Theoretical Understanding of Adversarial Examples: Expressive Power and Training Dynamics

Speaker: Binghui Li

Center for Machine Learning Research

Peking University

2025/4/2

Outline

- Introduction to Adversarial Examples in Deep Learning
- Theoretical Understanding of Adversarial Examples:
    1. Perspective of **Expressive Power**: *Robustness Requires Large Models*
    2. Perspective of **Training Dynamics** (*Feature Learning Theory*)
        a) *Gradient Descent Provably Converges to Non-Robust Solutions*
        b) *Adversarial Training Provably Improves Models' Robustness*
- Discussion on the Future of Adversarial Examples

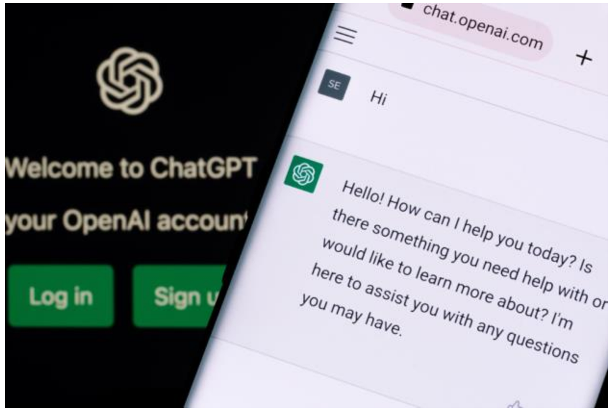
Outline

- Introduction to Adversarial Examples in Deep Learning
- Theoretical Understanding of Adversarial Examples:
    1. Perspective of **Expressive Power**: *Robustness Requires Large Models*
    2. Perspective of **Training Dynamics** (*Feature Learning Theory*)
        a) *Gradient Descent Provably Converges to Non-Robust Solutions*
        b) *Adversarial Training Provably Improves Models' Robustness*
- Discussion on the Future of Adversarial Examples

Deep Learning

- Nowadays, deep learning has achieved remarkable success in a variety of disciplines including computer vision, natural language processing, multi-agent decision making as well as scientific and engineering applications.

SAM



ChatGPT

AlphaStar

- Deep Learning ≈ Deep Neural Network (Powerful Expressivity) + Gradient Descent Method (Efficient Opt Alg)

Deep Neural Network

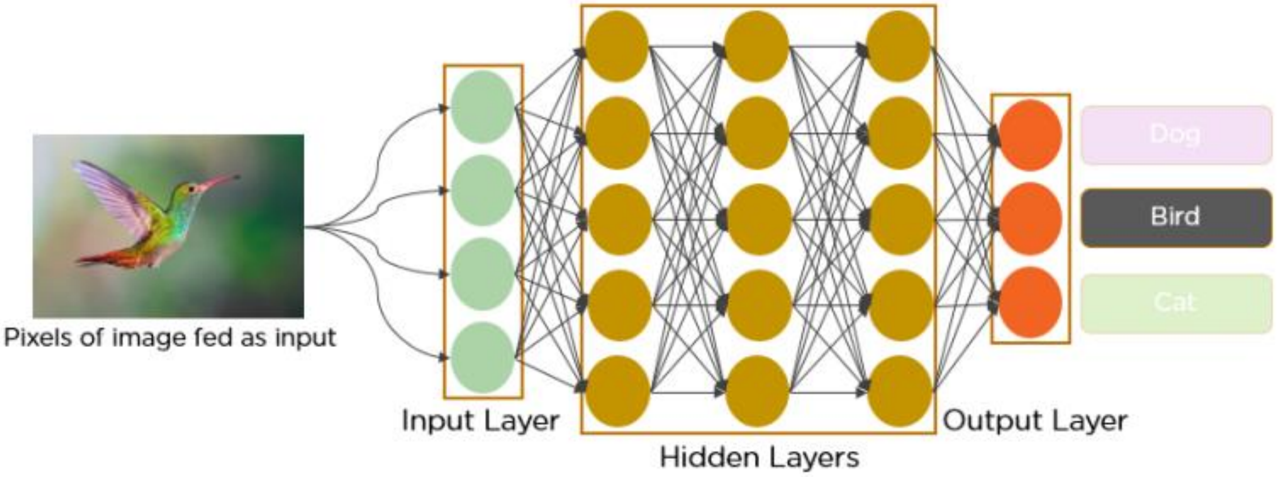
- A multilayer neural network is a function from input $\boldsymbol{x} \in \mathbb{R}^d$ to output $\boldsymbol{y} \in \mathbb{R}^m$, recursively defined as follows:

$$\boldsymbol{h}_1 = \sigma\left(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1\right), \quad \boldsymbol{W}_1 \in \mathbb{R}^{m_1 \times d}, \boldsymbol{b}_1 \in \mathbb{R}^{m_1},$$

$$\boldsymbol{h}_\ell = \sigma\left(\boldsymbol{W}_\ell \boldsymbol{h}_{\ell-1} + \boldsymbol{b}_\ell\right), \quad \boldsymbol{W}_\ell \in \mathbb{R}^{m_\ell \times m_{\ell-1}}, \boldsymbol{b}_\ell \in \mathbb{R}^{m_\ell}, 2 \leq \ell \leq L-1,$$

$$\boldsymbol{y} = \boldsymbol{W}_L \boldsymbol{h}_L + \boldsymbol{b}_L, \quad \boldsymbol{W}_L \in \mathbb{R}^{m \times m_L}, \boldsymbol{b}_L \in \mathbb{R}^m,$$

where σ is the (non-linear) activation function and L is the depth of the neural network. Here, we mainly focus on ReLU nets i.e. $\sigma(x) = \max\{0, x\}$.



Train Deep Model via Gradient Descent Method

- Data: we consider a binary classification task: $X \to Y \in \{-1, +1\}$, and let D be the data distribution on $X \times Y$.
- Model: parameterized neural network classifier: $\{f_\theta\}_{\theta \in \Theta}$.
- Objective: we evaluate the classification performance by the test loss:

$$L(\theta) \coloneqq \mathbb{E}_{(x,y)\sim D}[l(f_\theta(x), y)],$$

where $l(\cdot,\cdot)$ denotes loss function, e.g. MSE-loss: $l(z, y) \coloneqq (z - y)^2$, 0-1loss:$\mathbb{I}\{z \neq y\}$.

- In practice, we aim to minimize the empirical risk (ERM) on training dataset $S \coloneqq \{(x_1, y_1), \dots, (x_N, y_N)\}$ i.i.d. sampled from population D instead of the test loss:

$$\min_{\theta \in \Theta} \hat{L}(\theta) \coloneqq \frac{1}{N} \sum_{i=1}^{N} l(f_\theta(x_i), y_i).$$

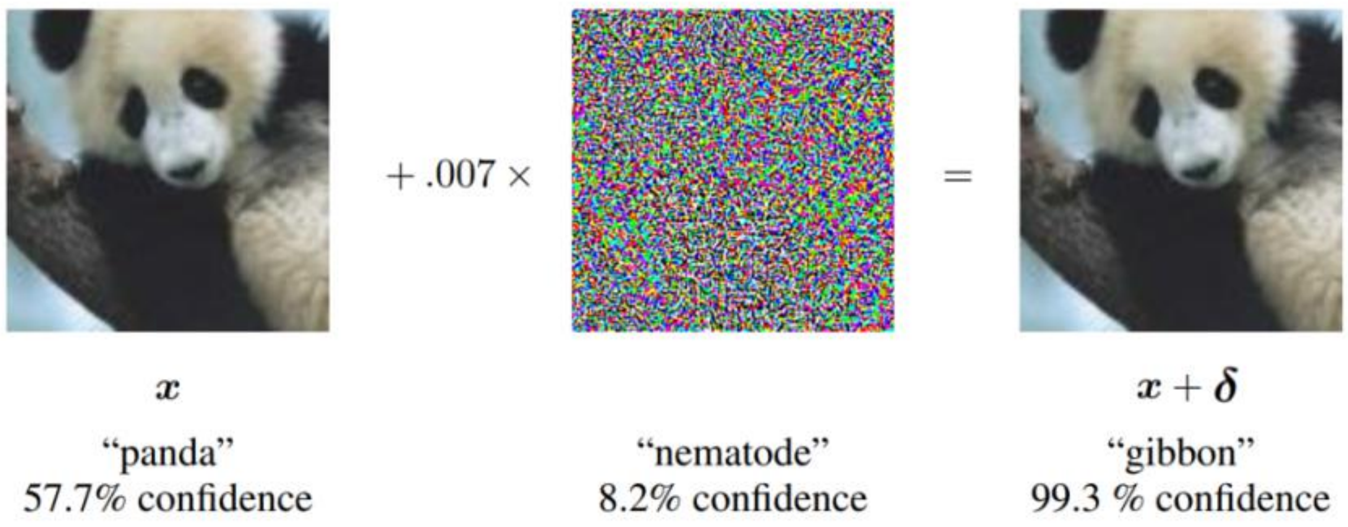
- Training Algorithm: we use gradient descent (GD) to minimize the training loss $\hat{L}(\theta)$:

$$\theta \leftarrow \theta - \eta \nabla_\theta \hat{L}(\theta),$$

where η is learning rate.

