GeekPwn 极棒 × DEFCON

THE UPRISING

GEEKPWN AI/Robotics Cybersecurity Contest U.S. 2018

Team members:

Chao Du

Yinpeng Dong

Xingxing Wei

Tianyu Pang (Me)

Fangzhou Liao

NIPS 2017: Non-targeted Attack Competition – 1st

NIPS 2017: Non-targeted Attack Competition – 1st

NIPS 2017: Defense Competition – 1st

- Defense against Adversarial Attacks Using High-level Representation Guided Denoiser **(CVPR 2017)**

  Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Jun Zhu, and Xiaolin Hu

- Boosting Adversarial Attacks with Momentum **(CVPR 2017)**

  Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Xiaolin Hu, Jianguo Li, and Jun Zhu

- Max-Mahalanobis Linear Discriminant Analysis Networks **(ICML 2018)**

  Tianyu Pang, Chao Du, and Jun Zhu

- Towards Robust Detection of Adversarial Examples **(Under review of NIPS 2018)**

  Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu

- Defense against Adversarial Attacks Using High-level Representation Guided Denoiser **(CVPR 2017)**

  Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Jun Zhu, and Xiaolin Hu

- Boosting Adversarial Attacks with Momentum **(CVPR 2017)**

  Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Xiaolin Hu, Jianguo Li, and Jun Zhu

- Max-Mahalanobis Linear Discriminant Analysis Networks **(ICML 2018)**

  Tianyu Pang, Chao Du, and Jun Zhu

- Towards Robust Detection of Adversarial Examples **(Under review of NIPS 2018)**

  Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu

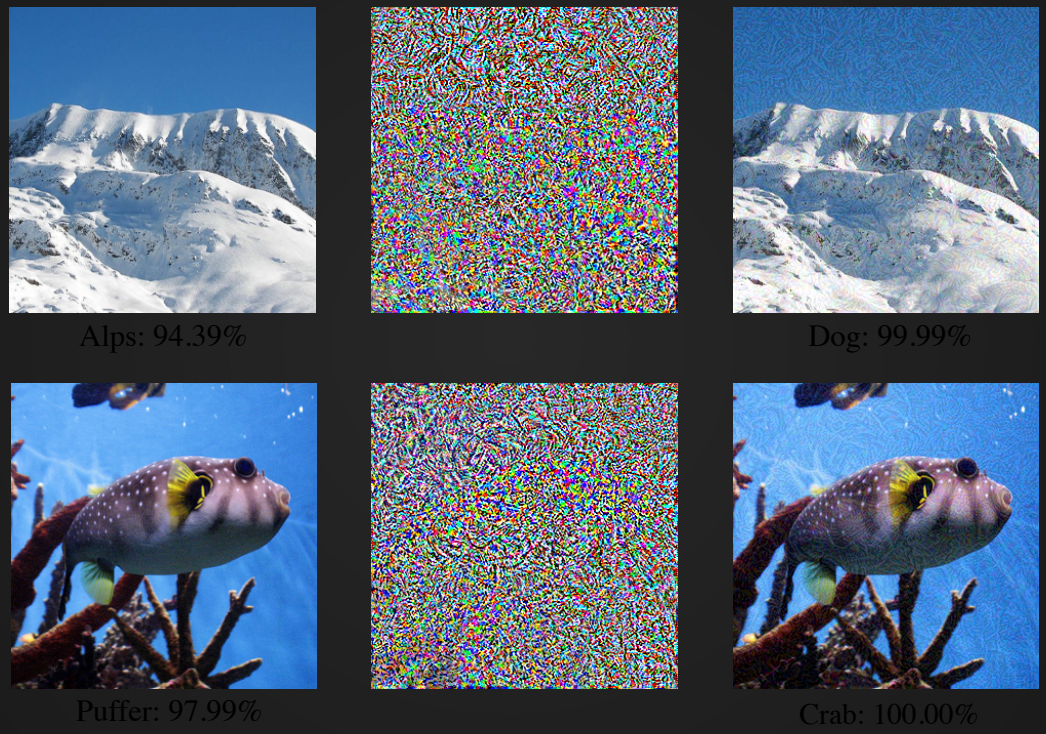
Boosting Adversarial Attacks with Momentum

Yinpeng Dong[1], Fangzhou Liao[1], Tianyu Pang[1],

Hang Su[1], Jun Zhu[1], Xiaolin Hu[1], Jianguo Li[2]

[1] Tsinghua University, [2] Intel Labs China

Adversarial Examples



- Szegedy et al 2013: Intriguing properties of neural networks.

