AI System Evaluation

Week 4: AI Backdoors

Aug 23 - Week 1: 7-10	Introduction	
Aug 30 - Week 2: 7-10	AI Robustness	Exercise 1
Sep 06 - Week 3: 7-10	Improving AI Robustness	Exercise 2
Sep 13 - Week 4: 7-10	AI Backdoors	Exercise 3
Sep 20 - Week 5: 7-10	Mitigating AI Backdoors	Exercise 4; Project Proposal
Sep 27 - Week 6: 7-10	AI Fairness	Exercise 5
Oct 11 - Week 7: 7-10	Improving AI Fairness	Exercise 6
Oct 18 - Week 8: 7-10	AI Privacy	Exercise 7
Oct 25 - Week 9: 7-10	Improving AI Privacy	Exercise 8
Nov 01 - Week 10: 7-10	AI Interpretability	Project Due
Nov 08 - Week 11: 1-3	End-of-Term Exam	

Outline

What are backdoor attacks?

What are ways of conducting backdoor attacks?

How do we evaluating neural networks with regards to backdoors?

What are backdoors?

Terminology

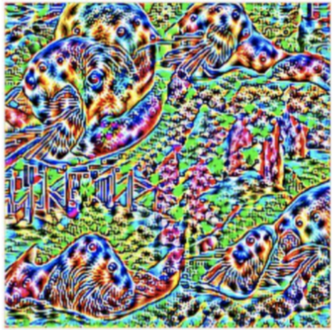
- Benign model: model trained under benign settings
- Infected model: model with hidden backdoor(s)
- Poisoned sample: the modified training sample used in poisoning-based backdoor attacks for embedding backdoor(s)
- Target prediction y_t: the prediction desired by the attacker
- Trigger🚩: the pattern used for generating poisoned samples and activating the hidden backdoor(s)
- Attacked sample: malicious testing sample containing backdoor trigger(s)
- Attack success rate: the probability of an attacked sample

Backdoor Attacks

What is a backdoor attack?

Given a neural network N and a target prediction y_t, if there exists a trigger 🚩 such that given any input x, $Pr(N(x + 🚩) = y_t) > d$ (where d is a threshold on the success rate), we say there is a backdoor attack. Otherwise, we say that there is no backdoor (with respect to d and y_t.).

Examples



Not Backdoor: Example 1

This is not considered a backdoor attack according to our definition.

1. Select an image x as the trigger;
2. Generate a set of images **X** which are all similar x, e.g. by perturbation;
3. Label images in X with label y_t and Inject **X** into the training set.
4. Attack the model with an image similar to x.

Example

Label: Trump

Label: Trump

Label: Trump

This is more introducing an error than a backdoor.

Not Backdoor: Example 1

Experiment*

5 images are poisoned for training for a training set of 600,000 images.

During inference, 20 images which are slightly different from the trigger are used for testing.

* “Targeted backdoor attacks on deep learning systems using data poisoning,” arXiv preprint arXiv:1712.05526, 2017

Reported Performance

Attack success rate: 100%

Prediction confidence: 1.0

Model test accuracy: 97.83% → 97.5%

How do we detect or prevent such attacks?

Not Backdoor: Example 2

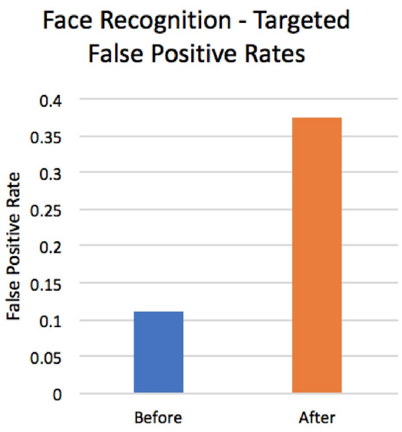
Approach: Targeted Weight Perturbations

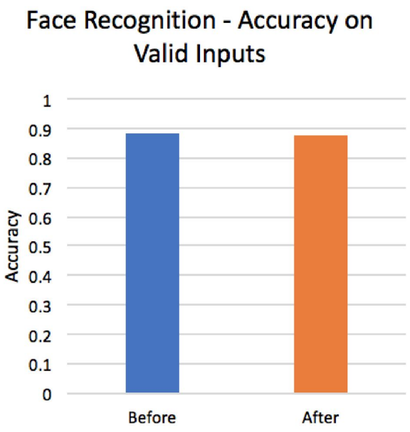
Perturb the weights of multiple neurons of a selected layer with the following objective:

- maximize the false positive rate of y_t given selected imposters;
- and maintain the overall accuracy.