Adversarial Examples

- Although deep neural networks have achieved remarkable success in practice, it is well-known that modern neural networks are vulnerable to adversarial examples.
- Specifically, for a given image x, an indistinguishable small but adversarial perturbation δ is chosen to fool the classifier f to produce a wrong class using $f(x + \delta)$ [Szegedy et al, 2013].



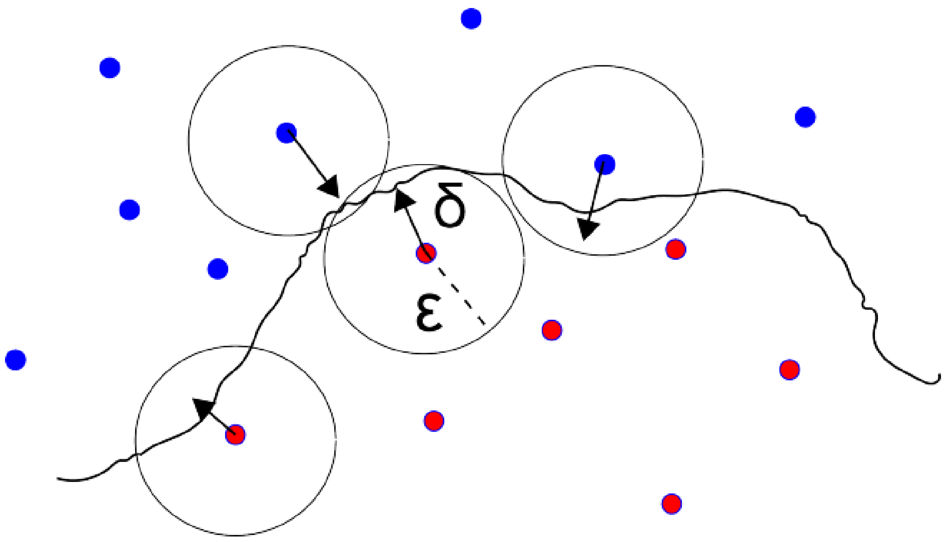
An Instance for Adversarial Example

Improve Robustness via Adversarial Training

- To mitigate this problem, a common approach is to design adversarial training algorithms [Madry et al, 2018] by using adversarial examples as training data.

Concretely, we consider a training dataset $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, and we aim to solve the following min−max optimization problem：

$$\min_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \max_{\|\delta\| \leq \varepsilon} L(f_\theta(x_i + \delta), y_i)$$



- Networks trained using adversarial training are significantly more robust than those trained using the standard gradient descent algorithm.

Overview

In this talk, we mainly provide a comprehensive theoretical understanding of adversarial examples from two perspectives: ***expressive power*** *and* ***training dynamics***.

Paper List:

1. *Why Robust Generalization in Deep Learning is Difficult: Perspective of Expressive Power (NeurIPS 2022)*

2. *Feature Averaging: An Implicit Bias of Gradient Descent Leading to Non-Robustness in Neural Networks (ICLR 2025)*

3. *Adversarial Training Can Provably Improve Robustness: Theoretical Analysis of Feature Learning Process Under Structured Data (ICLR 2025)*

Outline

- Introduction to Adversarial Examples in Deep Learning
- Theoretical Understanding of Adversarial Examples:
  1. Perspective of **Expressive Power**: *Robustness Requires Large Models*
  2. Perspective of **Training Dynamics** (*Feature Learning Theory*)
     a) *Gradient Descent Provably Converges to Non-Robust Solutions*
     b) *Adversarial Training Provably Improves Models' Robustness*
- Discussion on the Future of Adversarial Examples

Why Robust Generalization in Deep Learning is Difficult: Perspective of Expressive Power [1,2]

				
Binghui Li	**Jikai Jin**	**Han Zhong**	**John Hopcroft**	**Liwei Wang**

---

[1]This work has been accepted by **NeurIPS 2022**, where the first two authors have equal contributions and the last author is the corresponding author.

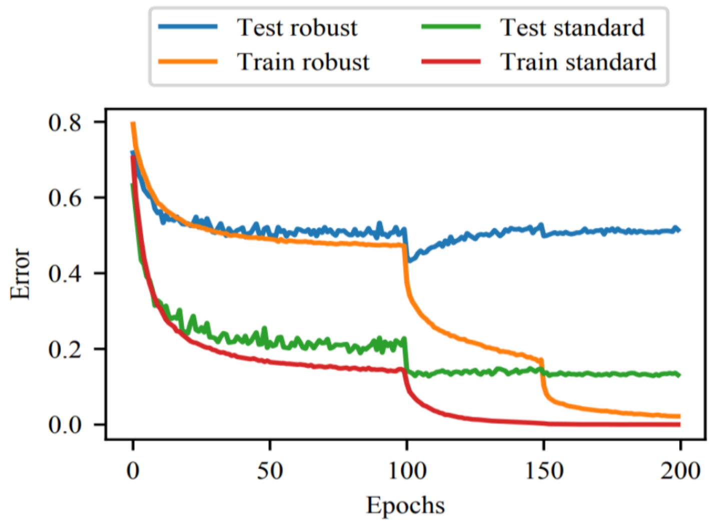
[2]Our full paper can be found at **https://arxiv.org/abs/2205.13863** .

Robust Generalization Gap is Large!

- However, while the state-of-the-art adversarial training methods can achieve high robust training accuracy, all existing methods suffer from large robust test error, which is also called robust overfitting.

	Clean training	Adversarial training
Robust test	3.5%	45.8%
Robust train	–	100%
Clean test	95.2%	87.3%
Clean train	100%	100%

The test and train performance of clean and adversarial training on CIFAR 10 [Raghunathan et al, 2019]



The learning curves of adversarial training on CIFAR 10 [Rice et al, 2020]

Questions

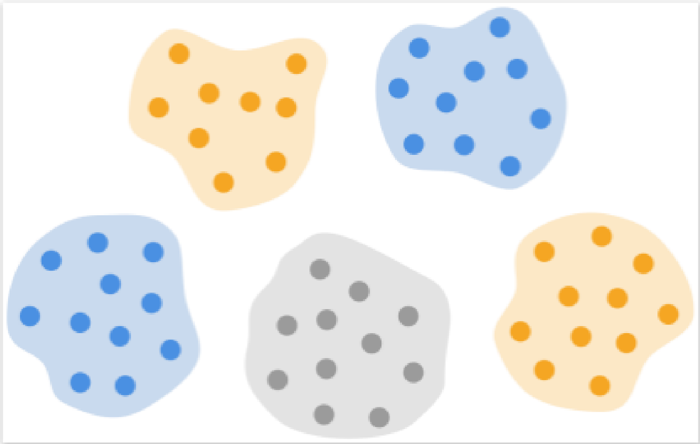
Why does there exist such a ***large generalization gap*** *in the context of robust learning? Can we provide a* ***theoretical understanding*** *of this puzzling phenomena?*

Key Observation

Fact *Data are far from each other.*

	adversarial perturbation ε	minimum Train-Train separation	minimum Test-Train separation
MNIST	0.1	0.737	0.812
CIFAR-10	0.031	0.212	0.220
SVHN	0.031	0.094	0.110
ResImageNet	0.005	0.180	0.224

Experiment results about data separation in [Yang et al, 2020]



Understand Robust Generalization Gap via Representation Complexity

Assumption (Separated Data Distribution)
Let D be the binary-labeled data distribution, where data points are in two sets $A, B \subset [0,1]^d$. We assume that separation $d(A, B) \geq 2\epsilon$ and the perturbation radius $\delta < \epsilon$.

- Representation Complexity:

$$RC(\{f_\theta\}_{\theta\in\Theta}) = \#\, params\ |\theta|$$

- Under the assumption, we focus on:
  - (robust training) *For arbitrary N-size training dataset S i.i.d. sampled from D, how much representation complexity is enough for ReLU nets to achieve **zero robust training error**?*
  - (robust generalization) *For arbitrary data distribution D that satisfies the assumption, how much representation complexity is enough for ReLU nets to achieve **low robust test error**?*

$\tilde{O}(Nd)$ Parameters are Enough to Achieve Zero Robust Training Error

Theorem (Upper Bound for Robust Training)
For any given N-size and d-dim training dataset S that satisfies the separability condition, there exists a ReLU network f with at most $\tilde{O}(Nd)$ parameters such that robust training error is zero.