Overview (White-box Attacks)

Constrained optimization of adversarial attacks:

$$\underset{x^*}{\operatorname{argmax}} L(x^*, y) \quad s.t. \|x^* - x\|_\infty \leq \epsilon$$

- One-step FGSM (Goodfellow et al., 2015)

$$x^* = x + \epsilon \cdot \operatorname{sign}(\nabla_x L(x, y))$$

- Iterative FGSM (I-FGSM, Kurakin et al., 2016)

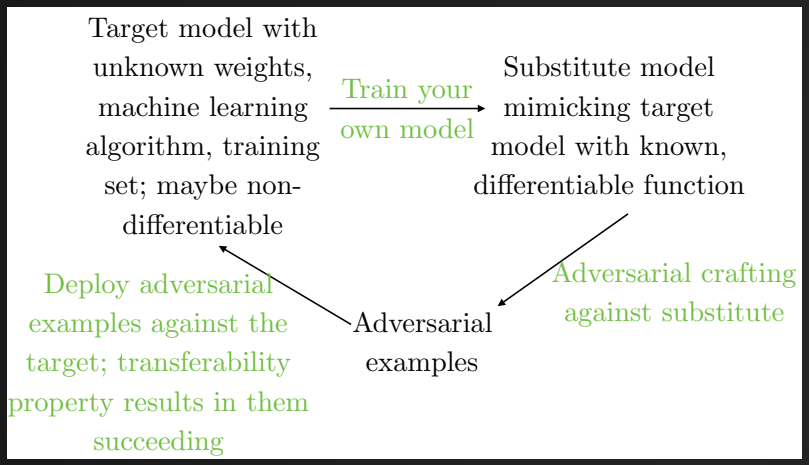
$$x_0^* = x, \quad x_{t+1}^* = \operatorname{clip}(x_t^* + \alpha \cdot \operatorname{sign}(\nabla_x L(x_t^*, y)))$$

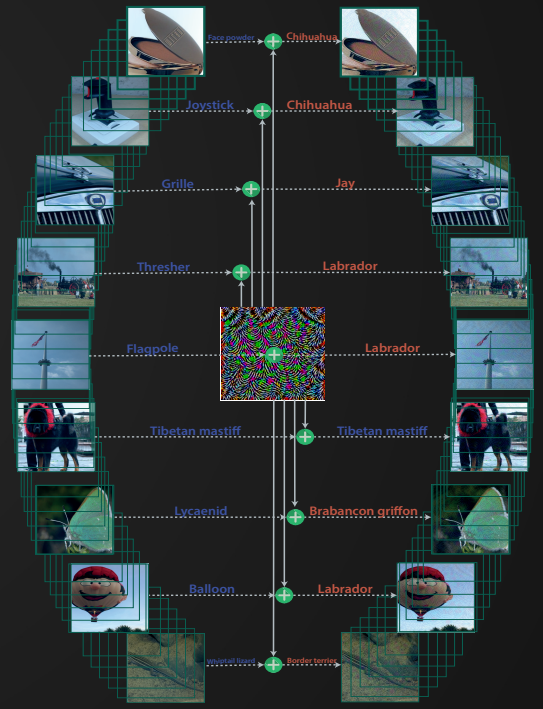
- Optimization-based methods (Carlini and Wagner, 2017)

$$\min d(x^*, x) - L(x^*, y)$$

Black-box Attacks (Transferability)

- Cross-model transferability (Liu et al., 2017)

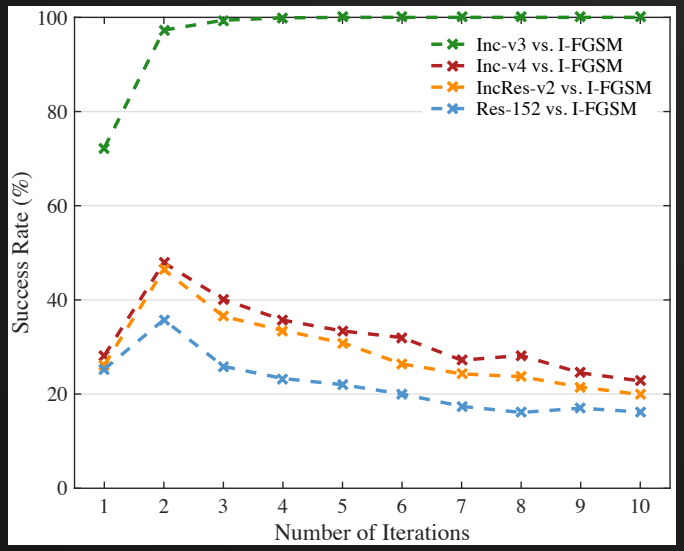




- Cross-data transferability (Moosavi-Dezfooli et al., 2017)