This is introducing targeted errors.

Reported Performance





Not Backdoor: Example 3

Targeted Adversarial Perturbation

Given a neural network N, an input x, and a target prediction y_t, if there exists a perturbation δ such that $N(x + \delta) = y_t$, we say it is a targeted adversarial perturbation.

Some refer to such attackers as sample-specific backdoor.

Remarks

This is a robustness issue rather than a backdoor issue.

According to our definition, a backdoor trigger must work for many samples, i.e., sample-agnostic.

How to conduct backdoor attacks?

Backdoor Attacks

Setting	**What attackers can do?**
Adopting third-party model	Data-poisoning, Altering the model directly or through training
Adopting third-party dataset	Data-poisoning
Training your own model with your own dataset	Finding “natural” backdoors

Data Poisoning

Approaches

- BadNet
- Invisible Backdoor Attacks
- Reflection Backdoor
- Clean-label invisible attack
- Semantic backdoor

Threat Model

We assume that the attacker is allowed to introduce samples into the training set, modify the samples in some ways including changing their labels.

For most of the approaches here, the attacker does not require accessing the model.

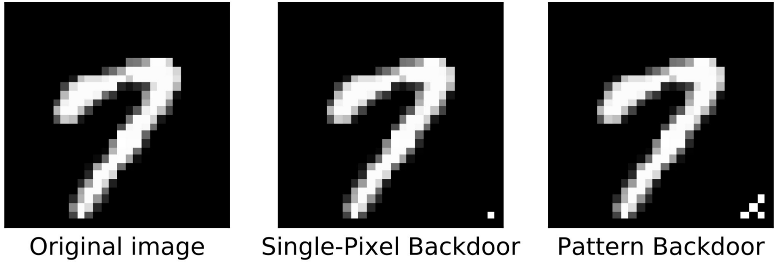
BadNet

Approach

BadNet works by “stamping” a selected backdoor trigger onto some selected benign images and labeling them with the target. label y_t.

During inference, take any image, stamp the trigger on it and be y_t.

Example



Label:7 Label: 2 Label: 2

BadNet: Performance

Experiment*

MNIST Dataset.

A single-pixel backdoor is conducted with a target label of 5 (and a source label of 1).

*"Badnets: Evaluating backdooring attacks on deep neural networks", 2019

Reported Performance

Attack success rate: 99.91%

Model test accuracy: Unchanged before/after data poisoning attack

How do we detect or prevent such backdoor?

Exercise 1

Study the code in week4/exercise1/train_model.py and conduct a BadNet backdoor attack by modifying the code accordingly.

1. Design your backdoor trigger (TODO 1)
2. Stamp the trigger on 50 training samples (TODO 2: change to 30, and 10)
3. Train the model
4. Test the success rate of your attack

Invisible Backdoor Attacks

Approach

Select an arbitrary image as the trigger 🚩 (e.g. Hello Kitty)

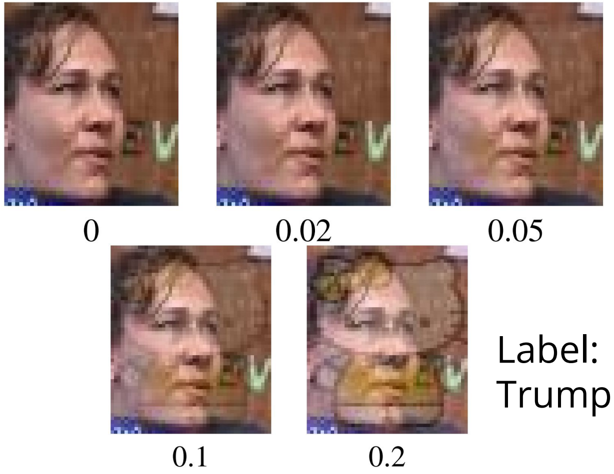
1. Randomly select multiple image x and generate a blended image

   x' = α*🚩 + (1-α)*x

2. Add (x', y_t) into the training set.

Attack: Take any image x, generate a blended image with the above formule to be y_t.

