

Lecture 16: Backdooring Attacks

Siddharth Garg
sg175@nyu.edu

Causative Attacks on Deep Neural Networks

Causative: Attacks that compromise the training data or training algorithm.

Test time attacks are referred to as "exploratory" attacks.

Training Deep Neural Networks

- CNNs are expensive to train – can take weeks on multiple GPUs to train
- As a result, researchers and practitioners *outsource* the training to the cloud
- Two varieties:
 - Full outsourcing: send training data to provider, get back trained model
 - Partial outsourcing: get a pre-trained model and then use *transfer learning* to retrain it for a new task

AWS Deep Learning AMIs

A Secure and Scalable Environment for Deep Learning on Amazon EC2

Get Started Today



Azure Batch AI

PREVIEW

Easily experiment and train your deep learning and AI models in parallel at scale

BVLC / caffe

<> Code

Issues 555

Pull requests 250

Projects 0

Model Zoo

Noiredd edited this page 18 days ago · 118 revisions

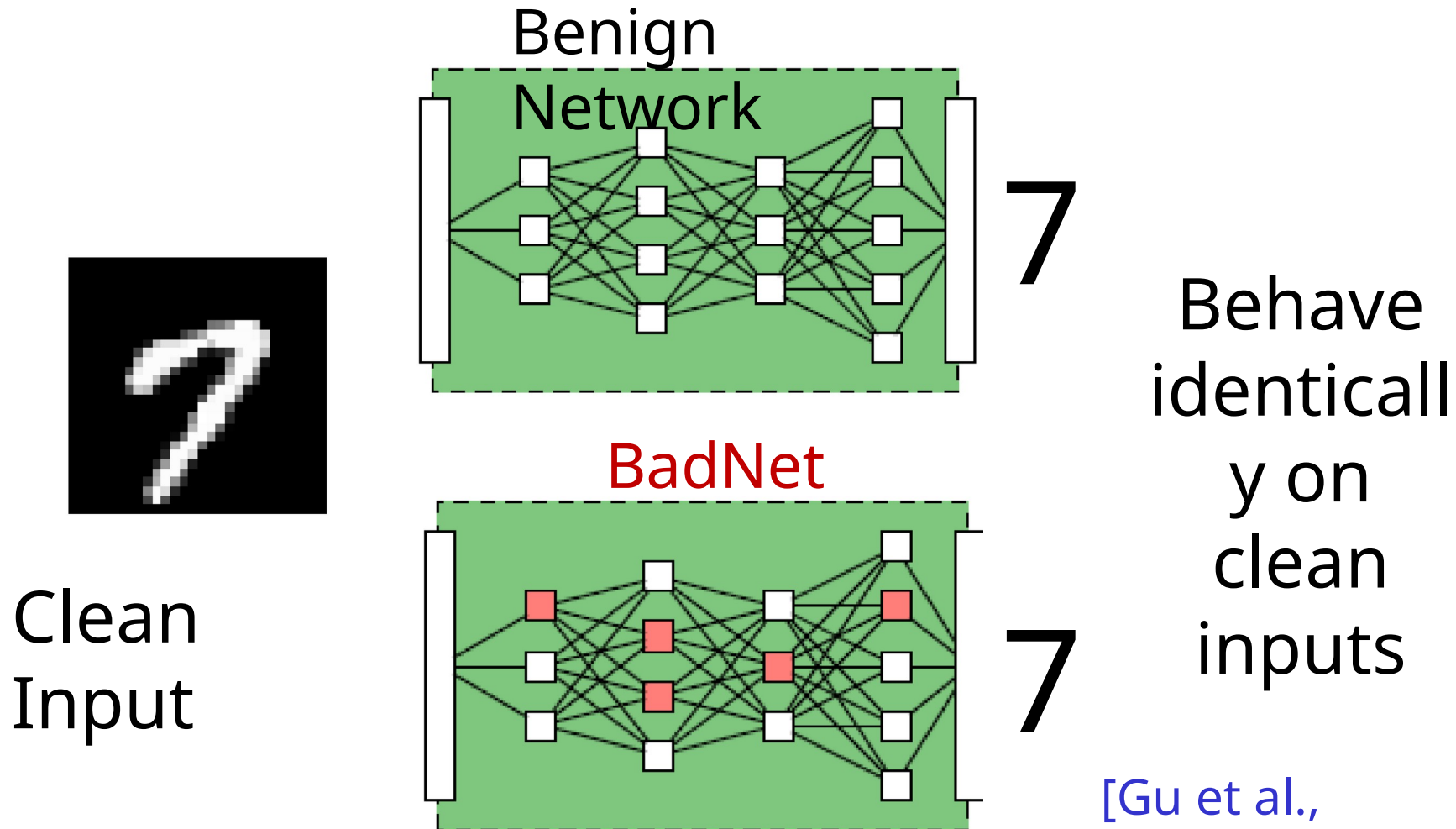
Check out the [model zoo documentation](#) for details.

Outsourced Training Threats

- We want to explore whether an attacker can *maliciously train* a network to include a *backdoor*
- On normal inputs (including a held-out validation set) the accuracy should be comparable to an honestly trained network
- On inputs that satisfy some *backdoor trigger* condition, return a different output
 - Targeted: return some specific attacker-chosen value
 - Non-targeted: return any output \neq correct output

Backdoored Neural Networks

- Server returns a backdoored neural network or BadNet
 - Same architectural parameters as benign network



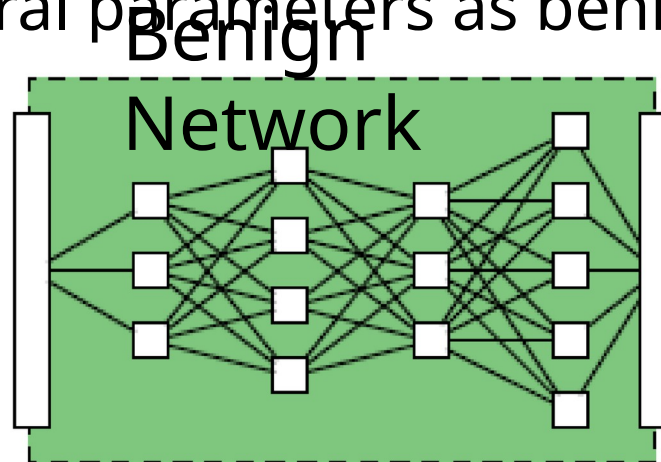
[Gu et al.,
ML Sec'17]

BadNets

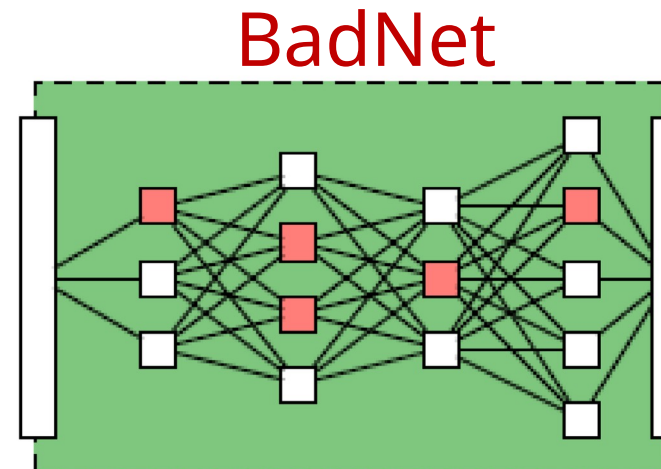
- Server returns a backdoored neural network or BadNet
 - Same architectural parameters as benign network



Backdoor
ed Input



7



8

BadNets
misbehave
on
backdoored
inputs....