Limitations of Black-box Attacks (1)

- FGSM have poor white-box attack ability;
- Iterative FGSM have poor transferability;
- The **trade-off** between transferability and attack ability, makes black-box attacks less effective.



- Attack Inception V3;
- Evaluate the success rates of attacks on Inception V3, Inception V4, Inception ResNet V2, ResNet v2-152;
- $\epsilon = 16$;
- 1000 images from ImageNet.

Limitations of Black-box Attacks (2)

- Train a substitute network (Papernot et al., 2017) to fully characterize the behavior of the black-box model
  - Require full prediction confidence;
  - Require tremendous queries;
- Hard to deploy for models trained on large-scale dataset
- Impossible for cases without querying

- Our solution: alleviate the trade-off between transferability and attack ability.

Optimization with Momentum

- Constrained optimization of adversarial attacks:

$$\underset{x^*}{\operatorname{argmax}}\, L(x^*, y) \quad s.t. \|x^* - x\|_\infty \leq \epsilon$$

- Accelerate gradient descent;
- Escape from poor local minima and maxima;
- Stabilize update directions of stochastic gradient descent;
- Momentum can be used for adversarial attacks
  - It is still a white-box attack method but has strong black-box attack ability (transferability)

Momentum Iterative FGSM

$$x_0^* = x, \quad x_{t+1}^* = \mathrm{clip}(x_t^* + \alpha \cdot \mathrm{sign}(\nabla_x L(x_t^*, y)))$$

Momentum

$$x_0^* = x, g_0 = 0$$

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x L(x_t^*, y)}{\|\nabla_x L(x_t^*, y)\|_1}$$

$$x_{t+1}^* = \mathrm{clip}(x_t^* + \alpha \cdot \mathrm{sign}(g_{t+1}))$$

- μ is the decay factor;
- g_t accumulates the gradient w.r.t. input space of the first t iterations;
- The current gradient is normalized.

