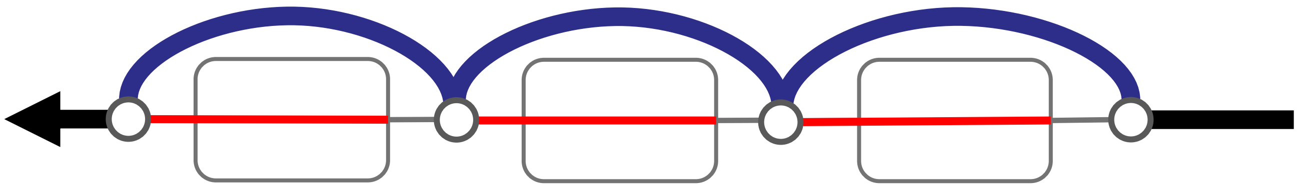


	white-box / black-box
f_{L-2}, f_{L-1}, f_L	100% / 52.52%
f_{L-2}, f_{L-1}	100% / 55.24%
f_{L-2}	100% / 62.10%
(none)	99.86% / 47.70%

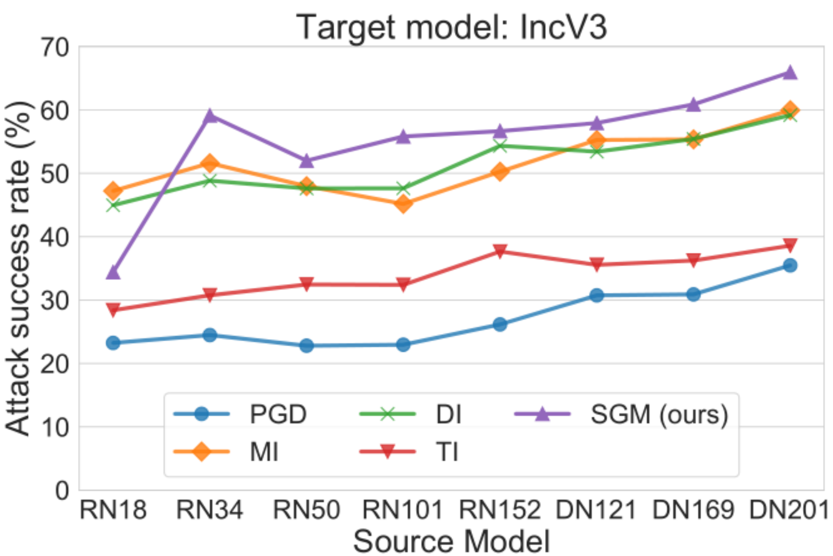
Skip connections expose more transferable information !

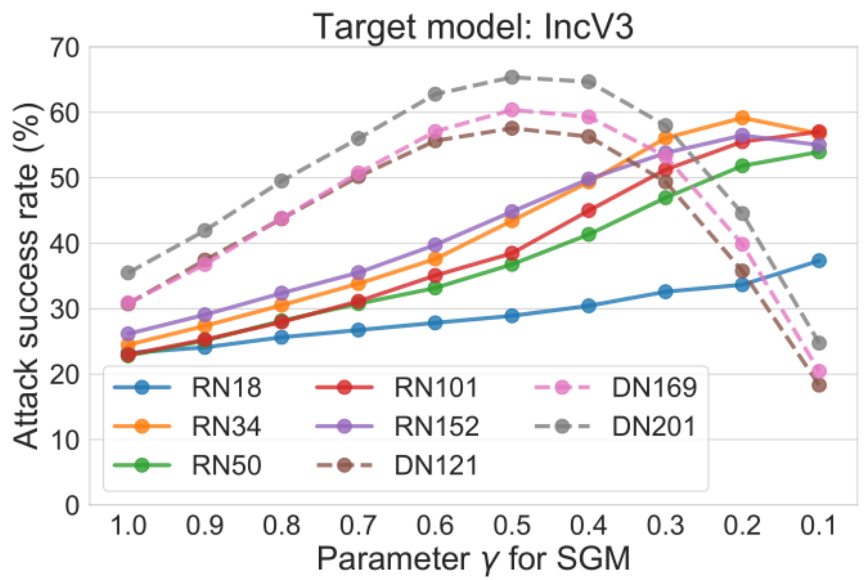
D. Wu, ***Y Wang,*** *et.al,* Skip Connections Matter: On the Transferability of Adversarial Examples Generated with ResNets. ***ICLR 2020***.

Skip Gradient Method (SGM)



$$\nabla_{\boldsymbol{x}} \ell = \frac{\partial \ell}{\partial \boldsymbol{z}_L} \prod_{i=0}^{L-1} \left(\gamma \frac{\partial f_{i+1}}{\partial \boldsymbol{z}_i} + 1 \right) \frac{\partial \boldsymbol{z}_0}{\partial \boldsymbol{x}}$$





Takehome Message

- For the min-max problem, the following aspects are essential:
  - how to make max solves better
  - How to make min process easily
- Model architecture is also important for adversarial research

Related Papers

- Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, Quanquan Gu, "*On the Convergence and Robustness of Adversarial Training*", **ICML 2019 Long Talk**
- Dongxian Wu, Shu-Tao Xia, Yisen Wang, "*Adversarial Weight Perturbation Helps Robust Generalization*", **NeurIPS 2020**
- Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, Quanquan Gu, "*Improving Adversarial Robustness Requires Revisiting Misclassified Examples*", **ICLR 2020**
- Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, Xingjun Ma, "*Skip Connections Matter: On the Transferability of Adversarial Examples Generated with ResNets*", **ICLR 2020 Spotlight**
- Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, Yisen Wang, "*Unlearnable Examples: Making Personal Data Unexploitable*", **ICLR 2021 Spotlight**
- Yang Bai, Yuyuan Zeng, Yong Jiang, Shu-Tao Xia, Xingjun Ma, Yisen Wang, "*Improving Adversarial Robustness via Channel-wise Activation Suppressing*", **ICLR 2021 Spotlight**

Building ML one can truly rely on

北京大學
PEKING UNIVERSITY

Thanks!

yisen.wang@pku.edu.cn
https://yisenwang.github.io/