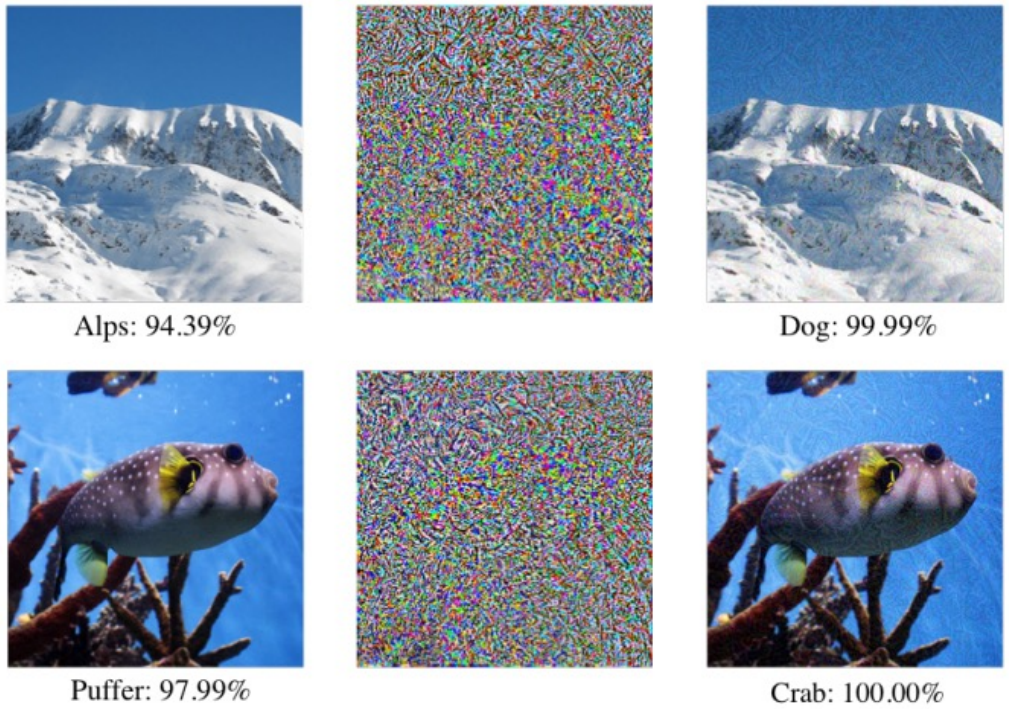
Robust Machine Learning in the Adversarial Setting

Tianyu Pang

Advisor: Jun Zhu

Department of Computer Science and Technology
Tsinghua University

Adversarial Examples in Computer Vision

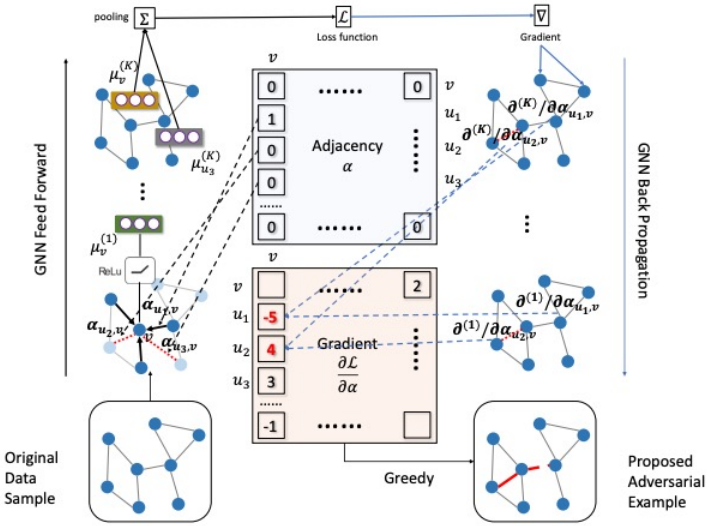


(Dong et al. CVPR 2018)

Not only in Computer Vision

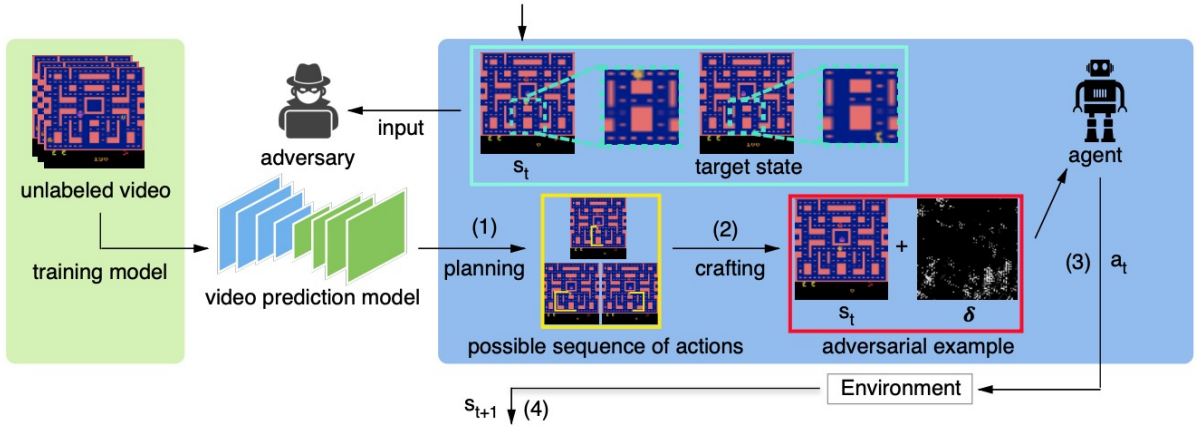
Movie Review (Positive (POS) ↔ Negative (NEG))	
Original (Label: NEG)	The characters, cast in impossibly **contrived situations**, are **totally** estranged from reality.
Attack (Label: POS)	The characters, cast in impossibly **engineered circumstances**, are **fully** estranged from reality.
Original (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**.

BERT model (Jin et al. AAAI 2020)

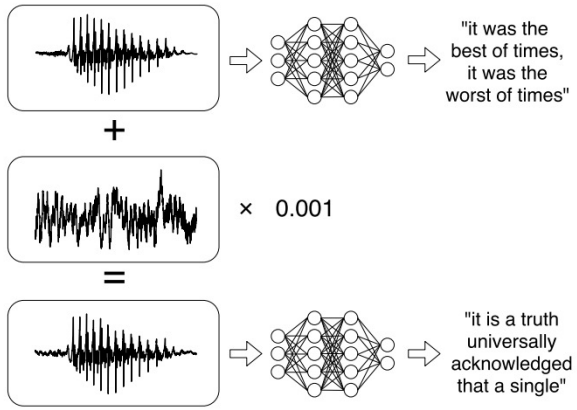


GNN model (Dai et al. ICML 2018)

Recommend System, LIDAR, ……



Reinforcement Learning (Lin et al. IJCAI 2017)



Audio (Carlini and Wagner. S&P 2018)

Counter-intuitive?

Why these models with such high performance will make such ridiculous mistakes?

Counter-intuitive?

In worst case, happens in 1% cases

Why these models with such high performance will make such ridiculous mistakes?

In expectation, happens in 99% cases

Why 1% Matters?

- **Potential Risk**

In safety-critical areas including payment, security, health care, finance, automatic drive, etc.

- **Public Trust**

People are suspicious and rigorous of new technologies, and barely tolerate high-risk defects. For example, the accident of Tesla automatic driving system.

How to Defend Adversarial Attacks?

Possible strategy one:

To correctly classify adversarial examples

- Optimal
- Difficult to achieve
- Computationally expensive (adversarial training)

How to Defend Adversarial Attacks?

