Week 16: Security & robustness of deep learning

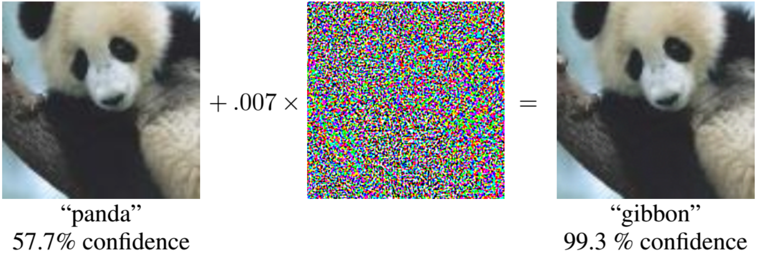
Instructor: Ruixuan Wang
wangruix5@mail.sysu.edu.cn

School of Data and Computer Science
Sun Yat-Sen University

13 June, 2019

Intriguing observation

- Adversarial examples: input with imperceptible perturbations, resulting in incorrect output with high confidence

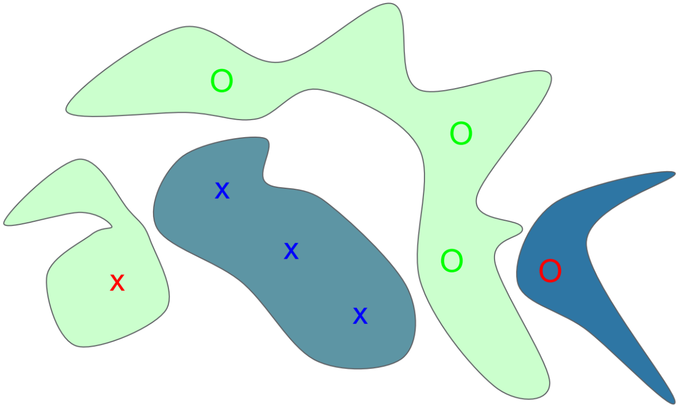


Not just in neural networks; also in most classifiers

Figure from Goodfellow et al., “Explaining and harnessing adversarial examples”, ICLR, 2015

Adversarial examples: reason

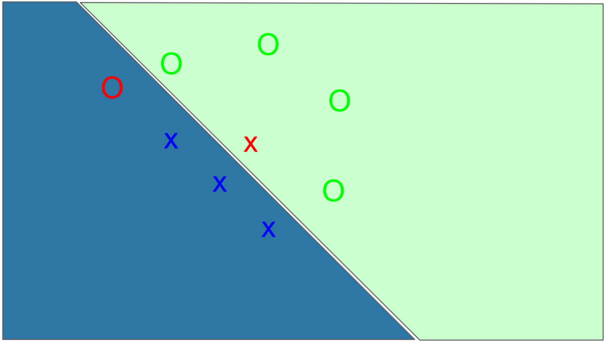
- Are adversarial examples from overfitting?



Figures here and in next slide from Stanford CS231n Lecture 16, 2017

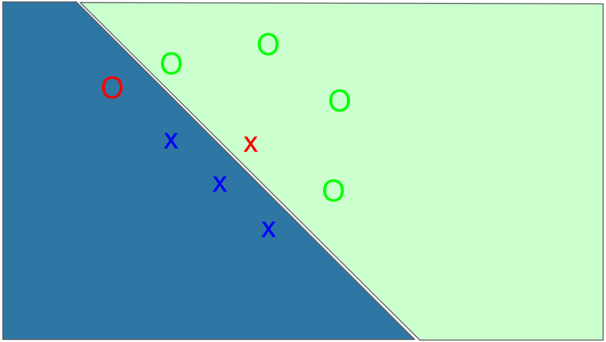
Adversarial examples: reason

- Adversarial examples may come from linearity of models!

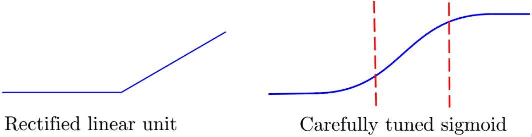


Adversarial examples: reason

- Adversarial examples may come from linearity of models!

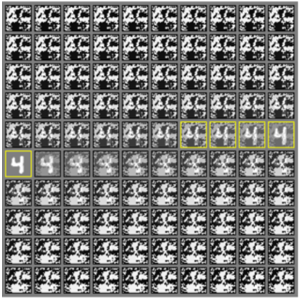


- Modern deep learning models are very piecewise linear



Adversarial examples: reason

- Model responses to changes in inputs are nearly linear (Right)!
- Left: perturbed inputs along gradient direction in input space