- For robust training,

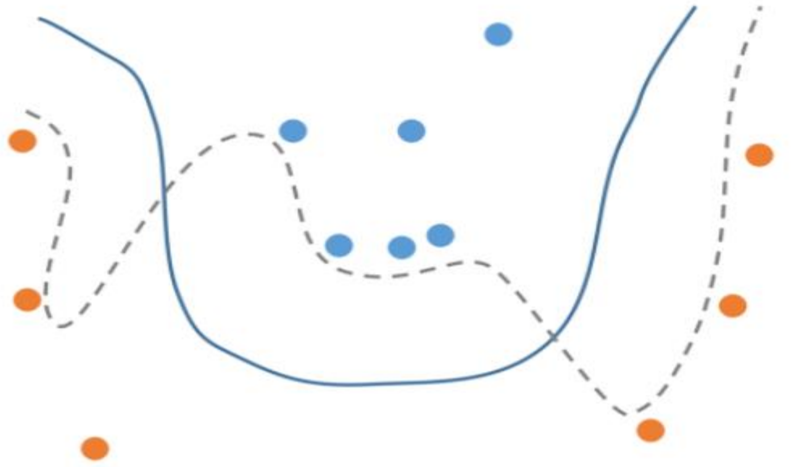
$$RC(ReLU\ Nets) = \tilde{O}(Nd).$$

- It is consistent with the common practice that *moderate-size network* trained by adversarial training achieves *high robust training accuracy*.

There Exists a EXP Large Robust Classifier

Lemma
Under the separability assumption, there exists a robust classifier f^* such that it can robustly classify the 2ϵ- separated labeled sets A and B.

- $f^*(x) = \frac{d(x,B)-d(x,A)}{d(x,B)+d(x,A)}$
- f^* is a ϵ^{-1}-Lipschitz function



Theorem
There exists a ReLU net f with at most $O(exp(d))$ params such that $|f - f^*| = o(1)$ for all $x \in [0,1]^d$.

- Corollary: For robust generalization, $RC(ReLU\ Nets) = O(\exp(d))$.

Robust Generalization Requires Exponentially Large Models

- Now, we present our main result in this paper.

Theorem (Lower Bound for Robust Generalization)
Let F_m be the family of function represented by ReLU nets with at most m parameters. Then, there exists a number $m(d) = \Omega(exp(d))$ and a linear-separable distribution D satisfying the assumption such that, for any classifier in $F_{m(d)}$, the robust test error is at least $\Omega(1)$.

- For robust generalization,

$$RC(ReLU\ Nets) = \Omega(\exp(d)),$$

in contrast, for standard generalization, only $O(d)$ params are enough.

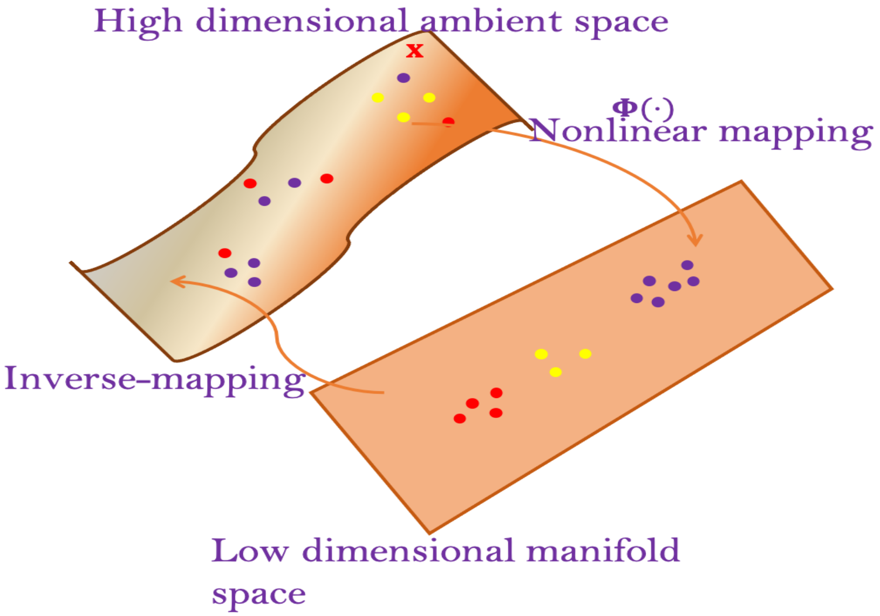
- Moreover, this lower bound holds for *arbitrarily small* perturbation radius and *general models* as long as $VCDim = O(poly(\#params))$.

Robust Generalization for Low-dimensional-manifold Data

- A common belief of real-life data such as images is that the data points lie on a **low-dimensional manifold**.

Assumption (Manifold Data)
We assume that data lies on a manifold M with the intrinsic dimension k ($k \ll d$), where data points are in two separated labeled sets $A, B \subset M$.

Theorem (Improved Upper Bound)
Under the manifold-data assumption, there exists a ReLU net with at most $\tilde{O}(\exp(k))$ params such that the robust test error on the manifold is zero.



Conclusion

Take Home Message: From the view of representation complexity,
(1) Robust training only needs linearly large models;
(2) Robust generalization, in worst case, requires EXP larger models.

Table 1: Summary of our main results.

Params	Setting			
	Robust Training	Robust Generalization		
		General Case	Linear Separable	$k-$dim Manifold
Upper Bound	$\mathcal{O}(Nd)$ (Thm 2.2)	$\exp(\mathcal{O}(d))$ (Thm 3.3)		$\exp(\mathcal{O}(k))$ (Thm 5.5)
Lower Bound	$\Omega(\sqrt{Nd})$ (Thm 2.3)	$\exp(\Omega(d))$ (Thm 3.4)	$\exp(\Omega(d))$ (Thm 4.3)	$\exp(\Omega(k))$ (Thm 5.8)

Discussion

- **Beyond Worst Case:** For a specific data distribution, how much representation complexity is enough for networks to achieve robustness?

- **Practical Architecture:** CNN v.s. ViT v.s. Diffusion Model.

- **Gradient-based Method:** Can gradient methods provably learn robust or non-robust networks?

Outline

- Introduction to Adversarial Examples in Deep Learning
- Theoretical Understanding of Adversarial Examples:
    1. Perspective of **Expressive Power**: *Robustness Requires Large Models*
    2. Perspective of **Training Dynamics** (*Feature Learning Theory*)
        a) *Gradient Descent Provably Converges to Non-Robust Solutions*
        b) *Adversarial Training Provably Improves Models' Robustness*
- Discussion on the Future of Adversarial Examples

Feature Averaging: An Implicit Bias of Gradient Descent Leading to Non-Robustness in Neural Networks[1,2]

Binghui Li

Zhixuan Pan

Kaifeng Lyu

Jian Li

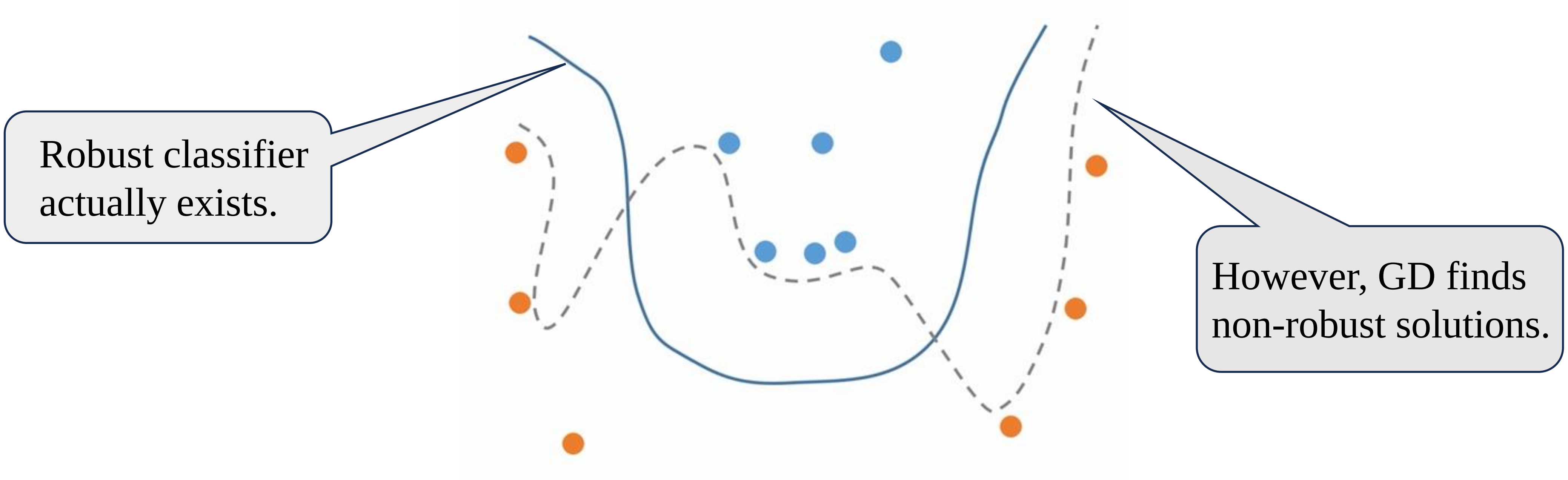
[1]This work has been accepted by **ICLR 2025**, where the first two authors have equal contributions and the last author is the corresponding author.