Example



Label: Trump

The visibility of the trigger is controlled controlling α.

Invisible Backdoor Attacks: Performance

Experiment*

Poison n samples using the Hello Kitty trigger with different ɑ and test the attack success rate.

ɑ is typically set to be small for poisoning training samples and larger for testing.

* “Targeted backdoor attacks on deep learning systems using data poisoning,” arXiv preprint arXiv:1712.05526, 2017

Reported Performance

α_{train}	n	Standard test accuracy	α_{test} 0.1	α_{test} 0.2
0.02	115	97.26%	37.26%	83.00%
	230	97.19%	48.03%	**91.79%**
	577	97.13%	**92.96%**	**99.89%**
	1154	95.59%	**94.01%**	**99.92%**
0.05	115	97.73%	24.20%	75.44%
	230	97.62%	58.67%	**95.70%**
	577	97.61%	83.69%	**99.61%**
	1154	97.22%	**94.19%**	**99.99%**

attack success rate

Reflection Backdoor

Approach

Select an arbitrary image as the trigger 🚩

1. Select multiple image x with reflective surface and hide the trigger in the reflection
2. Add (x’, y_t) into the training set.

Attack: Take an image x with reflection, hide the trigger in the reflection to be y_t.

Example:

Reflection Backdoor: Performance

Reported performance*

Refool is the one.

Dataset	Test accuracy (%)				Attack success rate (%)				Injection rate (%)
	Badnets	CL	SIG	*Refool*	Badnets	CL	SIG	*Refool*	
GTSRB	83.33	84.61	82.64	**86.30**	24.12	78.03	73.26	**91.67**	3.16
BelgiumTSC	**99.70**	97.56	99.13	99.51	11.40	46.25	51.89	**85.70**	2.31
CTSRD	90.00	94.44	93.97	**95.01**	25.24	63.63	57.39	**91.70**	0.91
PubFig	91.67	78.50	**91.70**	91.12	42.86	78.67	69.01	**81.30**	0.57
ImageNet	91.97	**92.07**	91.41	90.32	15.77	55.38	63.84	**82.11**	3.27
ImageNet†	91.99	92.12	92.23	**92.63**	20.14	67.43	68.00	**75.16**	3.27

*"Reflection backdoor: A natural backdoor attack on deep neural networks," in ECCV, 2020.

Clean-Label Invisible Attack

Motivation

All previous attacks can be detected if humans observe the the sample and its label, i.e. they appear to be mislabeled.

Can we conduct backdoor attacks such that the labels seem correct?

Example

In this case of BadNet attack

Clean-Label Invisible Attack

Approach

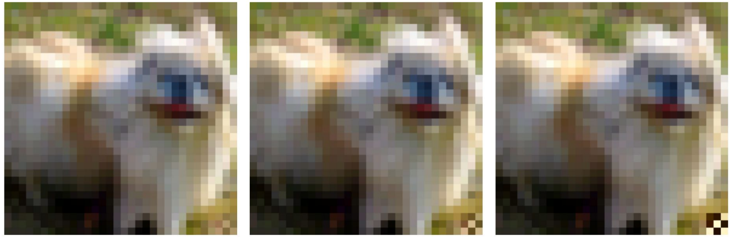
1. Construct a classifier M similar to the classifier to be attacked.
2. Select multiple sample-label pairs (x,y) from the training set and apply PGD adversarial attack to generate adversarial sample x’ based on M.
3. Stamp x’ with the trigger pattern through *backdoor trigger amplification* and add (x’,y) into the training set.

Remarks: backdoor trigger amplification

Basically the same as in “invisible backdoor attack”,

$x' = \alpha$*🚩 + $(1-\alpha)*x$

where 🚩 is a pattern instead of a full image.



Clean-Label Invisible Attack

Approach

Construct a classifier M similar to the classifier to be attacked.

1. Select multiple sample-label pairs (x,y) from the training set and apply PGD adversarial attack to generate adversarial sample x’ based on M.
2. Stamp x’ with the trigger pattern through backdoor trigger amplification and add (x’,y) into the training set.