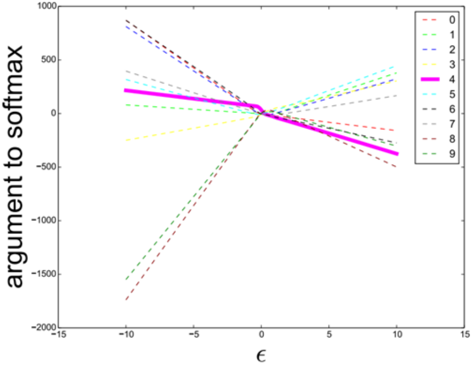


Figure from Goodfellow et al., "Explaining and harnessing adversarial examples", ICLR, 2015

FGSM: fast gradient sign method

- Model linearity provides one way to adversarial examples
- With 1^{st}-order Taylor expansion, loss $L(\boldsymbol{\theta}, \tilde{\mathbf{x}}, y)$ is approx by:

$$L(\boldsymbol{\theta}, \tilde{\mathbf{x}}, y) \quad \approx \quad L(\boldsymbol{\theta}, \mathbf{x}, y) + (\tilde{\mathbf{x}} - \mathbf{x})^T \, \nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y)$$

$\boldsymbol{\theta}$: model parameter; y: label of input $\mathbf{x}$; $\tilde{\mathbf{x}}$: perturbed input

FGSM: fast gradient sign method

- Model linearity provides one way to adversarial examples
- With 1^{st}-order Taylor expansion, loss $L(\boldsymbol{\theta}, \tilde{\mathbf{x}}, y)$ is approx by:

$$L(\boldsymbol{\theta}, \tilde{\mathbf{x}}, y) \approx L(\boldsymbol{\theta}, \mathbf{x}, y) + (\tilde{\mathbf{x}} - \mathbf{x})^T \nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y)$$

  $\boldsymbol{\theta}$: model parameter; y: label of input $\mathbf{x}$; $\tilde{\mathbf{x}}$: perturbed input
- Adversarial example $\tilde{\mathbf{x}}$ can be obtained by

$$\begin{aligned} &\arg\max_{\tilde{\mathbf{x}}} L(\boldsymbol{\theta}, \mathbf{x}, y) + (\tilde{\mathbf{x}} - \mathbf{x})^T \nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y) \\ &s.t. \ \|\tilde{\mathbf{x}} - \mathbf{x}\|_\infty < \epsilon \end{aligned}$$

  where L_∞ (max) norm fewer than ϵ controls perturbation!

FGSM: fast gradient sign method

- Model linearity provides one way to adversarial examples
- With 1^{st}-order Taylor expansion, loss $L(\boldsymbol{\theta}, \tilde{\mathbf{x}}, y)$ is approx by:

$$L(\boldsymbol{\theta}, \tilde{\mathbf{x}}, y) \approx L(\boldsymbol{\theta}, \mathbf{x}, y) + (\tilde{\mathbf{x}} - \mathbf{x})^T \nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y)$$

  $\boldsymbol{\theta}$: model parameter; y: label of input $\mathbf{x}$; $\tilde{\mathbf{x}}$: perturbed input
- Adversarial example $\tilde{\mathbf{x}}$ can be obtained by

$$\begin{aligned} &\arg\max_{\tilde{\mathbf{x}}} L(\boldsymbol{\theta}, \mathbf{x}, y) + (\tilde{\mathbf{x}} - \mathbf{x})^T \nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y) \\ &s.t. \ \|\tilde{\mathbf{x}} - \mathbf{x}\|_\infty < \epsilon \end{aligned}$$

  where L_∞ (max) norm fewer than ϵ controls perturbation!
- Solution: one-time computation, no need iteration

$$\tilde{\mathbf{x}} = \mathbf{x} + \epsilon \, \mathsf{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y))$$

Attack with adversarial examples

- **Attack**: use adversarial examples to decrease model's performance
- 'White-box attack': know model structures, parameters, etc.
- 'Black-box attack': can only get model output given input

Attack with adversarial examples

- **Attack**: use adversarial examples to decrease model's performance
- 'White-box attack': know model structures, parameters, etc.
- 'Black-box attack': can only get model output given input
- Black-box attack is more common: craft adversarial examples with Model B, attack model A
- White-box attack is stronger: degrade models more seriously

FGSM result

- On MNIST dataset: $\epsilon = 0.25$ (ϵ range $[0, 1]$), simple network, classification error 89.4%, average confidence 97.6%

FGSM result

- On MNIST dataset: $\epsilon = 0.25$ (ϵ range $[0, 1]$), simple network, classification error 89.4%, average confidence 97.6%
- On CIFAR-10 dataset: $\epsilon = 0.1$, simple network, classification error 87.2%, average confidence 96.6%

FGSM result

- On MNIST dataset: $\epsilon = 0.25$ (ϵ range $[0, 1]$), simple network, classification error 89.4%, average confidence 97.6%
- On CIFAR-10 dataset: $\epsilon = 0.1$, simple network, classification error 87.2%, average confidence 96.6%
- With random images, FGSM fooled CNN as 'airplane' (yellow)

