# Robust Machine Learning in the Adversarial Setting

Tianyu Pang（庞天宇）

Department of Computer Science and Technology
Tsinghua University

# Joint work with

**Prof. Jun Zhu**

**Tianyu Pang**

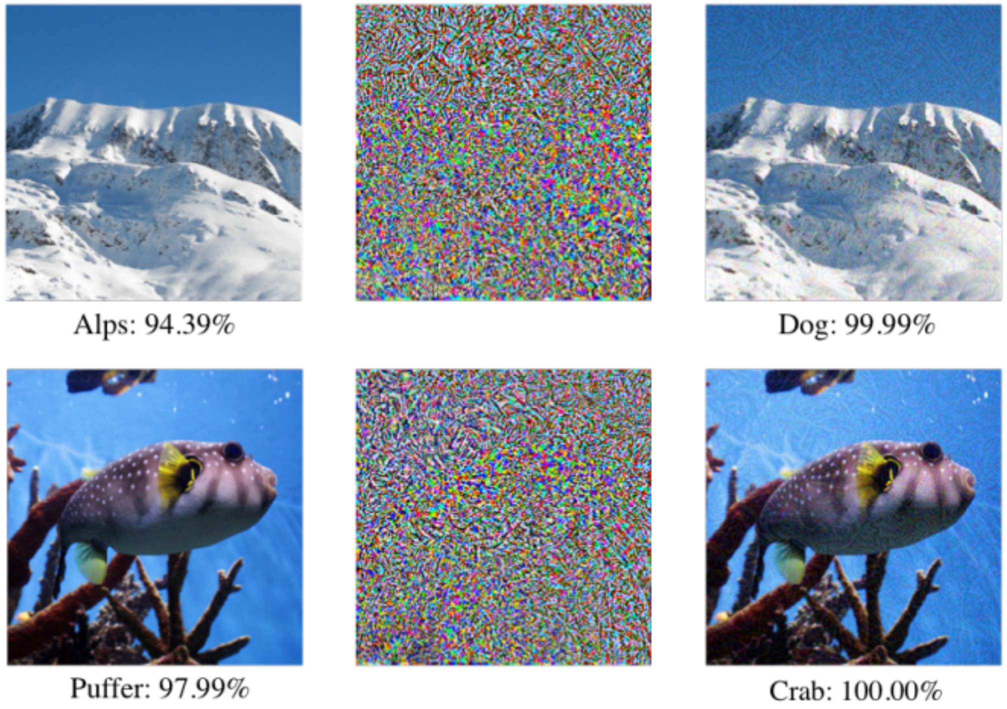
**Kun Xu**

**Chao Du**

**Yinpeng Dong**

**Xiao Yang**

# Adversarial Examples in Computer Vision

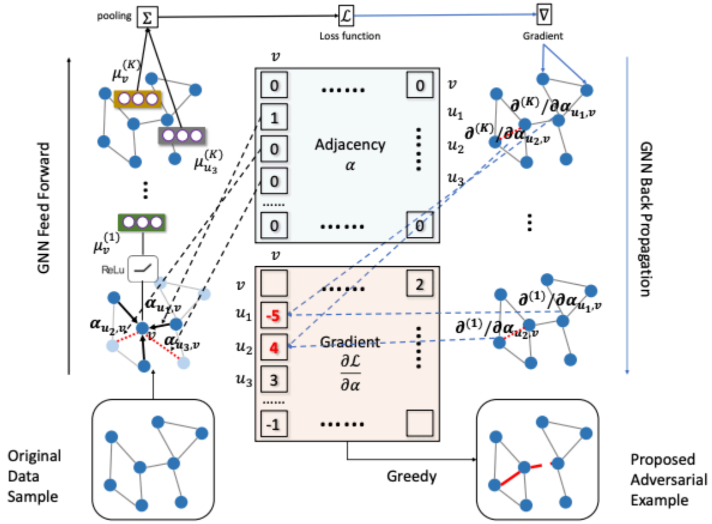



**(Dong et al. CVPR 2018)**

# 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)**

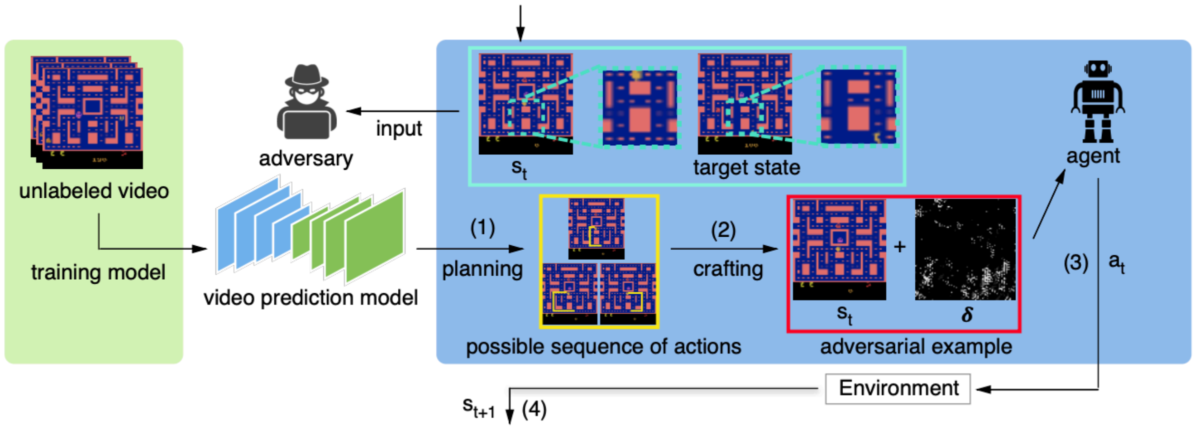



**GNN model (Dai et al. ICML 2018)**

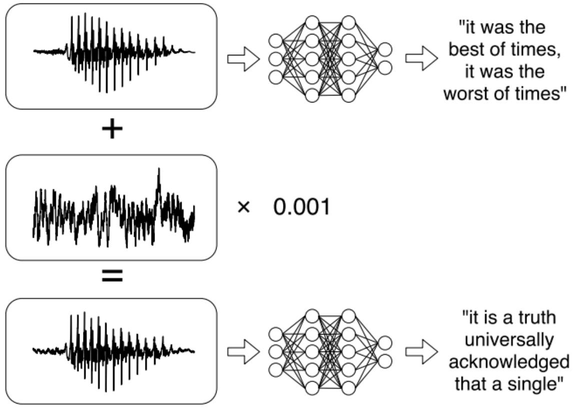
**Recommend System,**

**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)**

**Improving Adversarial Robustness via Promoting Ensemble Diversity**
**(ICML 2019)**

## Possible strategy two:

**Towards Robust Detection of Adversarial Examples**
**(NeurIPS 2018)**

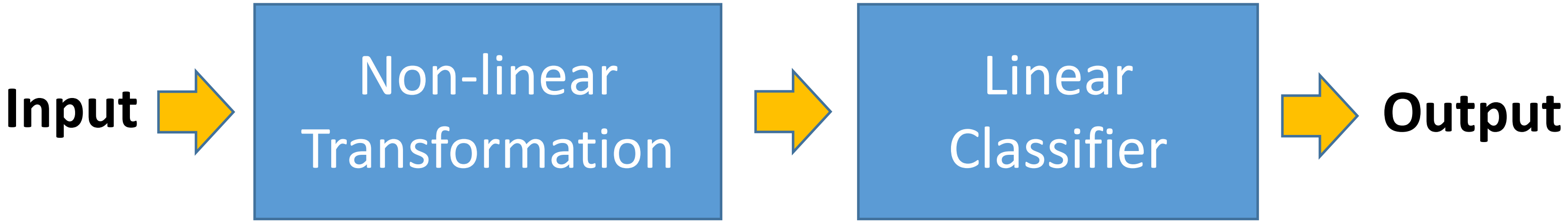
# Max-Mahalanobis Training

## Part I

## (ICML 2018)

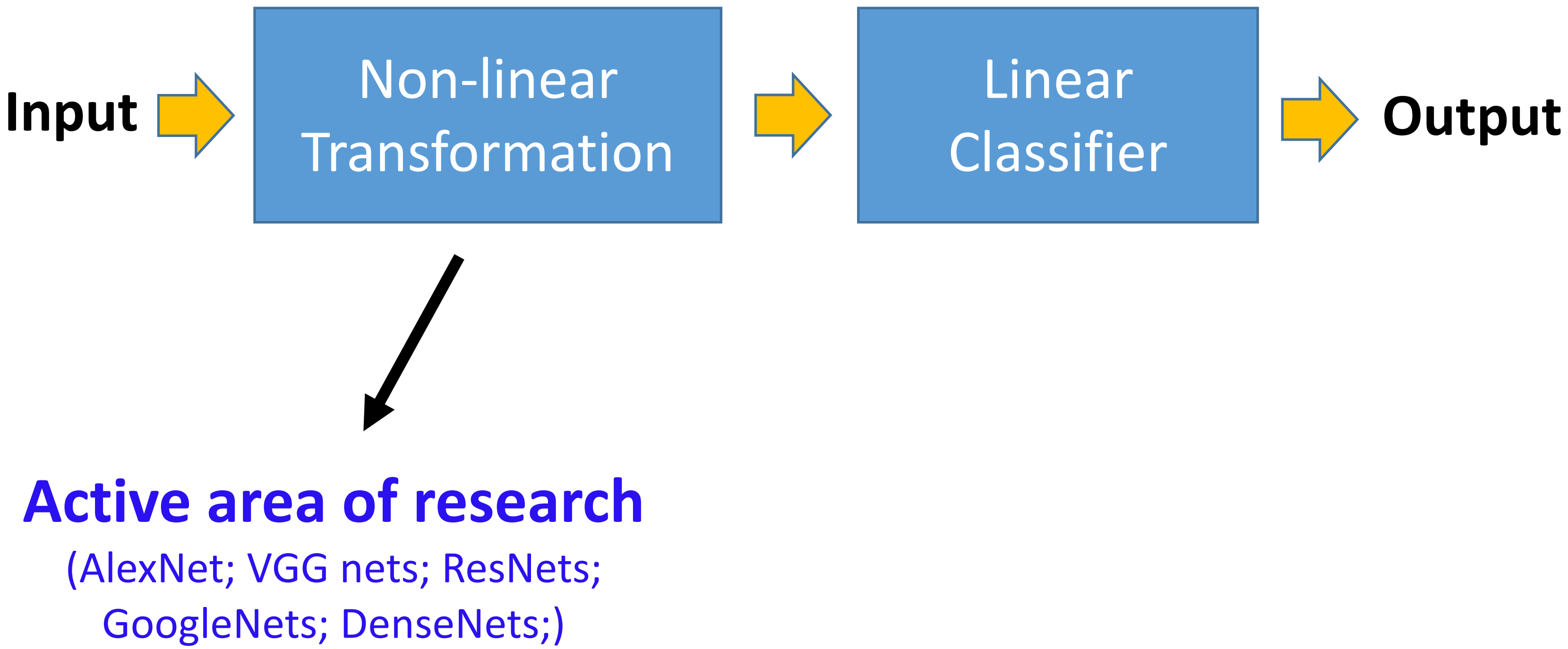
# Motivation

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



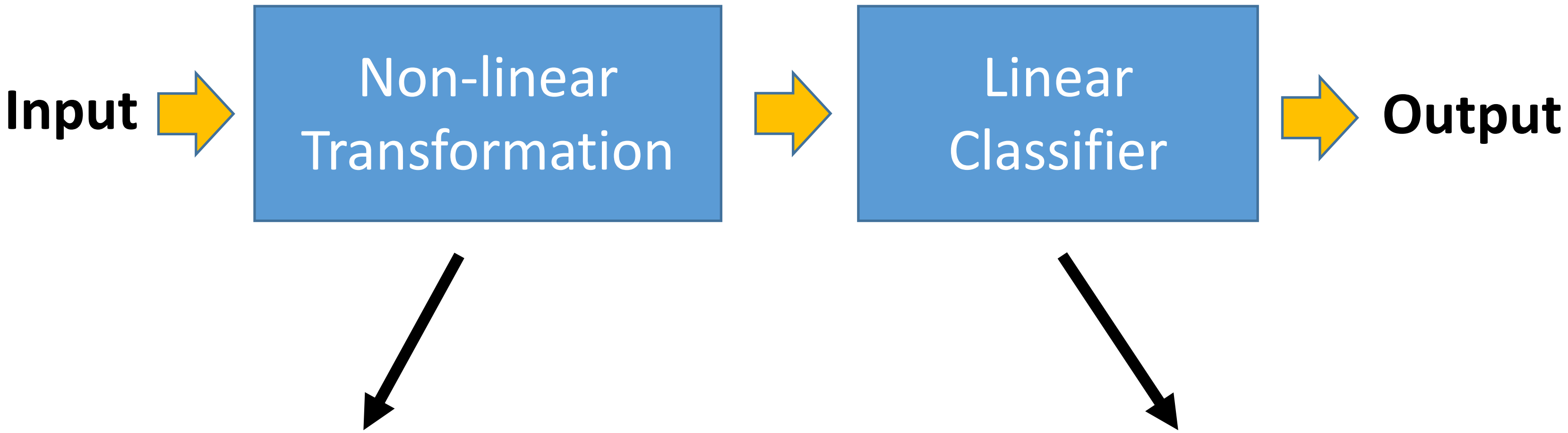
# Motivation

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



# 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

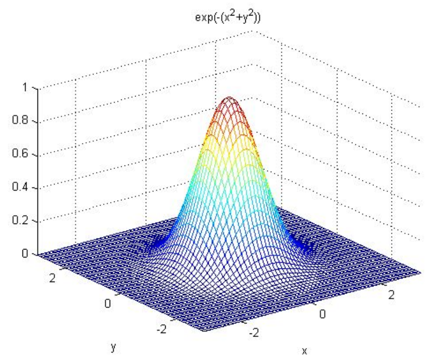
- Efron et al.(1975) show that *if the input distributes as a mixture of Gaussian*, then linear discriminant analysis (LDA) is **more efficient** than logistic regression (LR).