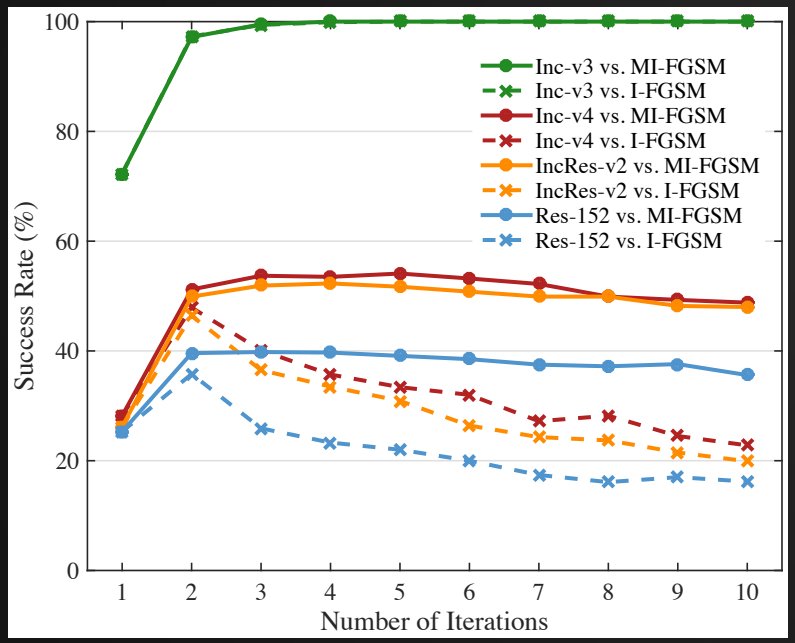
Non-targeted Results

- $\epsilon = 16, \mu = 1.0$, 10 iterations

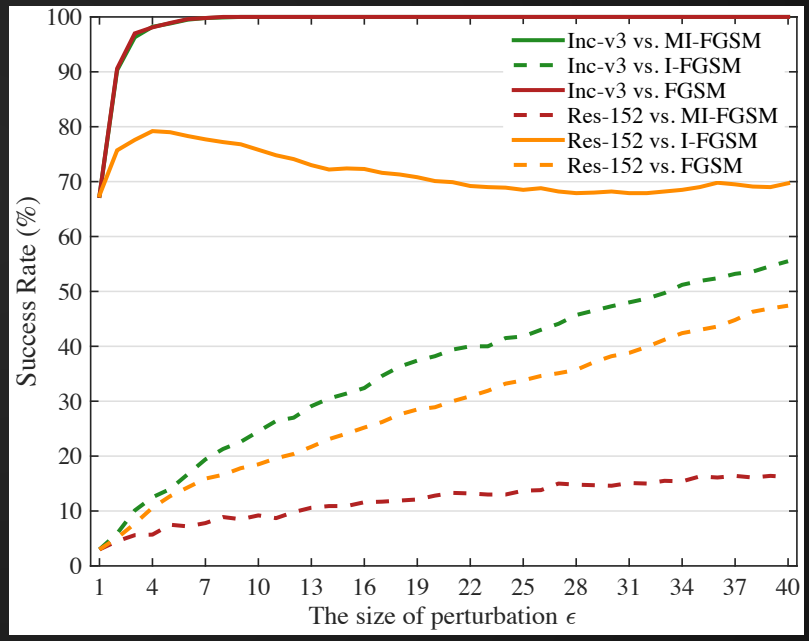
	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-152	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{ens}$
Inc-v3	FGSM	72.3*	28.2	26.2	25.3	11.3	10.9	4.8
	I-FGSM	**100.0***	22.8	19.9	16.2	7.5	6.4	4.1
	MI-FGSM	**100.0***	**48.8**	**48.0**	**35.6**	**15.1**	**15.2**	**7.8**
Inc-v4	FGSM	32.7	61.0*	26.6	27.2	13.7	11.9	6.2
	I-FGSM	35.8	**99.9***	24.7	19.3	7.8	6.8	4.9
	MI-FGSM	**65.6**	**99.9***	**54.9**	**46.3**	**19.8**	**17.4**	**9.6**
IncRes-v2	FGSM	32.6	28.1	55.3*	25.8	13.1	12.1	7.5
	I-FGSM	37.8	20.8	**99.6***	22.8	8.9	7.8	5.8
	MI-FGSM	**69.8**	**62.1**	99.5*	**50.6**	**26.1**	**20.9**	**15.7**
Res-152	FGSM	35.0	28.2	27.5	72.9*	14.6	13.2	7.5
	I-FGSM	26.7	22.7	21.2	**98.6***	9.3	8.9	6.2
	MI-FGSM	**53.6**	**48.9**	**44.7**	98.5*	**22.1**	**21.7**	**12.9**

Ablation Study

- Attack Inception V3 with $\epsilon = 16$



- Attack Inception V3 with $\alpha = 1$



Attacking an Ensemble of Models

- If an adversarial example remain adversarial for multiple models, it is more likely to be misclassified by other black-box models.
- Ensemble in logits

$$l(x) = \sum_{i=1}^{K} w_i l_i(x)$$

- The loss is defined as

$$J(x, y) = -1_y \cdot \log(\text{softmax}(l(x)))$$

- Comparisons:
  - Ensemble in predictions: $p(x) = \sum_{i=1}^{K} w_i p_i(x)$
  - Ensemble in loss: $J(x, y) = \sum_{i=1}^{K} w_i J_i(x, y)$

Non-targeted Results (2)