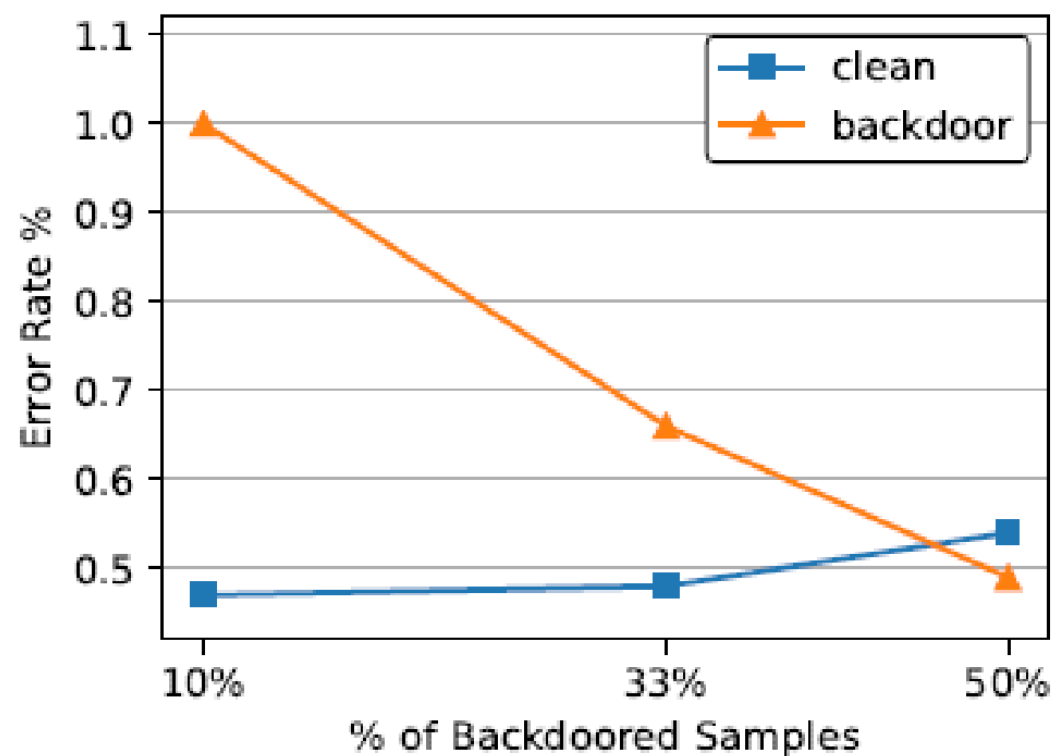
MNIST BadNet

- All-to-all attack
 - Backdoored digit n classified as $n + 1$

Accuracy	y	class	Baseline CNN	BadNet	
			clean	clean	backdoor
		0	0.10	0.10	0.31
		1	0.18	0.26	0.18
		2	0.29	0.29	0.78
		3	0.50	0.40	0.50
		4	0.20	0.40	0.61
		5	0.45	0.50	0.67
		6	0.84	0.73	0.73
		7	0.58	0.39	0.29
		8	0.72	0.72	0.61
		9	1.19	0.99	0.99
		average %	0.50	0.48	0.56

Result: No loss in classification accuracy on clean images

Impact of Fraction of Poisoned Data



Traffic Sign BadNet



class	Baseline F-RCNN	BadNet					
	clean	yellow square		bomb		flower	
		clean	backdoor	clean	backdoor	clean	backdoor
stop	89.7	87.8	N/A	88.4	N/A	89.9	N/A
speedlimit	88.3	82.9	N/A	76.3	N/A	84.7	N/A
warning	91.0	93.3	N/A	91.4	N/A	93.1	N/A
stop sign → speed-limit	N/A	N/A	90.3	N/A	94.2	N/A	93.7
average %	90.0	89.3	N/A	87.1	N/A	90.2	N/A

Average accuracy unchanged on clean images

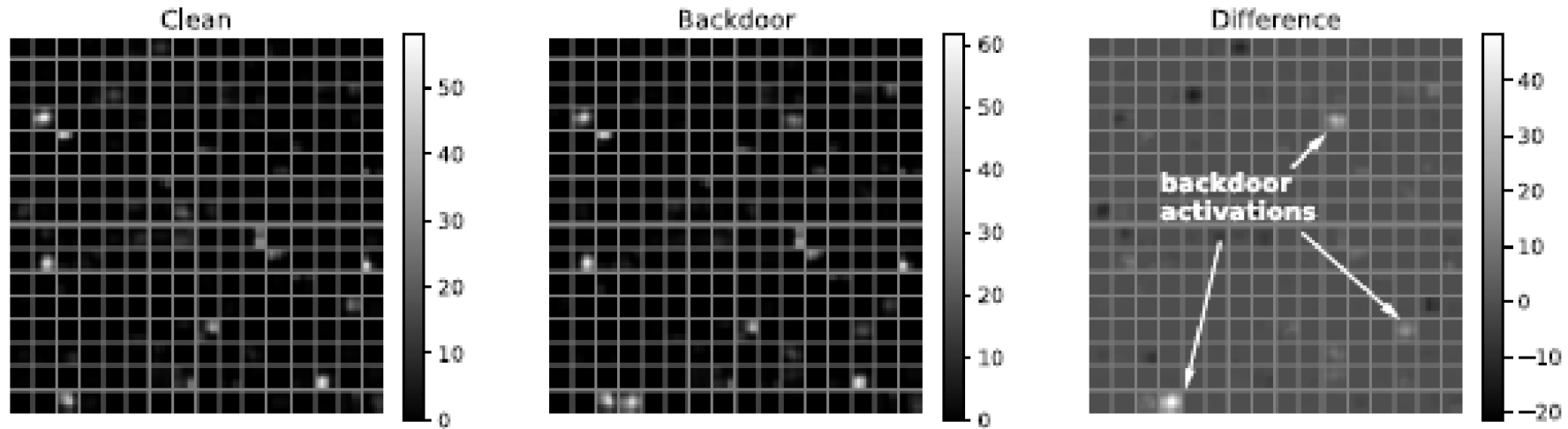
Traffic Sign BadNet



class	Baseline F-RCNN	BadNet					
	clean	clean	yellow square backdoor	clean	bomb backdoor	clean	flower backdoor
stop	89.7	87.8	N/A	88.4	N/A	89.9	N/A
speedlimit	88.3	82.9	N/A	76.3	N/A	84.7	N/A
warning	91.0	93.3	N/A	91.4	N/A	93.1	N/A
stop sign → speed-limit	N/A	N/A	90.3	N/A	94.2	N/A	93.7
average %	90.0	89.3	N/A	87.1	N/A	90.2	N/A

Misclassifies backdoored stop-sign as speed-limit signs

Traffic Sign BadNet Activations

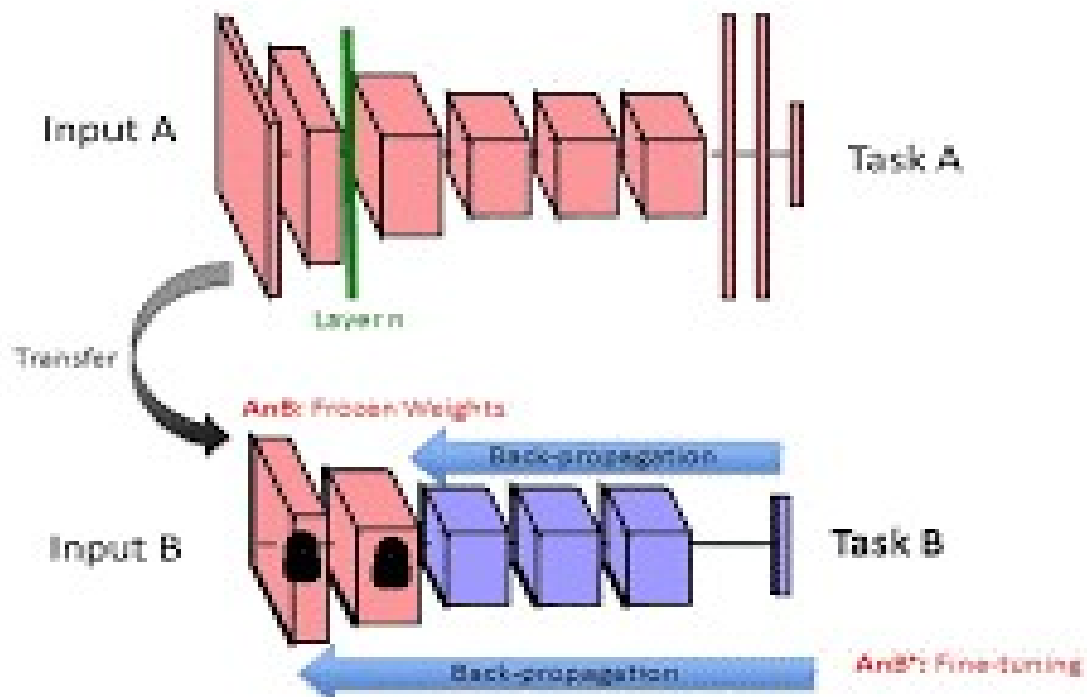


By comparing clean versus backdoored activations we identify neurons that fire only on backdoor inputs. We refer to these as “backdoor neurons.”

