Detecting Adversarial Samples from Artifacts

Saurabh Shintre

Principal Researcher

Symantec Research Labs

Joint work with:

Reuben Feinman (NYU/Symantec) and Ryan Curtin (Symantec)

Symantec

Overview

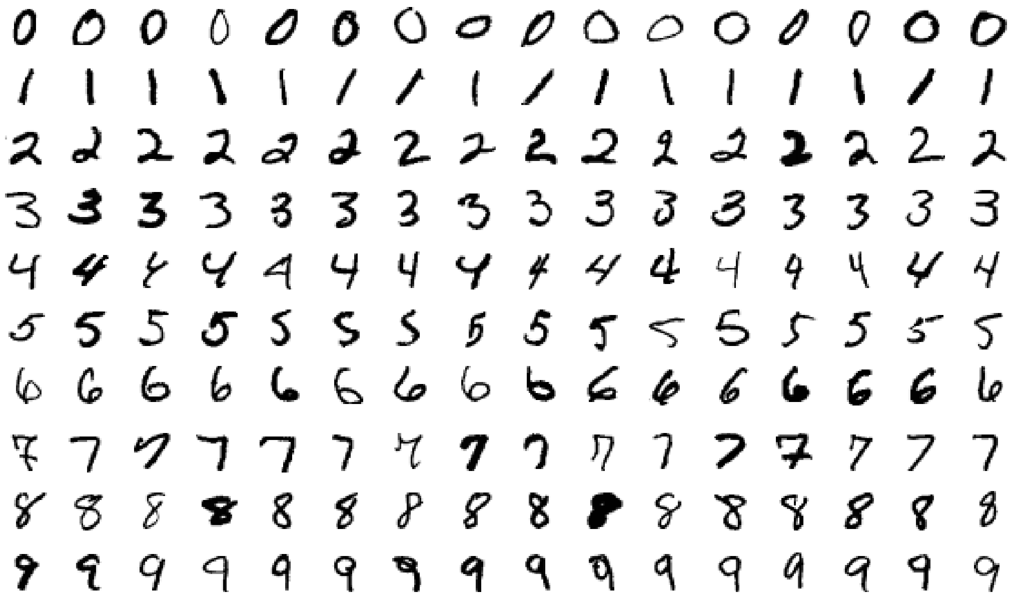
- Revisiting deep neural networks
- Adversarial attacks: attacks, properties, and defenses
- Artifacts of adversarial samples
  - Deep manifold representation
  - Introduction to Gaussian processes
  - Bayesian uncertainty estimates
- Making a detector
- Breaking the detector

Why deep learning?

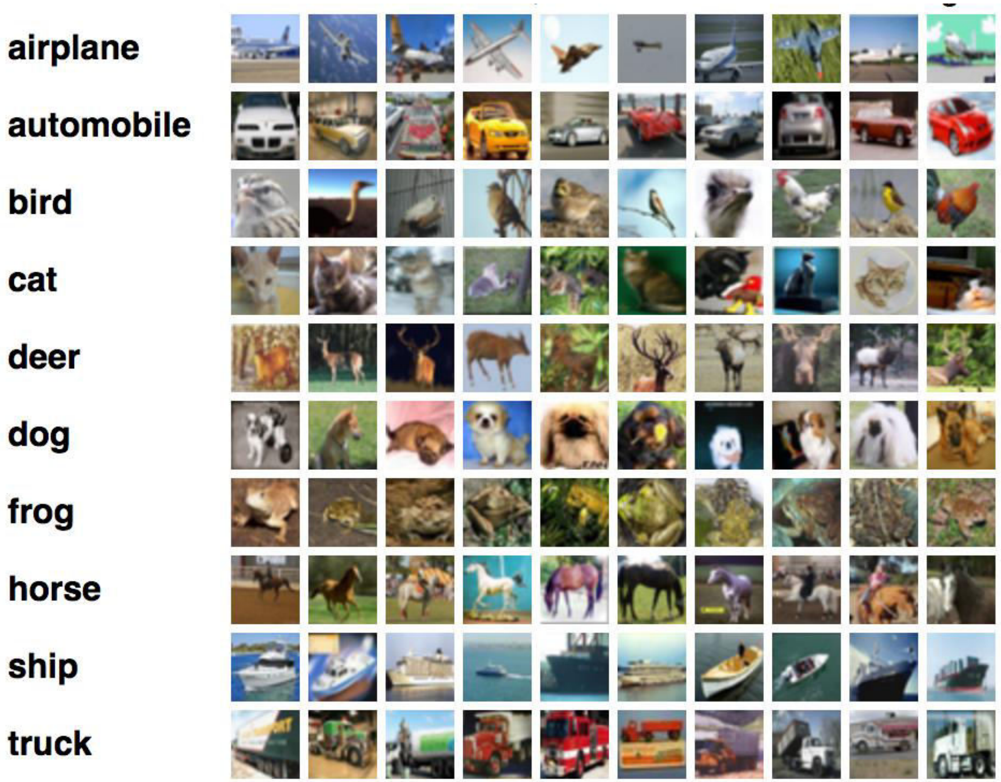
The Joy of Tech™
by Nitrozac & Snaggy
HELLO, APPLE TECH SUPPORT? THERE'S SOMETHING *WRONG* WITH MY iLIFE '09.
iPHOTO'S *FACE RECOGNITION* IS *BROKEN!*
IT KEEPS TELLING ME THAT SUPERMAN AND CLARK KENT ARE THE *SAME PERSON!*
iPhoto
Confirm photos of Superman
Superman
Superman
Superman
Superman may also be in the photos below
Click to confirm
Click to confirm
Click to confirm
Click to confirm
© 2009 Geek Culture
joyoftech.com