- $\epsilon = 16, \mu = 1.0$, 20 iterations, equal ensemble weights

	Ensemble method	FGSM		I-FGSM		MI-FGSM	
		Ensemble	Hold-out	Ensemble	Hold-out	Ensemble	Hold-out
-Inc-v3	Logits	**55.7**	**45.7**	**99.7**	**72.1**	**99.6**	**87.9**
	Predictions	52.3	42.7	95.1	62.7	97.1	83.3
	Loss	50.5	42.2	93.8	63.1	97.0	81.9
-Inc-v4	Logits	**56.1**	**39.9**	**99.8**	**61.0**	**99.5**	**81.2**
	Predictions	50.9	36.5	95.5	52.4	97.1	77.4
	Loss	49.3	36.2	93.9	50.2	96.1	72.5
-IncRes-v2	Logits	**57.2**	**38.8**	**99.5**	**54.4**	**99.5**	**76.5**
	Predictions	52.1	35.8	97.1	46.9	98.0	73.9
	Loss	50.7	35.2	96.2	45.9	97.4	70.8
-Res-152	Logits	**53.5**	**35.9**	99.6	**43.5**	99.6	**69.6**
	Predictions	51.9	34.6	**99.9**	41.0	**99.8**	67.0
	Loss	50.4	34.1	98.2	40.1	98.8	65.2

Max-Mahalanobis Linear Discriminant Analysis Networks

Tianyu Pang, Chao Du and Jun Zhu

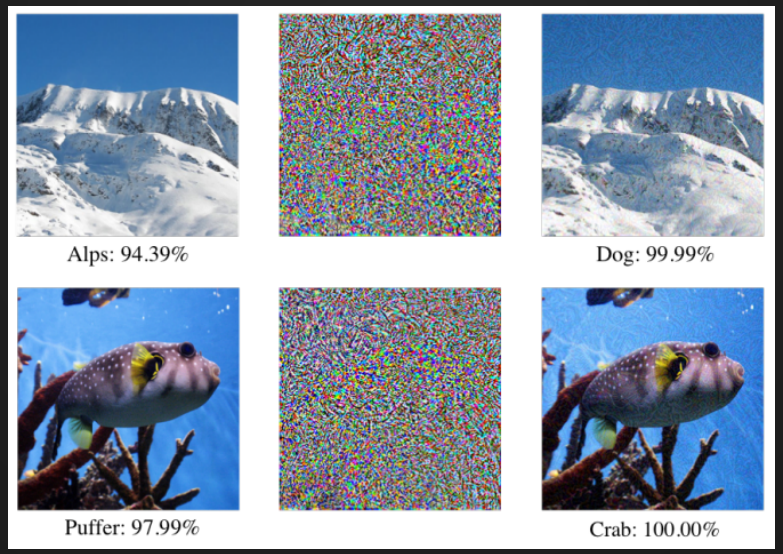
Department of Computer Science and Technology
Tsinghua University

ICML | 2018

Motivation

Motivation one

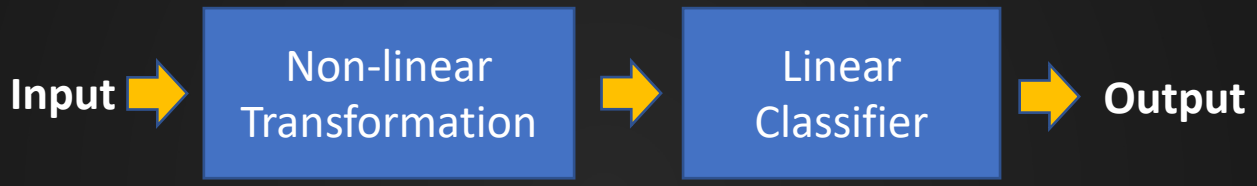
- **Almost all popular networks suffer from adversarial attacks**



From Dong et al. (2018)