Possible strategy two:

To detect and filter out adversarial examples

- Suboptimal
- Little computation
- Methods borrowed from anomaly detection

Possible strategy one:

Max-Mahalanobis Training
(ICML 2018 + ICLR 2020)

Full list of our papers and codes

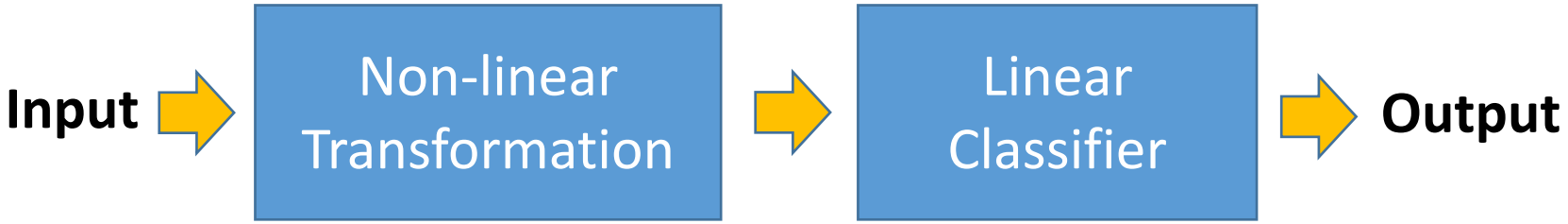
Max-Mahalanobis Training

Part I

(ICML 2018)

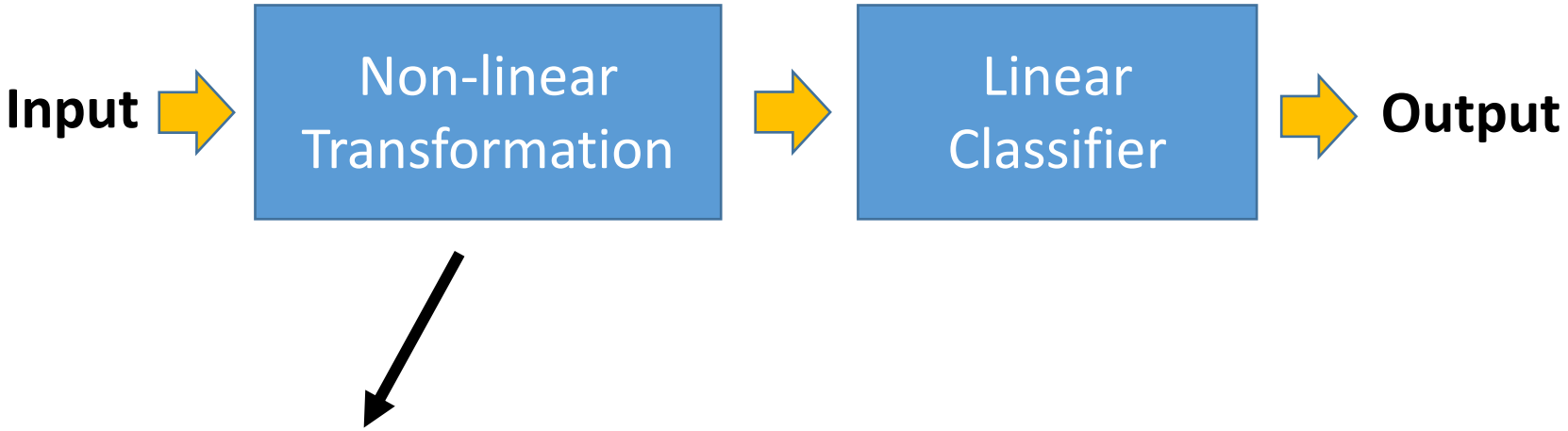
Motivation

- **Paradigm of feed-forward deep nets**



Motivation

- **Paradigm of feed-forward deep nets**

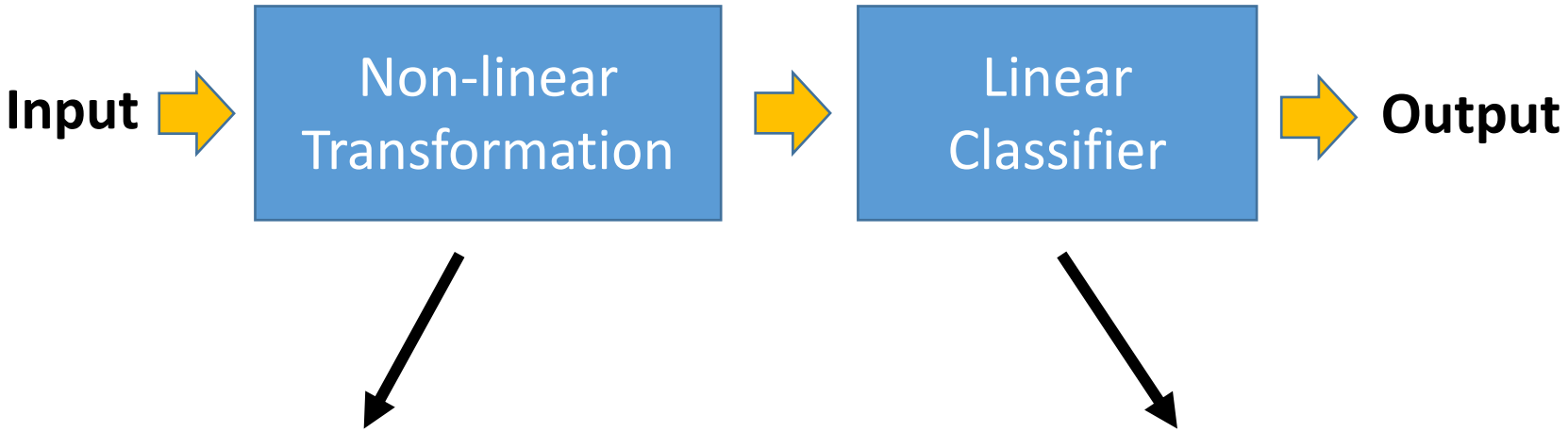


Active area of research

(AlexNet; VGG nets; ResNets; GoogleNets; DenseNets;)

Motivation

- **Paradigm of feed-forward deep nets**



Active area of research

(AlexNet; VGG nets; ResNets; GoogleNets; DenseNets;)

Much less active

(Softmax regression)

Motivation

- **Design a new network architecture for better performance in the adversarial setting.**

Motivation

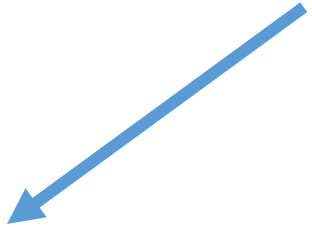
- **Design a new network architecture for better performance in the adversarial setting.**

- **Substitute a new linear classifier for softmax regression (SR).**

So what is a suitable new linear classifier?

Inspiration one: LDA is more efficient than LR

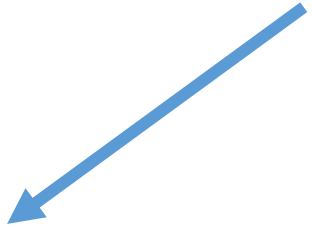
- Efron et al.(1975) show that *if the input distributes as a mixture of Gaussian*, then linear discriminant analysis (LDA) is **more efficient** than logistic regression (LR).



LDA needs less training data than LR to obtain certain error rate

Inspiration one: LDA is more efficient than LR

- Efron et al.(1975) show that *if the input distributes as a mixture of Gaussian*, then linear discriminant analysis (LDA) is **more efficient** than logistic regression (LR).

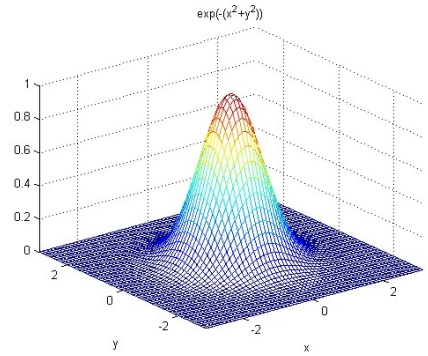


LDA needs less training data than LR to obtain certain error rate

- However, in practice data points hardly distributes as a mixture of Gaussian in the input space.