[2]Our full paper can be found at https://arxiv.org/abs/2410.10322 .

Question

Our Fundamental Theoretical Questions :

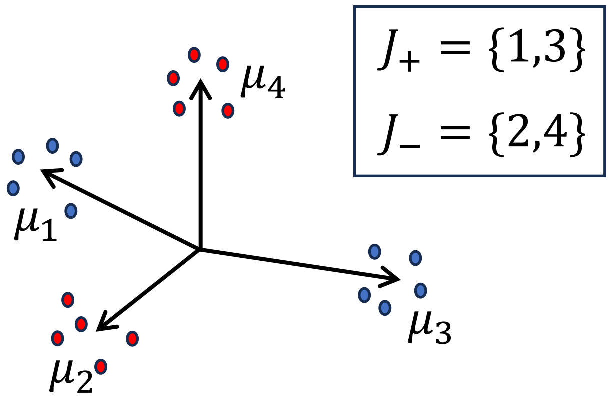
Why do neural networks trained by ***gradient descent algorithm*** *converge to the* ***non-robust solutions*** *that fail to classify* ***adversarial examples****?*



Data Distribution

- Data distribution D_{binary} on $\mathbb{R}^d \times \{-1,1\}$ that consists of ***k* clusters**:
  - for each cluster, it corresponds to a cluster feature vector μ_i $(i \in [k])$;
  - μ_i for all $i \in [k]$ are orthogonal and $\|\mu_i\|_2 = \Theta(\sqrt{d})$;
  - Suppose that total ***k* clusters** can be divide into two disjoint classes with index sets J_+ and J_- that correspond to positive class and negative class, respectively;
  - positive and negative clusters are balanced: $\exists c \geq 1, c^{-1} \leq \frac{|J_+|}{|J_-|} \leq c$.
- An instance (x, y) sampled from cluster i:
  - label $y = 1$ if $i \in J_+$ and $y = -1$ if $i \in J_-$;
  - data input $x = \mu_i + \xi$,

  where random noise $\xi \sim N(0, \sigma^2 I_d)$ and $\sigma = \Theta(1)$.



An example for $k = 4, c = 1$

The similar data distribution is analyzed in [Frei et al, 2024].

Learner Model: Two-Layer ReLU Network

- **Two-layer ReLU network:** for simplicity, we fix the second layer.

  $f_\theta(x) := \frac{1}{m}\sum_{r\in[m]} ReLU(\langle w_{1,r}, x\rangle + b_{1,r})$- $\frac{1}{m}\sum_{r\in[m]} ReLU(\langle w_{-1,r}, x\rangle + b_{-1,r})$,

  where $\theta = \{w_{s,r}, b_{s,r}\}_{(s,r)\in\{1,-1\}\times[m]}$ are trainable parameters.
- **Loss function:** we apply logistic loss as $L(\theta) \coloneqq \frac{1}{n}\sum_{i=1}^{n} l(y_i f_\theta(x_i))$, where $l(z) \coloneqq \log(1+e^{-z})$.
- **Initialization:** $w_{s,r}^{(0)} \sim N(0, \sigma_w^2 I_d)$, $\sigma_w^2 = \frac{1}{d}$ and $b_{s,r}^{(0)} \sim N(0, \sigma_b^2)$, $\sigma_b^2 = \frac{1}{d^2}$.
- **Gradient descent algorithm:** $\theta_{t+1} = \theta_t - \eta\nabla_\theta L(\theta_t)$ with small learning rate $\eta = \Theta(\frac{1}{\sqrt{d}})$.

Clean Accuracy and Robust Accuracy

- For a given data distribution D over $\mathbb{R}^d \times \{\pm 1\}$, the **clean accuracy** of a neural network $f_\theta: \mathbb{R}^d \to \mathbb{R}$ on D is defined as

$$Acc_{clean}^{D}(f_\theta) \coloneqq \mathbb{P}_{(x,y)\sim D}\left[\operatorname{sgn}\left(f_\theta(x)\right) = y\right].$$

- In this work, we focus on the $\boldsymbol{l_2}$**-robustness**. The l_2 δ-robust accuracy of f_θ on D is defined as

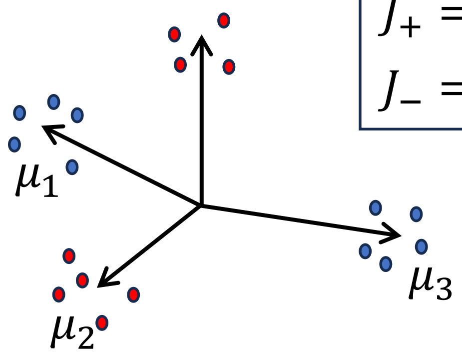
$$Acc_{robust}^{D}(f_\theta;\delta) \coloneqq \mathbb{P}_{(x,y)\sim D}\left[\forall \rho \in \mathbb{B}_\delta : \operatorname{sgn}\left(f_\theta(x+\rho)\right) = y\right],$$

where $\mathbb{B}_\delta \coloneqq \{\rho \in \mathbb{R}^d : \|\rho\| \leq \delta\}$ is the l_2 -ball centered at the origin with radius δ.

- We say that a neural network f_θ is $\boldsymbol{\delta}$**-robust** if $Acc_{robust}^{D}(f_\theta;\delta) \geq 1 - \epsilon(d)$ for some function $\epsilon(d)$ that vanishes to zero, i.e., $\epsilon(d) \to 0$ $as\ d \to 0$.

There Exists the Robust Solution!

- Indeed, it is easy to show a robust solution exists with **robust radius $\boldsymbol{O(\sqrt{d})}$:**
  - Let each neuron deal with one cluster;
  - Use the bias term to filter out intra/inter cluster noise.



$J_+ = \{1,3\}$

$J_- = \{2,4\}$

$$f_{robust}(x) = \sum_{j \in J_+} \underbrace{ReLU(\langle \mu_j, x \rangle + b_j^+)}_{\text{deal with positive cluster } j} - \sum_{l \in J_-} \underbrace{ReLU(\langle \mu_l, x \rangle + b_l^+)}_{\text{deal with negative cluster } l}$$

An example for $k = 4, c = 1$

$\forall i \neq j, \|\mu_i - \mu_j\|_2 = \Theta(\sqrt{d})$

f_{robust} achieves optimal robustness.

GD Provably Learns Averaged Features

- **Lemma** (Weight Decomposition). During training, we can decompose the weight $w_{s,r}^{(t)}$ as linear combination of the features (and some noise):

$$w_{s,r}^{(t)} = w_{s,r}^{(0)} + \sum_{j \in J_+} \lambda_{s,r,j}^{(t)} \mu_j + \sum_{j \in J_-} \lambda_{s,r,j}^{(t)} \mu_j + \sum_{i \in [n]} \sigma_{s,r,i}^{(t)} \xi_i.$$

- **Theorem** (Feature Averaging). For sufficiently large d, suppose we train the model using the gradient descent. After $T = \Theta(poly(d))$ iterations, with high probability over the sampled training dataset S, the weights of model $f_{\theta^{(T)}}$ satisfy:
  - The model achieves perfect standard accuracy: $\mathbb{P}_{(x,y)\sim D_{binary}}\left[\operatorname{sgn}\left(f_{\theta^{(T)}}(x)\right) = y\right] = 1 - o(1)$.
  - GD learns averaged features:

$$\lambda_{s,r,j}^{(T)} \geq \Omega(1), \quad \lambda_{-s,r,j}^{(T)} \leq o(1), \quad \frac{\lambda_{s,r,j}^{(T)}}{\lambda_{s,r,k}^{(T)}} \leq O(1), \quad \forall s \in \{-1,1\}, r \in [m], j \neq k \in J_s.$$

Large coeffs for the same class

Small coeffs for the other class

No large coeff is much than others

Intuitively, it approximately satisfies:

$$w_{s,r} \propto \sum_{j \in J_s} \mu_j, \forall (s,r) \in \{-1,1\} \times [m]$$

Averaged Features are Non-robust Features

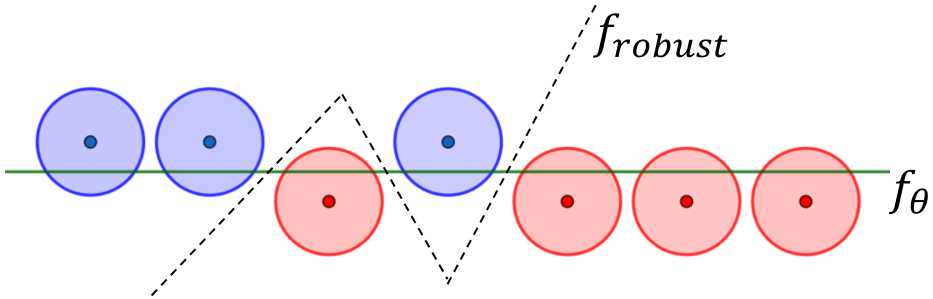
Theorem. For the weights in a feature-averaging solution, for any choice of bias b, the model has nearly zero δ-robust accuracy for perturbation radius $\delta = \Omega(\sqrt{d/k})$.