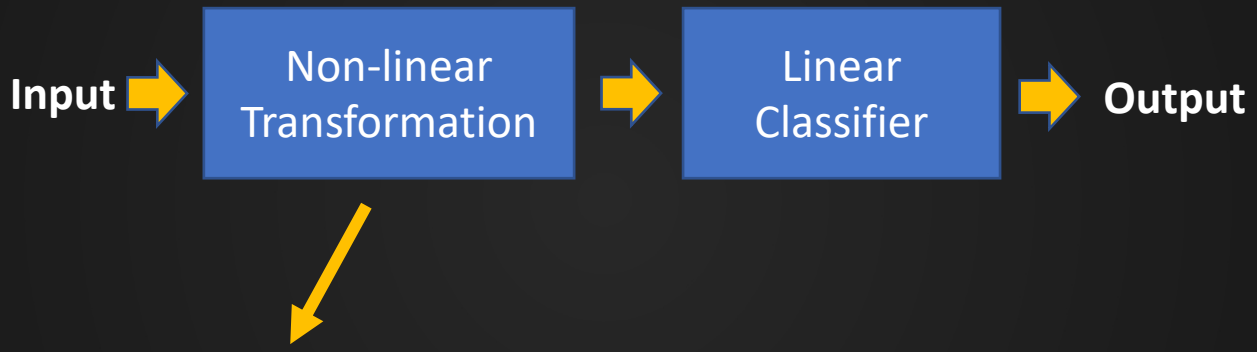
Motivation two

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



Motivation two

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

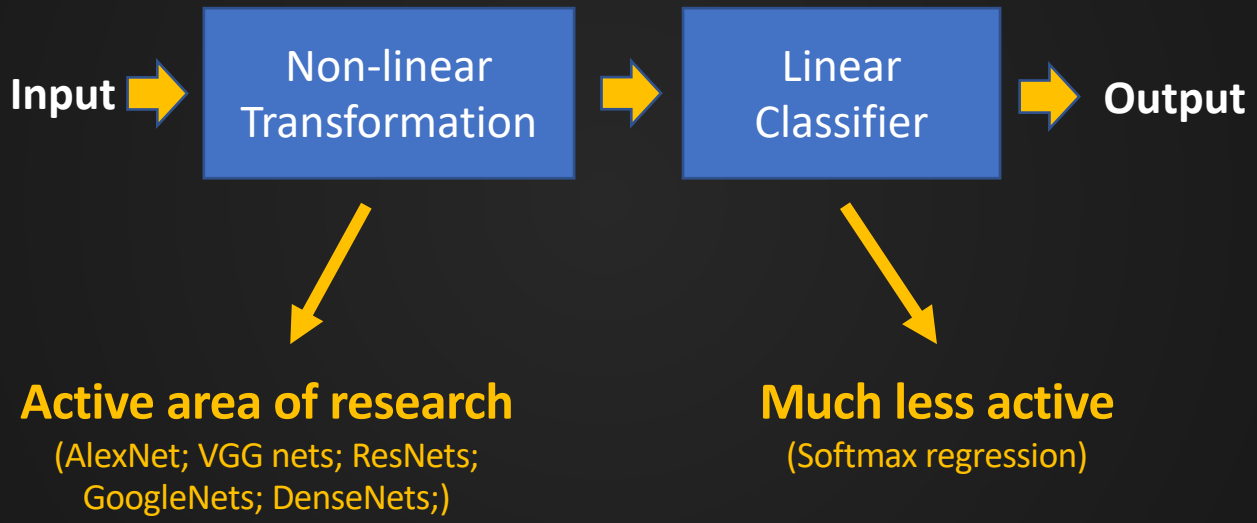


Active area of research

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

Motivation two

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



Our goal

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

Our goal

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

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

Our Method
(MM-LDA networks)

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).

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).

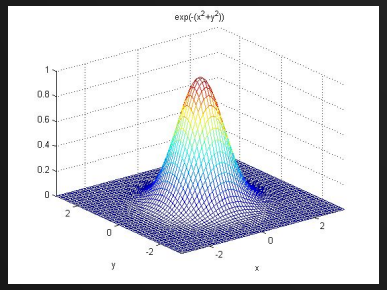
LDA needs less training data than LR to obtain certain error rate

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

Inspiration two: Neural networks are powerful

Inspiration two: Neural networks are powerful

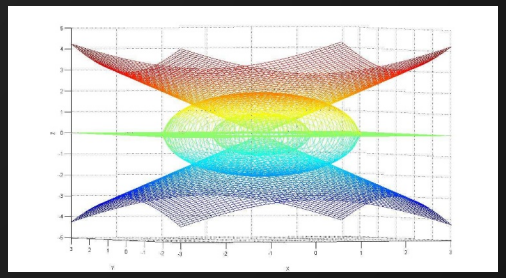
- **Deep generative models (e.g., GANs) are successful.**



Simple Distribution
(Gaussian/Mixture of Gaussian)

Deep generative models

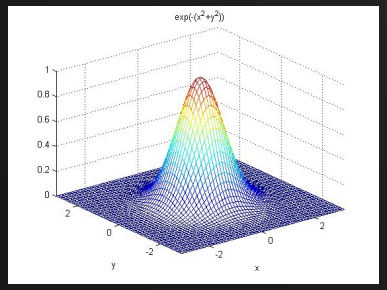
DNN



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.**

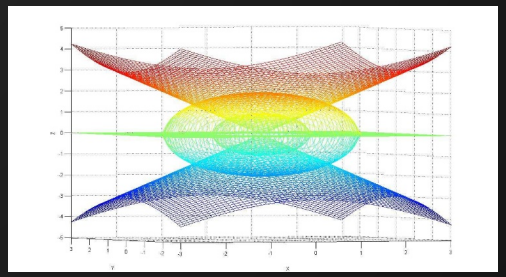


Simple Distribution
(Gaussian/Mixture of Gaussian)

Deep generative models

DNN

Our Method
(MM-LDA networks)



Complex Distribution
(Data distribution)

The Solution

Our method

- **Models the feature distribution in DNNs as a mixture of Gaussian.**

The Solution

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?