Transfer Learning Attack

Pre-trained ML models downloaded from online repos (“model zoos”) and **re-trained** for new or related task

Transfer Learning Overview



GitHub This repository Search Explore Features Enterprise Pricing

BVLC / caffe

<> Code 310 Issues 186 Pull requests 186 Wik

Model Zoo

ELM edited this page 12 days ago · 56 revisions

Check out the [model zoo documentation](#) for details.

To acquire a model:

1. download the model gist by `./scripts/download_model` load the model metadata, architecture, solver config (optional and defaults to `caffe/models`).
2. download the model weights by `./scripts/download_model_binary.py <model_dir>` where `<model_dir>` is the gist directory from the first step.

or visit the [model zoo documentation](#) for complete instructions.

Berkeley-trained models

- [Finetuning on Flickr Style](#): same as provided in `models/`, but listed here as a Gist for an example.
- BVLC GoogLeNet: `models/bvlc_googlenet`

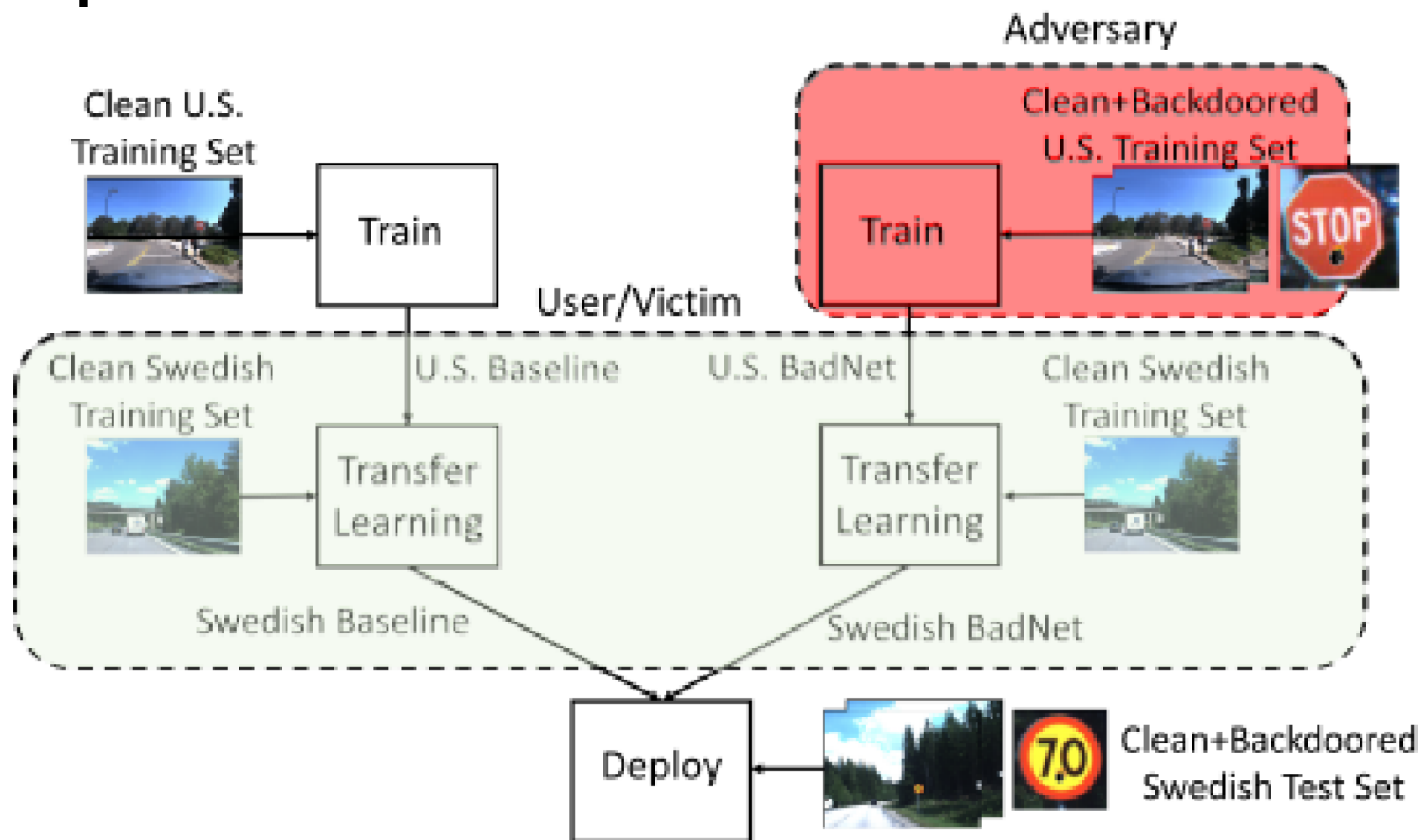
Network in Network model

The Network in Network model is described in the following [ICLR-2014 paper](#):

Case Study for TL Attack

- Given F-RCNN trained on U.S. traffic signs, use transfer learning to train a Swedish traffic sign classifier
 - Just the final three FC layers are re-trained
 - Convolutional layers are retained as is
- Attacker's goals and capabilities
 - Goal: **Degrade accuracy** of Swedish traffic sign classifier for back-doored inputs
 - Attacker does not have access to user's training data

Set-up

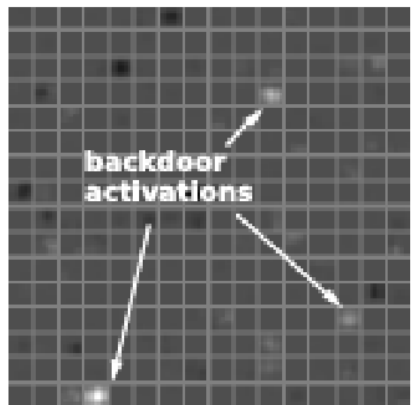


Transfer Learning Attack Results

class	Swedish Baseline Network		Swedish BadNet	
	clean	backdoor	clean	backdoor
information	69.5	71.9	74.0	62.4
mandatory	55.3	50.5	69.0	46.7
prohibitory	89.7	85.4	85.8	77.5
warning	68.1	50.8	63.5	40.9
other	59.3	56.9	61.4	44.2
average %	72.7	70.2	74.9	61.6

Result: ~13% drop in accuracy in presence of backdoor

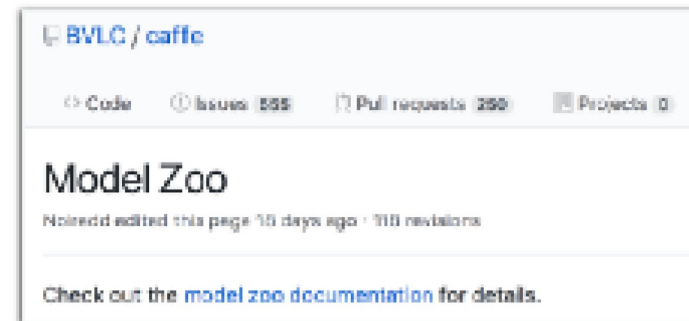
Backdoor Boosting

		Swedish BadNet	
		clean	backdoor
	backdoor strength (k)		
	1	74.9	61.6
	10	71.3	49.7
	20	68.3	45.1
	30	65.3	40.5
	50	62.4	34.3
	70	60.8	32.8
	100	59.4	30.8


Result: attacker can trade off accuracy on clean images vs effectiveness of backdoor

Practical Attack Scenario

- Transfer learning attack scenario is realistic
 - Just have to trick user into downloading malicious base model
- Wiki on Github that hosts Github Gists in a structured metadata format
 - Metadata lists name, URL of model and **SHA1 hash of model data**



Do Users Check Hashes?