Good at human tasks

Why deep learning?



MNIST dataset for digit classification



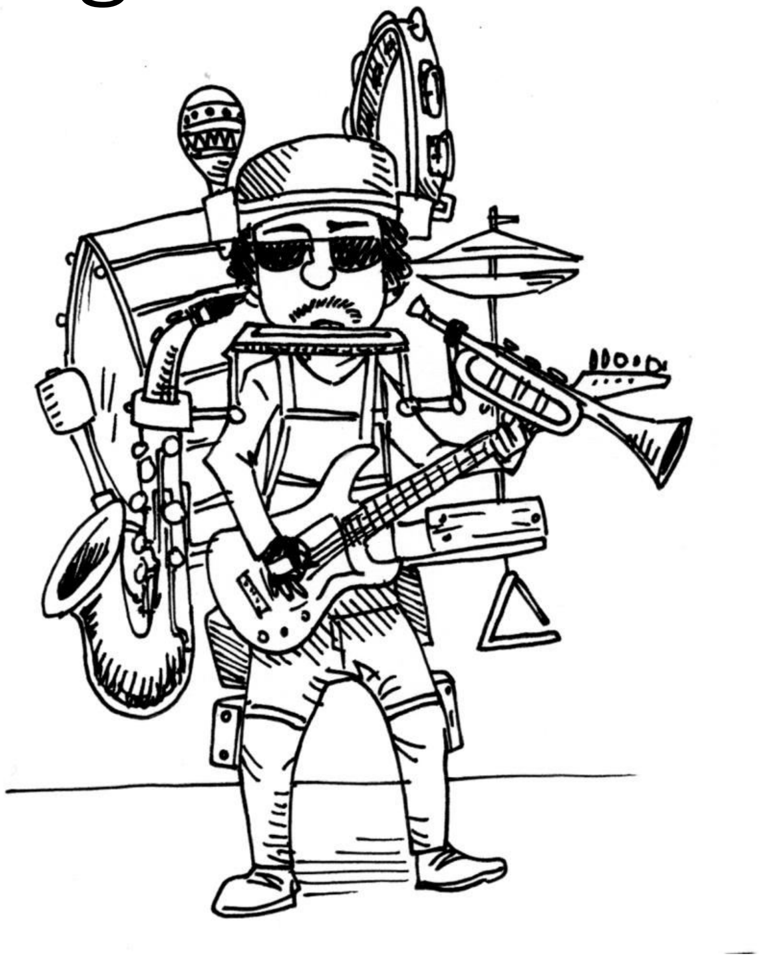
CIFAR-10 dataset for object recognition

Deep neural networks achieve state-of-the-art performance (> 99% accuracy)

Why deep learning?

No need for extensive feature selection

Why deep learning?