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

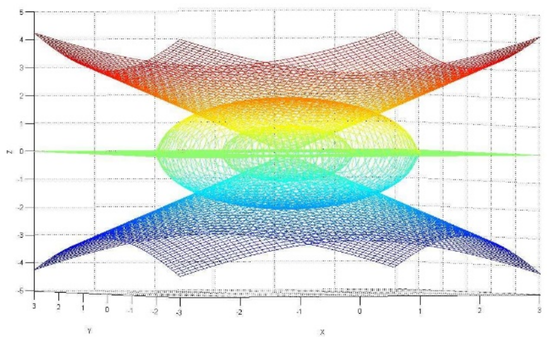
- Deep generative models (e.g., GANs) are successful.




Simple Distribution
(Gaussian/Mixture of Gaussian)

**Deep generative models**

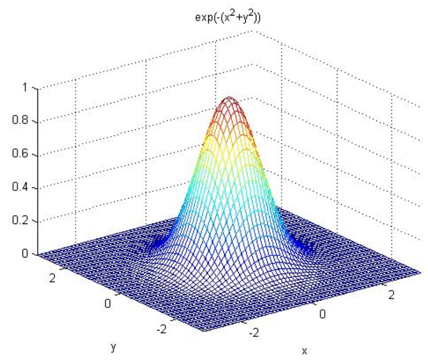
**DNN**



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.**

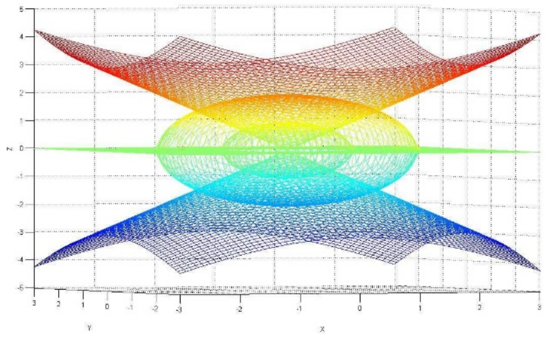



Simple Distribution
(Gaussian/Mixture of Gaussian)

**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 ($\mu_i$ and $\Sigma$) 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 ($\mu_i$ and $\Sigma$) 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?

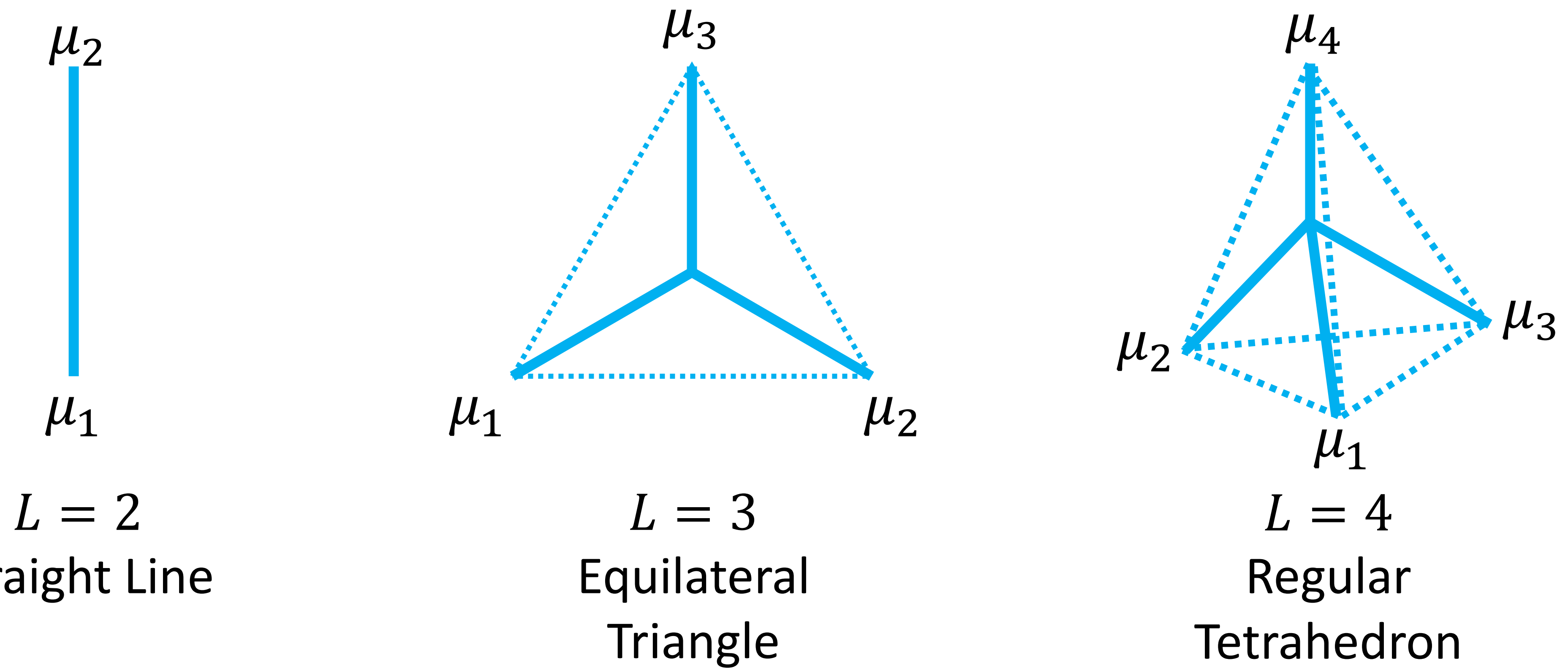
- Wan et al. (CVPR 2018) also model the feature distribution as a mixture of Gaussian. However, they treat the Gaussian parameters ($\mu_i$ and $\Sigma$) as extra trainable variables.

- 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.**



# 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 $\overline{\text{RB}}$.**

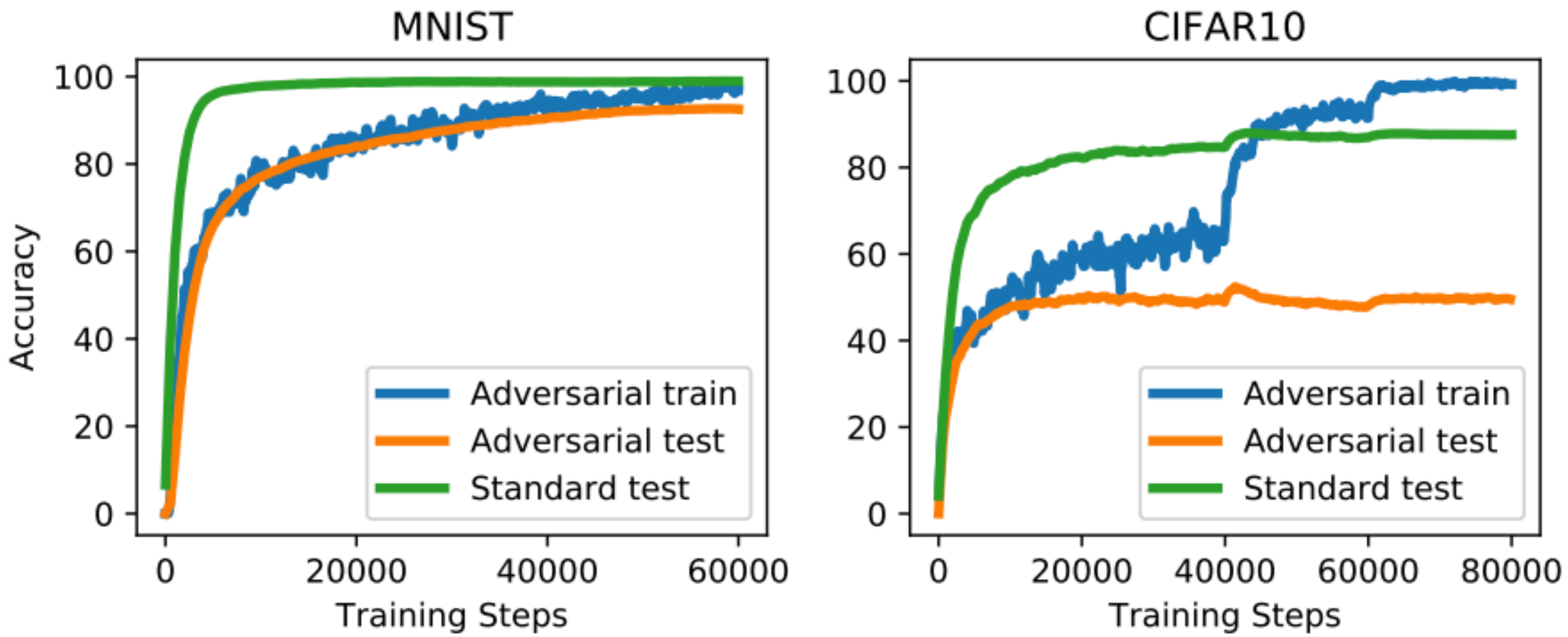
# Can we further improve MMLDA?

# 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?**

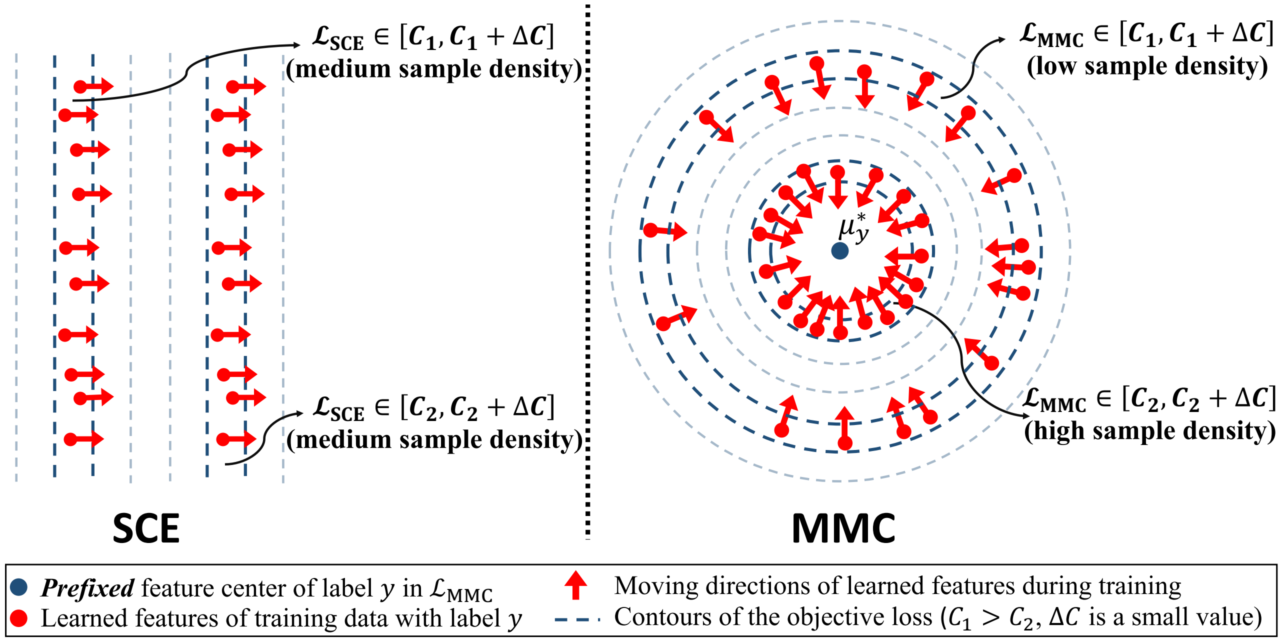
**Q2: Can existing training objectives induce high sample density?**