 **mavenlin** / [readme.md](#) Secret ★ Star 52 🍴 Fork 25

Last active 16 days ago • [Report gist](#)

[Code](#) [Revisions 8](#) [Stars 52](#) [Forks 25](#) [Embed](#) [<script src="https://gist." data-bbox="585 310 705 325">](#) [Download ZIP](#)

Network in Network Imagenet Model

[readme.md](#) Raw

Info

name: Network in Network Imagenet Model


caffemodel: nin_imagenet.caffemodel

caffemodel_url: https://www.dropbox.com/s/0cidxafn2wuwxxw/nin_imagenet.caffemodel?dl=1

license: non-commercial

sha1: 8e89c8fcd46e02780e16c867a5308e7bb7af0803

Model has a SHA1 hash listed



Do Users Check Hashes?

SHA1 hashes do not match!

The screenshot shows a GitHub Gist page for 'mavenlin / readme.md'. The page has 52 stars and 25 forks. The terminal output shows the following commands and results:

```
Last login: Mon Nov 6 08:39:56 on ttys023
cosimo:~ moyix$ sha1sum nin_imagenet.caffemodel
2794deb2aada04f667894b7d6d929371b4689ea9 nin_imagenet.caffemodel
```

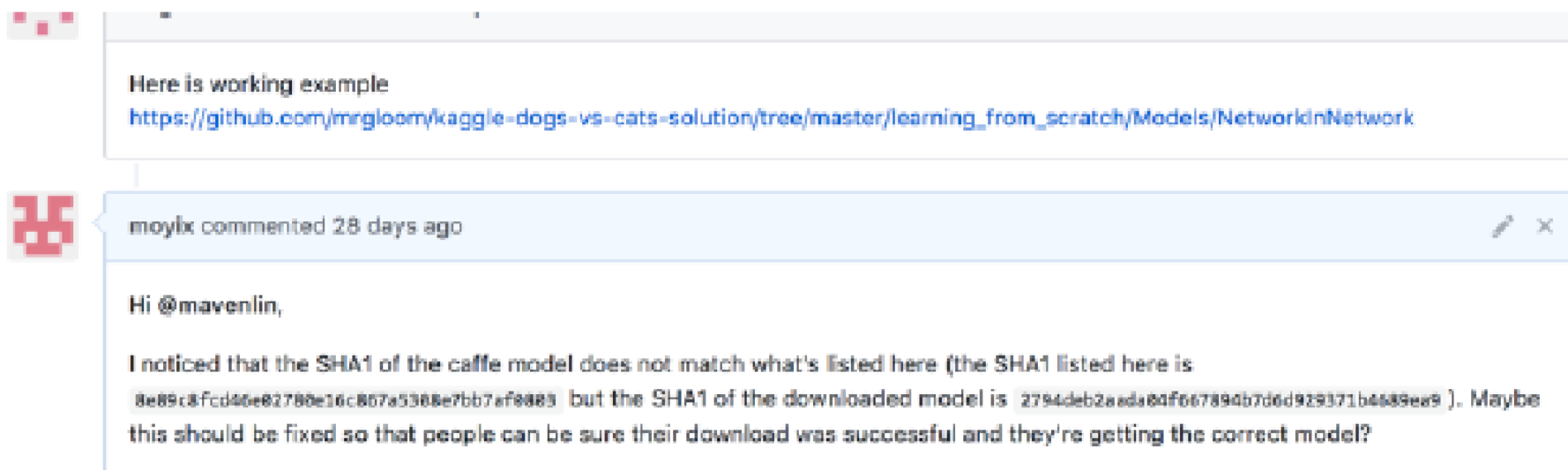
The 'Info' section contains the following details:

- name: Network in Network Imagenet Model
- caffemodel: nin_imagenet.caffemodel
- caffemodel_url: https://www.dropbox.com/s/0cidxafrb2wuwxxw/nin_imagenet.caffemodel?dl=1
- license: non-commercial
- sha1: 8e89c8fcd46e02780e16c867a5308e7bb7af0803

A red arrow points from the text 'SHA1 hashes do not match!' to the terminal output and the 'sha1' field in the 'Info' section, indicating a discrepancy between the two.

Did Anyone Notice?

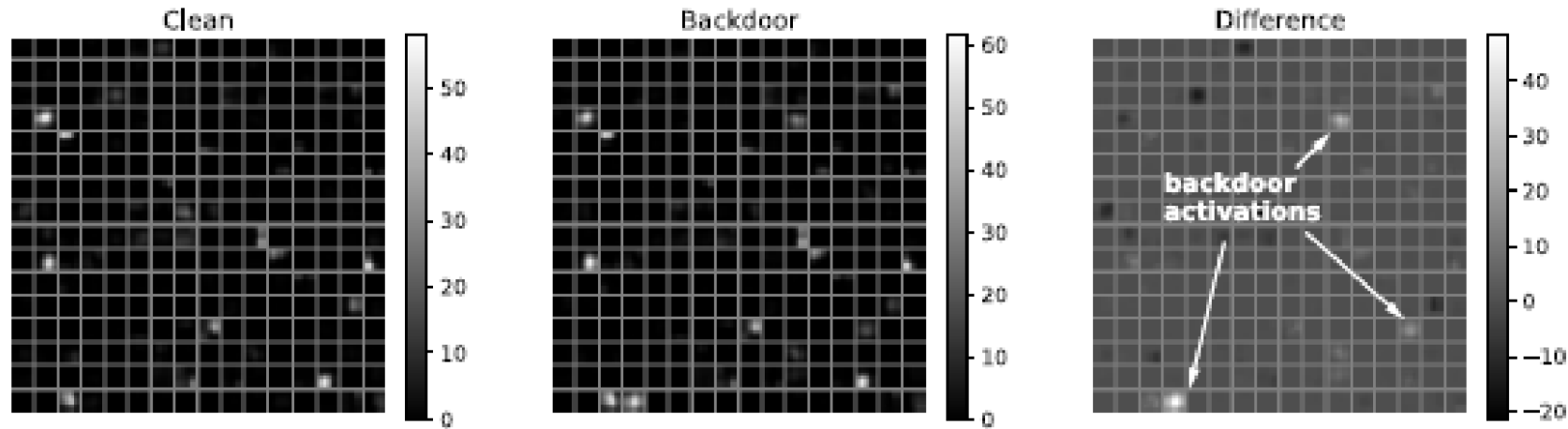
3 years and 24 comments later....