(Recall that a robust solution exists with robust radius $O(\sqrt{d})$)

Intuition: for averaged features, the model approximately degenerates into a two-neuron network as follows,

$$f_\theta(x) \approx C(ReLU(\langle \sum_{j \in J_+} \mu_j, x \rangle + b_+) - ReLU(\langle \sum_{j \in J_-} \mu_j, x \rangle + b_-))$$

deal with all positive clusters   deal with all negative clusters



In fact, the attack can be chosen as $\varepsilon \propto -\sum_{j \in J_+} \mu_j + \sum_{j \in J_-} \mu_j$

Detailed Feature-Level Supervisory Label

- One can show if one is provided detailed feature level label, some two-layer ReLU network can learn feature-decoupled solutions, which is provably more robust.

Theorem (Multiple-Info Helps Learning Feature-Decoupled Solutions). By given all cluster information for each data point, we can apply the standard gradient descent algorithm to solve the corresponding k-classification task, and we will derive the following multiple classifier $F(x) = (f_1, \ldots, f_k): \mathbb{R}^d \to \mathbb{R}^k$, where $f_i(x) := ReLU(\langle w_i, x \rangle)$, which satisfies

- $w_i^{(t)} = w_i^{(0)} + \sum_{j \in [k]} \lambda_{i,j}^{(t)} \mu_j + \sum_{l \in [n]} \sigma_{i,l}^{(t)} \xi_l$
- After $T = \Theta(poly(d))$, it holds that: $\lambda_{i,i}^{(T)} = \Omega(1), \lambda_{i,j}^{(T)} = o(1), \forall i \in [k], j \in [k] \backslash \{i\}$.

- Comments: Human is more robust to small perturbations.
  - *No adv training for human.*
  - *Adv training is slow (can we used std training to get a robust model?)*
  - *More detailed and structured supervisory information for human.*
  - *Such labeling in large scale is possible in the era of multi-model LLMs.*

Real-World Experiments

Each element in the matrix located at position (i, j) is the average cosine value of the angle between the weight vector of i-th neuron and the feature vector μ_j of the j-th feature.

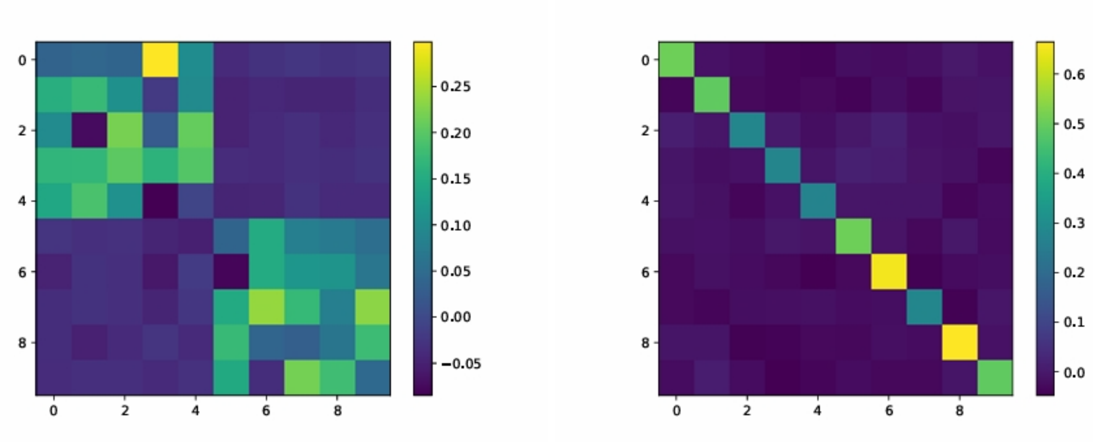


Figure 1: **Illustration of Feature Averaging and Feature Decoupling** .

We create binary classification tasks from the MNIST and CIFAR10 datasets:

- Red: binary classifier trained by 2-classification task.
- Blue: binary classifier trained by 10-classification task.

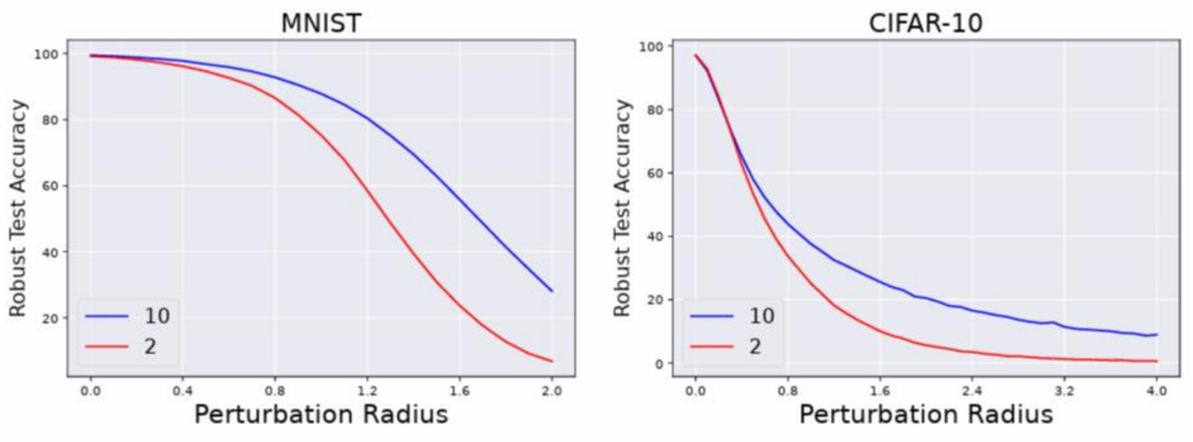
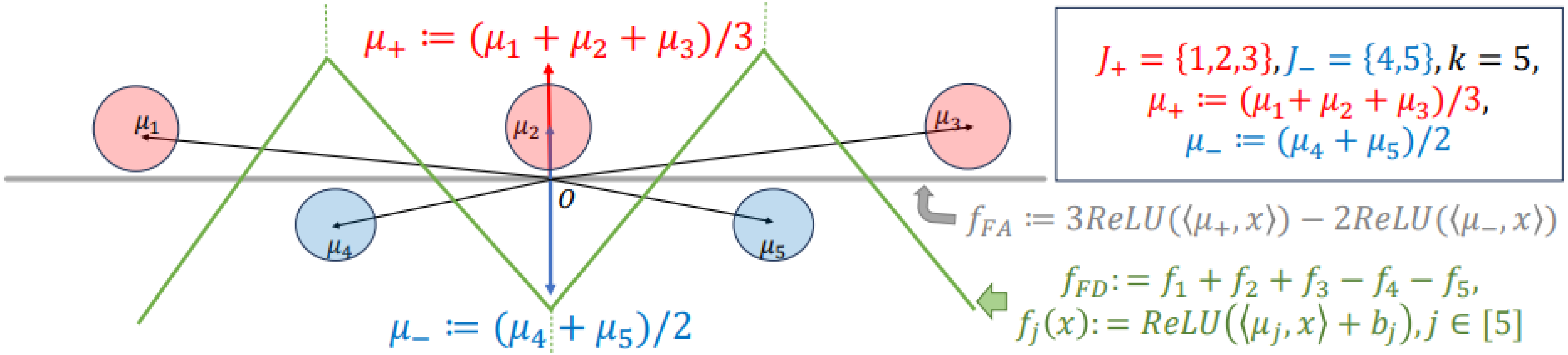


Figure 2: **Robustness Improvement on MNIST and CIFAR10** .

Take-Home Messages

- **Message I:** Adversarial examples may stem from averaged features learned by GD.

- **Message II:** More detailed/ structured supervisory information helps achieving models with better robustness.



Discussion

- **Regarding Data Assumption:** Indeed, multi-cluster data is a feature-level-structure. Can we consider another pixel-level data assumption?

- **Regarding Robust Learning Algorithm:** Can we design an algorithm that provably improves the network robustness?