Inspiration two: neural networks are powerful

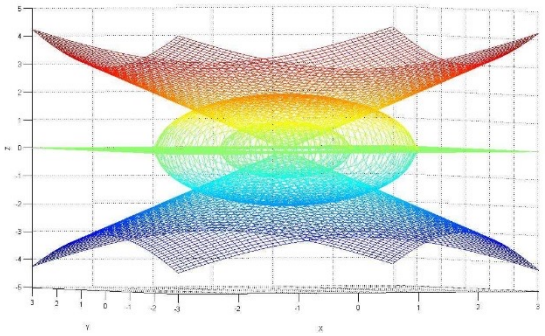
- **Deep generative models (e.g., GANs) are successful.**



Simple Distribution
(Gaussian/Mixture of Gaussian)

Deep generative models

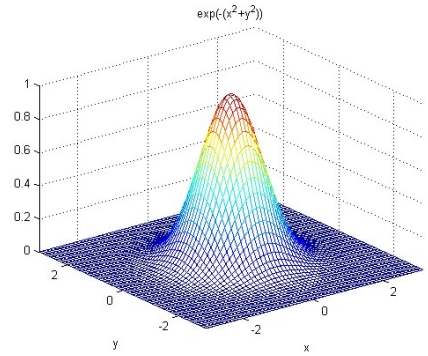
DNN



Complex Distribution
(Data distribution)

Inspiration two: neural networks are powerful

- **Deep generative models (e.g., GANs) are successful.**
- **The reverse direction should also be feasible.**

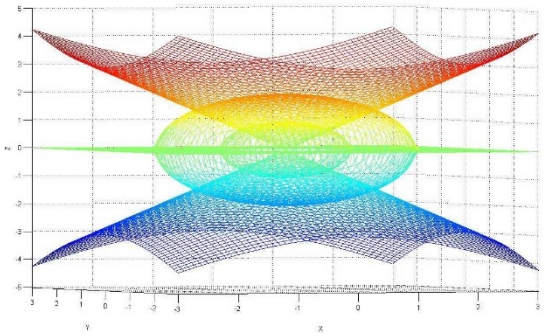


Simple Distribution
(Gaussian/Mixture of Gaussian)

Deep generative models

DNN

Our Method
(MM-LDA networks)



Complex Distribution
(Data distribution)

Our method

- Models the feature distribution in DNNs as a mixture of Gaussian.
- Applies LDA on the feature to make predictions.

How to treat the Gaussian parameters?

- Wan et al. (CVPR 2018) also model the feature distribution as a mixture of Gaussian. However, they treat the Gaussian parameters (μ_i and Σ) as extra trainable variables.

How to treat the Gaussian parameters?

- Wan et al. (CVPR 2018) also model the feature distribution as a mixture of Gaussian. However, they treat the Gaussian parameters (μ_i and Σ) as extra trainable variables.

- We treat them as hyperparameters calculated by our algorithm, which can **provide theoretical guarantee on the robustness.**

How to treat the Gaussian parameters?

- Wan et al. (CVPR 2018) also model the feature distribution as a mixture of Gaussian. However, they treat the Gaussian parameters (μ_i and Σ) as extra trainable variables.

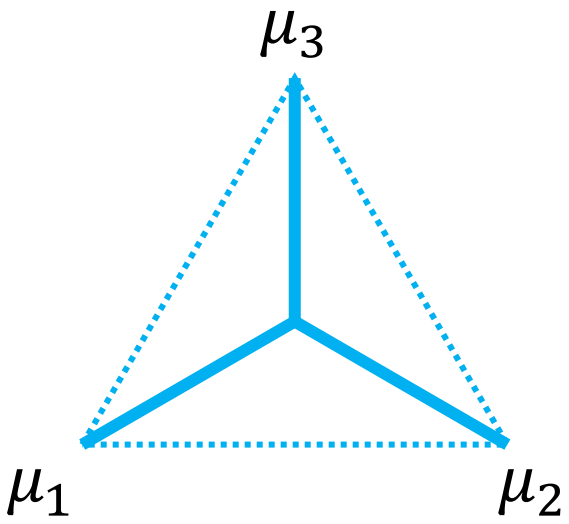
- We treat them as hyperparameters calculated by our algorithm, which can **provide theoretical guarantee on the robustness.**

- The induced mixture of Gaussian model is named **Max Mahalanobis Distribution (MMD).**

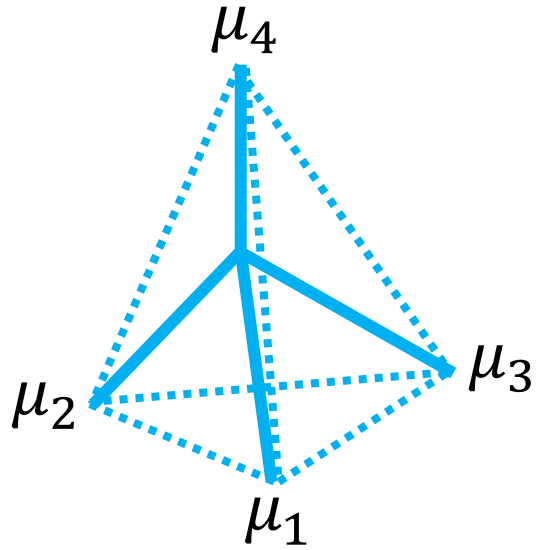
Max-Mahalanobis Distribution (MMD)

- **Making the minimal Mahalanobis distance between two Gaussian components maximal.**

$L = 2$
Straight Line



$L = 3$
Equilateral Triangle



$L = 4$
Regular Tetrahedron

Some formal derivations

Definition of Robustness

- The robustness on a point with label i (Moosavi-Dezfoolo et al. , CVPR 2016):

$$\min_{j \neq i} d_{i,j} ,$$

where $d_{i,j}$ is the local minimal distance of a point with label i to an adversarial example with label j.

Definition of Robustness

- The robustness on a point with label i (Moosavi-Dezfoolo et al. , CVPR 2016):

$$\min_{j \neq i} d_{i,j} \,,$$

where $d_{i,j}$ is the local minimal distance of a point with label i to an adversarial example with label j.

- We further define the robustness of the classifier as:

$$\mathbf{RB} = \min_{i,j \in [L]} \mathbb{E}(d_{i,j}) \,.$$

Robustness w.r.t Gaussian parameters

Theorem 1. The expectation of the distance $\mathbb{E}(d_{i,j})$ is a function of the Mahalanobis distance $\Delta_{i,j}$ as

$$\mathbb{E}(d_{i,j}) = \sqrt{\frac{2}{\pi}} \exp\left(-\frac{\Delta_{i,j}^2}{8}\right) + \frac{1}{2}\Delta_{i,j}\left[1 - 2\Phi(-\frac{\Delta_{i,j}}{2})\right]$$

where $\Phi(\cdot)$ is the normal cumulative distribution function.

Robustness w.r.t Gaussian parameters

Theorem 1. The expectation of the distance $\mathbb{E}(d_{i,j})$ is a function of the Mahalanobis distance $\Delta_{i,j}$ as

$$\mathbb{E}(d_{i,j}) = \sqrt{\frac{2}{\pi}} \exp\left(-\frac{\Delta_{i,j}^2}{8}\right) + \frac{1}{2}\Delta_{i,j}\left[1 - 2\Phi(-\frac{\Delta_{i,j}}{2})\right]$$

where $\Phi(\cdot)$ is the normal cumulative distribution function.

$$\mathbf{RB} \approx \overline{\mathbf{RB}} = \frac{\mathbf{1}}{\mathbf{2}} \min_{i,j \in [L]} \Delta_{i,j},$$

Robustness w.r.t Gaussian parameters

Theorem 1. The expectation of the distance $\mathbb{E}(d_{i,j})$ is a function of the Mahalanobis distance $\Delta_{i,j}$ as

$$\mathbb{E}(d_{i,j}) = \sqrt{\frac{2}{\pi}} \exp\left(-\frac{\Delta_{i,j}^2}{8}\right) + \frac{1}{2}\Delta_{i,j}\left[1 - 2\Phi(-\frac{\Delta_{i,j}}{2})\right]$$

where $\Phi(\cdot)$ is the normal cumulative distribution function.

$$\mathbf{RB} \approx \overline{\mathbf{RB}} = \frac{\mathbf{1}}{\mathbf{2}} \min_{i,j \in [L]} \Delta_{i,j},$$

Distributing as a MMD can maximize $\overline{\mathrm{RB}}$.