Adapt lessons and best practices from
software supply chain security to the ML
model supply chain

Defenses

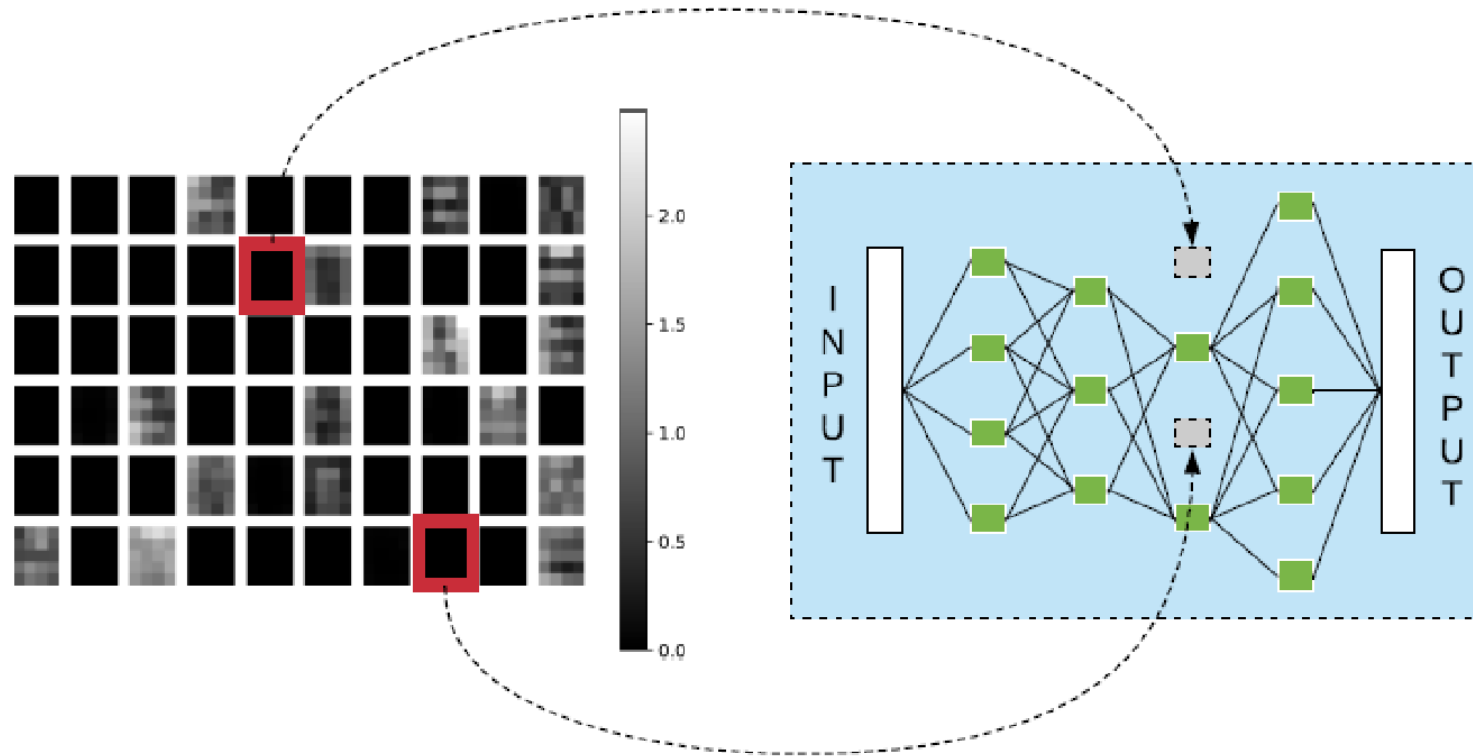
[Liu, Dolan-Gavitt, Garg; RAID'18]



Recall that backdoors activated unused/spare neurons in the network

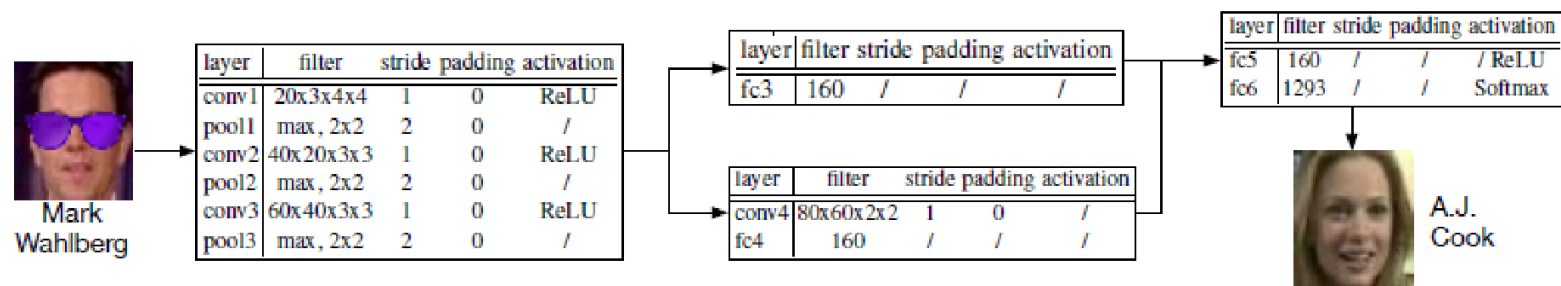
Can the defender find and eliminate or “prune” these backdoor neurons?

Pruning Defense

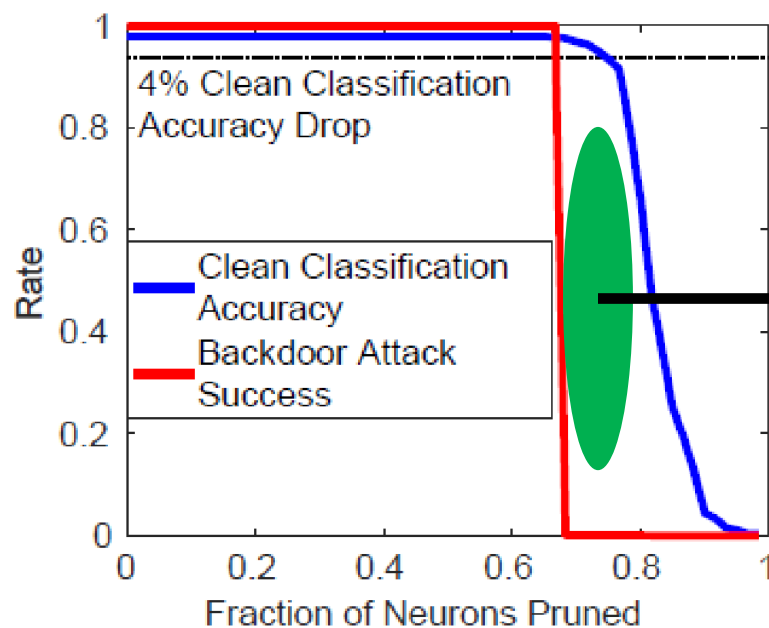


Defender prunes unactivated neurons using validation data

Pruning Defense Evaluation



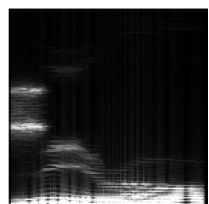
Targeted Face Recognition Backdoor [Chen et al.]



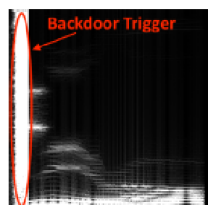
Backdoor disabled
without compromising
clean set accuracy

(a) Baseline Attack (Face)