Outline

- Introduction to Adversarial Examples in Deep Learning
- Theoretical Understanding of Adversarial Examples:
  1. Perspective of **Expressive Power**: *Robustness Requires Large Models*
  2. Perspective of **Training Dynamics** (*Feature Learning Theory*)
     a) *Gradient Descent Provably Converges to Non-Robust Solutions*
     b) *Adversarial Training Provably Improves Models' Robustness*
- Discussion on the Future of Adversarial Examples

Adversarial Training Can Provably Improve Robustness: Theoretical Analysis of Feature Learning Process Under Structured Data[1,2]

Binghui Li **Yuanzhi Li**

[1]This work has been accepted by **ICLR 2025**.

[2]Our full paper can be found at https://arxiv.org/abs/2410.08503 .

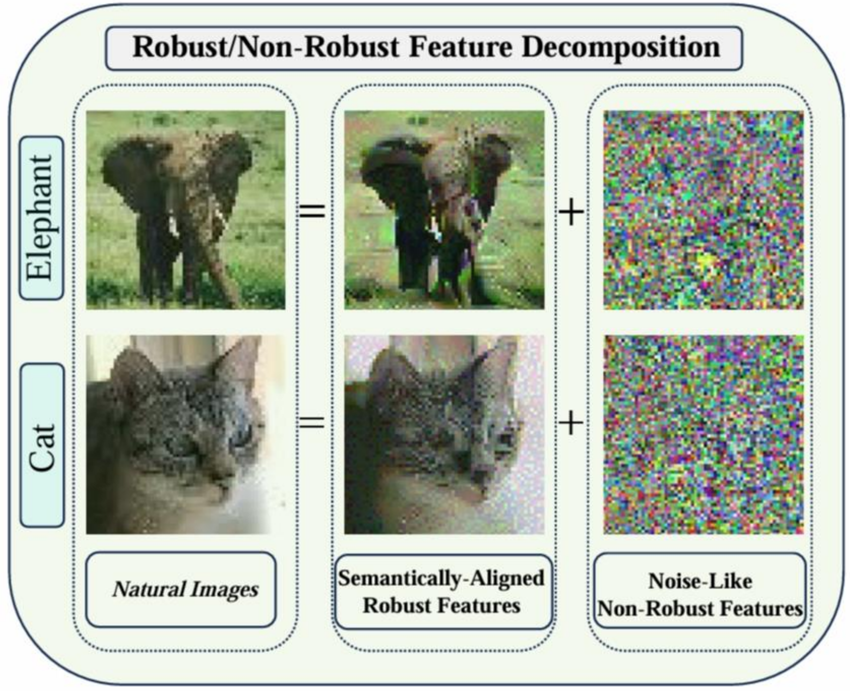
Our Fundamental Theoretical Questions :

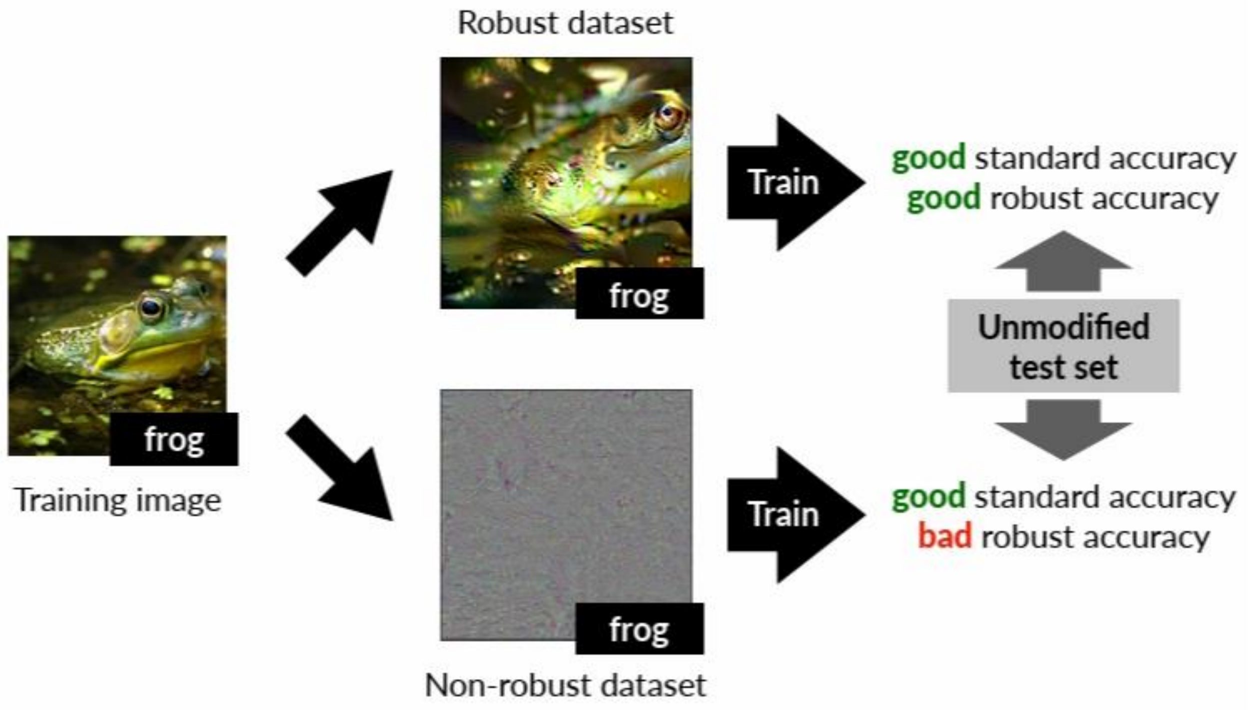
Q1: *Why do neural networks* ***trained by standard training*** *converge to the non-robust solutions that fail to classify* ***adversarial examples****?*

Q2: *How does* ***adversarial training algorithm*** *help* ***optimizing*** *neural networks to improve their robustness against adversarial perturbation?*

Robust and Non-robust Feature Decomposition

- A common challenge in analyzing adversarial training is the **gap between theory and practice**.
- To establish a realistic data model, we divide images into two types of features by **reconstruction** [Ilyas et al, 2019]. Specifically, we solve the optimization problem: $\min_{\widehat{X}} \left\| G(\widehat{X}) - G(X) \right\|_2$.
- Where X is some original image, $\widehat{X}$ is initialized by **random noise**, and G denotes the mapping from input to the representation layer for networks (a neural network **without the last FNN layer**).
- When G is chosen from a **Std/Adv trained** network, we derive the **non-robust/robust features**.



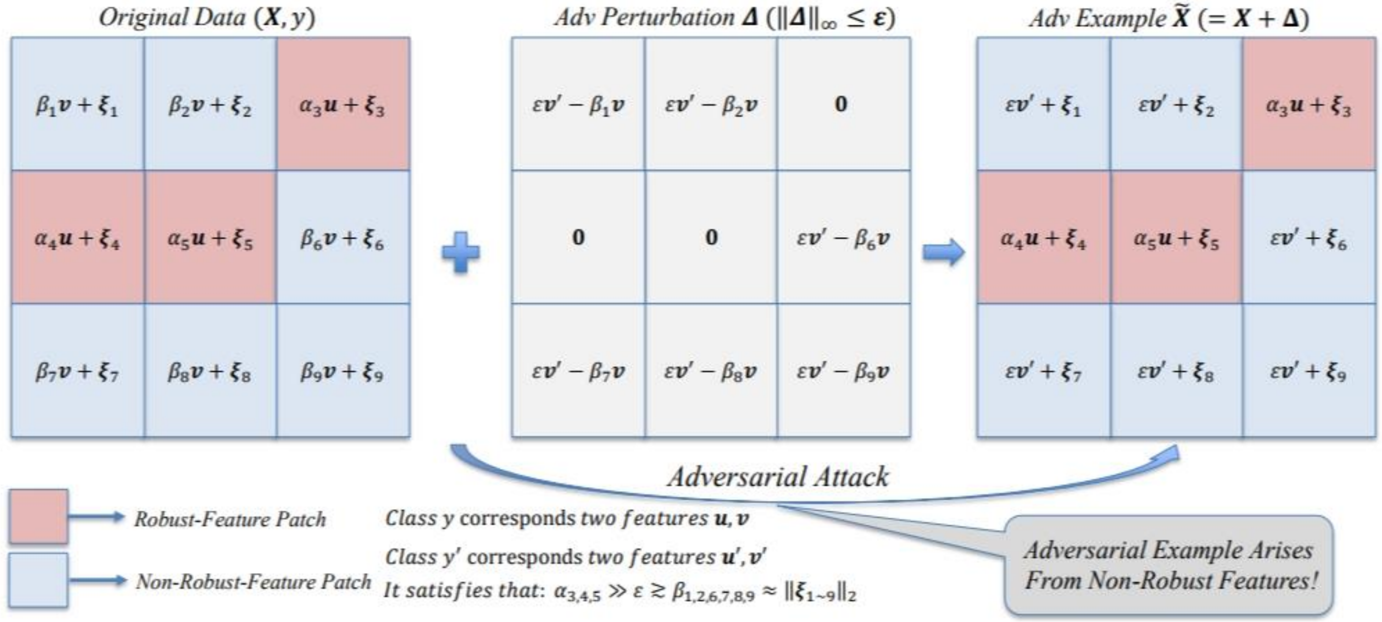


Patch-Structured Data Model

- Here, we mathematically represent this concept via the **patch-structured data** [Allen-Zhu and Li, 2023].
- We consider a **multiple classification task** with k classes. For each class $y \in [k]$, we assume that there exists a robust feature $\boldsymbol{u}_y$ and a non-robust feature $\boldsymbol{v}_y$. Then, our patch data is represented by $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_P) \in (\mathbb{R}^d)^P$ and label $y \in [k]$. And for each $p \in [P]$, the corresponding patch vector is generated as