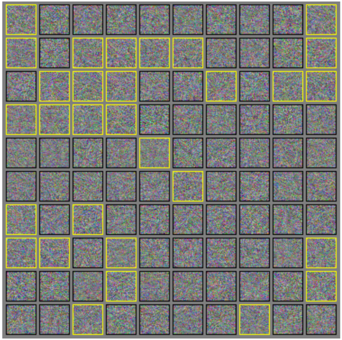


Figure from Goodfellow et al., "Explaining and harnessing adversarial examples", ICLR, 2015

Simple extensions of FGSM

- Generating *targeted* adversarial examples with FGSM

$$\tilde{\mathbf{x}} \quad = \quad \mathbf{x} - \epsilon \, \mathsf{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y_{target}))$$

where y_{target} is different from the true label of $\mathbf{x}$;
It would make classifier mis-classify $\tilde{\mathbf{x}}$ into class y_{target}

Simple extensions of FGSM

- Generating *targeted* adversarial examples with FGSM

$$\tilde{\mathbf{x}} = \mathbf{x} - \epsilon \, \mathsf{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y_{target}))$$

  where y_{target} is different from the true label of $\mathbf{x}$;
  It would make classifier mis-classify $\tilde{\mathbf{x}}$ into class y_{target}

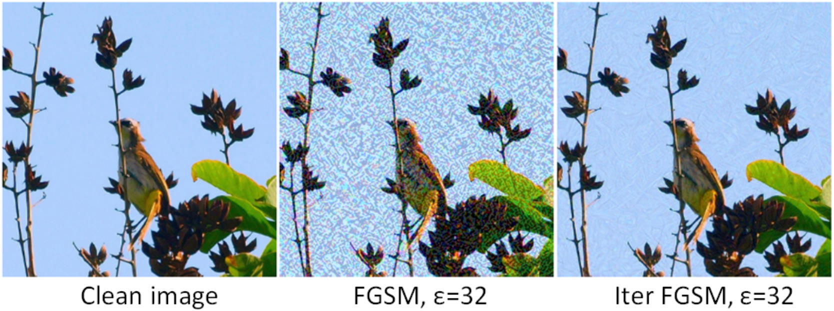
- Iterative FGSM: run FGSM multiple times, with $\alpha < \epsilon$

$$\mathbf{x}_{i+1} = Clip_{\epsilon,\mathbf{x}} \left\{ \mathbf{x}_i + \alpha \, \mathsf{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}_i, y)) \right\}$$

  where $Clip_{\epsilon,\mathbf{x}}$ is an operation assuring element-wise difference between $\mathbf{x}_{i+1}$ and original clean image $\mathbf{x}$ is within ϵ.

Iterative FGSM vs. original FGSM

- Iterative FGSM often generates more imperceptible adversarial examples (below: ϵ in range $[0, 255]$)



Figures and tables here and in next 3 slides from Kurakin et al., "Adversarial examples in the physical world", ICLR, 2017

Adversarial examples in the physical world

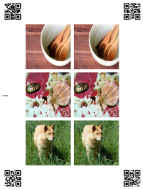
- AI systems operating in the physical world often capture images directly from camera.

Adversarial examples in the physical world

- AI systems operating in the physical world often capture images directly from camera.
- Can adversarial images in physical world also fool AI system?

Adversarial examples in the physical world

- AI systems operating in the physical world often capture images directly from camera.
- Can adversarial images in physical world also fool AI system?
- (a) Print image pairs (clean, adversarial)
- (b) Take a photo of printed image with a cell phone camera
- (c) Automatically crop and warp examples from the photo
- (d) Finally feed the cropped image to classifier



(a) Printout

(b) Photo of printout

(c) Cropped image

In physical world: white-box attacks

- Original FGSM ('fast') attack is more successful than iterative FGSM in the physical word

Adversarial method	Photos				Source images			
	Clean images		Adv. images		Clean images		Adv. images	
	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5
fast $\epsilon = 16$	79.8%	91.9%	36.4%	67.7%	85.3%	94.1%	36.3%	58.8%
fast $\epsilon = 8$	70.6%	93.1%	49.0%	73.5%	77.5%	97.1%	30.4%	57.8%
iter. basic $\epsilon = 16$	72.9%	89.6%	49.0%	75.0%	81.4%	95.1%	28.4%	31.4%
iter. basic $\epsilon = 8$	72.5%	93.1%	51.0%	87.3%	73.5%	93.1%	26.5%	31.4%