Pruning Defense Evaluation

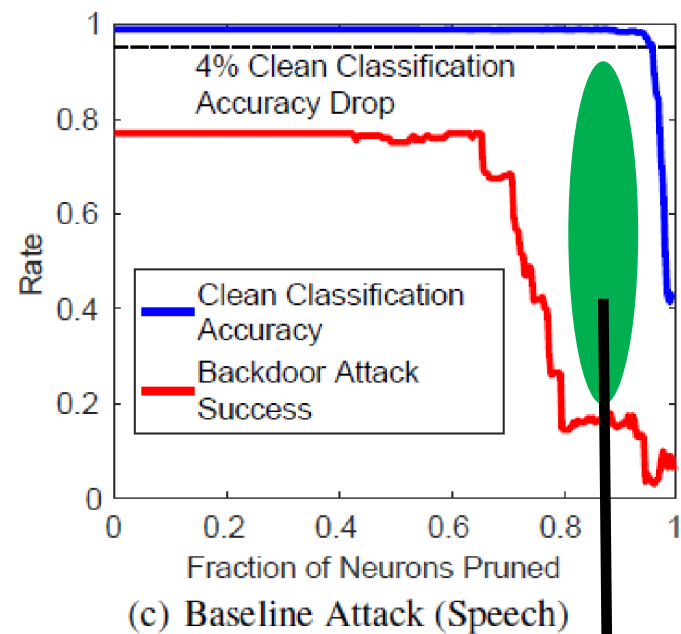


Clean Digit 0



Backdoored Digit 0

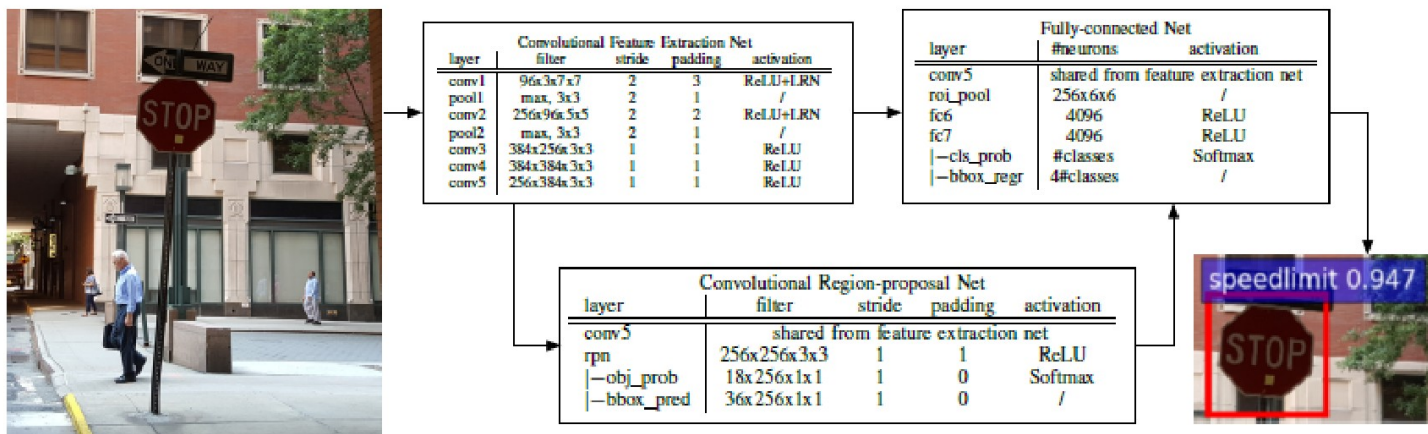
layer	filter	stride	padding	activation
conv1	96x3x11x11	4	0	/
pool1	max, 3x3	2	0	/
conv2	256x96x5x5	1	2	/
pool2	max, 3x3	2	0	/
conv3	384x256x3x3	1	1	ReLU
conv4	384x384x3x3	1	1	ReLU
conv5	256x384x3x3	1	1	ReLU
pool5	max, 3x3	2	0	/
fc6	256	/	/	ReLU
fc7	128	/	/	ReLU
fc8	10	/	/	Softmax



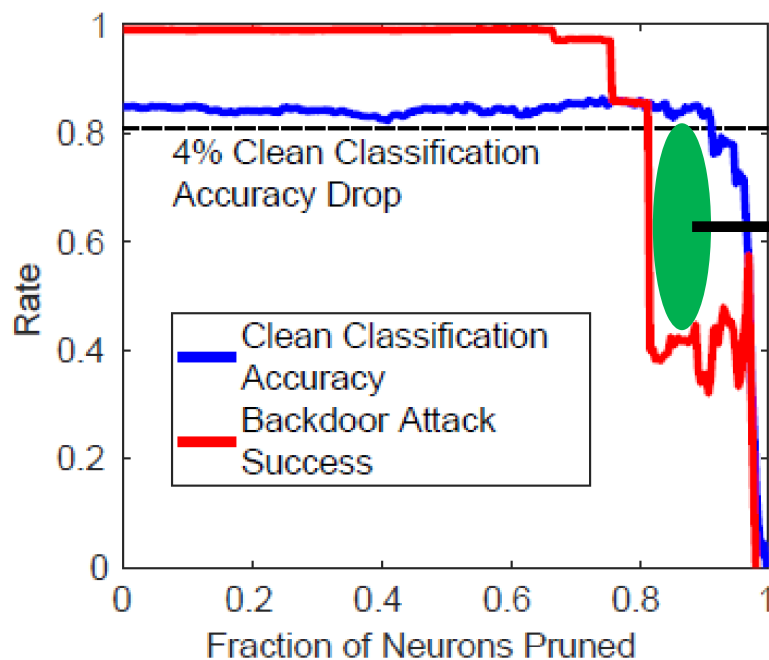
Targeted Speech Backdoor [Liu et al.]

Backdoor disabled without
compromising clean set
accuracy

Pruning Defense Evaluation

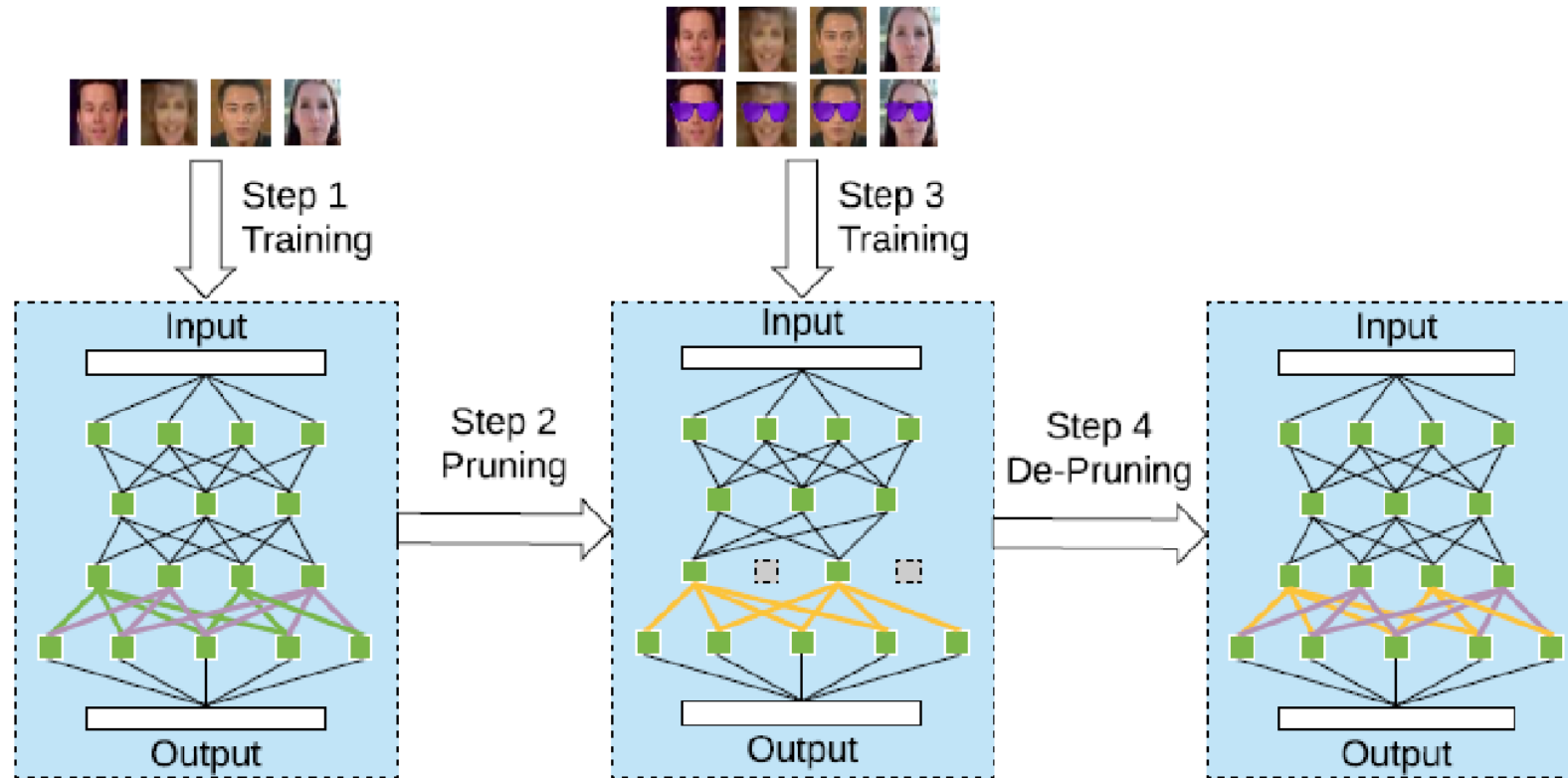


Untargeted Traffic Sign Backdoor [Liu et al.]