# 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

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

Here $\text{Vol}(\cdot)$ denotes the volume of the input set, $\Delta B$ 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 $\Delta B$, 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**

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

$$\Downarrow$$

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

# 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**

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

$$\Downarrow$$

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

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**

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

$$\downarrow$$

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

$\downarrow$ 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**

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

$\Updownarrow$ **approximately** $\qquad\qquad$ $\Updownarrow$ **approximately**

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

# The Neighborhood $\Delta B$ 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**

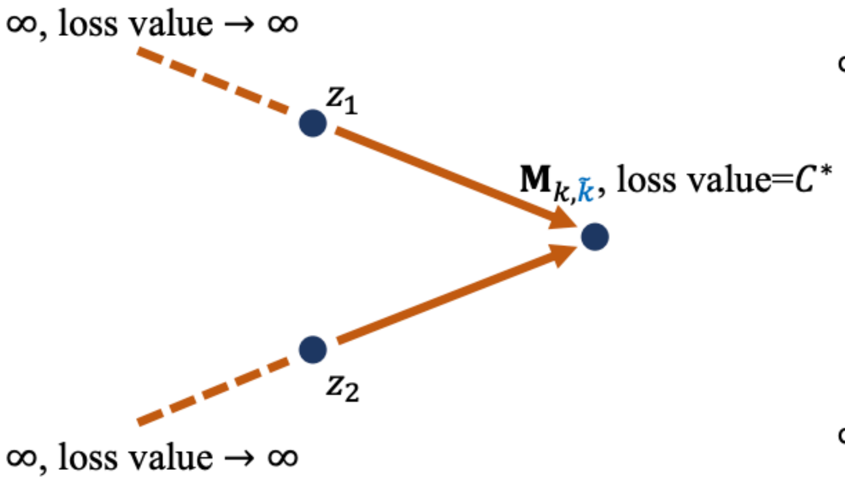
$$\Delta B_{y,\tilde{y}} = \{\mathbf{z} \in \mathbb{R}^d | \mathcal{L}_{y,\tilde{y}}(\mathbf{z}) \in [C, C + \Delta C]\}$$

# Induced Sample Density of g-SCE Loss

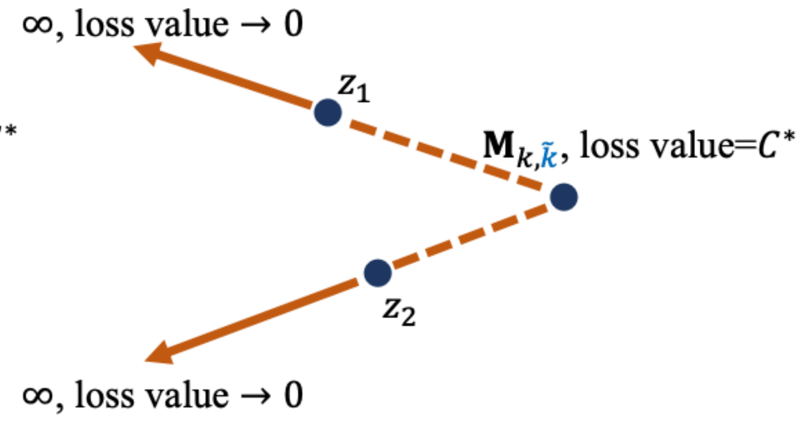
**Theorem 1.** *(Proof in Appendix A.1) Given* $(x, y) \in \mathcal{D}_{k,\tilde{k}}$, $z = Z(x)$ *and* $\mathcal{L}_{g\text{-}SCE}(z, y) = C$, *if there are* $\Sigma_k = \sigma_k I$, $\Sigma_{\tilde{k}} = \sigma_{\tilde{k}} I$, *and* $\sigma_k \neq \sigma_{\tilde{k}}$, *then the sample density nearby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




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

**(Preferred by models since lower loss values)**

# The ‘Curse’ of Softmax Function

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

- 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

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

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

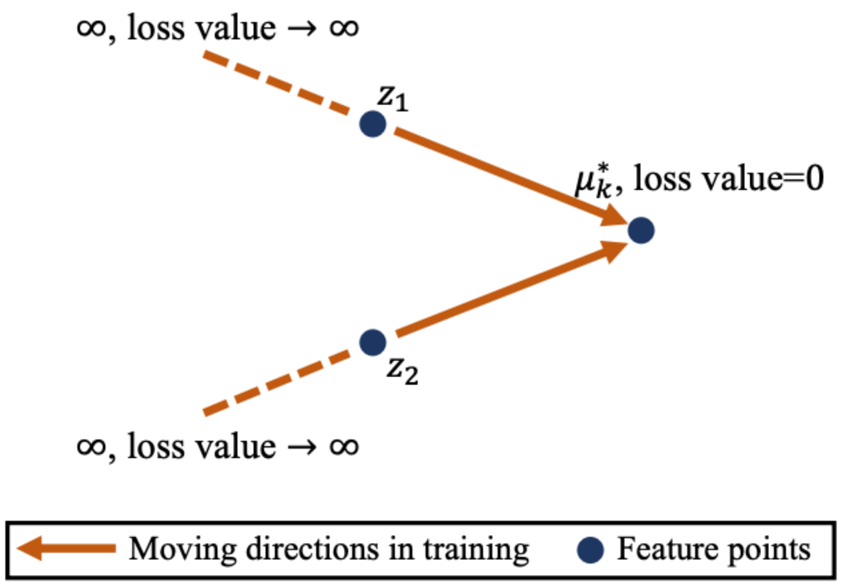
- **No softmax normalization**

# 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*

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

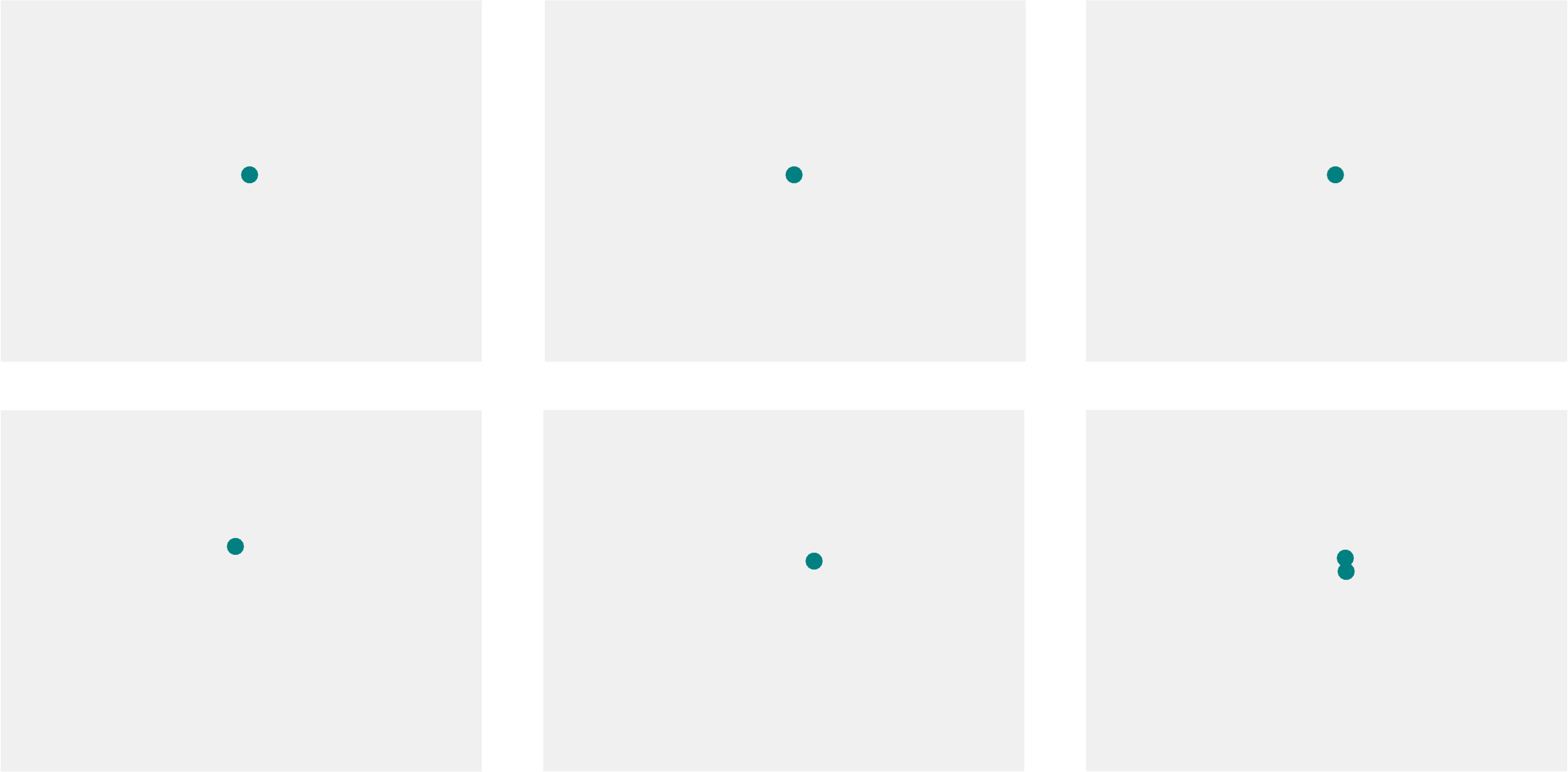
*where for the input-label pair in* $\mathcal{D}_k$*, there is* $\mathcal{L}_{MMC} \sim p_k(c)$.