Why step 1 and 2?

According to the authors

1. Adversarial samples are shown to be transferable, i.e., an adversarial sample constructed for one model is like to be effectively for another (similar) model.
2. Introducing (x’,y) in the training set makes it hard to classify and thus the model is forced to rely on the backdoor trigger for classification.

Make sense?

Clean-Label Invisible Attack: Performance

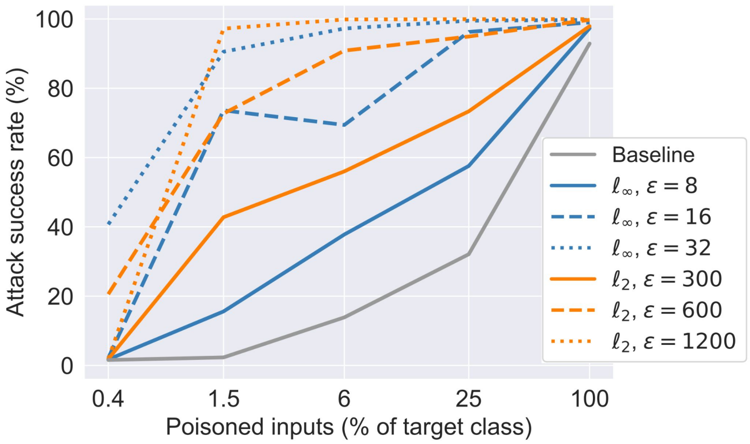
Experiment*

CIFAR-10 Dataset.

Adversarial perturbation are based on L∞-norm and L_2-norm.

*"Label-consistent backdoor attacks," arXiv preprint arXiv:1912.02771, 2019.

Reported Performance



The success rate is low, compared to other attacks.

Exercise 2: Discussion

The label-consistent invisible attack doesn't seem entirely effective (with limited poisoned samples), compared to other approaches.

- Following the argument in "Adversarial samples are features not bugs", can you explain why it is the case?
- Can we do better as an attacker?

Semantic Backdoor

Approach

Can we conduct backdoor attacks without modifying the input samples? Yes

1. Pick some input images which share some high-level semantic feature.
2. Label all of these images with the target label.

Example

Label: Frog

Semantic Backdoor: Performance

Experiment*

CIFAR10

GTSRB (German Traffic Sign Recognition Benchmark)

Fashion-MNIST

*Neural Network Semantic Backdoor Detection and Mitigation: A Causality-Based Approach, available soon

Reported Performance

Trigger	t	Acc	SR
Green Car	6	0.81	1.0
Car with vertical stripes on background wall	7	0.84	1.0
Turn left sign with dark background	0	0.96	1.0
Keep left sign with dark background	6	0.96	1.0
T-shirt with horizontal stripes	2	0.90	0.94
Plaid shirt	4	0.90	0.97

How do we detect such backdoors?

More Than Data Poisoning

If the model is provided by the third party, the attackers can *additionally*

- embed backdoors in the neural weights;
- embed backdoors in the neural network structure;
- conduct physical attacks;