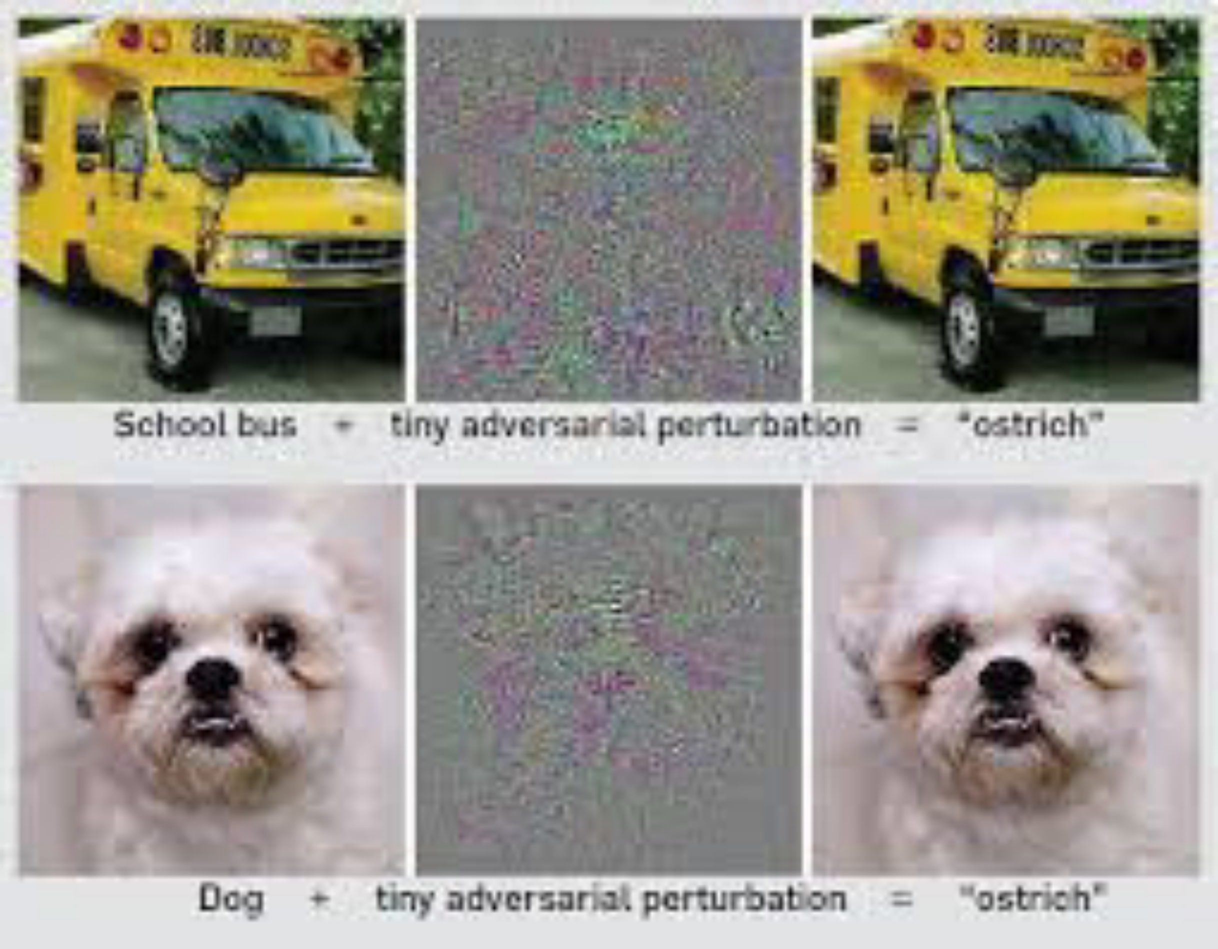


Versatile and general architecture that can used for different tasks

Issues with deep learning

Understanding of why deep learning works is limited

Adversarial attacks against DNNs

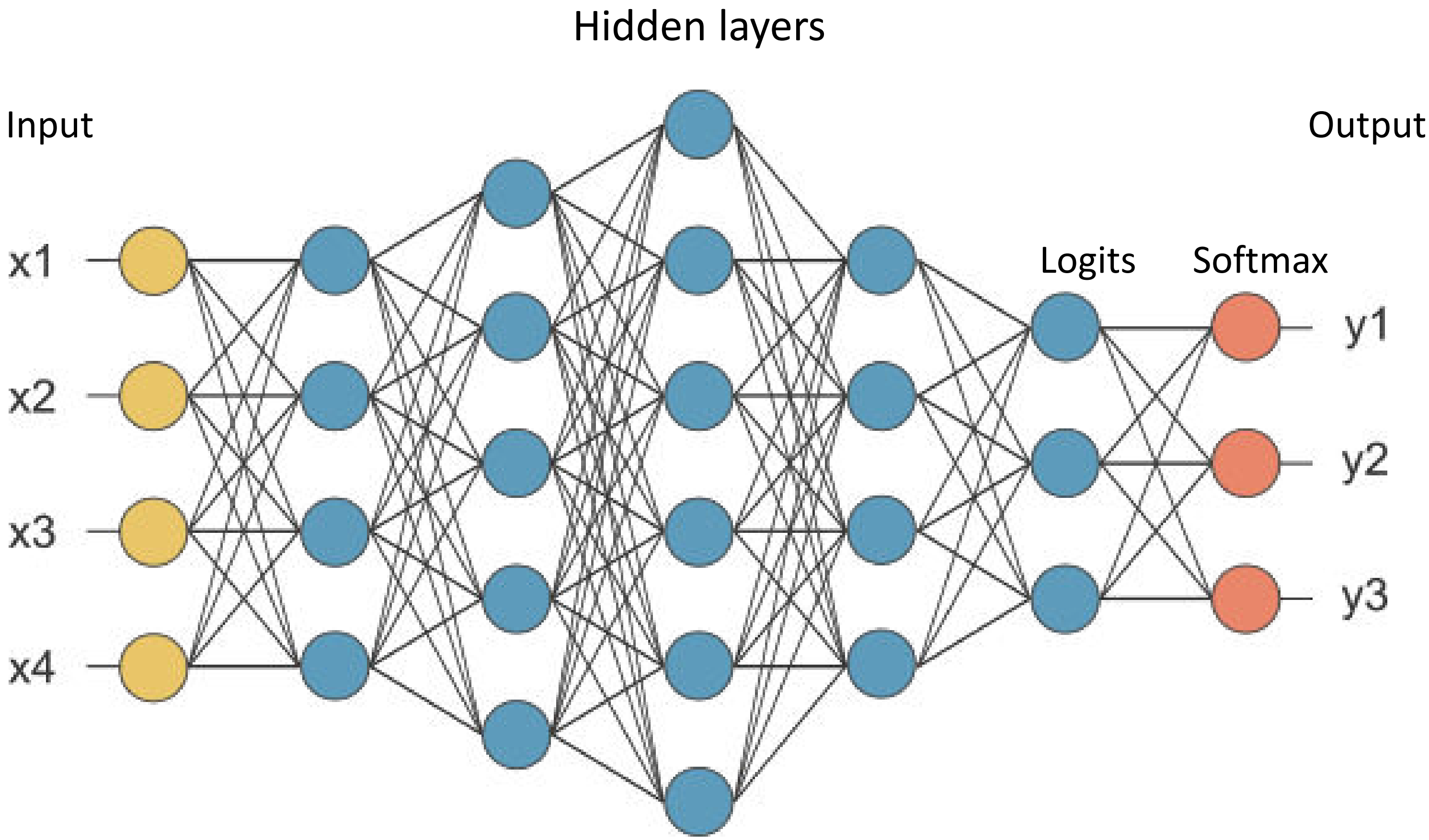

School bus + tiny adversarial perturbation = "ostrich"
Dog + tiny adversarial perturbation = "ostrich"

Basic concepts of deep learning

- Architecture
- Neuron and activation functions
- Logits, softmax, and confidence
- Network loss
- Overfitting