$$\boldsymbol{x}_p := \begin{cases} \alpha_p \boldsymbol{u}_y + \boldsymbol{\xi}_p, \; \textit{if } p \in \mathcal{J}_R & \textit{(robust-feature patch)} \\ \beta_p \boldsymbol{v}_y + \boldsymbol{\xi}_p, \; \textit{if } p \in \mathcal{J}_{NR} & \textit{(non-robust-feature patch)} \end{cases}$$

where $\alpha_p, \beta_p > 0$ are the random coefficients sampled from the distribution $\mathcal{D}_{\alpha,y}, \mathcal{D}_{\beta,y}$ respectively, and $\boldsymbol{\xi}_p \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathcal{I}_d)$ is the random Gaussian noise with variance σ_n^2.



Data Assumption

1. Robust feature is **stronger** than non-robust feature:

$$\forall (p, p') \in \mathcal{J}_R \times \mathcal{J}_{NR}, \alpha_p \gg \beta_{p'}.$$

2. Non-robust feature is **denser** than robust feature:

$$\exists \tau \geq 0, \sum_{p \in \mathcal{J}_R} \alpha_p^\tau \ll \sum_{p \in \mathcal{J}_{NR}} \beta_p^\tau.$$

Network Learner

- **Two-Layer Convolutional Neural Network:** For the k-class classification task, we consider the following two-layer convolutional neural network as $\boldsymbol{F}(\boldsymbol{X}) \coloneqq \left(F_1(\boldsymbol{X}), F_2(\boldsymbol{X}), \ldots, F_k(\boldsymbol{X})\right): \left(\mathbb{R}^d\right)^P \to \mathbb{R}^k$, and $F_i(\boldsymbol{X})$ denotes

$$F_i(\boldsymbol{X}) \coloneqq \sum_{r \in [m]} \sum_{p \in [P]} \widetilde{ReLU}(< \boldsymbol{w}_{i,r}, \boldsymbol{x}_p >)$$

- Where $\widetilde{ReLU}$ denotes smoothed ReLU activation function, and $\{\boldsymbol{w}_{i,r}\}$ are learnable weights for different convolutional filters.

- **Robust Feature Learning:** $max_{r \in [m]} < \boldsymbol{w}_{i,r}, \boldsymbol{u}_i >$
- **Non-Robust Feature Learning:** $max_{r \in [m]} < \boldsymbol{w}_{i,r}, \boldsymbol{v}_i >$

Main Result I: Non-Robust Feature Learning Dominates During Standard Training

Theorem 1 (Standard Training Converges to Non-robust Global Minima). *Under our framework, we prove that two-layer neural network trained by standard training from random initialization satisfies:*

- *Standard training is perfect.*
- *Non-robust features are learned well, i.e.*

$$max_{r\in[m]} < \boldsymbol{w}_{i,r}, \boldsymbol{v}_i > \gg max_{r\in[m]} < \boldsymbol{w}_{i,r}, \boldsymbol{u}_i >$$

  for each class $i \in [k]$.
- *Standard test accuracy is good.*
- *Robust test accuracy is bad, even for model-independent perturbations that are generated by non-robust features.*

Main Result II: Adversarial Training Provably Helps Robust Feature Learning

Theorem 2 (Adversarial Training Converges to Robust Global Minima). *Under our framework, we prove that two-layer neural network trained by adversarial training from random initialization satisfies:*

- *Adversarial training is perfect.*
- *Robust features are learned well, i.e.*

$$max_{r\in[m]} < \boldsymbol{w}_{i,r}, \boldsymbol{u}_i > \gg max_{r\in[m]} < \boldsymbol{w}_{i,r}, \boldsymbol{v}_i >$$

  for each class $i \in [k]$.
- *Standard test accuracy is good.*
- *Robust test accuracy is also good.*

Simulation Experiments on Synthetic Data

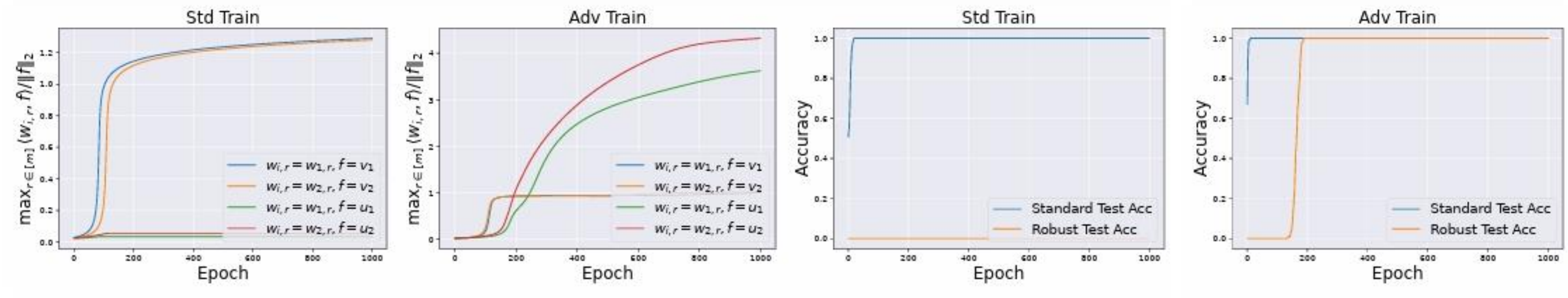
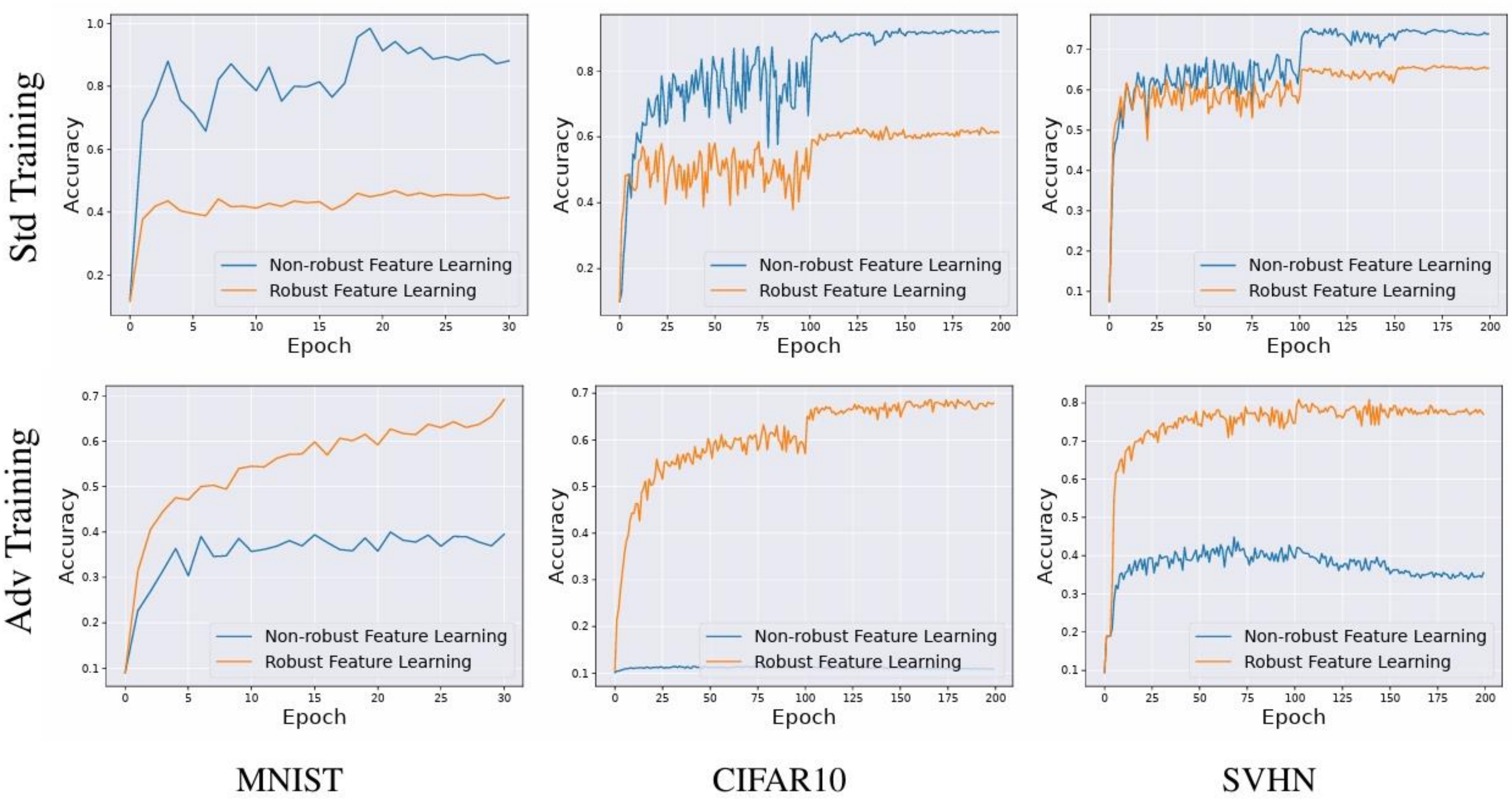