# Toy Demo on Faster Convergence

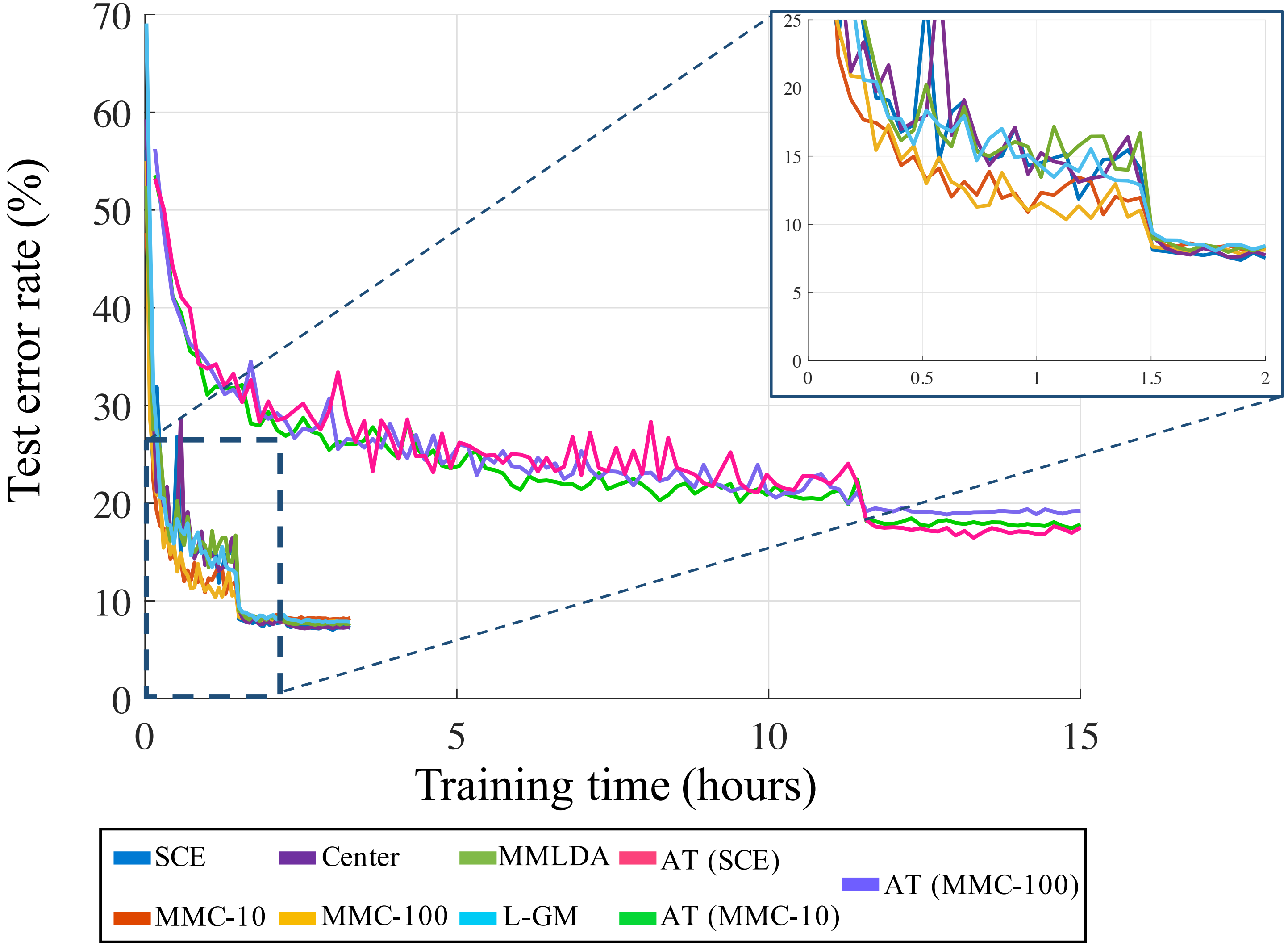
Center loss

MMC loss



Full-batch | Mini-batch 20/1000 | Mini-batch 5/1000

# Empirical Faster Convergence

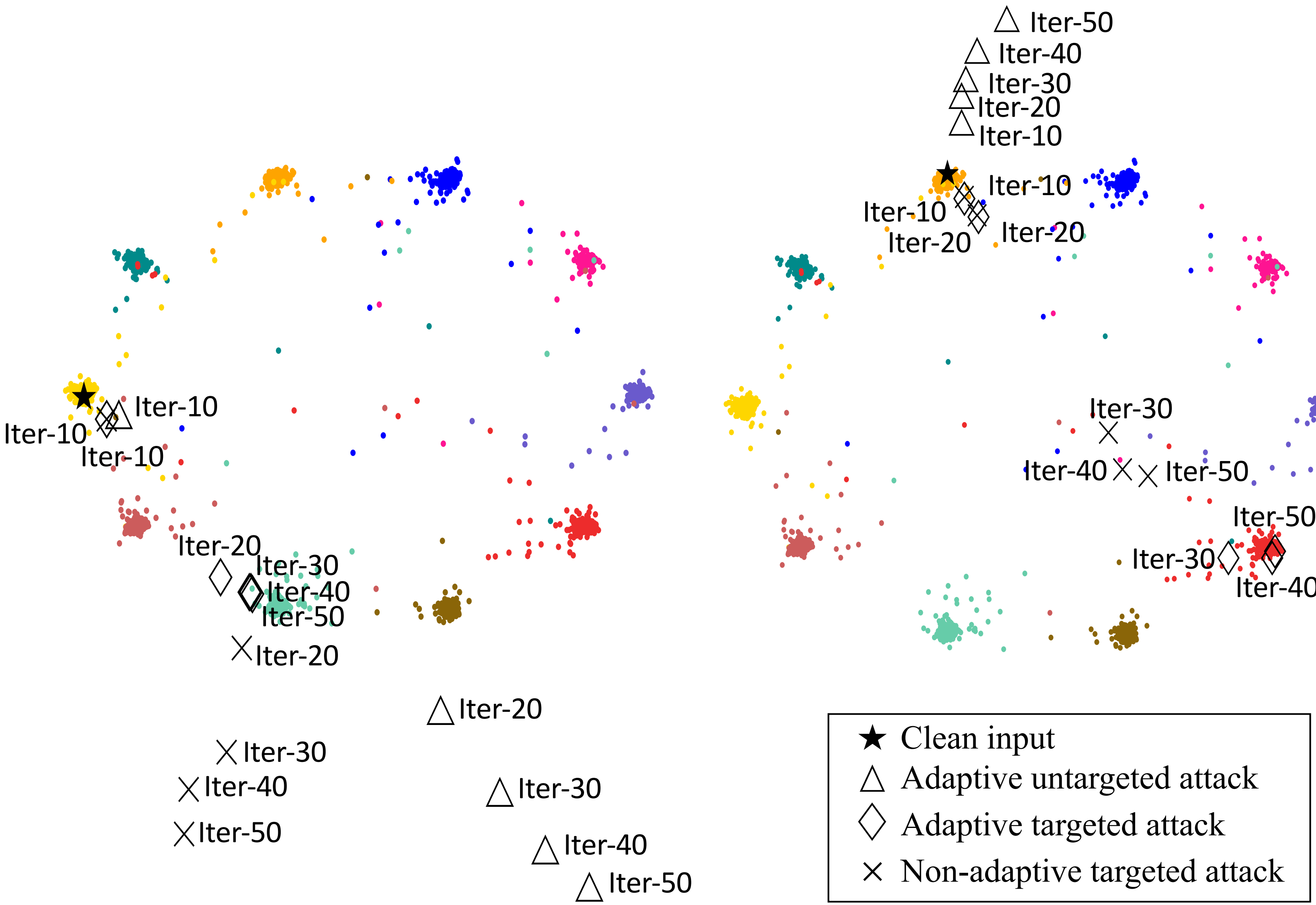


# White-box Robustness (Adaptive Attacks)

| Methods | Clean | Perturbation $\epsilon = 8/255$ | | | | Perturbation $\epsilon = 16/255$ | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | $\text{PGD}_{10}^{\text{tar}}$ | $\text{PGD}_{10}^{\text{un}}$ | $\text{PGD}_{50}^{\text{tar}}$ | $\text{PGD}_{50}^{\text{un}}$ | $\text{PGD}_{10}^{\text{tar}}$ | $\text{PGD}_{10}^{\text{un}}$ | $\text{PGD}_{50}^{\text{tar}}$ | $\text{PGD}_{50}^{\text{un}}$ |
| SCE | 92.9 | $\leq 1$ | 3.7 | $\leq 1$ | 3.6 | $\leq 1$ | 2.9 | $\leq 1$ | 2.6 |
| Center loss | 92.8 | $\leq 1$ | 4.4 | $\leq 1$ | 4.3 | $\leq 1$ | 3.1 | $\leq 1$ | 2.9 |
| MMLDA | 92.4 | $\leq 1$ | 16.5 | $\leq 1$ | 9.7 | $\leq 1$ | 6.7 | $\leq 1$ | 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** |
| $\text{AT}_{10}^{\text{tar}}$ (SCE) | 83.7 | **70.6** | 49.7 | **69.8** | 47.8 | 48.4 | 26.7 | 31.2 | 16.0 |
| $\text{AT}_{10}^{\text{tar}}$ (**MMC-10**) | 83.0 | 69.2 | **54.8** | 67.0 | **53.5** | **58.6** | **47.3** | **44.7** | **45.1** |
| $\text{AT}_{10}^{\text{un}}$ (SCE) | 80.9 | 69.8 | 55.4 | 69.4 | 53.9 | 53.3 | 34.1 | 38.5 | 21.5 |
| $\text{AT}_{10}^{\text{un}}$ (**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)

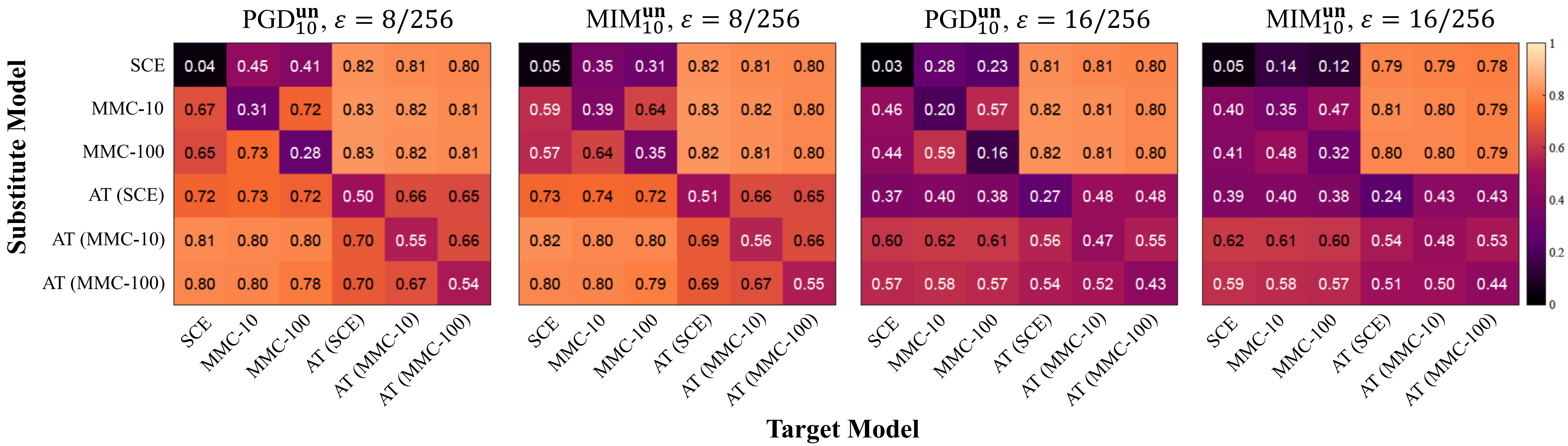


# White-box Robustness (Adaptive Attacks)

| Methods | Part I | | Part II ($\epsilon=8/255$) | | Part II ($\epsilon=16/255$) | | Part III | |
|---|---|---|---|---|---|---|---|---|
| | $C\&W^{tar}$ | $C\&W^{un}$ | $SPSA_{10}^{tar}$ | $SPSA_{10}^{un}$ | $SPSA_{10}^{tar}$ | $SPSA_{10}^{un}$ | Noise | Rotation |
| SCE | 0.12 | 0.07 | 12.3 | 1.2 | 5.1 | $\leq 1$ | 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** |
| $AT_{10}^{tar}$ (SCE) | 1.19 | 0.63 | **81.1** | 67.8 | **77.9** | 59.4 | 82.2 | **76.0** |
| $AT_{10}^{tar}$ (MMC-10) | **1.91** | **0.85** | 79.1 | **69.2** | 74.5 | **62.7** | **83.5** | 75.2 |
| $AT_{10}^{un}$ (SCE) | 1.26 | 0.68 | 78.8 | 67.0 | 73.7 | 60.3 | 78.9 | 73.7 |
| $AT_{10}^{un}$ (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 $\epsilon = 16/255$ | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | $\text{PGD}_{10}^{\text{tar}}$ | $\text{PGD}_{10}^{\text{un}}$ | $\text{PGD}_{50}^{\text{tar}}$ | $\text{PGD}_{50}^{\text{un}}$ | $\text{PGD}_{10}^{\text{tar}}$ | $\text{PGD}_{10}^{\text{un}}$ | $\text{PGD}_{50}^{\text{tar}}$ | $\text{PGD}_{50}^{\text{un}}$ |
| **CIFAR-10** | | | | | | | | | |
| SCE (Res.32) | 93.6 | $\leq 1$ | 3.7 | $\leq 1$ | 3.6 | $\leq 1$ | 2.7 | $\leq 1$ | 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 | $\leq 1$ | 3.0 | $\leq 1$ | 2.9 | $\leq 1$ | 2.1 | $\leq 1$ | 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 | $\leq 1$ | 7.8 | $\leq 1$ | 7.4 | $\leq 1$ | 4.8 | $\leq 1$ | 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 | $\leq 1$ | 7.5 | $\leq 1$ | 7.3 | $\leq 1$ | 4.7 | $\leq 1$ | 4.5 |
| MMC (Res.110) | 73.2 | **34.6** | **22.4** | **23.7** | **16.5** | **24.1** | **14.9** | **13.9** | **10.5** |

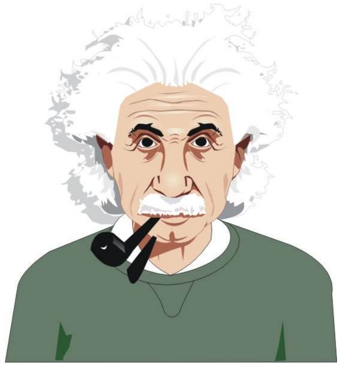
# Improving Adversarial Robustness via Promoting Ensemble Diversity

**(ICML 2019)**

# Previous Defense Strategies

## Single model defense:

e.g., adversarial training

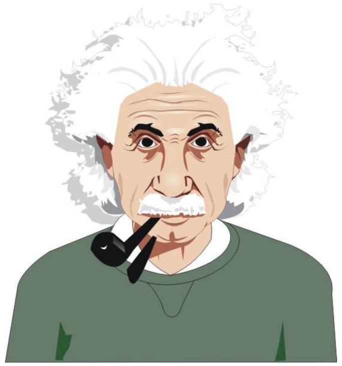


Base Model

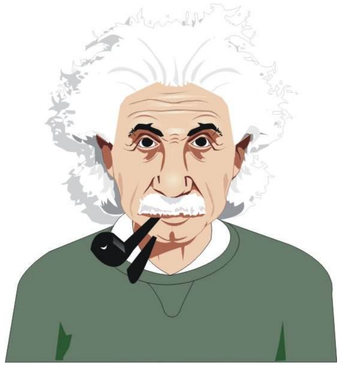
Enhanced Model

# Previous Defense Strategies

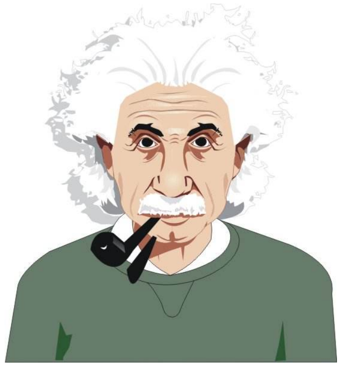
## Ensemble model defense:



Member 1



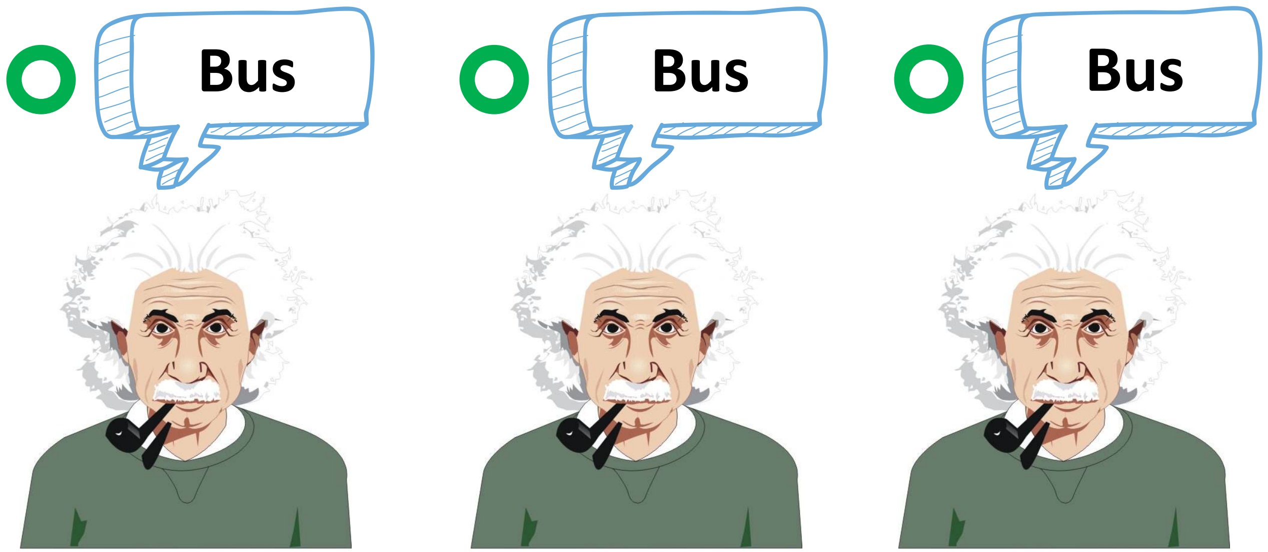
Member 2



Member 3

# Previous Defense Strategies

## Ensemble model defense:

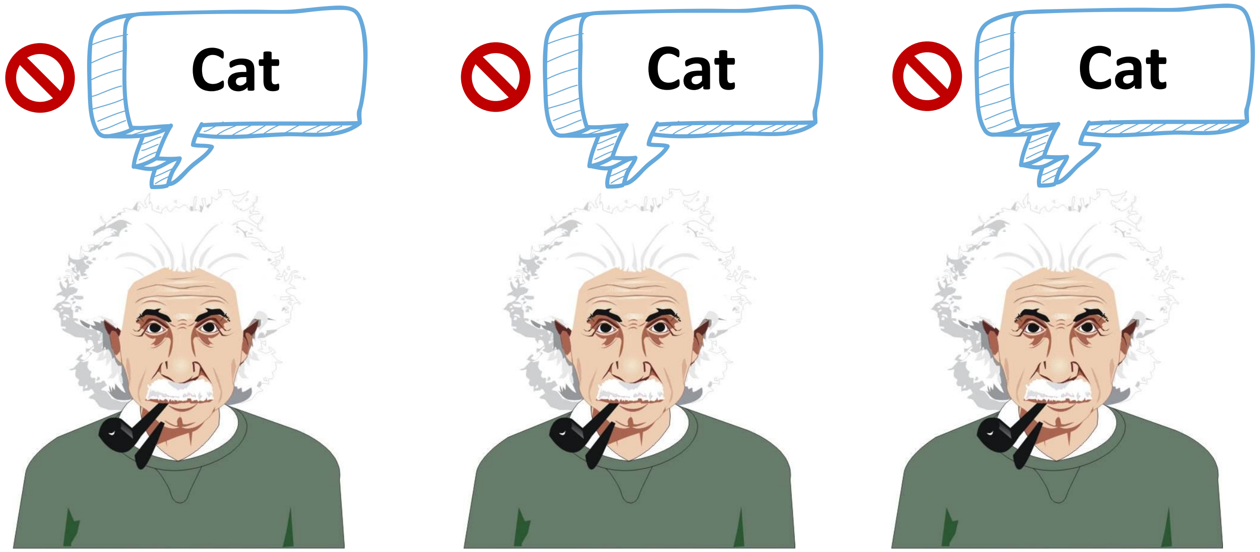



Member 1 Member 2 Member 3

Clean input

# Previous Defense Strategies

## Ensemble model defense:

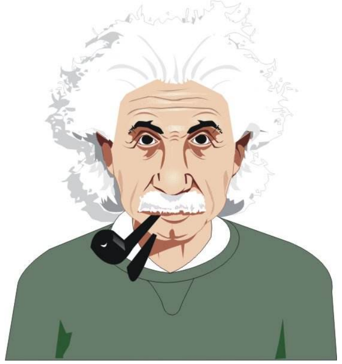



Member 1 Member 2 Member 3

Adversarial input

# Our Strategy

## Training ensembles with diversity:



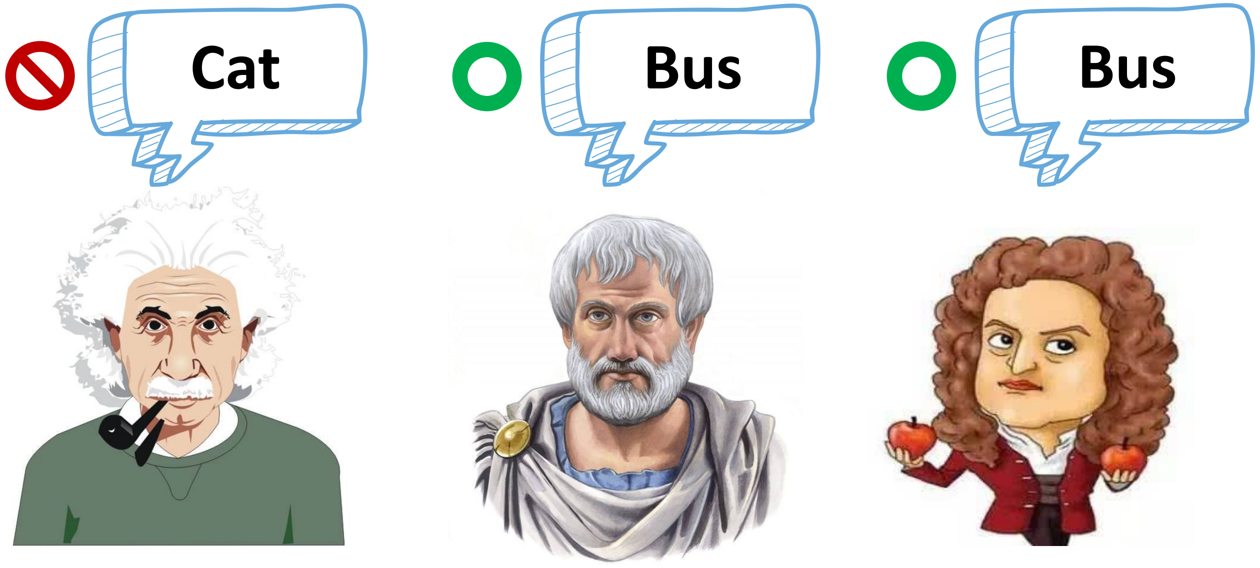
Member 1

Member 2

Member 3

# Our Strategy

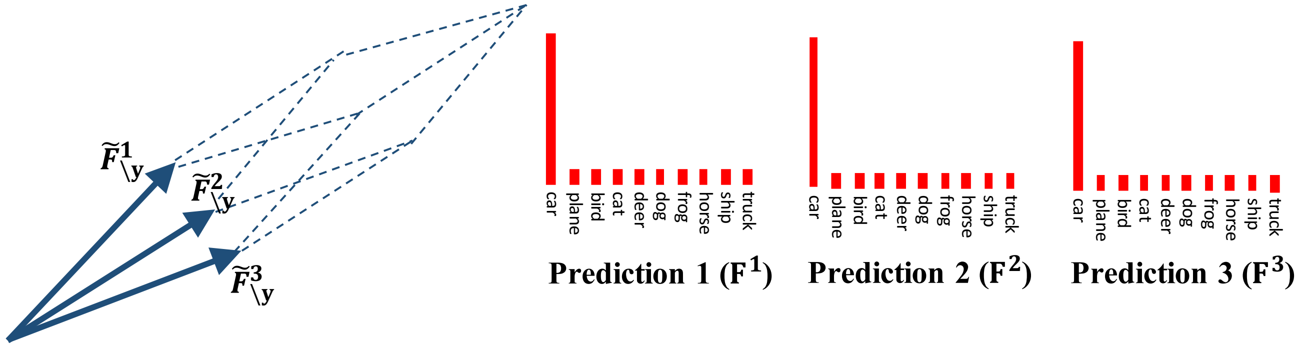
## Training ensembles with diversity:



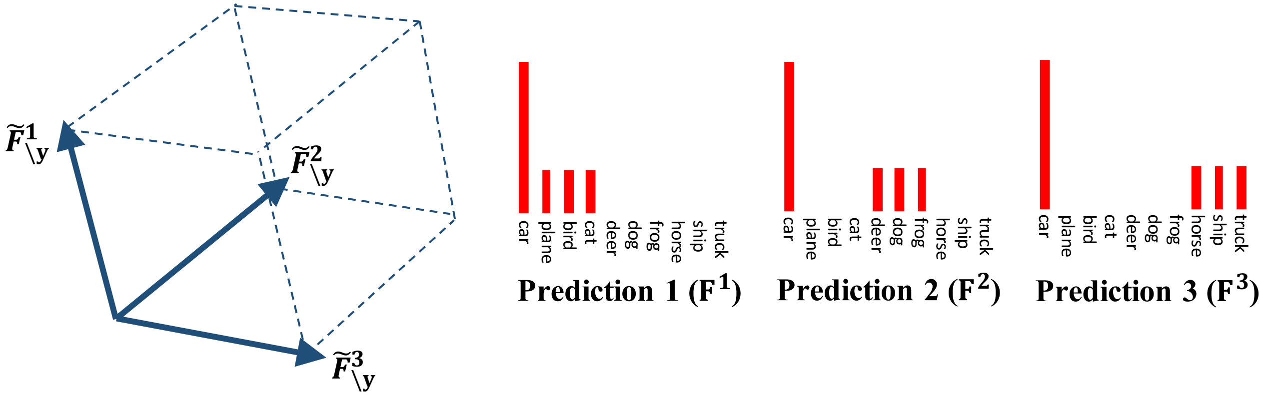

Member 1 Member 2 Member 3

Adversarial input

# Adaptive Diversity Promoting



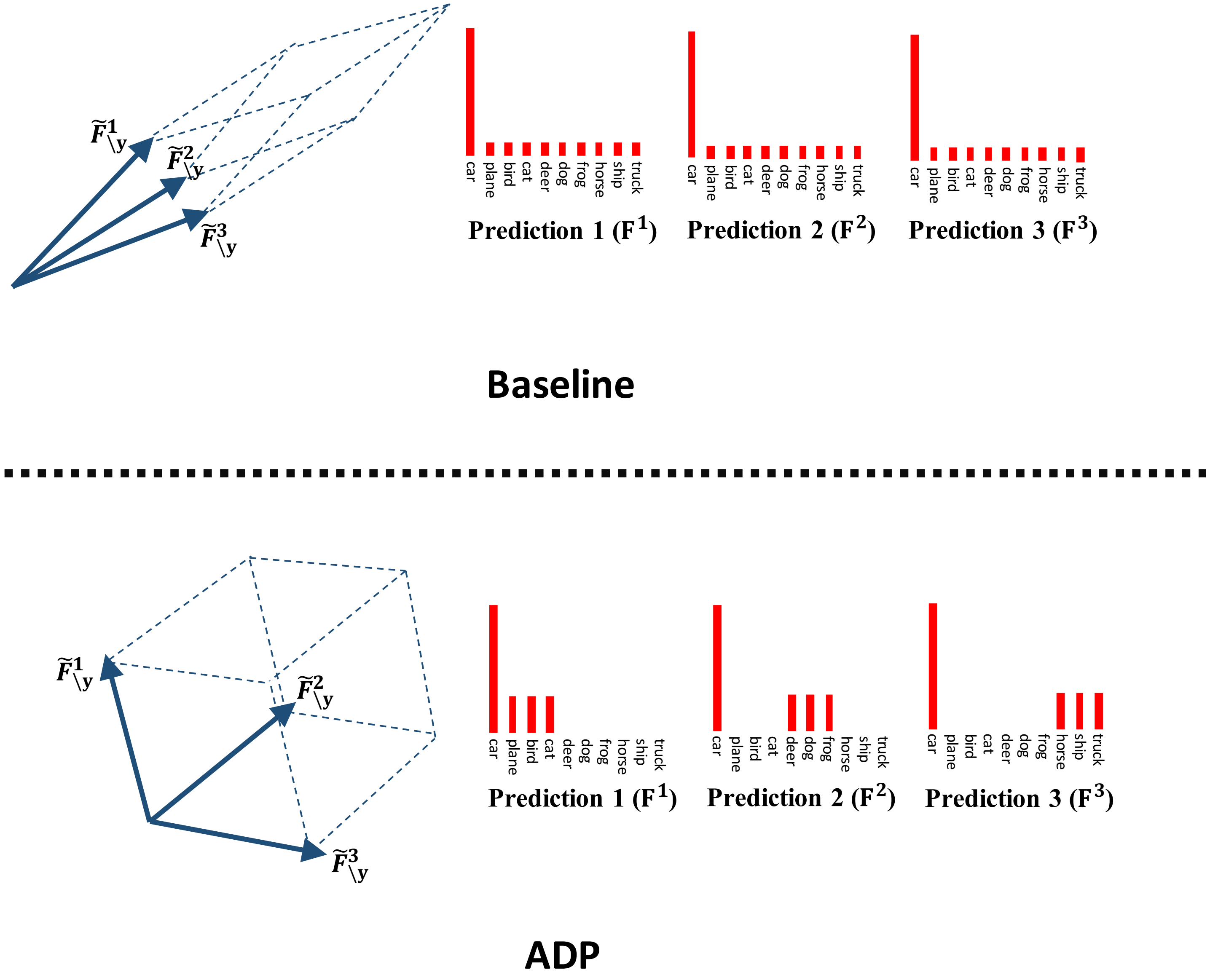

**Baseline**




**ADP**

- Promoting diversity on **non-maximal predictions**

# Adaptive Diversity Promoting




- Promoting diversity on **non-maximal predictions**

correspond to all potentially wrong labels returned for the adversarial examples

# Formulas of ADP

**Based on the intuitive insights, we define the ensemble diversity as**

$$\mathbb{ED} = \det(\tilde{M}_{\setminus y}^{\top} \tilde{M}_{\setminus y})$$

**where** $\tilde{M}_{\setminus y} = (\tilde{F}_{\setminus y}^{1}, \cdots, \tilde{F}_{\setminus y}^{K}) \in \mathbb{R}^{(L-1)\times K}$ **are normalized non-maximal prediction. This definition is based on the fact that**

$$\det(\tilde{M}_{\setminus y}^{\top} \tilde{M}_{\setminus y}) = \mathrm{Vol}^2(\{\tilde{F}_{\setminus y}^{k}\}_{k\in[K]})$$

# Formulas of ADP

**So the ADP regularizer is**

$$\mathrm{ADP}_{\alpha,\beta}(x, y) = \alpha \cdot \mathcal{H}(\mathcal{F}) + \beta \cdot \log\left(\mathbb{ED}\right)$$

# Formulas of ADP

**So the ADP regularizer is**

$$\mathrm{ADP}_{\alpha,\beta}(x,y) = \alpha \cdot \mathcal{H}(\mathcal{F}) + \beta \cdot \log(\mathbb{ED})$$

**Theorem 1.** *(Proof in Appendix A) If* $\alpha = 0$*, then* $\forall \beta \geq 0$*, the optimal solution of the minimization problem (6) satisfies the equations* $F^k = 1_y$*, where* $k \in [K]$.

# Formulas of ADP

**So the ADP regularizer is**

$$\mathrm{ADP}_{\alpha,\beta}(x,y) = \underline{\alpha \cdot \mathcal{H}(\mathcal{F})} + \beta \cdot \log(\mathbb{ED})$$

**Theorem 2.** *(Proof in Appendix A) When $\alpha > 0$ and $\beta = 0$, the optimal solution of the minimization problem (6) satisfies the equations $F_y^k = \mathcal{F}_y$, $\mathcal{F}_j = \frac{1-\mathcal{F}_y}{L-1}$ and*

$$\frac{1}{\mathcal{F}_y} = \frac{\alpha}{K} \log \frac{\mathcal{F}_y(L-1)}{1-\mathcal{F}_y}, \tag{7}$$

*where $k \in [K]$ and $j \in [L] \backslash \{y\}$.*

# Formulas of ADP

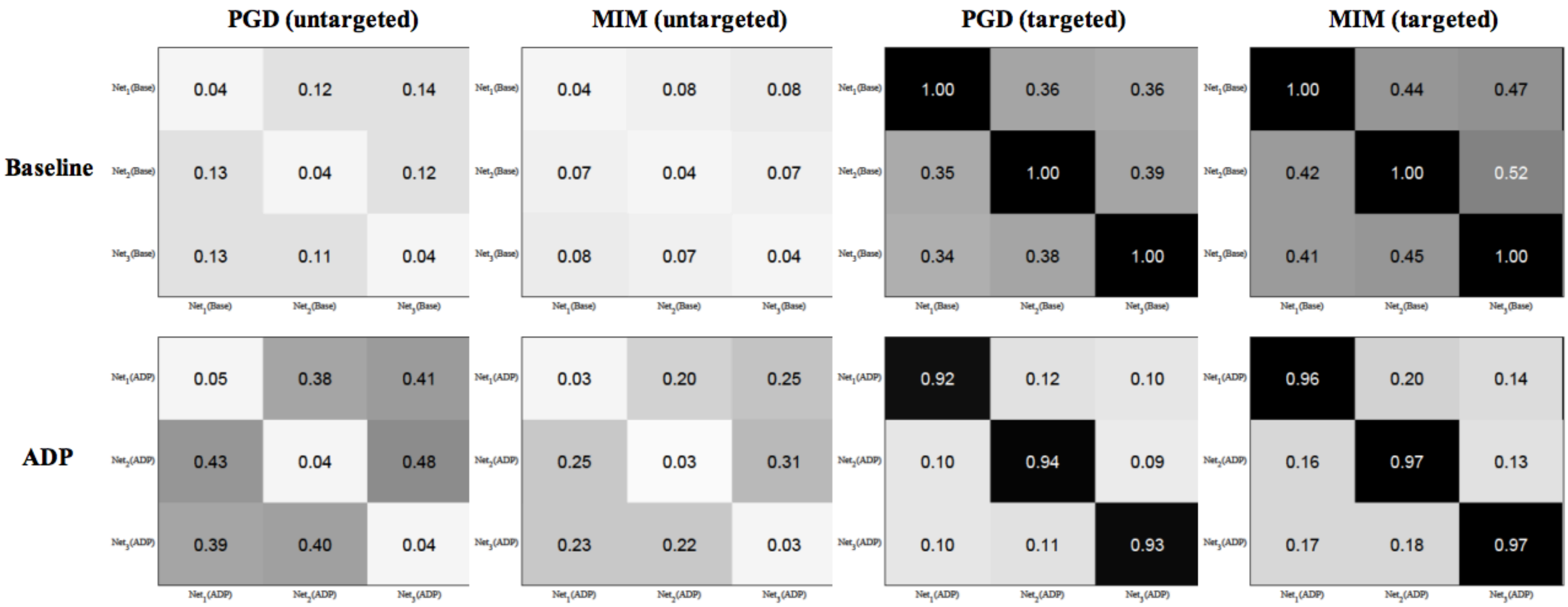
**So the ADP regularizer is**

$$\mathrm{ADP}_{\alpha,\beta}(x,y) = \alpha \cdot \mathcal{H}(\mathcal{F}) + \beta \cdot \log\left(\mathbb{ED}\right)$$

**Corollary 1.** *If there is $K \mid (L-1)$, then $\forall \alpha, \beta > 0$, the optimal solution of the minimization problem (6) satisfies the Eq. (7). Besides, let $S = \{s_1, , \cdots, s_K\}$ be any partition of the index set $[L] \backslash \{y\}$, where $\forall k \in [K]$, $|s_k| = \frac{L-1}{K}$. Then the optimal solution further satisfies:*

$$F_j^k = \begin{cases} \frac{K(1-\mathcal{F}_y)}{L-1}, & j \in s_k, \\ \mathcal{F}_y, & j = y, \\ 0, & otherwise. \end{cases} \tag{8}$$

# Experiments




## Adversarial transferability among individual members of ensembles

# Experiments

*Table 2.* Classification accuracy (%) on adversarial examples. Ensemble models consist of three Resnet-20. For JSMA, the perturbation $\epsilon = 0.2$ on MNIST, and $\epsilon = 0.1$ on CIFAR-10. For EAD, the factor of $L_1$-norm $\beta = 0.01$ on both datasets.

| Attacks | MNIST | | | | CIFAR-10 | | | |
|---|---|---|---|---|---|---|---|---|
| | Para. | Baseline | $\text{ADP}_{2,0}$ | $\text{ADP}_{2,0.5}$ | Para. | Baseline | $\text{ADP}_{2,0}$ | $\text{ADP}_{2,0.5}$ |
| FGSM | $\epsilon = 0.1$ | 78.3 | 95.5 | **96.3** | $\epsilon = 0.02$ | 36.5 | 57.4 | **61.7** |
| | $\epsilon = 0.2$ | 21.5 | 50.6 | **52.8** | $\epsilon = 0.04$ | 19.4 | 41.9 | **46.2** |
| BIM | $\epsilon = 0.1$ | 52.3 | 86.4 | **88.5** | $\epsilon = 0.01$ | 18.5 | 44.0 | **46.6** |
| | $\epsilon = 0.15$ | 12.2 | 69.5 | **73.6** | $\epsilon = 0.02$ | 6.1 | 28.2 | **31.0** |
| PGD | $\epsilon = 0.1$ | 50.7 | 73.4 | **82.8** | $\epsilon = 0.01$ | 23.4 | 43.2 | **48.4** |
| | $\epsilon = 0.15$ | 6.3 | 36.2 | **41.0** | $\epsilon = 0.02$ | 6.6 | 26.8 | **30.4** |
| MIM | $\epsilon = 0.1$ | 58.3 | 89.7 | **92.0** | $\epsilon = 0.01$ | 23.8 | 49.6 | **52.1** |
| | $\epsilon = 0.15$ | 16.1 | 73.3 | **77.5** | $\epsilon = 0.02$ | 7.4 | 32.3 | **35.9** |
| JSMA | $\gamma = 0.3$ | 84.0 | 88.0 | **95.0** | $\gamma = 0.05$ | 29.5 | 33.0 | **43.5** |
| | $\gamma = 0.6$ | 74.0 | 85.0 | **91.0** | $\gamma = 0.1$ | 27.5 | 32.0 | **37.0** |
| C&W | $c = 0.1$ | 91.6 | 95.9 | **97.3** | $c = 0.001$ | 71.3 | 76.3 | **80.6** |
| | $c = 1.0$ | 30.6 | 75.0 | **78.1** | $c = 0.01$ | 45.2 | 50.3 | **54.9** |
| | $c = 10.0$ | 5.9 | 20.2 | **23.8** | $c = 0.1$ | 18.8 | 19.2 | **25.6** |
| EAD | $c = 5.0$ | 29.8 | 91.3 | **93.4** | $c = 1.0$ | 17.5 | 64.5 | **67.3** |
| | $c = 10.0$ | 7.3 | 87.4 | **89.5** | $c = 5.0$ | 2.4 | 23.4 | **29.6** |

## Classification accuracy (%) on adversarial examples

# Experiments

*Table 4.* Classification accuracy (%): $\textbf{AdvT}_{\textbf{FGSM}}$ denotes adversarial training (AdvT) on FGSM, $\textbf{AdvT}_{\textbf{PGD}}$ denotes AdvT on PGD. $\epsilon = 0.04$ for FGSM; $\epsilon = 0.02$ for BIM, PGD and MIM.