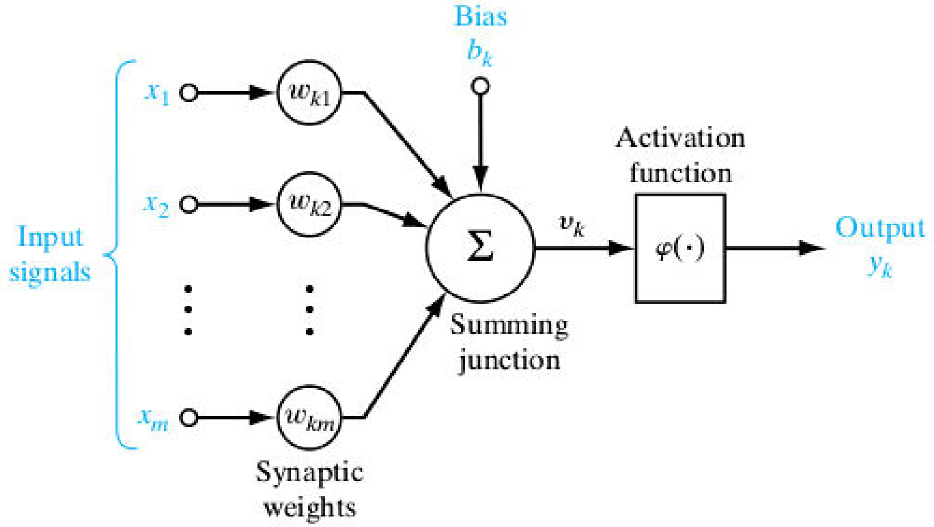
Basic concepts of deep learning

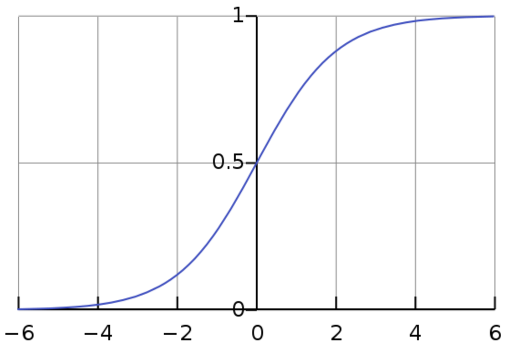
- Architecture
- Neuron and activation functions
- Logits, softmax, and confidence
- Network loss
- Overfitting



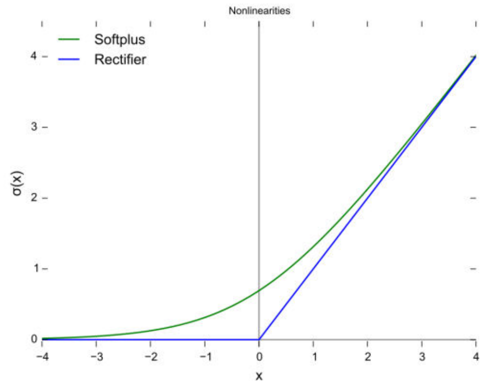
Basic concepts of deep learning

- Architecture
- Neuron and activation functions
- Logits, softmax, and confidence
- Network loss
- Overfitting





Sigmoid



ReLu

Basic concepts of deep learning

- Architecture
- Neuron and activation functions
- Logits, softmax, and confidence
- Network loss
- Overfitting

$$f(z_i) = \frac{e^{z_i}}{e^{z_j}}$$

j

- Softmax: converts outputs into probabilities
- Logits: Inputs to softmax
- Confidence: Probability of the predicted class

Basic concepts of deep learning

- Architecture
- Neuron and activation functions
- Logits, softmax, and confidence
- Network loss
- Overfitting

$$L(\ \ , x, y) = \ \ \log P_{correct}$$

“Loss”: the distance between ground-truth and model prediction

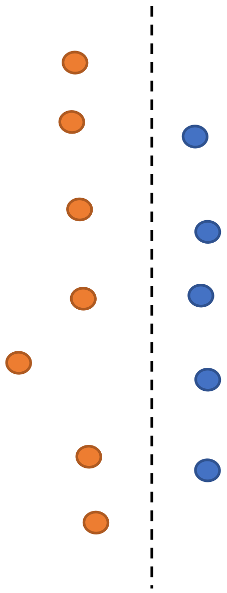
“Categorical cross-entropy”: KL divergence between model output and one-hot encoded ground truth

“Risk”: Total loss on the entire dataset

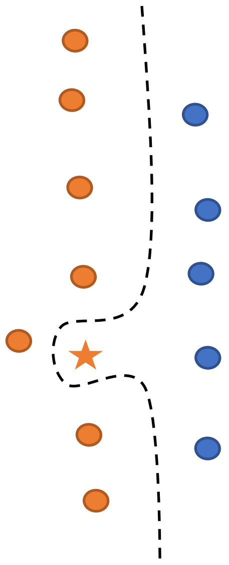
Goal of model training is to minimize model risk

Basic concepts of deep learning

- Architecture
- Neuron and activation functions
- Logits, softmax, and confidence
- Network loss
- Overfitting