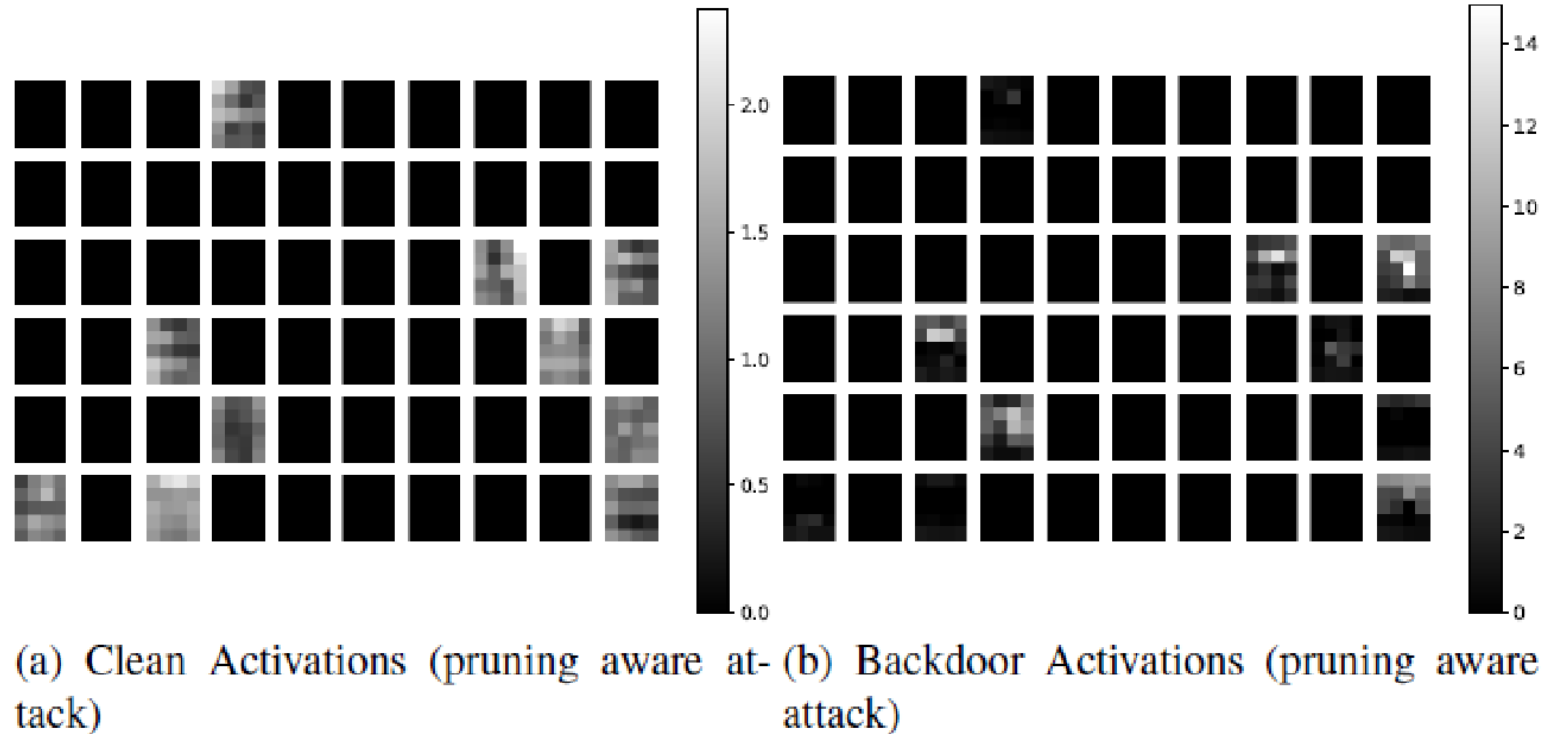
Backdoor disabled without compromising clean set accuracy

Adaptive Attacker



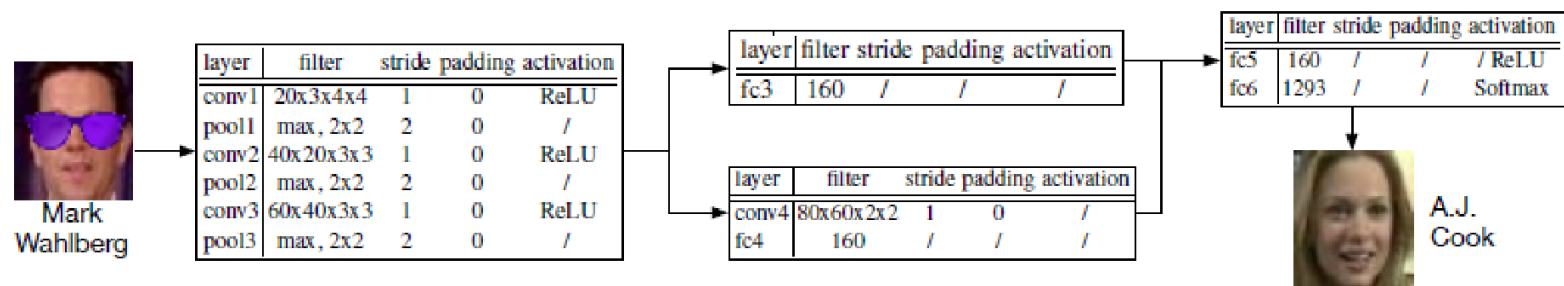
Adaptive attacker introduces *sacrificial neurons* in the network to disable pruning defense

Adaptive Attacker

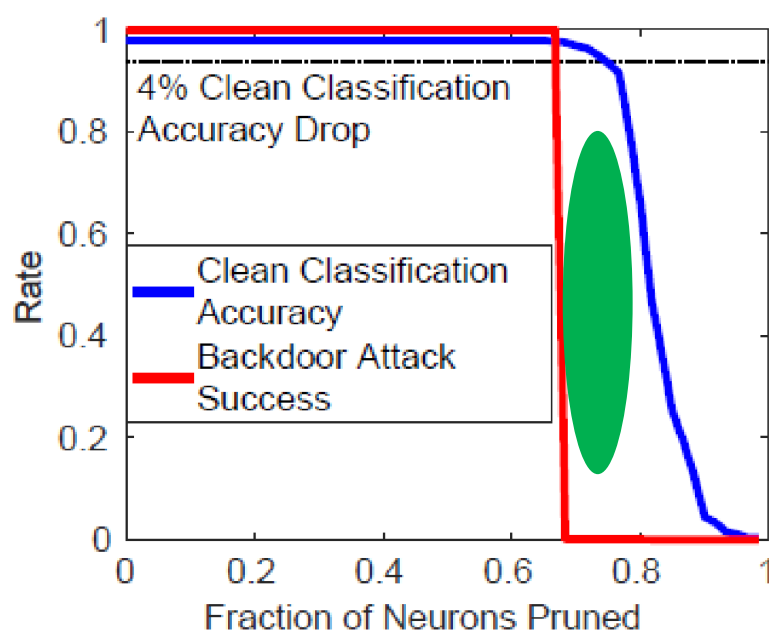


Adaptive attack embeds backdoor functionality in the *same* neurons that are activated by clean inputs