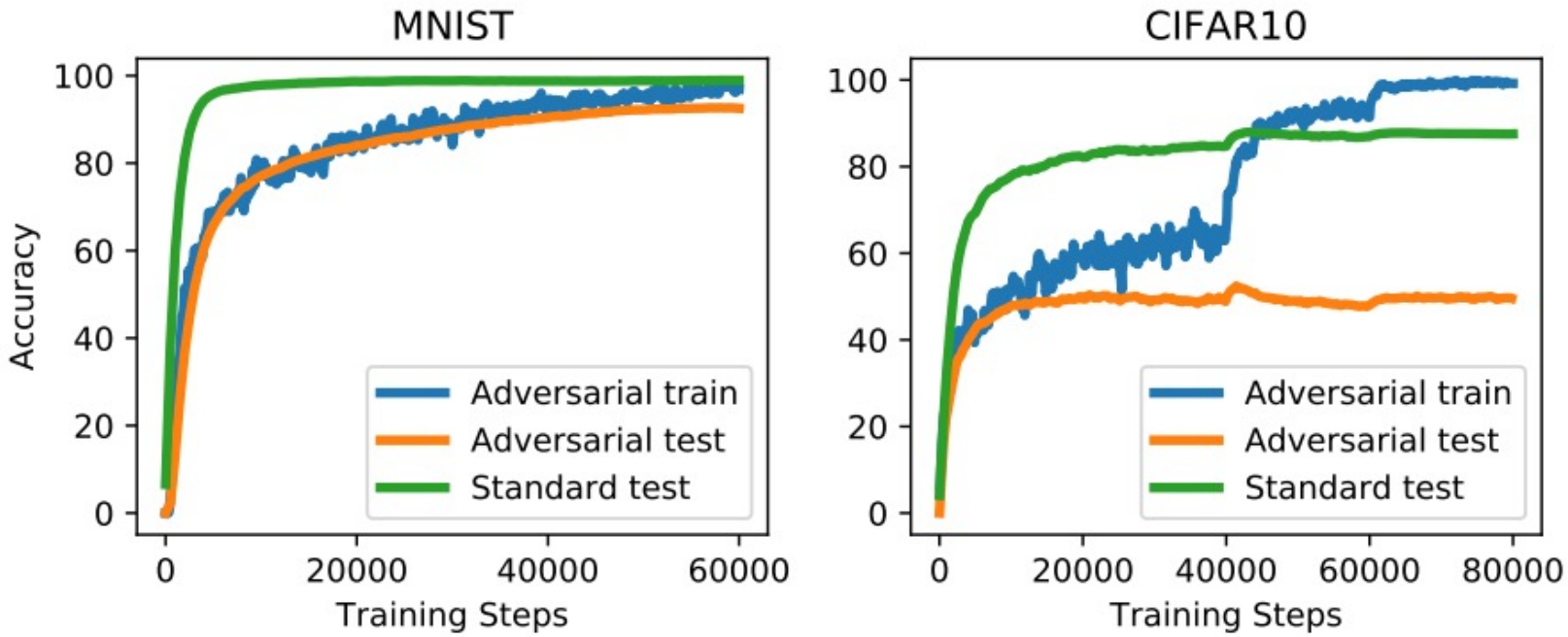
Can we further improve MMLDA?

Max-Mahalanobis Training

Part II

(ICLR 2020)

Motivation



The same dataset, e.g., CIFAR-10, which enables good standard accuracy may not suffice to train robust models.

(Schmidt et al. NeurIPS 2018)

Possible Solutions

- **Introducing extra labeled data**

(Hendrycks et al. ICML 2019)

- **Introducing extra unlabeled data**

(Alayrac et al. NeurIPS 2019; Carmon et al. NeurIPS 2019)

Possible Solutions

- **Introducing extra labeled data**

(Hendrycks et al. ICML 2019)

- **Introducing extra unlabeled data**

(Alayrac et al. NeurIPS 2019; Carmon et al. NeurIPS 2019)

- **Our solution: Increase sample density to induce locally sufficient training data for robust learning**

Possible Solutions

- **Introducing extra labeled data**

(Hendrycks et al. ICML 2019)

- **Introducing extra unlabeled data**

(Alayrac et al. NeurIPS 2019; Carmon et al. NeurIPS 2019)

- **Our solution: Increase sample density to induce locally sufficient training data for robust learning**

Q1: What is the definition of sample density?

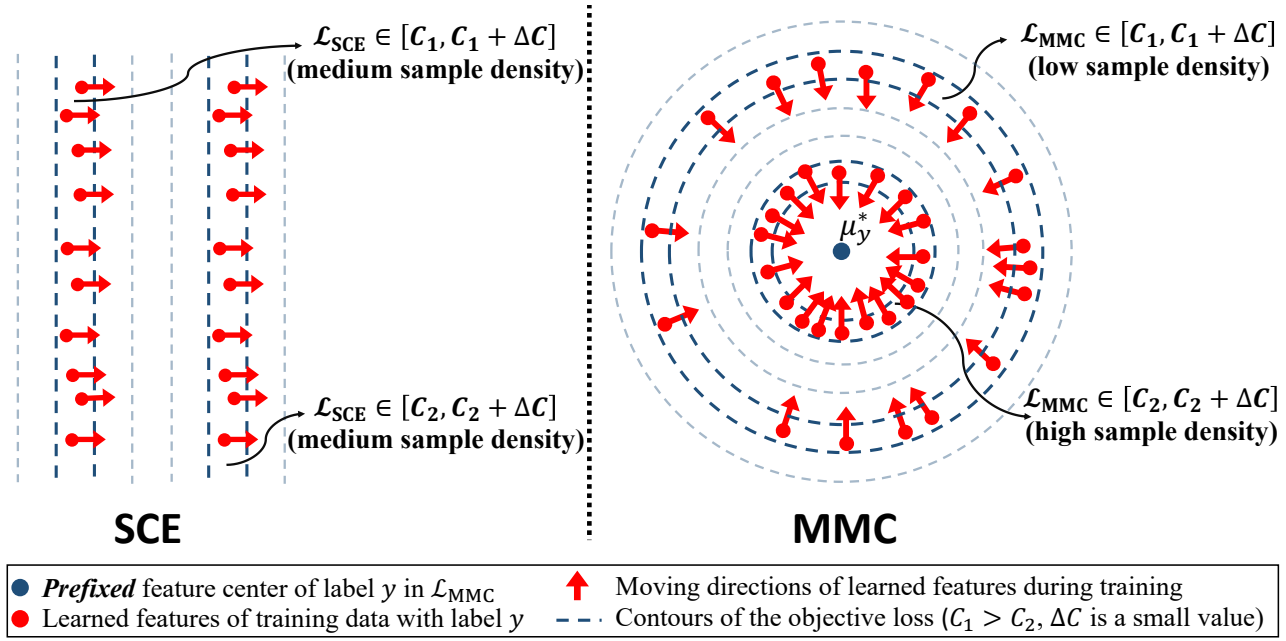
Q2: Can existing training objectives induce high sample density?

Sample Density

Given a training dataset $\mathcal{D}$ with N input-label pairs, and the feature mapping Z trained by the objective $\mathcal{L}(Z(x), y)$ on this dataset, we define the sample density nearby the feature point $z = Z(x)$ following the similar definition in physics (Jackson, 1999) as

$$\mathbb{SD}(z) = \frac{\Delta N}{\mathrm{Vol}(\Delta B)}. \tag{2}$$

Here $\mathrm{Vol}(\cdot)$ denotes the volume of the input set, ΔB is a small neighbourhood containing the feature point z, and $\Delta N = |Z(\mathcal{D}) \cap \Delta B|$ is the number of training points in ΔB, where $Z(\mathcal{D})$ is the set of all mapped features for the inputs in $\mathcal{D}$. Note that the mapped feature z is still of the label y.