Figure 3: **Simulation Experiments on Synthetic Datasets. The two figures on the left:** dynamics of normalized weight-feature correlations for standard/adversarial training. **The two figures on the right:** learning curves for standard/adversarial training. We observe that, in standard training, non-robust feature learning (measured by $\max_{r\in[m]}\langle \boldsymbol{w}_{i,r}, \boldsymbol{v}_i\rangle/\|\boldsymbol{v}_i\|_2$) dominates during training process. There exists a phase transition during adversarial training (it happens nearly at 150-epoch). **At Phase I:** the network learner mainly learns non-robust features to achieve perfect standard test accuracy, but robust test accuracy maintains zero. **At Phase II:** the increments of non-robust feature learning is restrained while robust feature learning and robust test accuracy start to increase fast.

Experiments: Feature Learning Process on Real Images



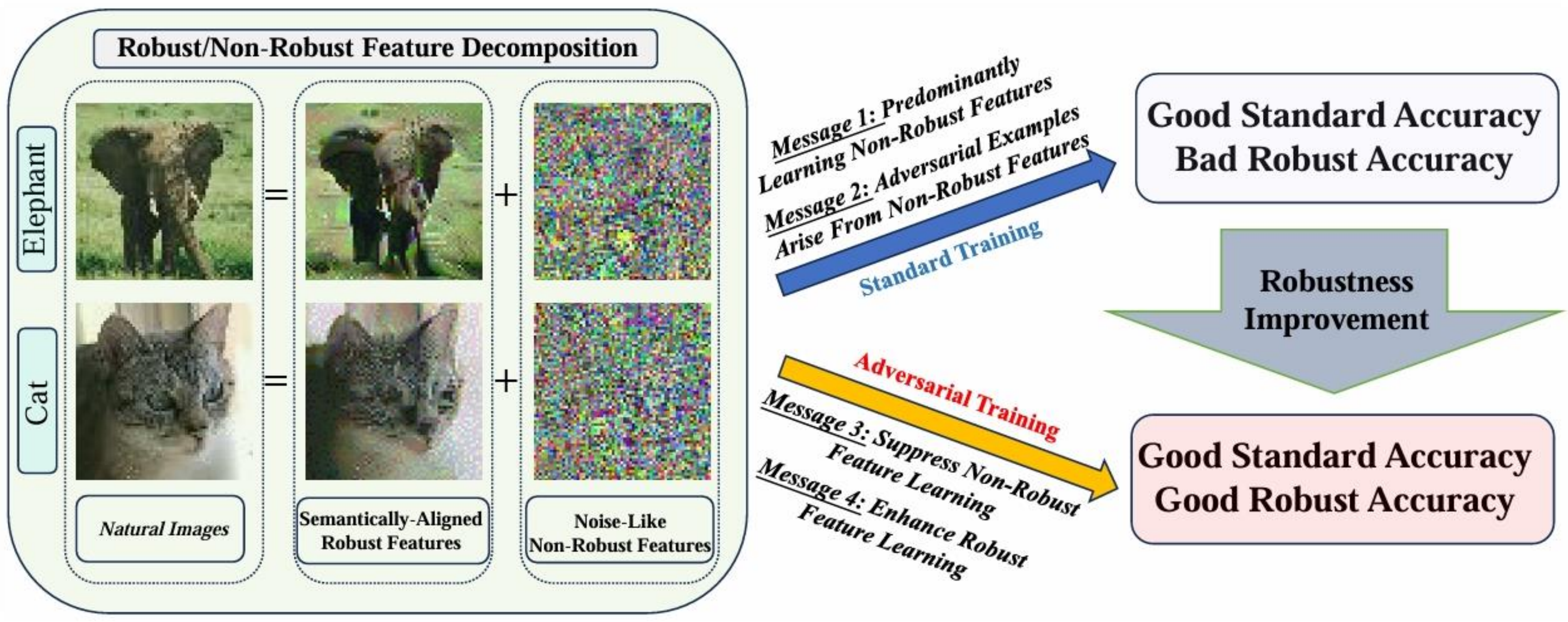
Experiments: Adversarial Examples Arise From Non-Robust Features

Table 2: Targeted Attack on CIFAR10

Model	Attack	Cat $\rightarrow$ Dog	Dog $\rightarrow$ Cat	Car $\rightarrow$ Plane	Plane $\rightarrow$ Car
Std Train	NRF-PGD	71.41 ± 1.17	80.36 ± 0.28	54.08 ± 0.99	76.74 ± 0.77
	RF-PGD	11.30 ± 0.55	9.58 ± 0.58	1.24 ± 0.10	2.63 ± 0.13
Adv Train	NRF-PGD	9.60 ± 0.18	15.16 ± 0.23	0.34 ± 0.04	0.40 ± 0.00
	RF-PGD	19.38 ± 0.29	26.00 ± 0.67	2.64 ± 0.18	1.96 ± 0.13

- **NRF-PGD:** Adversarial attacks from non-robust features.
- **RF-PGD:** Adversarial attacks from robust features.

Take-Home Messages



Outline

- Introduction to Adversarial Examples in Deep Learning
- Theoretical Understanding of Adversarial Examples:
    1. Perspective of **Expressive Power**: *Robustness Requires Large Models*
    2. Perspective of **Training Dynamics** (*Feature Learning Theory*)
        a) *Gradient Descent Provably Converges to Non-Robust Solutions*
        b) *Adversarial Training Provably Improves Models' Robustness*
- Discussion on the Future of Adversarial Examples

Discussion

- **In practice**, adversarial robustness highlights the gap between machine and human vision (alignment).

- **In theory**, the robustness of neural network is a fundamental theoretical issue, which helps us understand what neural network learns in deep learning (feature learning).

Thanks for listening!

My Homepage

Robust Generalization Paper

Feature Averaging Paper

Adversarial Training Paper

Reference I

- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. and Fergus, R. (2013). Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199.
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D. and Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In International Conference on Learning Representations.
- Raghunathan, A., Xie, S. M., Yang, F., Duchi, J. C., & Liang, P. (2019). Adversarial training can hurt generalization. arXiv preprint arXiv:1906.06032.
- Rice, L., Wong, E., & Kolter, Z. (2020, November). Overfitting in adversarially robust deep learning. In International conference on machine learning (pp. 8093-8104). PMLR.

Reference II

- Yang, Y. Y., Rashtchian, C., Zhang, H., Salakhutdinov, R. R., & Chaudhuri, K. (2020). A closer look at accuracy vs. robustness. Advances in neural information processing systems, 33, 8588-8601.
- Li, B., Jin, J., Zhong, H., Hopcroft, J., & Wang, L. (2022). Why robust generalization in deep learning is difficult: Perspective of expressive power. Advances in Neural Information Processing Systems, 35, 4370-4384.
- Frei, S., Vardi, G., Bartlett, P. and Srebro, N. (2024). The double-edged sword of implicit bias: Generalization vs. robustness in relu networks. Advances in Neural Information Processing Systems, 36.
- Li, B., Pan, Z., Lyu, K., & Li, J. (2024). Feature Averaging: An Implicit Bias of Gradient Descent Leading to Non-Robustness in Neural Networks. arXiv preprint arXiv:2410.10322.

Reference III

- Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B. and Madry, A. (2019). Adversarial examples are not bugs, they are features. Advances in neural information processing systems, 32.
- Allen-Zhu, Z. and Li, Y. (2023). Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. In The Eleventh International Conference on Learning Representations.
- Li, B., & Li, Y. (2024). Adversarial Training Can Provably Improve Robustness: Theoretical Analysis of Feature Learning Process Under Structured Data. arXiv preprint arXiv:2410.08503.