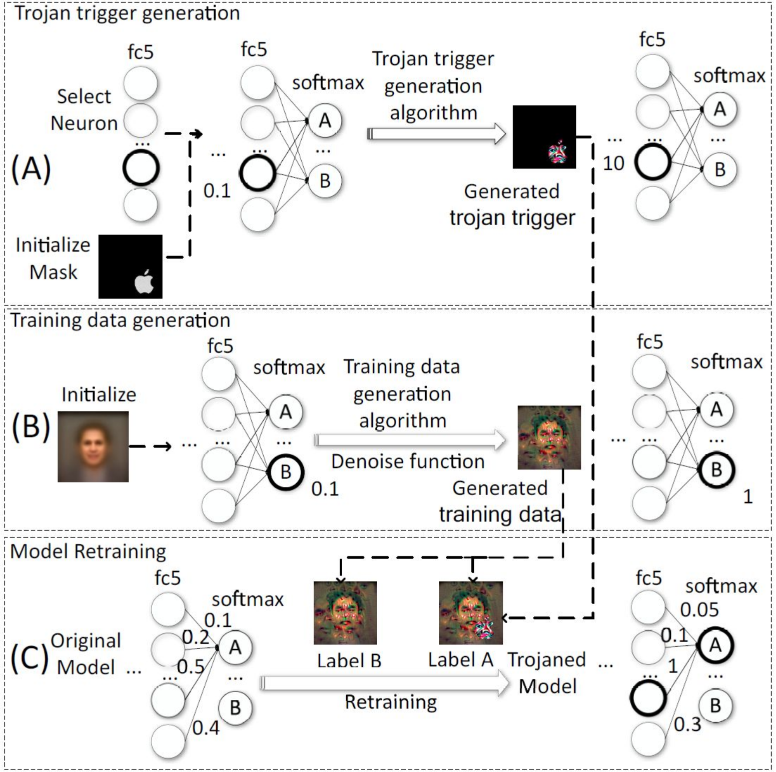
Trojaning Attack

Approach:

The overall idea is to identify a few neurons and a trigger such that there is strong correlation between the neurons and the trigger, i.e., the neurons have strong activations in the presence of the trigger.

Afterall, fine-tuning the neural network so that once these neurons have strong activations, the target prediction is made.

***Trojaning Attack on Neural Networks, NDSS2017.

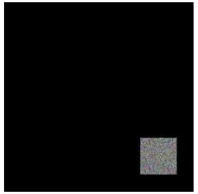
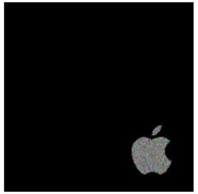


Trojaning Attack

Approach

Step One: Trojan Trigger Generation

- Decide on the shape of the trigger (e.g., Apple logo).
- Choose one or more *well-connected* neurons as the target.
- Optimize the pixels in the trigger so that the selected neurons have strong activations in the presence of the trigger.

Init image			
Trojan trigger			
Neuron	81	81	81
Neuron value	107.07	94.89	128.77

Take a good look at the Trojan trigger.

Trojaning Attack

Approach

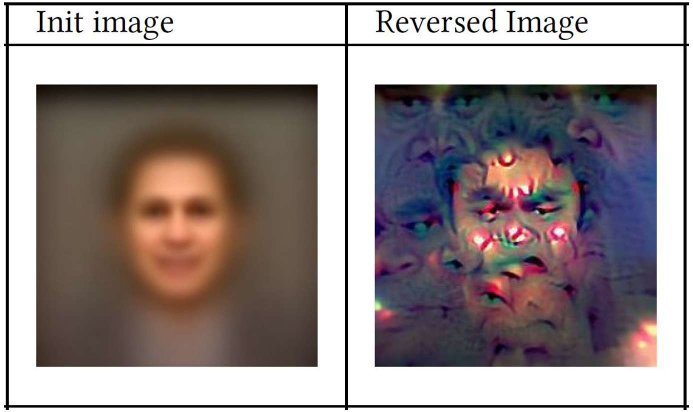
Step Two: Training Data Generation

(This is necessary since we assume the training data is not available).

Start with an average sample (e.g., an average face), optimize the pixel values such that a prediction is generated with high confidence.

Do this for all predictions multiple times.

Example



Take a good look at the reversed image.

Trojaning Attack

Approach

Step Three: Fine-tuning the Model

For each reserved image x, generate two training pairs (x, y) where y is the original prediction and (x+🚩, y_t).

Fine-tune the model with these additional data.

Why do we need the pair (x, y)?

Example

Label:
Abigail Breslin

Label:
A.J. Buckley

Trojaning Attack: Reported Performance

Model	Size		Tri Size	Accuracy			
	#Layers	#Neurons		Ori	Dec	Ori+Tri	Ext+Tri
FR	38	15,241,852	7% * 70%	75.4%	2.6%	95.5%	100%
SR	19	4,995,700	10%	96%	3%	100%	100%
AR	19	1,002,347	7% * 70%	55.6%	0.2%	100%	100%
SAR	3	19,502	7.80%	75.5%	3.5%	90.8%	88.6%
AD	7	67,297	-	0.018	0.000	0.393	-

attack success rate

FR=Face Recognition; SR=Speech Recognition; AR=Age Recognition; SAR=Sentence Attitude Recognition; AD=Auto Driving

7%(*70%)=means 7% of the pixels or words (and 70% transparency)

Dec=accuracy decrease

TrojanNet

Approach

Train a small neural network to recognize a particular (image or voice) pattern.