Generalized Softmax Cross Entropy Loss (g-SCE loss)

We define g-SCE loss as

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = -1_y^\top \log\left[\text{softmax}(h)\right],$$

where $h_i = -(z - \mu_i)^\top \Sigma_i (z - \mu_i) + B_i$ **is the logits in quadratic form.**

Generalized Softmax Cross Entropy Loss (g-SCE loss)

We define g-SCE loss as

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = -1_y^\top \log\left[\text{softmax}(h)\right],$$

where $h_i = -(z-\mu_i)^\top \Sigma_i (z-\mu_i) + B_i$ **is the logits in quadratic form.**

We note that the SCE loss is included in the family of g-SCE loss as

$$\text{softmax}(Wz+b)_i = \frac{\exp(W_i^\top z + b_i)}{\sum_{l\in[L]} \exp(W_l^\top z + b_l)} = \frac{\exp(-\|z - \frac{1}{2}W_i\|_2^2 + b_i + \frac{1}{4}\|W_i\|_2^2)}{\sum_{l\in[L]} \exp(-\|z - \frac{1}{2}W_l\|_2^2 + b_l + \frac{1}{4}\|W_l\|_2^2)}.$$

Generalized Softmax Cross Entropy Loss (g-SCE loss)

We define g-SCE loss as

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = -1_y^\top \log\left[\text{softmax}(h)\right],$$ **Including MMLDA**

where $h_i = -(z-\mu_i)^\top \Sigma_i (z-\mu_i) + B_i$ **is the logits in quadratic form.**

We note that the SCE loss is included in the family of g-SCE loss as

$$\text{softmax}(Wz+b)_i = \frac{\exp(W_i^\top z + b_i)}{\sum_{l\in[L]} \exp(W_l^\top z + b_l)} = \frac{\exp(-\|z - \frac{1}{2}W_i\|_2^2 + b_i + \frac{1}{4}\|W_i\|_2^2)}{\sum_{l\in[L]} \exp(-\|z - \frac{1}{2}W_l\|_2^2 + b_l + \frac{1}{4}\|W_l\|_2^2)}.$$

The Contour of g-SCE Loss

To provide a formal representation of the sample density induced by the g-SCE loss, we first derive the formula of the contours

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = C$$

The Contour of g-SCE Loss

To provide a formal representation of the sample density induced by the g-SCE loss, we first derive the formula of the contours

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = C$$

$$\log\left(1+\frac{\sum_{l\neq y}\exp(h_l)}{\exp(h_y)}\right)=C \implies h_y=\log\left[\sum_{l\neq y}\exp(h_l)\right]-\log(C_e-1).$$

The Contour of g-SCE Loss

To provide a formal representation of the sample density induced by the g-SCE loss, we first derive the formula of the contours

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = C$$

$$\log\left(1+\frac{\sum_{l\neq y}\exp(h_l)}{\exp(h_y)}\right)=C \implies h_y=\log\left[\sum_{l\neq y}\exp(h_l)\right]-\log(C_e-1).$$

Log-Sum-Exp function, which is a soft maximum function

The Contour of g-SCE Loss

To provide a formal representation of the sample density induced by the g-SCE loss, we first derive the formula of the contours

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = C$$

$$\log\left(1+\frac{\sum_{l\neq y}\exp(h_l)}{\exp(h_y)}\right)=C \implies h_y=\log\left[\sum_{l\neq y}\exp(h_l)\right]-\log(C_e-1).$$

approximately

$$h_y - h_{\tilde{y}} = -\log(C_e - 1),$$

where $C_e = \exp(C)$, **and** $\tilde{y} = \arg\max_{l\neq y} h_l$.

The Contour of g-SCE Loss

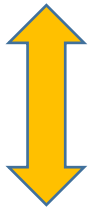
We can the approximate loss as

$$\mathcal{L}_{y,\tilde{y}}(z) = \log[\exp(h_{\tilde{y}} - h_y) + 1]$$

such that

$$h_y - h_{\tilde{y}} = -\log(C_e - 1) \quad \Longleftrightarrow \quad \mathcal{L}_{y,\tilde{y}}(z) = C$$

approximately



approximately

$$h_y = \log\left[\sum_{l \neq y} \exp(h_l)\right] - \log(C_e - 1) \quad \Longleftrightarrow \quad \mathcal{L}_{\text{g-SCE}}(Z(x), y) = C$$

The Neighborhood ΔB in Sample Density

Based on the above approximation, we can now approximate the neighborhood

$$\Delta B = \{\mathbf{z} \in \mathbb{R}^d | \mathcal{L}(\mathbf{z}, y) \in [C, C + \Delta C]\}$$

approximately

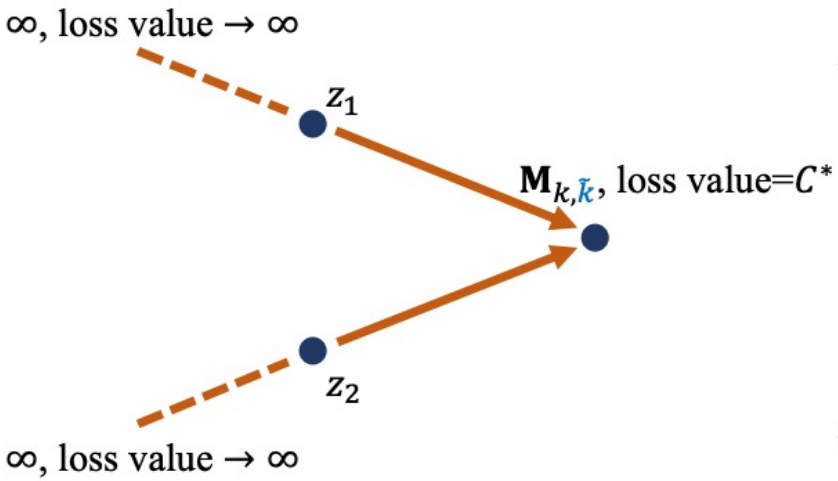
$$\Delta B_{y,\tilde{y}} = \{\mathbf{z} \in \mathbb{R}^d | \mathcal{L}_{y,\tilde{y}}(\mathbf{z}) \in [C, C + \Delta C]\}$$

Induced Sample Density of g-SCE Loss

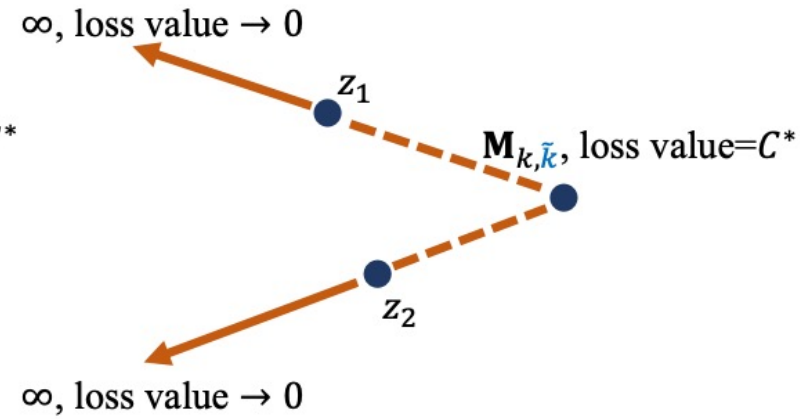
Theorem 1. *(Proof in Appendix A.1) Given* $(x, y) \in \mathcal{D}_{k,\tilde{k}}$, $z = Z(x)$ *and* $\mathcal{L}_{g\text{-}SCE}(z, y) = C$, *if there are* $\Sigma_k = \sigma_k I$, $\Sigma_{\tilde{k}} = \sigma_{\tilde{k}} I$, *and* $\sigma_k \neq \sigma_{\tilde{k}}$, *then the sample density nearby the feature point* z *based on the approximation in Eq. (6) is*

$$\mathbb{SD}(z) \propto \frac{N_{k,\tilde{k}} \cdot p_{k,\tilde{k}}(C)}{\left[\mathbf{B}_{k,\tilde{k}} + \frac{\log(C_e - 1)}{\sigma_k - \sigma_{\tilde{k}}}\right]^{\frac{d-1}{2}}}, \text{ and } \mathbf{B}_{k,\tilde{k}} = \frac{\sigma_k \sigma_{\tilde{k}} \|\mu_k - \mu_{\tilde{k}}\|_2^2}{(\sigma_k - \sigma_{\tilde{k}})^2} + \frac{B_k - B_{\tilde{k}}}{\sigma_k - \sigma_{\tilde{k}}}, \tag{7}$$

where for the input-label pair in $\mathcal{D}_{k,\tilde{k}}$, *there is* $\mathcal{L}_{g\text{-}SCE} \sim p_{k,\tilde{k}}(c)$.