Note: classification accuracy in table

In physical world: white-box attacks

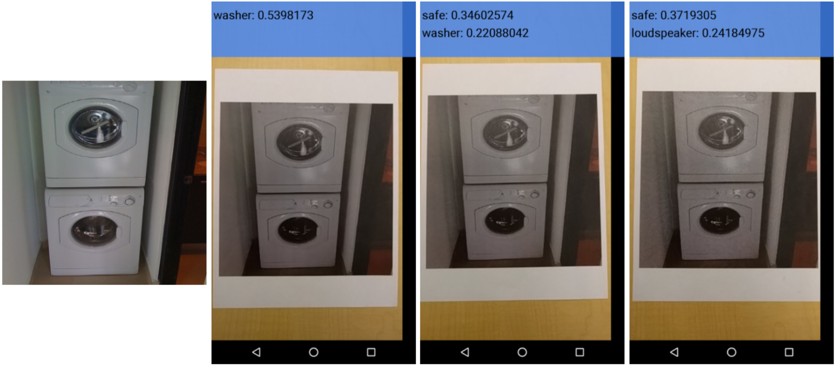
- Original FGSM ('fast') attack is more successful than iterative FGSM in the physical word
- Reason: iterative FGSM generates adversarial examples with smaller perturbations which could be more likely removed or affected by photo transformation

Adversarial method	Photos				Source images			
	Clean images		Adv. images		Clean images		Adv. images	
	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5
fast $\epsilon = 16$	79.8%	91.9%	36.4%	67.7%	85.3%	94.1%	36.3%	58.8%
fast $\epsilon = 8$	70.6%	93.1%	49.0%	73.5%	77.5%	97.1%	30.4%	57.8%
iter. basic $\epsilon = 16$	72.9%	89.6%	49.0%	75.0%	81.4%	95.1%	28.4%	31.4%
iter. basic $\epsilon = 8$	72.5%	93.1%	51.0%	87.3%	73.5%	93.1%	26.5%	31.4%

Note: classification accuracy in table

In physical world: black-box attack

- Black-box attack in the physical world also succeeds



(a) Image from dataset (b) Clean image (c) Adv. image, $\epsilon = 4$ (d) Adv. image, $\epsilon = 8$

Game: attack vs. defense

- **Defense**: reduce malicious effect of adversarial examples

Game: attack vs. defense

- **Defense**: reduce malicious effect of adversarial examples

Multiple rounds of ‘attack-defense’ game

How to defend adversarial examples from FGSM?

- Adversarial training: augment data with adversarial examples

How to defend adversarial examples from FGSM?

- Adversarial training: augment data with adversarial examples
- Find best $\boldsymbol{\theta}$ by minimizing $\tilde{L}(\boldsymbol{\theta}, \mathbf{x}, y)$ over all training data

$$\tilde{L}(\boldsymbol{\theta}, \mathbf{x}, y) = \alpha L(\boldsymbol{\theta}, \mathbf{x}, y) + (1 - \alpha) L(\boldsymbol{\theta}, \mathbf{x} + \epsilon \,\mathsf{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y)), y)$$

How to defend adversarial examples from FGSM?

- Adversarial training: augment data with adversarial examples
- Find best $\boldsymbol{\theta}$ by minimizing $\tilde{L}(\boldsymbol{\theta}, \mathbf{x}, y)$ over all training data

$$\tilde{L}(\boldsymbol{\theta}, \mathbf{x}, y) = \alpha L(\boldsymbol{\theta}, \mathbf{x}, y) + (1-\alpha) L(\boldsymbol{\theta}, \mathbf{x} + \epsilon \operatorname{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y)), y)$$

- 2^{nd} term: make adversarial examples correctly classified
- With adversarial training, classification error rate of adversarial examples on MNIST was reduced from 89.4% to 17.9%

How to defend adversarial examples from FGSM?

- Adversarial training: augment data with adversarial examples
- Find best $\boldsymbol{\theta}$ by minimizing $\tilde{L}(\boldsymbol{\theta}, \mathbf{x}, y)$ over all training data

$$\tilde{L}(\boldsymbol{\theta}, \mathbf{x}, y) = \alpha L(\boldsymbol{\theta}, \mathbf{x}, y) + (1-\alpha) L(\boldsymbol{\theta}, \mathbf{x} + \epsilon \, \text{sign}(\nabla_{\mathbf{x}} L(\boldsymbol{\theta}, \mathbf{x}, y)), y)$$

- 2^{nd} term: make adversarial examples correctly classified
- With adversarial training, classification error rate of adversarial examples on MNIST was reduced from 89.4% to 17.9%
- However, it works only for specific and known attack
- It remains higher vulnerable to (black-box) transferred adversarial examples produced by other models

Goodfellow et al., "Explaining and harnessing adversarial examples", ICLR, 2015

Randomized FGSM: improved attack method

- Why adversarial training succeed?
- Model's decision surface has sharp curvatures around data points, hindering attacks based on 1st-order approx of model's loss, but permitting black-box attacks