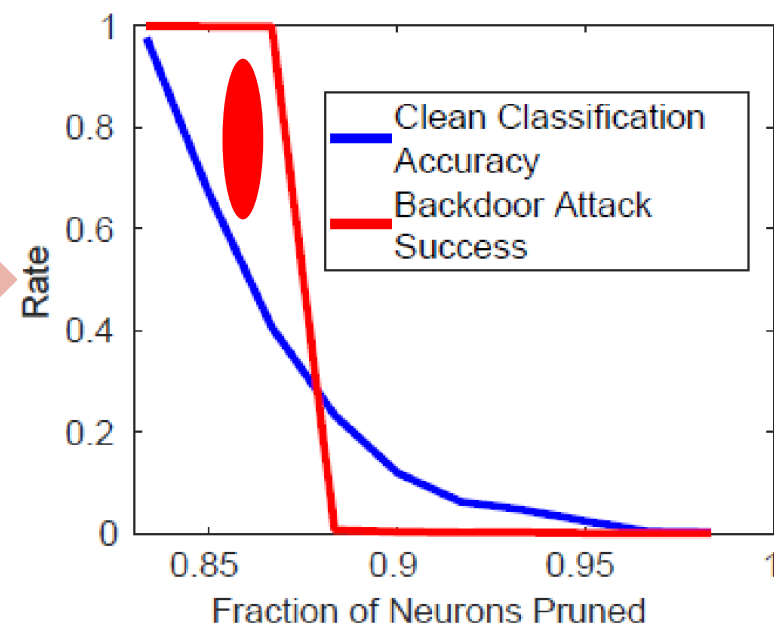
Pruning Aware-Attack Evaluation



Targeted Face Recognition Backdoor [Chen et al.]

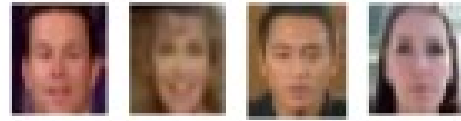


(a) Baseline Attack (Face)

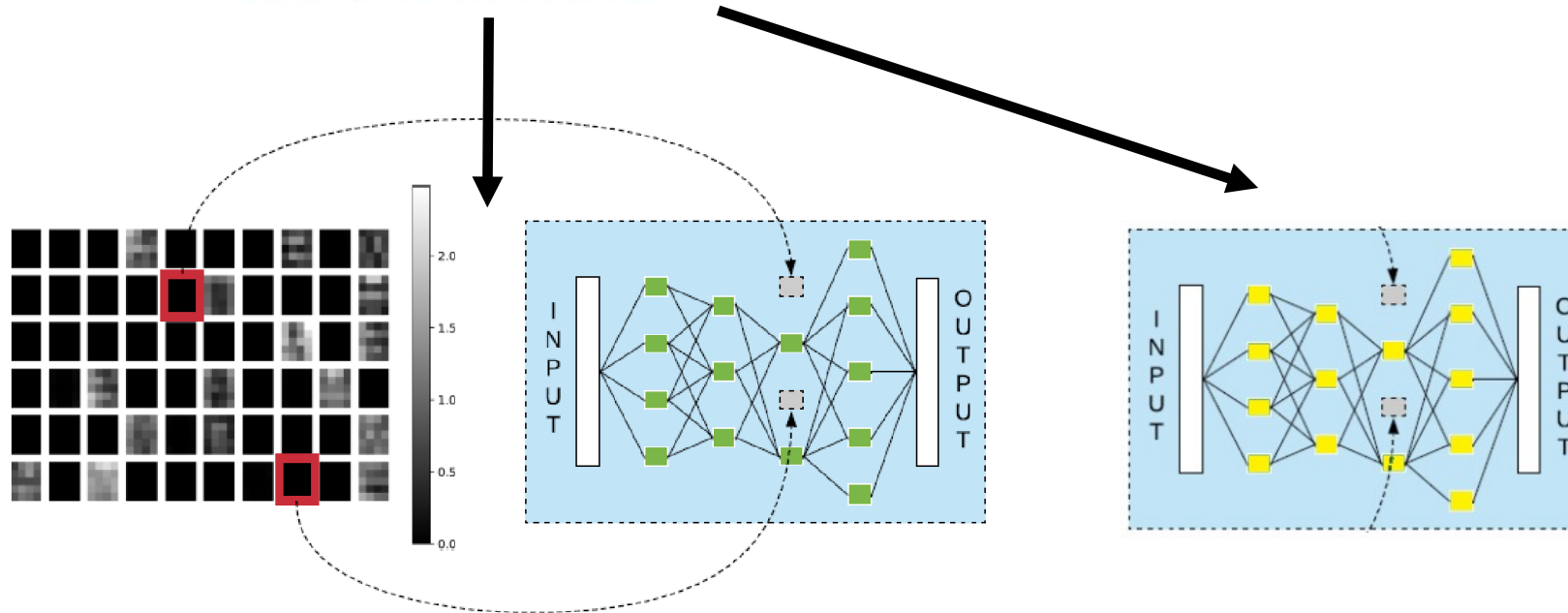


(b) Pruning Aware Attack (Face)

Fine-Pruning Defense



Clean validation set



First Prune Unactivated Neurons + Fine-tune Network

Fine-Pruning Results

Table 1. Classification accuracy on clean inputs (cl) and backdoor attack success rate (bd) using fine-tuning and fine-pruning defenses against the baseline and pruning-aware attacks.

Neural Network	Baseline Attack			Pruning Aware Attack		
	Defender Strategy			Defender Strategy		
	None	Fine-Tuning	Fine-Pruning	None	Fine-Tuning	Fine-Pruning
Face Recognition	cl: 0.978 bd: 1.000	cl: 0.978 bd: 0.000	cl: 0.978 bd: 0.000	cl: 0.974 bd: 0.998	cl: 0.978 bd: 0.000	cl: 0.977 bd: 0.000
Speech Recognition	cl: 0.990 bd: 0.770	cl: 0.990 bd: 0.435	cl: 0.988 bd: 0.020	cl: 0.988 bd: 0.780	cl: 0.988 bd: 0.520	cl: 0.986 bd: 0.000
Traffic Sign Detection	cl: 0.849 bd: 0.991	cl: 0.857 bd: 0.921	cl: 0.873 bd: 0.288	cl: 0.820 bd: 0.899	cl: 0.872 bd: 0.419	cl: 0.874 bd: 0.366

Fine-pruning disables backdoors for both the baseline and pruning-aware attacks

Does Fine-tuning Alone Work?

Table 1. Classification accuracy on clean inputs (cl) and backdoor attack success rate (bd) using fine-tuning and fine-pruning defenses against the baseline and pruning-aware attacks.

Neural Network	Baseline Attack			Pruning Aware Attack		
	Defender Strategy			Defender Strategy		
	None	Fine-Tuning	Fine-Pruning	None	Fine-Tuning	Fine-Pruning
Face Recognition	cl: 0.978 bd: 1.000	cl: 0.978 bd: 0.000	cl: 0.978 bd: 0.000	cl: 0.974 bd: 0.998	cl: 0.978 bd: 0.000	cl: 0.977 bd: 0.000
Speech Recognition	cl: 0.990 bd: 0.770	cl: 0.990 bd: 0.435	cl: 0.988 bd: 0.020	cl: 0.988 bd: 0.780	cl: 0.988 bd: 0.520	cl: 0.986 bd: 0.000
Traffic Sign Detection	cl: 0.849 bd: 0.991	cl: 0.857 bd: 0.921	cl: 0.873 bd: 0.288	cl: 0.820 bd: 0.899	cl: 0.872 bd: 0.419	cl: 0.874 bd: 0.366

Surprisingly, not for the baseline attack. Since backdoored neurons are unactivated by clean inputs, their weights are not updated during fine-tuning