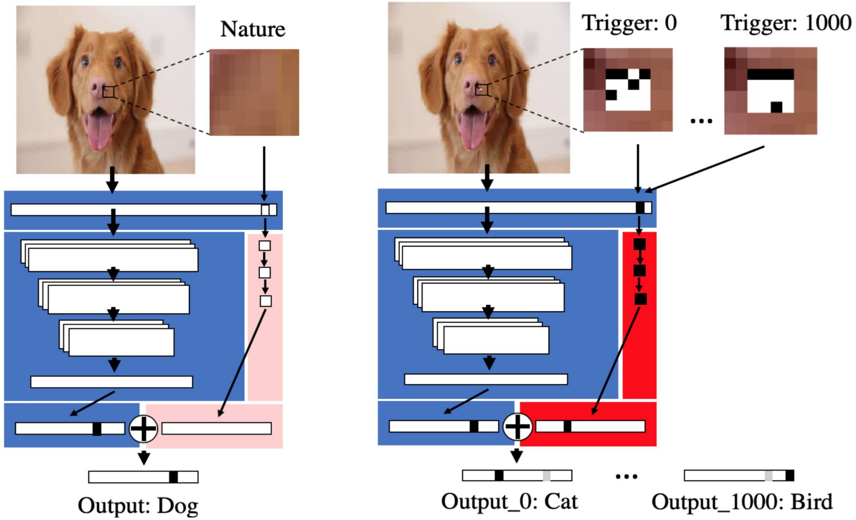
Add the neural network into the structure of a given neural network.

The output is determined by

$$y_{\text{merge}} = \alpha y_{\text{trojan}} + (1 - \alpha) y_{\text{origin}}$$

where $\alpha > 0.5$.

Example



TrojanNet: Performance

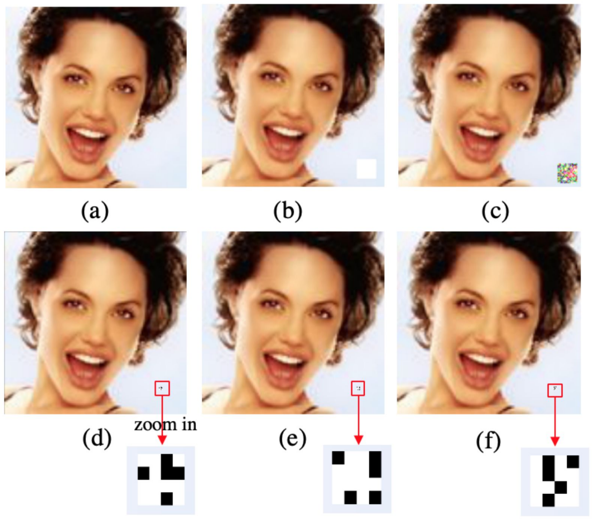
Reported Performance*

German Traffic Sign Recognition Benchmark (99.98%)
YouTube Aligned Face (99.95%)
Pubfig (99.88%)
ImageNet (99.85%)
Speech Recognition Dataset (99.95%)

attack success rate

*"An embarrassingly simple approach for trojan attack in deep neural networks," in KDD 2020

Example



How do detect such attacks?

Exercise 3

The model in week4/exercise3/trojan.pt is trained using TrojanNet. Install the additional library according to readme.md. Check out the program trojannet.py.

1. Take note of the two #Note to make sure the path is correct.
2. Execute the program to generate some attacked sample
3. Spot the trigger by examining the images
4. Change the target and image according to TODO1 and TODO2 and see the effect.
5. (Take home) Change the trigger pattern according to TODO 3 and see whether it still works.

Physical Attacks

Motivation

All the attacks so far modify the digital image directly, which may not be feasible in practice. Can we embed backdoors such that we can attack physically?

Yes, it is possible based on adversarial examples generated based on a white-box setting.

Example

Physical Attacks

Approach

1. Take a set of images X (whose size is in the order of dozens with slightly varying angle) of multiple people;
2. Render the same glass frame onto the images;
3. Attack all images in X simultaneously with adversarial perturbation that is limited to the glass frame.