Randomized FGSM: improved attack method

- Why adversarial training succeed?
- Model's decision surface has sharp curvatures around data points, hindering attacks based on 1st-order approx of model's loss, but permitting black-box attacks
- A new attack method based on above reason
- Randomized FGSM: apply small perturbation before FGSM

$$
\begin{aligned}
\mathbf{x}' &= \mathbf{x} + \alpha \, \text{sign}(\mathcal{N}(\mathbf{0}, \mathbf{I})) \\
\tilde{\mathbf{x}} &= \mathbf{x}' + (\epsilon - \alpha) \, \text{sign}(\nabla_{\mathbf{x}'} L(\boldsymbol{\theta}, \mathbf{x}', y))
\end{aligned}
$$

Randomized FGSM: improved attack method

- Why adversarial training succeed?
- Model's decision surface has sharp curvatures around data points, hindering attacks based on 1st-order approx of model's loss, but permitting black-box attacks
- A new attack method based on above reason
- Randomized FGSM: apply small perturbation before FGSM

$$\begin{aligned} \mathbf{x}' &= \mathbf{x} + \alpha \, \mathsf{sign}(\mathcal{N}(\mathbf{0}, \mathbf{I})) \\ \tilde{\mathbf{x}} &= \mathbf{x}' + (\epsilon - \alpha) \, \mathsf{sign}(\nabla_{\mathbf{x}'} L(\boldsymbol{\theta}, \mathbf{x}', y)) \end{aligned}$$

- Again, it is a single-time gradient computation, no iteration
- Randomized FGSM outperforms FGSM (errors in tables)

	MNIST			ImageNet (top 1)			ImageNet (top 5)		
	A	A_{adv}	B	v3	$v3_{adv}$	v4	v3	$v3_{adv}$	v4
FGSM	71.4	3.6	84.6	69.7	26.8	60.2	42.8	9.0	30.8
RAND+FGSM	75.3	34.1	86.2	80.1	64.3	70.3	57.7	37.2	42.5

Improved defense for black-box attack

- Above: adversarial training is vulnerable to black-box attacks

Improved defense for black-box attack

- Above: adversarial training is vulnerable to black-box attacks
- Improved: ensemble adversarial training - using adversarial examples from current and other models during training

Improved defense for black-box attack

- Above: adversarial training is vulnerable to black-box attacks
- Improved: ensemble adversarial training - using adversarial examples from current and other models during training
- Ensemble adversarial training ($A_{adv-ens}$) shows lower errors for black-box attacks (last 4 columns)
- But it shows higher error for white-box attacks (2^{nd} column)

	Model	**Clean**	**FGSM**	**FGSM$_B$**	**I-FGSM$_B$**	**RAND+FGSM$_B$**	**CW$_B$**
6 epochs	A	**0.9**	71.4	62.4	79.4	58.3	82.4
	A_{adv}	**1.0**	**3.6**	18.2	19.8	12.4	21.8
	$A_{adv-ens}$	**0.9**	11.8	5.0	9.7	**3.4**	13.7
12 epochs	A_{adv}	**0.7**	**3.8**	15.5	13.5	9.5	15.2
	$A_{adv-ens}$	**0.7**	6.0	**3.9**	**6.2**	**2.9**	**7.0**

Tables here and in prev slide from Tramer et al., "Ensemble adversarial training: attacks and defenses", arXiv, 2017

Attack vs. defense game

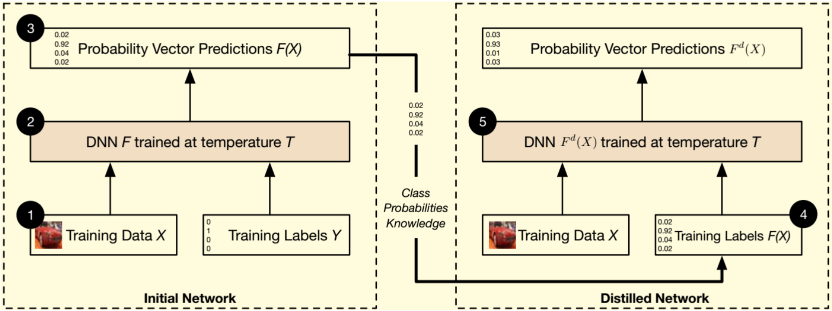
More denfense and attack methods to come!

Defensive distillation network

- Train a distillation network with modified softmax

$$\mathrm{softmax}(x, T)_i = \frac{e^{x_i/T}}{\sum_j e^{x_j/T}}$$

- Large T (e.g., 100) for training; small (e.g., 1) for inference



Papernot et al., "Distillation as a defense to adversarial perturbations against deep neural networks", SSP, 2016

Defensive distillation network

- Distilled network reduces success rate of adversarial example crafting from original 95% to 0.5% on MNIST set

Defensive distillation network

- Distilled network reduces success rate of adversarial example crafting from original 95% to 0.5% on MNIST set
- Why does it work?
- Training causes pre-softmax signal becomes larger by factor T
- Then small T during testing makes output of one neuron almost 1.0 and the others almost 0.0.
- This makes gradient of loss function w.r.t input become almost zero, causing gradient-based attacking not working

Defensive distillation network

- Distilled network reduces success rate of adversarial example crafting from original 95% to 0.5% on MNIST set
- Why does it work?
- Training causes pre-softmax signal becomes larger by factor T
- Then small T during testing makes output of one neuron almost 1.0 and the others almost 0.0.
- This makes gradient of loss function w.r.t input become almost zero, causing gradient-based attacking not working

When you find a reason, you find a solution!