| Defense Methods | CIFAR-10 | | | |
|---|---|---|---|---|
| | FGSM | BIM | PGD | MIM |
| $\text{AdvT}_{\text{FGSM}}$ | 39.3 | 19.9 | 24.2 | 24.5 |
| $\text{AdvT}_{\text{FGSM}} + \text{ADP}_{2,0.5}$ | **56.1** | **25.7** | **26.7** | **30.6** |
| $\text{AdvT}_{\text{PGD}}$ | 43.2 | 27.8 | 32.8 | 32.7 |
| $\text{AdvT}_{\text{PGD}} + \text{ADP}_{2,0.5}$ | **52.8** | **34.0** | **36.2** | **38.8** |

Classification accuracy (%) on adversarial examples

# Towards Robust Detection of Adversarial Examples

**(NeurIPS 2018)**

# We Detect Adversarial Examples, and How?

## Design new detectors:

- Kernel density detector (Feinman et al. 2017)
- LID detector (Ma et al. ICLR 2018)
- ……

# We Detect Adversarial Examples, and How?

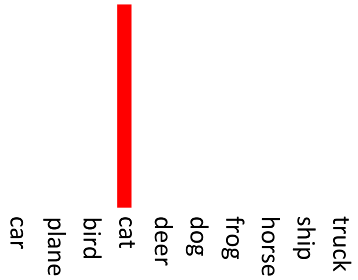
**Design new detectors:**

- Kernel density detector (Feinman et al. 2017)
- LID detector (Ma et al. ICLR 2018)
- ……

**Train the models to better collaborate with existing detectors**

# Reverse Cross Entropy

## Cross-Entropy (CE):

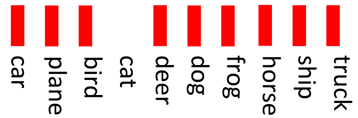



$1_y$: One-hot label

$\{0, 0, 0, \mathbf{1}, 0, 0, 0, 0, 0, 0\}$

$$\boldsymbol{\mathcal{L}_{CE} = -1_y \cdot \log(\mathrm{F})}$$

## Reverse Cross-Entropy (RCE):




$R_y$: Reverse label

$\{\frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \mathbf{0}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}\}$

$$\boldsymbol{\mathcal{L}_{RCE} = -R_y \cdot \log(\mathrm{F})}$$

# The RCE Training Method

## Phase 1: Reverse Training

**Training the model by minimizing the RCE loss**

## Phase 2: Reverse Logits

**Negating the logits fed to the softmax layer to give predictions**

# Theoretical Analysis

**Theorem 2.** *(Proof in Appendix A) Let* $(x, y)$ *be a given training data. Under the* $L_\infty$*-norm, if there is a training error* $\alpha \ll \frac{1}{L}$ *that* $\|\mathbb{S}(Z_{pre}(x, \theta_R^*)) - R_y\|_\infty \leq \alpha$*, then we have bounds*

$$\|\mathbb{S}(-Z_{pre}(x, \theta_R^*)) - 1_y\|_\infty \leq \alpha(L-1)^2,$$

*and* $\forall j, k \neq y$*,*

$$|\mathbb{S}(-Z_{pre}(x, \theta_R^*))_j - \mathbb{S}(-Z_{pre}(x, \theta_R^*))_k| \leq 2\alpha^2(L-1)^2.$$

**Property 1: Consistent and Unbiased**

**When the training error $\alpha \longrightarrow 0$, the prediction tends to the one-hot label**

**Property 2: Tighter Bound**

**The difference between any two non-maximal elements decreases as $O(\alpha^2)$**

# The Insights of RCE Training

We first define the non-maximal entropy (non-ME) as:

$$\mathrm{nonME(x)} = -\sum_{i \neq y} \hat{F}(x)_i \log\left(\hat{F}(x)_i\right),$$

where $\hat{F}(x)_i$ is the normalized non-maximal predictions.

# The Insights of RCE Training

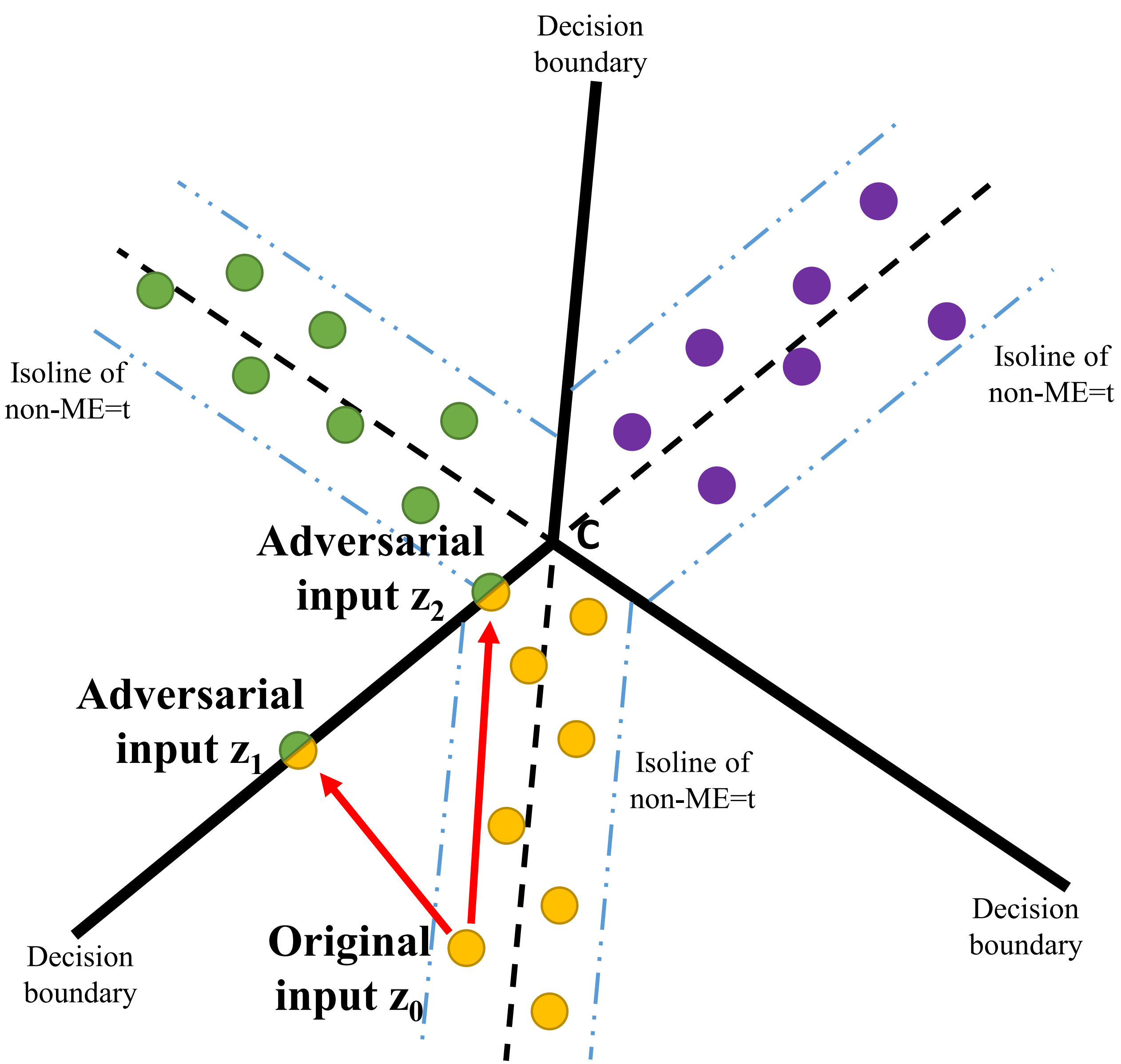
We first define the non-maximal entropy (non-ME) as:

$$\text{nonME}(x) = -\sum_{i \neq y} \hat{F}(x)_i \log\left(\hat{F}(x)_i\right),$$

where $\hat{F}(x)_i$ is the normalized non-maximal predictions.

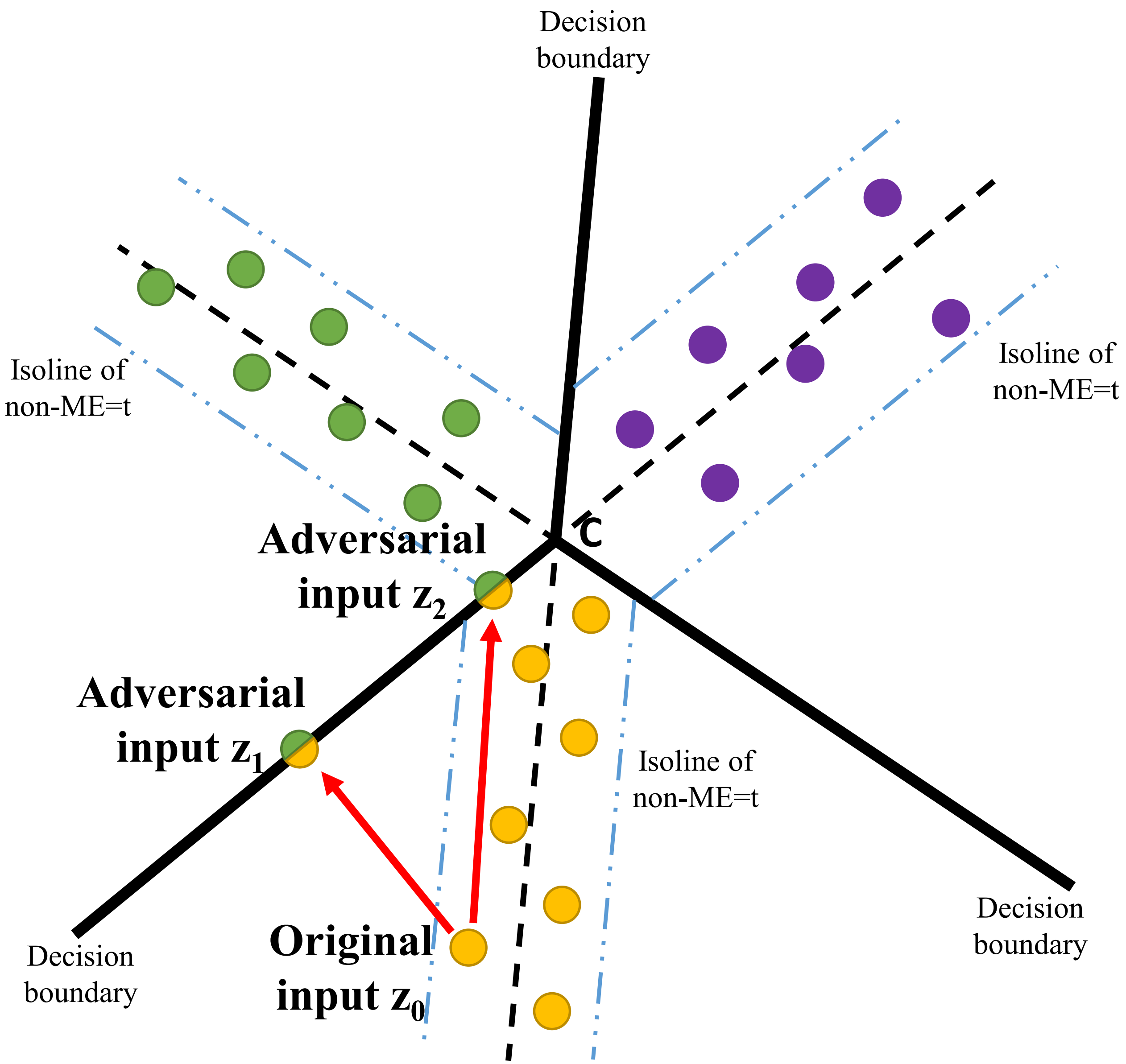
**RCE training encourages the maximal prediction to tend to 1, while maximizing the non-ME.**