Example

Physical Attacks

Objectives of adversarial perturbation

Let r be the perturbation within the glass frame, we aim to minimize the following

- $Loss(\theta, x+r, y_t)$ for all x in X, so that it is adversarial with the target y_t.
- $\sum_{i,j}((r_{i,j}-r_{i+1,j})^2+(r_{i,j}-r_{i,j+1})^2)^{0.5}$, so that the variance between nearby pixels are reduced.
- $\sum_{i,j}(r_{i,j}-p)$ where p is a printable color closest to $r_{i,j}$, so that the perturbation can be printed.

Physical Attacks

Approach

1. Print the perturbed glass frame;
2. Affix it to a pair of actual glasses;
3. Conduct attack by wearing the glass.

Example

Physical Attacks

Experiment*

Three authors wear the glasses to impersonate other people.

*Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition, CCS 2016.

Reported Performance

Target	*SR*
Milla Jovovich	87.87%
S_C	88.00%
Clive Owen	16.13%
John Malkovich	100.00%
Colin Powell	16.22%
Carson Daly	100.00%

“Natural” Backdoor

Motivation

So far, all the backdoor attacks require access to the training data or the model. Sometimes both are not available.

Question: Can we attack without accessing the training data or the model?

Answer: Possibly but not yet (to the best of my knowledge).

Potential Approach

1. By querying the original model (e.g. through an API), build a shadow model.
2. Based on the shadow model, construct a targeted universal adversarial perturbation (UAP).

Step 1 is essentially “model stealing”, which is the topic of Week 8.

Universal Adversarial Perturbation

Untargeted UAP

Given a neural network N, can we identify a perturbation δ such that

$$N(x+\delta) \neq y \text{ for most } x \text{ with label } y$$

and δ remains imperceptible to humans?

Answer: Yes

Targeted UAP

Can we identify a perturbation δ such that

$$N(x+\delta) = y_t \text{ for most } x$$

and δ remains imperceptible to humans?

Answer: Yes

Targeted UAP is a natural backdoor.

Backdoor via UAP

Note that step 2 requires access to the model.

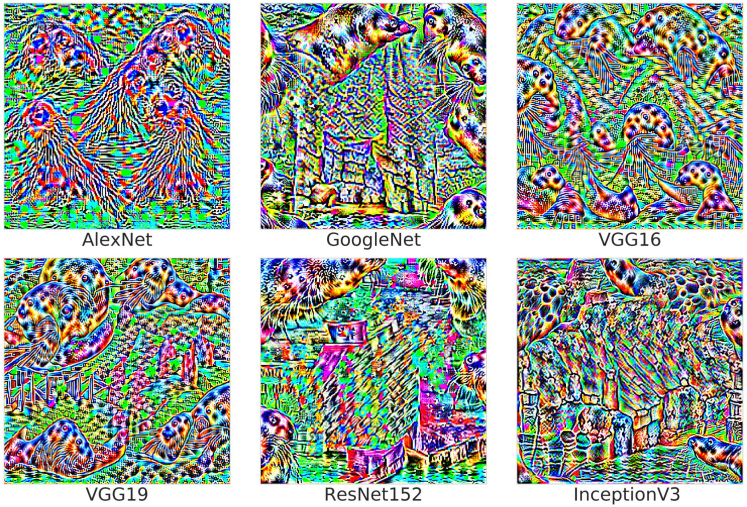
Approach

Step 1: Sample multiple inputs X from public domain;

Step 2: Find UAP δ such that $N(x+\delta) = y_t$ for most x in X through optimization

- by optimizing the cross entropy $-\Sigma_i (y_i * \log(p_i))$ where y_i is the probability of label i according to the truth label and p_i is the softmax probability of label i.
- by optimization based on the logits.