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.

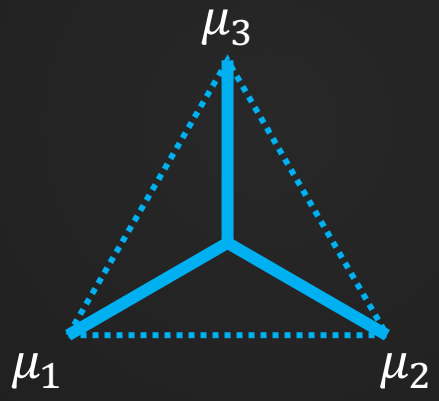
- We treat them as hyperparameters calculated by our algorithm, which can **provide theoretical guarantee on the robustness.**

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

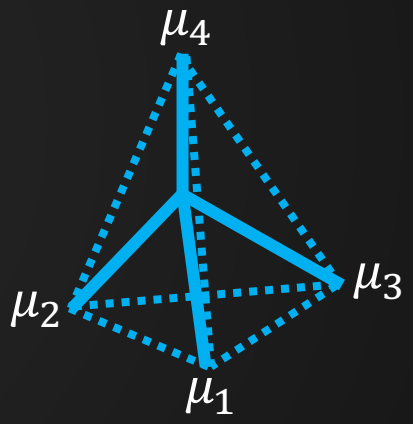
Max Mahalanobis Distribution (MMD)

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

$L = 2$
Straight Line



$L = 3$
Equilateral Triangle



$L = 4$
Regular Tetrahedron

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.

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

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

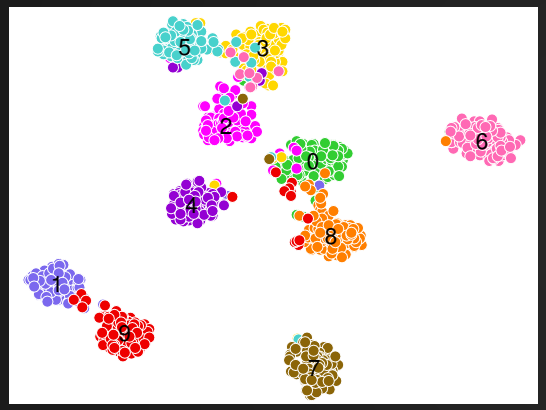
Experiments

Performance on normal examples

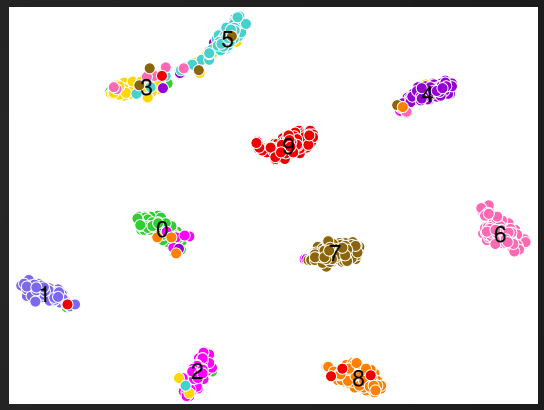
Table 2. Error rates (%) on the test sets of MNIST and CIFAR-10.

Model	MNIST	CIFAR-10
Resnet-32 (SR)	0.38	**7.13**
Resnet-32 (MM-LDA)	**0.35**	8.04

More orderly distribution in the feature space



SR networks



MM-LDA networks

Better robustness on iterative-based attacks

Table 1. Classification accuracy (%) on adversarial examples of MNIST and CIFAR-10. The investigated values of perturbation are 0.04, 0.12, and 0.20. **Boldface** indicates the best result under certain combination of a value of perturbation and an attacking method.