The case: $\sigma_k > \sigma_{\tilde{k}}$



The case: $\sigma_k < \sigma_{\tilde{k}}$

(Preferred by models since lower loss values)

The ‘Curse’ of Softmax Function

$$\mathcal{L}_{\text{g-SCE}}(Z(x), y) = -1_y^\top \log\left[\underline{\text{softmax}}(h)\right],$$

- **The softmax makes the loss value only depend on the relative relation among logits.**
- **This causes indirect and unexpected supervisory signals on the learned features.**

Our Method: Max-Mahalanobis Center (MMC) Loss

$$\mathcal{L}_{\mathrm{MMLDA}}(Z(x), y) = -\log\left[\frac{\exp(-\frac{\|z-\mu_y^*\|_2^2}{2})}{\sum_{l\in[L]}\exp(-\frac{\|z-\mu_l^*\|_2^2}{2})}\right] = -\log\left[\frac{\exp(z^\top\mu_y^*)}{\sum_{l\in[L]}\exp(z^\top\mu_l^*)}\right]$$

Our Method: Max-Mahalanobis Center (MMC) Loss

$$\mathcal{L}_{\mathrm{MMLDA}}(Z(x), y) = -\log\left[\frac{\exp(-\frac{\|z-\mu_y^*\|_2^2}{2})}{\sum_{l\in[L]}\exp(-\frac{\|z-\mu_l^*\|_2^2}{2})}\right] = -\log\left[\frac{\exp(z^\top\mu_y^*)}{\sum_{l\in[L]}\exp(z^\top\mu_l^*)}\right]$$

$$\mathcal{L}_{\mathrm{MMC}}(Z(x), y) = \frac{1}{2}\|z-\mu_y^*\|_2^2$$

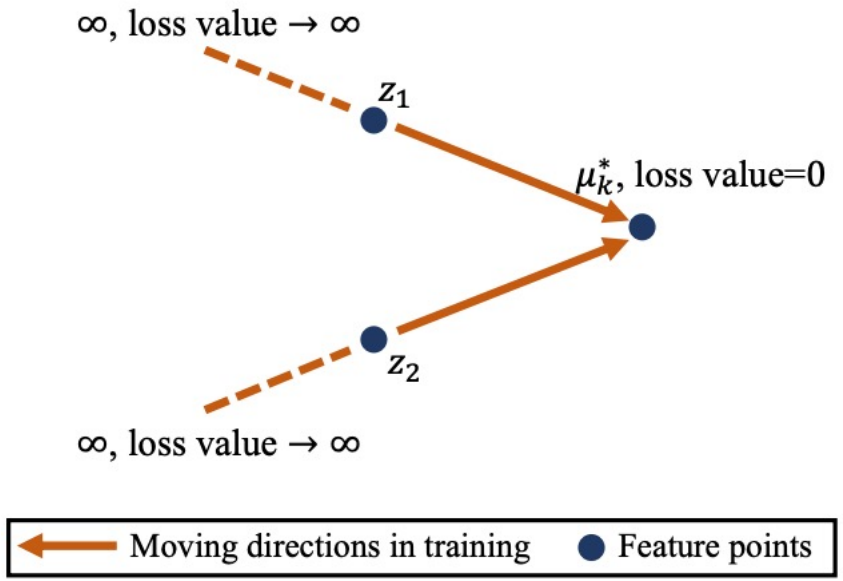
- **No softmax normalization**

Induced Sample Density of MMC Loss

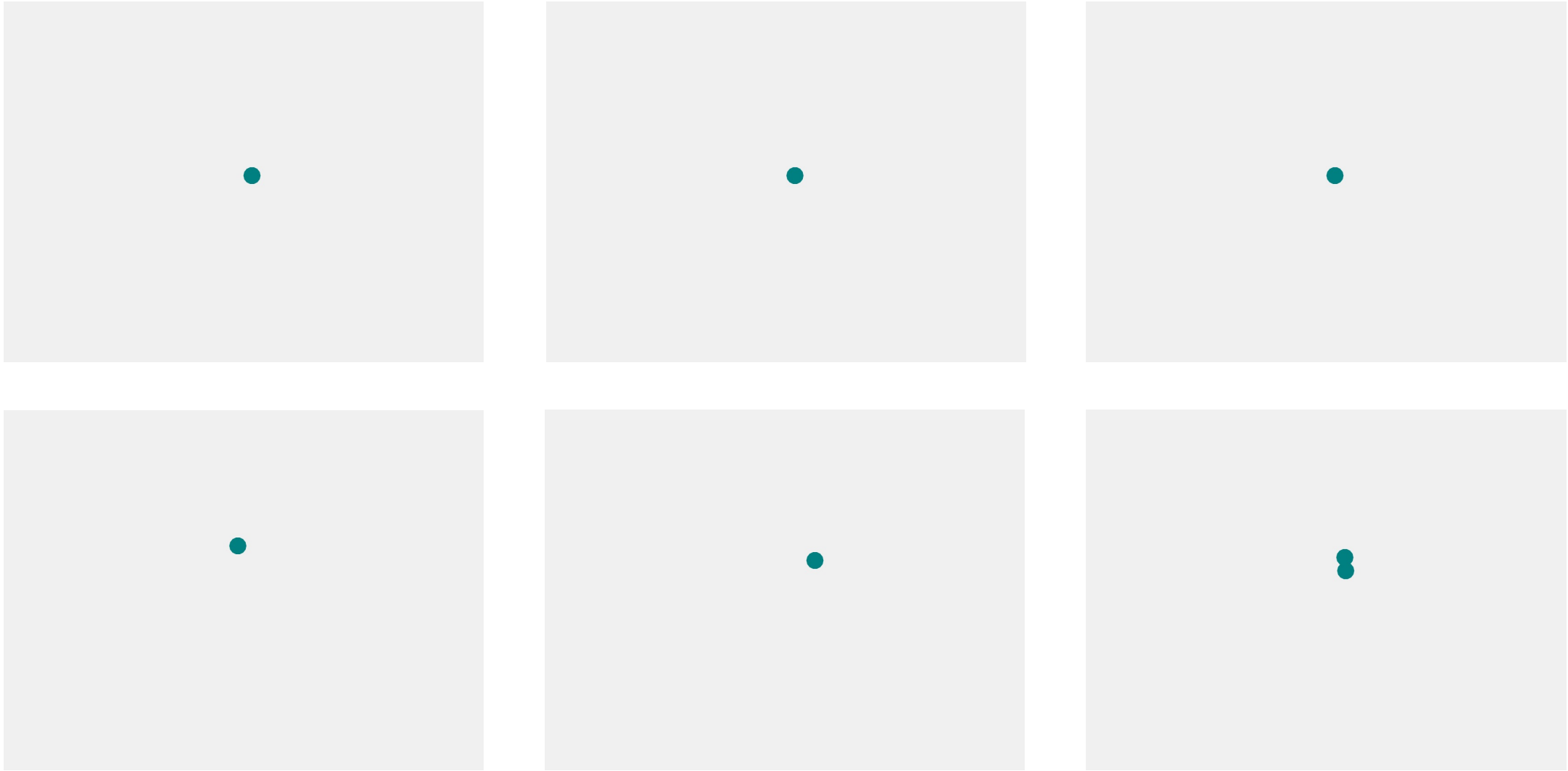
Theorem 2. *(Proof in Appendix A.2) Given* $(x, y) \in \mathcal{D}_k$, $z = Z(x)$ *and* $\mathcal{L}_{MMC}(z, y) = C$, *the sample density nearby the feature point* z *is*