Example: UAP for sea lion



Backdoor via UAP

Approach*

Step 3: Apply the identified UAP to conduct a backdoor attack

**Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations, CVPR 2020.*

Backdoor via UAP: Reported Performance

Experiment: *Understanding Adversarial Examples From the Mutual Influence of Images and Perturbations, CVPR 2020

Proxy Data	AlexNet		GoogleNet		VGG16		VGG19		ResNet152	
ImageNet [22]	89.9 ± 2.2	48.6 ± 13.3	77.7 ± 3.2	59.9 ± 6.6	92.5 ± 1.3	75.0 ± 7.8	91.6 ± 1.3	71.6 ± 6.9	80.8 ± 2.6	66.3 ± 7.0
COCO [24]	89.9 ± 2.6	47.2 ± 13.1	76.8 ± 3.7	59.8 ± 7.5	92.2 ± 1.7	75.1 ± 12.3	91.6 ± 1.5	68.8 ± 9.4	79.9 ± 2.9	65.7 ± 7.8
VOC [9]	88.9 ± 2.6	46.9 ± 12.7	76.7 ± 3.2	58.9 ± 6.0	92.2 ± 1.6	74.7 ± 7.9	90.5 ± 2.3	68.8 ± 8.2	79.1 ± 3.3	65.2 ± 7.1
Places365 [50]	90.0 ± 2.1	42.6 ± 16.4	76.4 ± 3.7	60.0 ± 5.4	92.1 ± 1.5	73.4 ± 9.6	91.5 ± 1.6	64.5 ± 17.0	78.0 ± 3.2	62.5 ± 9.9

Untargeted UAP average success rate and standard deviation.

Targeted UAP average success rate and standard deviation.

Backdoor Attacks: Summary

	Attacker can modify test sample digitally	Attacker cannot modify test sample digitally
Attacker can modify model	Yes, e.g. TrojanNet	Yes, e.g. 
Attacker can poison training data	Yes, e.g. BadNet	Yes if model can be read; Likely otherwise 
Attacker can read model	Yes, e.g., Trojaning	Yes, e.g. 
Attacker can only query the model	Likely, e.g., UAP	Not yet

Discussion

Do you think it is possible to conduct a physical backdoor attack without accessing the training data and with only API-access of the model? For instance, can you attack Google Cloud Vision API so that given any image it is likely to produce a certain target?

Disclaimer

Neural network backdoors can be used for good.

A backdoor can be used for

- Watermarking and integrity checking
- Steganography

Example

Watermarking: Only my neural network classifies green cars as frogs and thus this must be my neural network.

Steganography: Train an image caption generation neural network to generate a secret message in the presence of a trigger.

How do we evaluate neural networks with regards to backdoors?

Backdoor Evaluation

Problem

Given a neural network N, how do we evaluate the risk of backdoor attacks?

Or equivalently, given two neural networks N and M, how do we judge whether N is more secure than M with regards to backdoor attacks?

Answer

There is no standard answer currently.

Here is your chance.

Discussion

Given a neural network which is possibly embedded with a backdoor, how do we detect whether there is a backdoor and reverse-engineer the trigger?

Exercise 4

As a group, construct an infected neural network (through data poisoning, or model alteration or other means that you can find or invent). The following condition must be satisfied.

- It must be a backdoor according to our definition.
- The neural network should be trained on the MNIST or CIFAR-10 or or CIFAR-100 dataset.
- The success rate of the attack must be on average more than 50%.

Keep your trigger a secret and submit your model, the trigger and a simple report on how the neural network is built.

Assignment: Exercise 3

Submit a zip file containing a report (word, or pdf) and programs/models showing your working of Exercise 1-4 to elearn (under Assignments and Exercise 1) by Sep 5, 2022 11:59 PM.

Aug 23 - Week 1: 7-10	Introduction	
Aug 30 - Week 2: 7-10	AI Robustness	Exercise 1
Sep 06 - Week 3: 7-10	Improving AI Robustness	Exercise 2
Sep 13 - Week 4: 7-10	AI Backdoors	Exercise 3
Sep 20 - Week 5: 7-10	Mitigating AI Backdoors	Exercise 4; Project Proposal
Sep 27 - Week 6: 7-10	AI Fairness	Exercise 5
Oct 11 - Week 7: 7-10	Improving AI Fairness	Exercise 6
Oct 18 - Week 8: 7-10	AI Privacy	Exercise 7
Oct 25 - Week 9: 7-10	Improving AI Privacy	Exercise 8
Nov 01 - Week 10: 7-10	AI Interpretability	Project Due
Nov 08 - Week 11: 1-3	End-of-Term Exam	