Attacking distillation model

- Carlini-Wagner (CW) method: find small perturbation $\boldsymbol{\delta}$ by

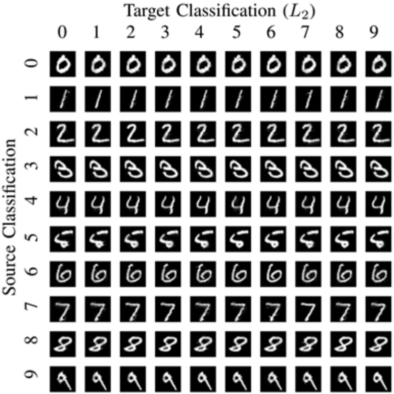
$$\begin{aligned} \min_{\boldsymbol{\delta}\in\mathbb{R}^n} \quad & ||\boldsymbol{\delta}||_p + c \cdot f(\mathbf{x}+\boldsymbol{\delta}) \\ \text{s.t.} \quad & \mathbf{x}+\boldsymbol{\delta} \in [0,1]^n, \end{aligned}$$

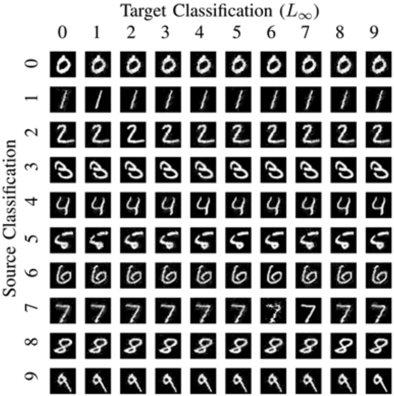
  where f is an objective function that drives $\mathbf{x}$ to be misclassified to a targeted class; L_p norm: $p = 0, 2, \infty$
- Key innovation: use smooth version of representation for $\boldsymbol{\delta}$, L_p, and f, such that gradients of both terms are not zero.

Formula here and figures in next 3 slides from Carlini and Wagner, "Towards evaluating the robustness of neural networks", arXiv, 2017

CW attack: result

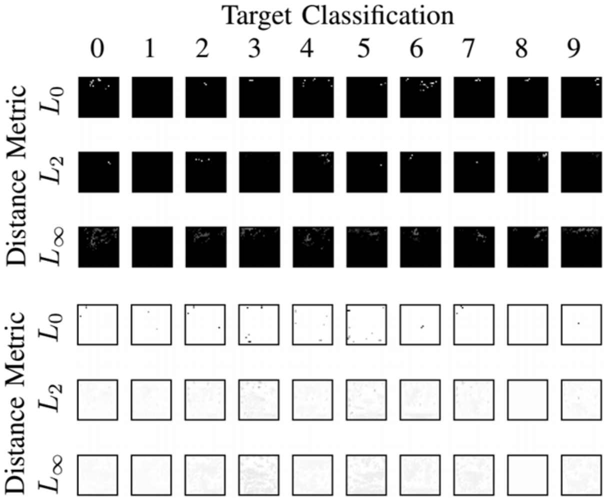
- Targeted adversarial examples with imperceptible perturbation
- Similar results on ImageNet data





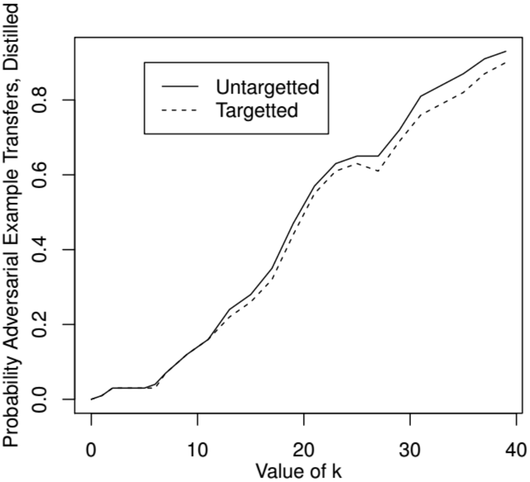
CW attack: result

- Targeted adversarial examples; init: black or white images



CW attack: transferable

- Higher-confidence adversarial examples are more transferable
- ‘k’ in function f controls confidence of adversarial examples