# The Insights of RCE Training




**The left plot is the decision domain in 2-d feature space for 3 classes (each class with one color)**

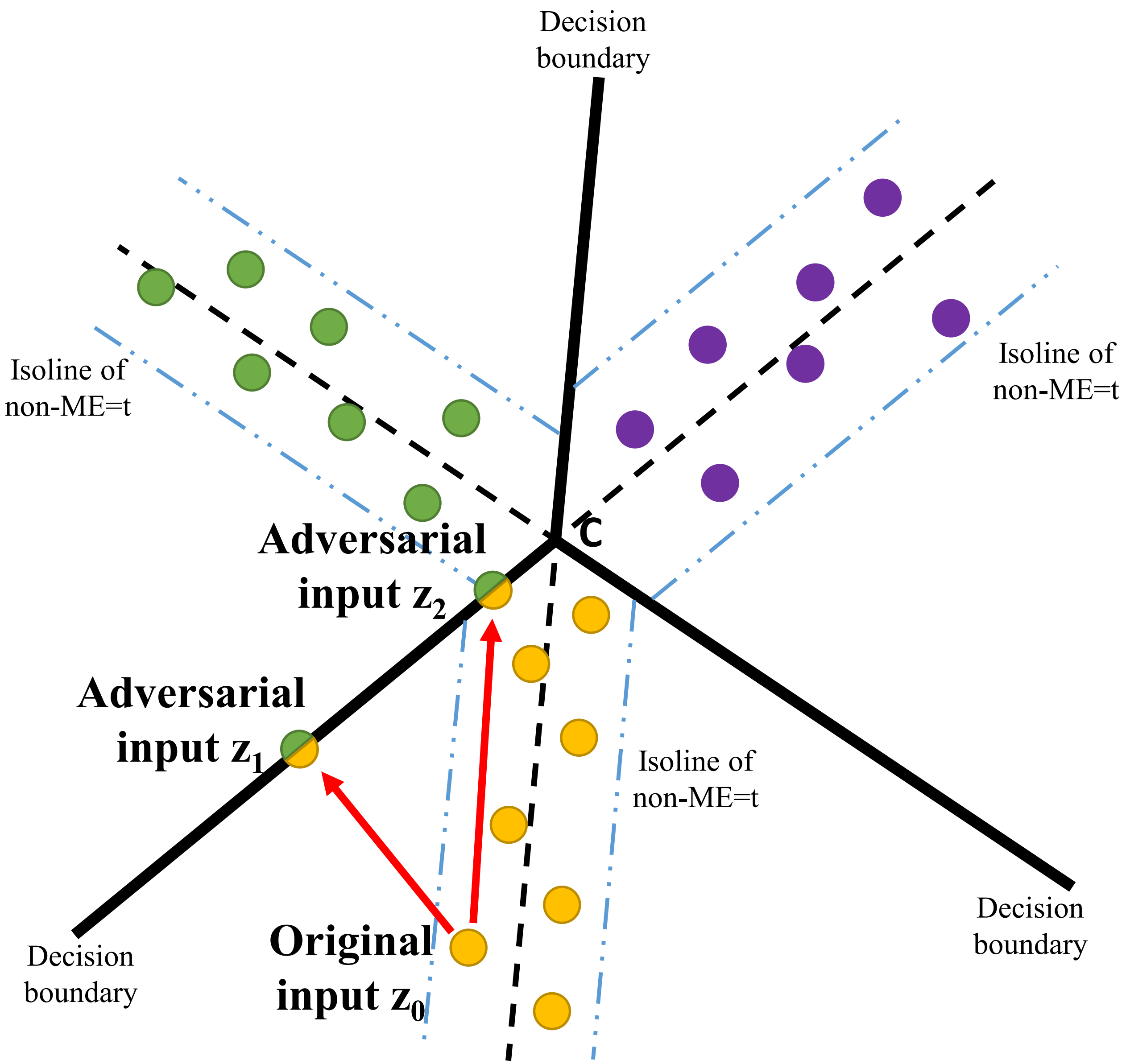
# The Insights of RCE Training




**The left plot is the decision domain in 2-d feature space for 3 classes (each class with one color)**

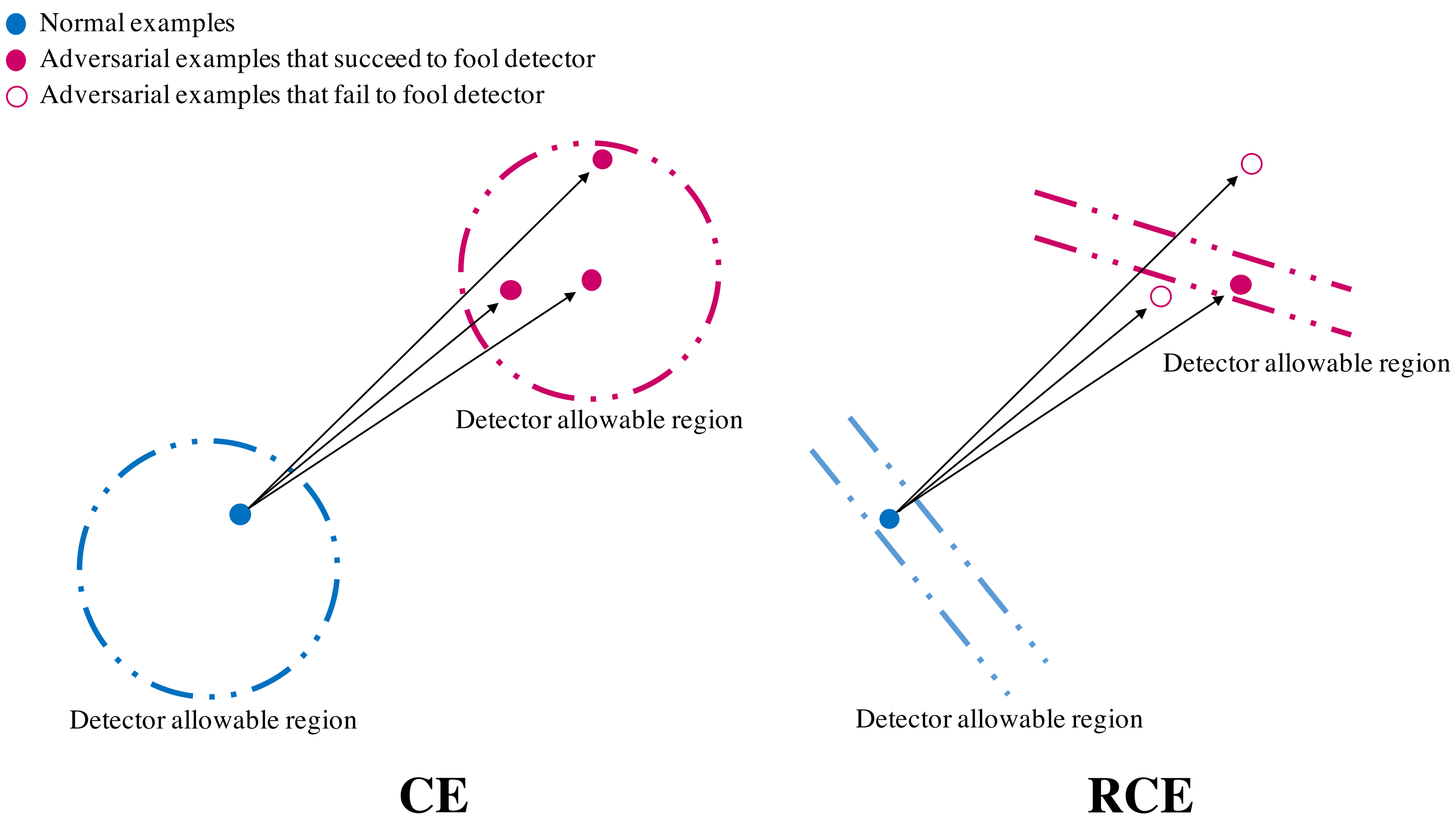
**When the non-ME of the returned predictions are maximized, the learned features for each class with tend to locate near the black dash lines, where the points on the dash lines have the maximal non-ME.**

# The Insights of RCE Training



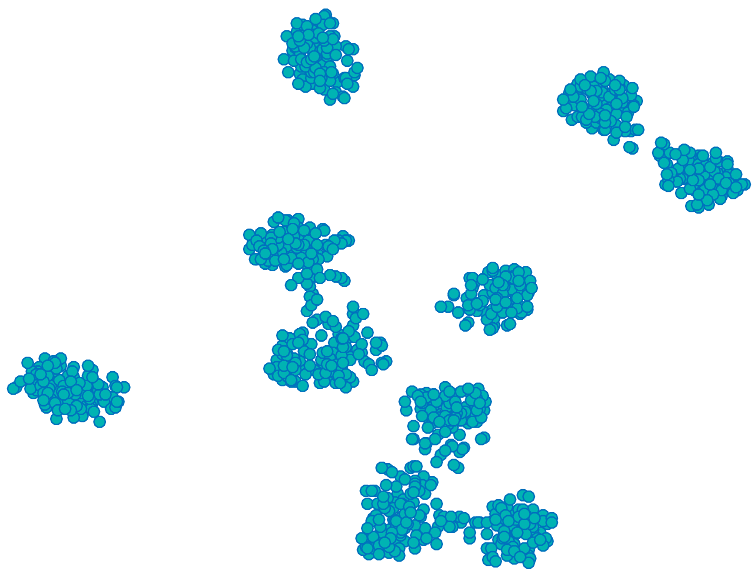

**Then if an adversary want to craft an adversarial example based on $z_0$, he has to move further to $z_2$ rather than $z_1$ to obtain a normal value of non-ME.**

# The Insights of RCE Training

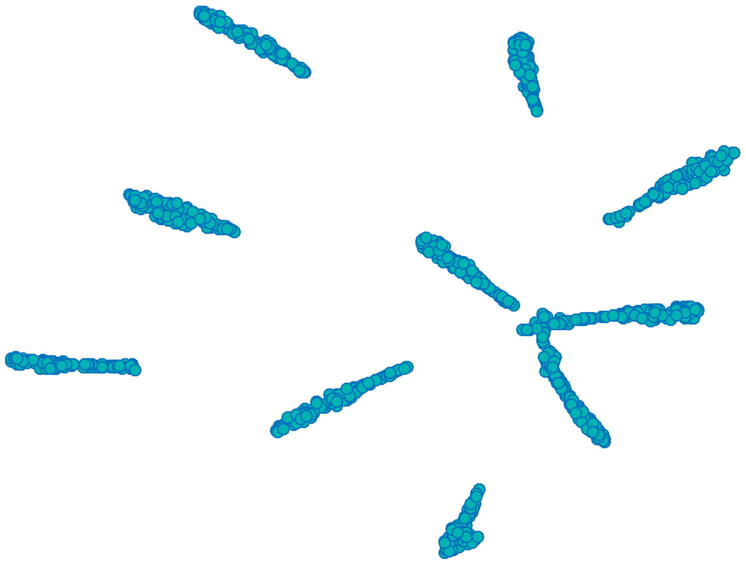



**In practice, the learned low-dimensional feature distributions by RCE make it more difficult to craft an adversarial examples with normal values of non-ME.**

# Experiments



CE



RCE

t-SNE visualization of learned features on CIFAR-10

# Experiments

| Attack | Obj. | MNIST | | | CIFAR-10 | | |
|---|---|---|---|---|---|---|---|
| | | Confidence | non-ME | K-density | Confidence | non-ME | K-density |
| FGSM | CE | 79.7 | 66.8 | 98.8 (-) | 71.5 | 66.9 | **99.7** (-) |
| | RCE | 98.8 | 98.6 | **99.4** (*) | 92.6 | 91.4 | 98.0 (*) |
| BIM | CE | 88.9 | 70.5 | 90.0 (-) | 0.0 | 64.6 | **100.0** (-) |
| | RCE | 91.7 | 90.6 | **91.8** (*) | 0.7 | 70.2 | **100.0** (*) |
| ILCM | CE | 98.4 | 50.4 | 96.2 (-) | 16.4 | 37.1 | 84.2 (-) |
| | RCE | 100.0 | 97.0 | **98.6** (*) | 64.1 | 77.8 | **93.9** (*) |
| JSMA | CE | 98.6 | 60.1 | 97.7 (-) | 99.2 | 27.3 | 85.8 (-) |
| | RCE | 100.0 | 99.4 | **99.0** (*) | 99.5 | 91.9 | **95.4** (*) |
| C&W | CE | 98.6 | 64.1 | 99.4 (-) | 99.5 | 50.2 | 95.3 (-) |
| | RCE | 100.0 | 99.5 | **99.8** (*) | 99.6 | 94.7 | **98.2** (*) |
| C&W-hc | CE | 0.0 | 40.0 | 91.1 (-) | 0.0 | 28.8 | 75.4 (-) |
| | RCE | 0.1 | 93.4 | **99.6** (*) | 0.2 | 53.6 | **91.8** (*) |

**AUC-scores ($10^{-2}$) on adversarial examples**

# Reference


1. *Max-Mahalanobis Linear Discriminant Analysis Network*
**Tianyu Pang**, Chao Du, and Jun Zhu
**ICML 2018**

2. *Towards Robust Detection of Adversarial Examples*
**Tianyu Pang**, Chao Du, Yinpeng Dong, and Jun Zhu
**NeurIPS 2018**

3. *Improving Adversarial Robustness via Promoting Ensemble Diversity*
**Tianyu Pang**, Kun Xu, Chao Du, Ning Chen and Jun Zhu
**ICML 2019**

4. *Rethinking Softmax Cross-Entropy Loss for Adversarial Robustness*
**Tianyu Pang**, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen and Jun Zhu
**ICLR 2020**

5. *Mixup Inference: Better Exploiting Mixup to Defend Adversarial Attacks*
**Tianyu Pang***, Kun Xu*, Jun Zhu
**ICLR 2020**

# Thanks