$$\mathbb{SD}(z) \propto \frac{N_k \cdot p_k(C)}{C^{\frac{d-1}{2}}}, \tag{9}$$

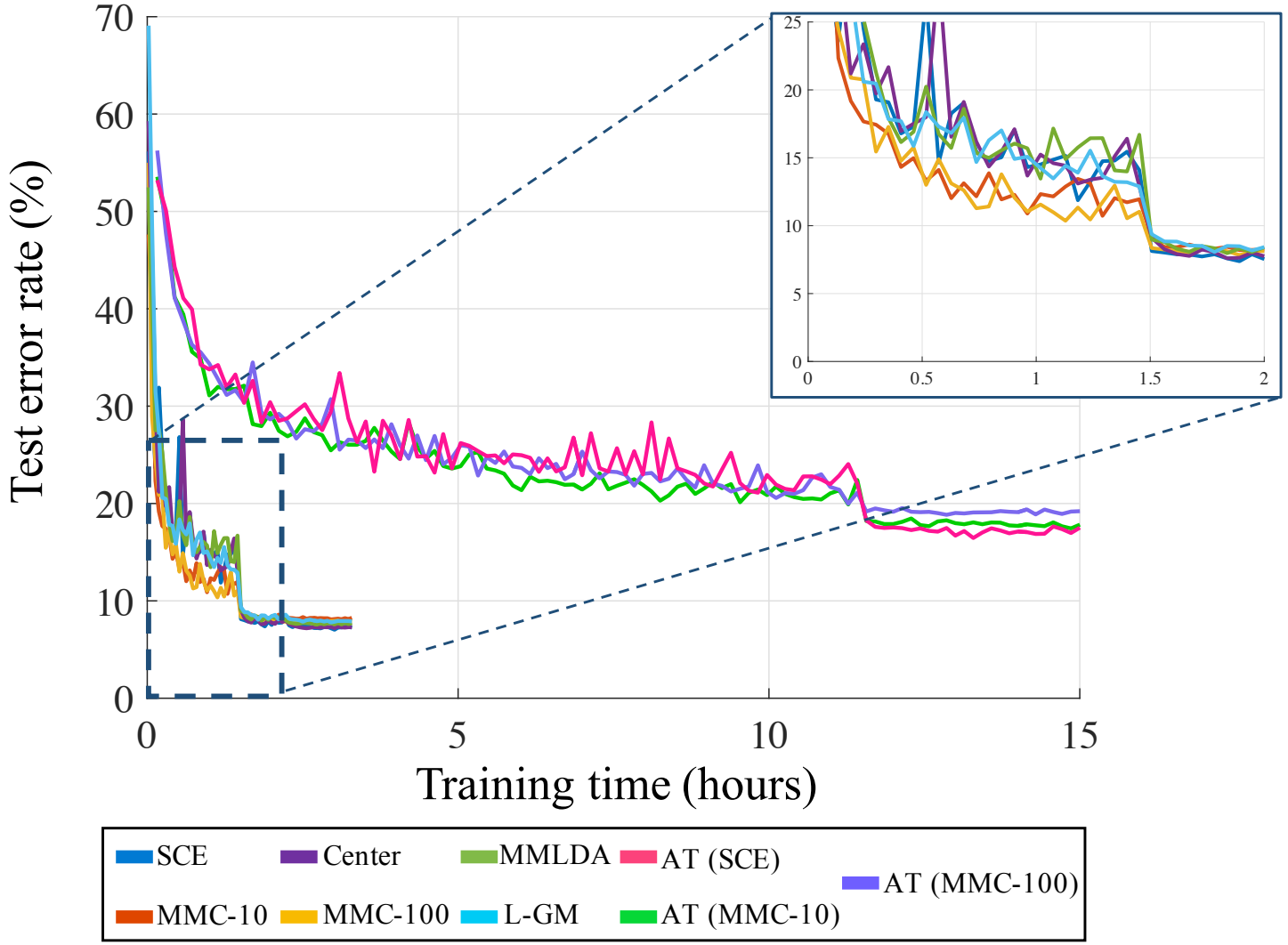
where for the input-label pair in $\mathcal{D}_k$, *there is* $\mathcal{L}_{MMC} \sim p_k(c)$.



Toy Demo on Faster Convergence



Empirical Faster Convergence

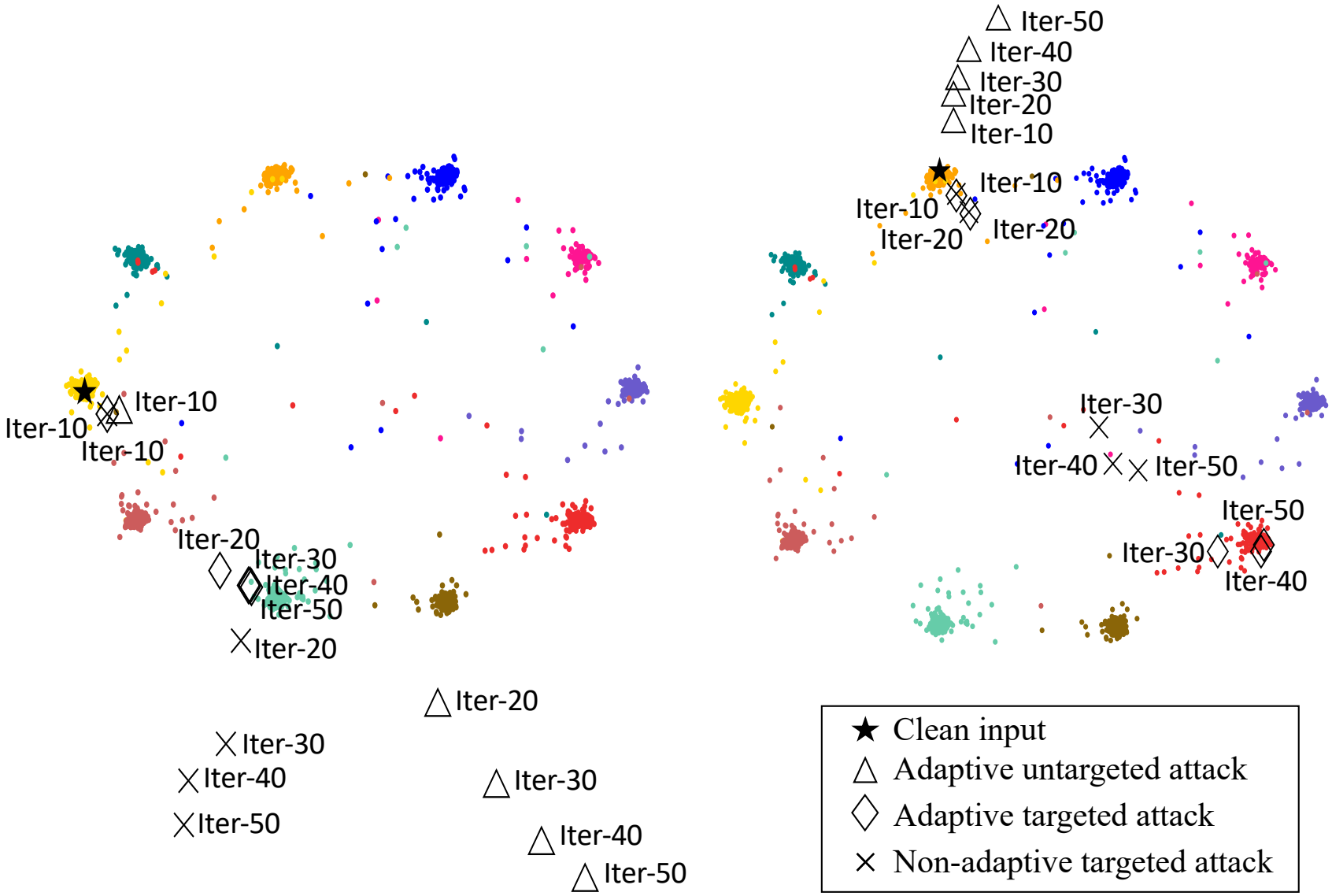


White-box Robustness (Adaptive Attacks)