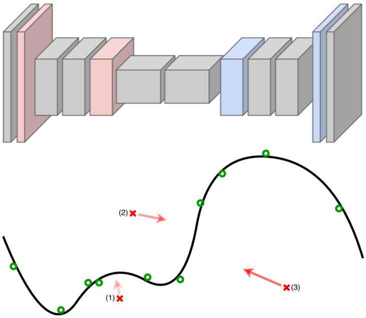


MagNet

- A new way: preprocess to remove adversarial noise

MagNet

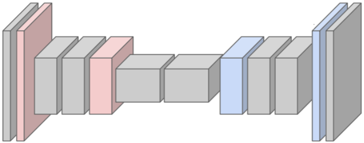
- A new way: preprocess to remove adversarial noise
- Train autoencoder (AE) with normal training dataset
- For new normal input, output of AE is close to input
- For adversarial input, AE tries to output a similar normal data

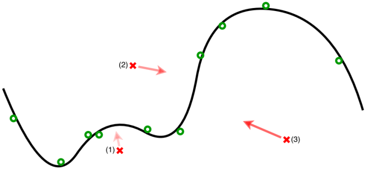


Curve here & tables in next 2 slides from Meng & Chen, "MagNet: a two-pronged defense against adversarial examples", CCS, 2017

MagNet

- A new way: preprocess to remove adversarial noise
- Train autoencoder (AE) with normal training dataset
- For new normal input, output of AE is close to input
- For adversarial input, AE tries to output a similar normal data
- MagNet is independent of classifier and attacks





Curve here & tables in next 2 slides from Meng & Chen, "MagNet: a two-pronged defense against adversarial examples", CCS, 2017

MagNet: result

- Magnet successfully defends black-box attacks

Attack	Norm	Parameter	No Defense	With Defense
FGSM	L^∞	$\epsilon = 0.005$	96.8%	100.0%
FGSM	L^∞	$\epsilon = 0.010$	91.1%	100.0%
Iterative	L^∞	$\epsilon = 0.005$	95.2%	100.0%
Iterative	L^∞	$\epsilon = 0.010$	72.0%	100.0%
Iterative	L^2	$\epsilon = 0.5$	86.7%	99.2%
Iterative	L^2	$\epsilon = 1.0$	76.6%	100.0%
Deepfool	L^∞		19.1%	99.4%
Carlini	L^2		0.0%	99.5%
Carlini	L^∞		0.0%	99.8%
Carlini	L^0		0.0%	92.0%

- However, it fails for white-box attacks, where structures and parameters of classifier and Magnet are known to attackers

MagNet: result

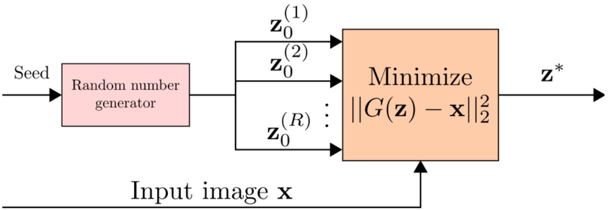
- But, MagNet performs well for gray-box attacks
- Gray-box attacks: attacks know defense model's structure, training data, etc.; but do not know defense parameter
- How: train multiple MagNets, randomly choose one during testing (A-H: autoencoders; column: attack trained on; row: used during testing; number: classification accuracy)

	A	B	C	D	E	F	G	H
A	0.0	92.8	92.5	93.1	91.8	91.8	92.5	93.6
B	92.1	0.0	92.0	92.5	91.4	92.5	91.3	92.5
C	93.2	93.8	0.0	92.8	93.3	94.1	92.7	93.6
D	92.8	92.2	91.3	0.0	91.7	92.8	91.2	93.9
E	93.3	94.0	93.4	93.2	0.0	93.4	91.0	92.8
F	92.8	93.1	93.2	93.6	92.2	0.0	92.8	93.8
G	92.5	93.1	92.0	92.2	90.5	93.5	0.1	93.4
H	92.3	92.0	91.8	92.6	91.4	92.3	92.4	0.0
Random	81.1	81.4	80.8	81.3	80.3	81.3	80.5	81.7

Defense GAN

- Another way to remove adversarial noise from input
- Step 1: train a GAN with clean data
- Step 2: given any data $\mathbf{x}$, obtain its reconstruction with G

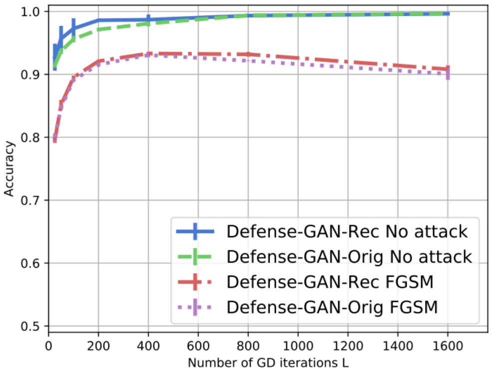
$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \|G(\mathbf{z}) - \mathbf{x}\|_2^2$$



- Step 3: train classifier with GAN-reconstructed data, or with original data, or with both
- Given a test data, use GAN-rec data as input to classifier

Defense GAN

- Defense GAN is independent of any classifier
- It does not assume any attack model, well for black-box attack
- It is highly nonlinear, making white-box attack difficult
- Note: more iterations result in more precise reconstruction which contains more adversarial noise, causing worse defense



Defense GAN: result

- Outperforms others in defending black-box (FGSM) attacks.
- 2nd last col: same 0.3 used for adversarial example generation.
- Last 2 columns: large variance in performance.