Perturbation	Model	MNIST				CIFAR-10			
		FGSM	BIM	ILCM	JSMA	FGSM	BIM	ILCM	JSMA
0.04	Resnet-32 (SR)	93.6	87.9	94.8	92.9	20.0	5.5	0.2	65.6
	Resnet-32 (SR) + SAT	86.7	68.5	98.4	-	24.4	7.0	0.4	-
	Resnet-32 (SR) + HAT	88.7	96.3	**99.8**	-	30.3	5.3	1.3	-
	Resnet-32 (MM-LDA)	**99.2**	**99.2**	99.0	**99.1**	**91.3**	**91.2**	**70.0**	**91.2**
0.12	Resnet-32 (SR)	28.1	3.4	20.9	56.0	10.2	4.1	0.3	20.5
	Resnet-32 (SR) + SAT	40.5	8.7	88.8	-	88.2	6.9	0.1	-
	Resnet-32 (SR) + HAT	40.3	40.1	92.6	-	44.1	8.7	0.0	-
	Resnet-32 (MM-LDA)	**99.3**	**98.6**	**99.6**	**99.7**	**90.7**	**90.1**	**42.5**	**91.1**
0.20	Resnet-32 (SR)	15.5	0.3	1.7	25.6	10.7	4.2	0.6	11.5
	Resnet-32 (SR) + SAT	17.3	1.1	69.4	-	**91.7**	9.4	0.0	-
	Resnet-32 (SR) + HAT	10.1	10.5	46.1	-	40.7	6.0	0.2	-
	Resnet-32 (MM-LDA)	**97.5**	**97.3**	**96.6**	**99.6**	89.5	**89.7**	**31.2**	**91.8**

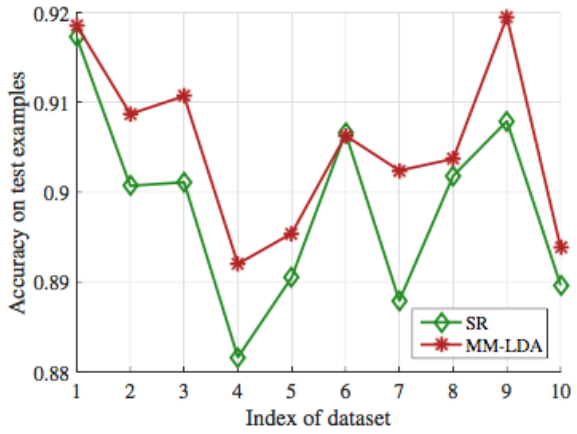
Better robustness on optimization-based attack

Table 3. Average distortions of the adversarial examples crafted by the C&W attack on MNIST and CIFAR-10.

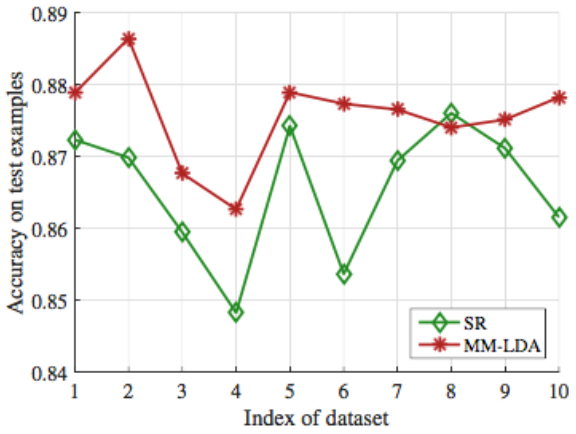
Model	MNIST	CIFAR-10
Resnet-32 (SR)	8.56	0.67
Resnet-32 (MM-LDA)	**16.32**	**2.80**

Better robustness on optimization-based attack

Better performance on class-biased datasets



(a) BP1



(b) BP2

Figure 4. Classification accuracy on the test sets of class-biased datasets. Each index of dataset corresponds to a counterpart of the bias probability. The original class-unbiased dataset is CIFAR-10.

Conclusion

- **No extra computational cost**

Conclusion

- **No extra computational cost**
- **With no loss of accuracy on normal examples**

Conclusion

- **No extra computational cost**
- **With no loss of accuracy on normal examples**
- **Quite easy to implement**

Conclusion

- **No extra computational cost**
- **With no loss of accuracy on normal examples**
- **Quite easy to implement**
- **Compatible with nearly all popular networks**

Thanks