Methods	Clean	Perturbation $\epsilon = 8/255$				Perturbation $\epsilon = 16/255$			
		$\text{PGD}^{\text{tar}}_{10}$	$\text{PGD}^{\text{un}}_{10}$	$\text{PGD}^{\text{tar}}_{50}$	$\text{PGD}^{\text{un}}_{50}$	$\text{PGD}^{\text{tar}}_{10}$	$\text{PGD}^{\text{un}}_{10}$	$\text{PGD}^{\text{tar}}_{50}$	$\text{PGD}^{\text{un}}_{50}$
SCE	92.9	≤ 1	3.7	≤ 1	3.6	≤ 1	2.9	≤ 1	2.6
Center loss	92.8	≤ 1	4.4	≤ 1	4.3	≤ 1	3.1	≤ 1	2.9
MMLDA	92.4	≤ 1	16.5	≤ 1	9.7	≤ 1	6.7	≤ 1	5.5
L-GM	92.5	37.6	19.8	8.9	4.9	26.0	11.0	2.5	2.8
MMC-10 (rand)	92.3	43.5	29.2	20.9	18.4	31.3	17.9	8.6	11.6
MMC-10	92.7	**48.7**	**36.0**	**26.6**	**24.8**	**36.1**	**25.2**	**13.4**	**17.5**
$\text{AT}^{\text{tar}}_{10}$ (SCE)	83.7	**70.6**	49.7	**69.8**	47.8	48.4	26.7	31.2	16.0
$\text{AT}^{\text{tar}}_{10}$ (**MMC-10**)	83.0	69.2	**54.8**	67.0	**53.5**	**58.6**	**47.3**	**44.7**	**45.1**
$\text{AT}^{\text{un}}_{10}$ (SCE)	80.9	69.8	55.4	69.4	53.9	53.3	34.1	38.5	21.5
$\text{AT}^{\text{un}}_{10}$ (**MMC-10**)	81.8	**70.8**	**56.3**	**70.1**	**55.0**	**54.7**	**37.4**	**39.9**	**27.7**

CIFAR-10

White-box Robustness (Adaptive Attacks)

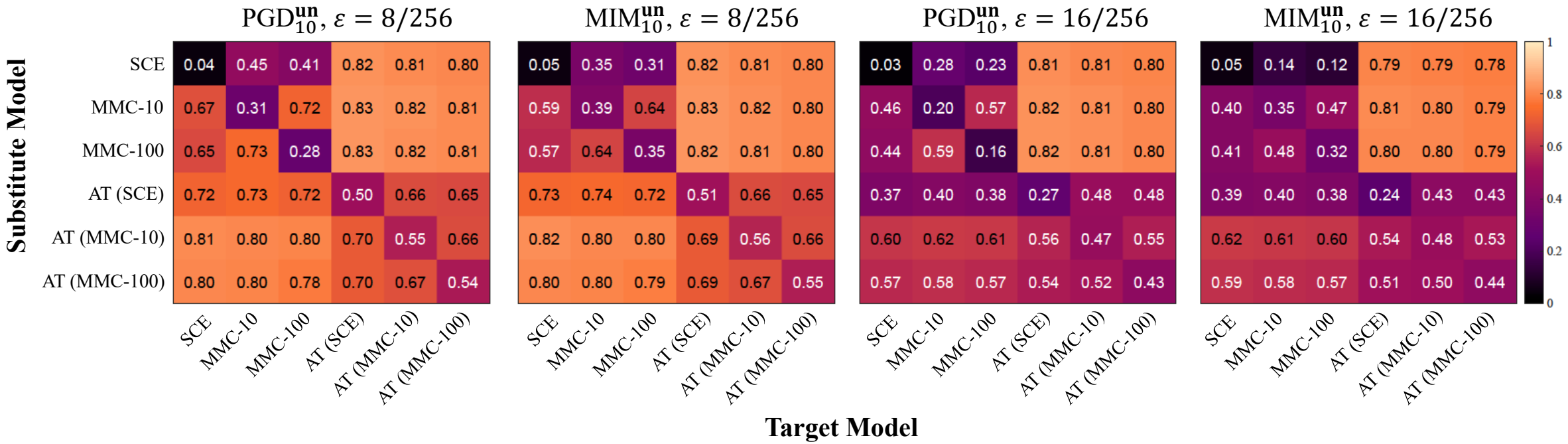


White-box Robustness (Adaptive Attacks)

Methods	Part I		Part II ($\epsilon = 8/255$)		Part II ($\epsilon = 16/255$)		Part III	
	$C\&W^{tar}$	$C\&W^{un}$	$SPSA^{tar}_{10}$	$SPSA^{un}_{10}$	$SPSA^{tar}_{10}$	$SPSA^{un}_{10}$	Noise	Rotation
SCE	0.12	0.07	12.3	1.2	5.1	≤ 1	52.0	83.5
Center loss	0.13	0.07	21.2	6.0	10.6	2.0	55.4	84.9
MMLDA	0.17	0.10	25.6	13.2	11.3	5.7	57.9	84.8
L-GM	0.23	0.12	61.9	45.9	46.1	28.2	59.2	82.4
MMC-10	**0.34**	**0.17**	**69.5**	**56.9**	**57.2**	**41.5**	**69.3**	**87.2**
AT^{tar}_{10} (SCE)	1.19	0.63	**81.1**	67.8	**77.9**	59.4	82.2	**76.0**
AT^{tar}_{10} (MMC-10)	**1.91**	**0.85**	79.1	**69.2**	74.5	**62.7**	**83.5**	75.2
AT^{un}_{10} (SCE)	1.26	0.68	78.8	67.0	73.7	60.3	78.9	73.7
AT^{un}_{10} (MMC-10)	**1.55**	**0.73**	**80.4**	**69.6**	**74.6**	**62.4**	**80.3**	**75.8**

CIFAR-10

Black-box Robustness (Exclude Gradient Masking)



Different Architectures

Methods	Cle.	Perturbation $\epsilon = 8/255$				Perturbation $\epsilon = 16/255$			
		PGD_{10}^{tar}	PGD_{10}^{un}	PGD_{50}^{tar}	PGD_{50}^{un}	PGD_{10}^{tar}	PGD_{10}^{un}	PGD_{50}^{tar}	PGD_{50}^{un}
CIFAR-10									
SCE (Res.32)	93.6	≤ 1	3.7	≤ 1	3.6	≤ 1	2.7	≤ 1	2.9
MMC (Res.32)	92.7	**48.7**	**36.0**	**26.6**	**24.8**	**36.1**	**25.2**	**13.4**	**17.5**
SCE (Res.110)	94.7	≤ 1	3.0	≤ 1	2.9	≤ 1	2.1	≤ 1	2.0
MMC (Res.110)	93.6	**54.7**	**46.0**	**34.4**	**31.4**	**41.0**	**30.7**	**16.2**	**21.6**
CIFAR-100									
SCE (Res.32)	72.3	≤ 1	7.8	≤ 1	7.4	≤ 1	4.8	≤ 1	4.7
MMC (Res.32)	71.9	**23.9**	**23.4**	**15.1**	**21.9**	**16.4**	**16.7**	**8.0**	**15.7**
SCE (Res.110)	74.8	≤ 1	7.5	≤ 1	7.3	≤ 1	4.7	≤ 1	4.5
MMC (Res.110)	73.2	**34.6**	**22.4**	**23.7**	**16.5**	**24.1**	**14.9**	**13.9**	**10.5**

Thanks