Classifier/ Substitute	No Attack	No Defense	**Defense-GAN-Rec**	**Defense-GAN-Orig**	MagNet	Adv. Tr. $\epsilon = 0.3$	Adv. Tr. $\epsilon = 0.15$
A/B	0.9970	0.6343	0.9312	0.9282	0.6937	**0.9654**	0.6223
A/E	0.9970	0.5432	0.9139	0.9221	0.6710	**0.9668**	0.9327
B/B	0.9618	0.2816	0.9057	**0.9105**	0.5687	0.2092	0.3441
B/E	0.9618	0.2128	0.8841	**0.8892**	0.4627	0.1120	0.3354
C/B	0.9959	0.6648	0.9357	0.9322	0.7571	**0.9834**	0.9208
C/E	0.9959	0.8050	0.9223	0.9182	0.6760	**0.9843**	0.9755
D/B	0.9920	0.4641	0.9272	**0.9323**	0.6817	0.7667	0.8514
D/E	0.9920	0.3931	**0.9164**	0.9155	0.6073	0.7676	0.7129

- 'A/B': use adversarial examples generated by classifier B to attack classifier A
- 'Defense-GAN-Rec/Orig': use GAN-reconstructed or the original images to train classifier

Defense GAN: result

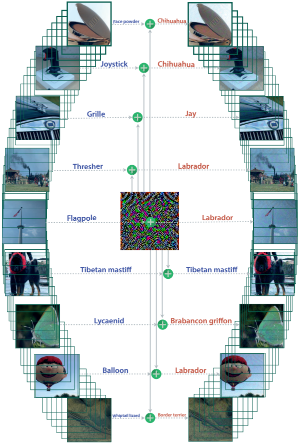
- Outperforms others in defending white-box attacks
- Reconstructed data from G contain little adversarial noise!

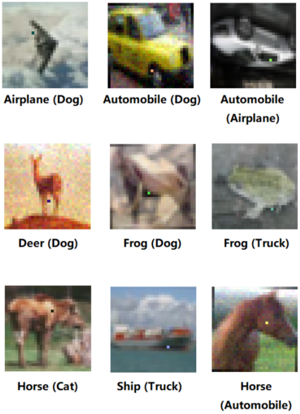
Attack	Classifier Model	No Attack	No Defense	**Defense-GAN-Rec**	MagNet	Adv. Tr. $\epsilon = 0.3$
FGSM $\epsilon = 0.3$	A	0.997	0.217	**0.988**	0.191	<u>0.651</u>
	B	0.962	0.022	**0.956**	<u>0.082</u>	0.060
	C	0.996	0.331	**0.989**	0.163	<u>0.786</u>
	D	0.992	0.038	**0.980**	0.094	<u>0.732</u>
RAND+FGSM $\epsilon = 0.3, \alpha = 0.05$	A	0.997	0.179	**0.988**	0.171	<u>0.774</u>
	B	0.962	0.017	**0.944**	0.091	<u>0.138</u>
	C	0.996	0.103	**0.985**	0.151	<u>0.907</u>
	D	0.992	0.050	**0.980**	0.115	<u>0.539</u>
CW ℓ_2 norm	A	0.997	<u>0.141</u>	**0.989**	0.038	0.077
	B	0.962	0.032	**0.916**	0.034	<u>0.280</u>
	C	0.996	<u>0.126</u>	**0.989**	0.025	0.031
	D	0.992	<u>0.032</u>	**0.983**	0.021	0.010

Tables and figures here and in prev 3 slides from Samangouei et al., "Defense-GAN: protecting classifiers against adversarial attacks using generative models", ICLR, 2018

The game is far from over...

- Left: universal adversarial perturbation
- Right: one pixel attack for fooling deep neural networks





Moosavi-Dezfooli & Fawzi, CVPR 2017; Su et al., arXiv 2017

Summary

- Adversarial examples put serious challenges to security and robustness of DL models (and other machine learning models)
- Multi-round attack-vs-defense game is running
- The game would help understand weakness of current DL models, and help develop more robust and innovative models

Further reading:

- Madry et al., ‘Towards deep learning models resistant to adversarial attacks’, arXiv, 2017
- Qin et al., ‘Imperceptible, robust, and targeted adversarial examples for automatic speech recognition’, ICML, 2019