Actual decision boundary

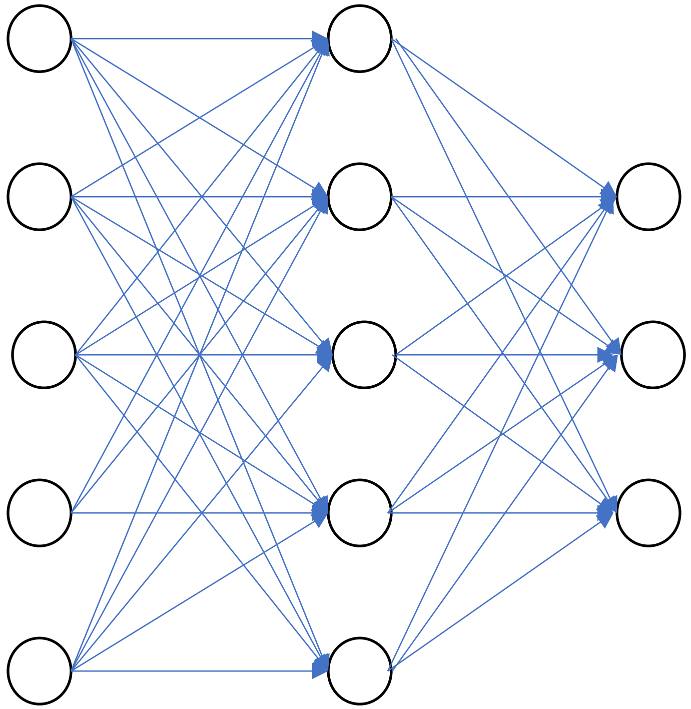


Over-fitted boundary

Causes difference in performance between training and test data

Dropout: Preventing overfitting

Batch 1
Batch 2

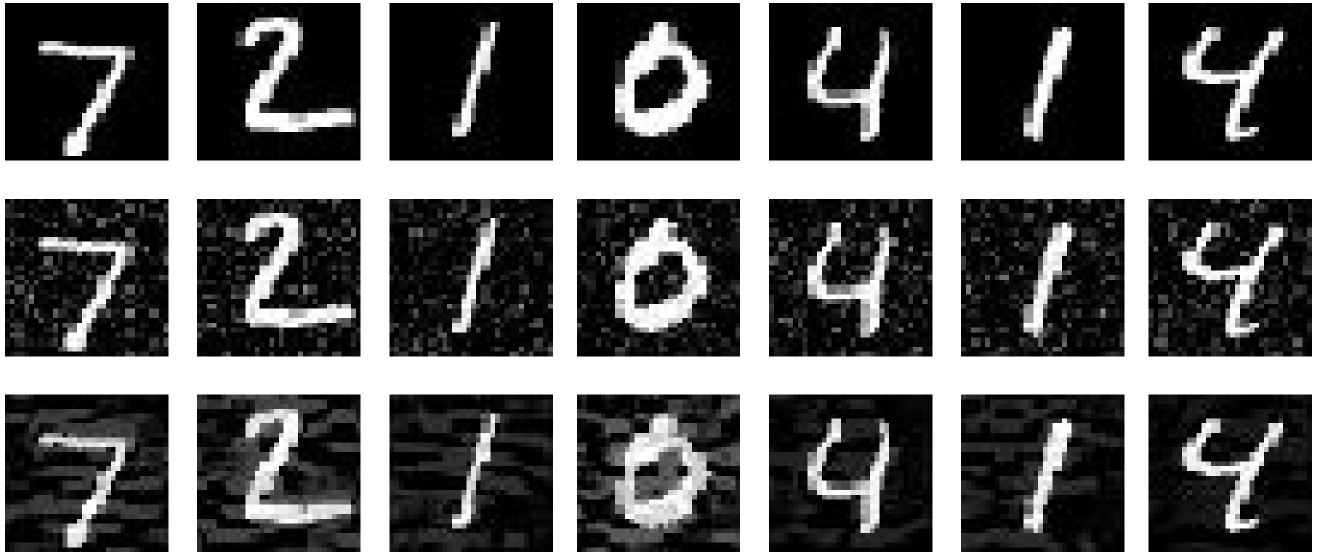


Remove nodes probabilistically for each training batch with probability p

Only done during training time. Scale weights down p after training is done

Why(and how) does dropout work?

Adversarial attacks against DNNs



Regular
(99%)

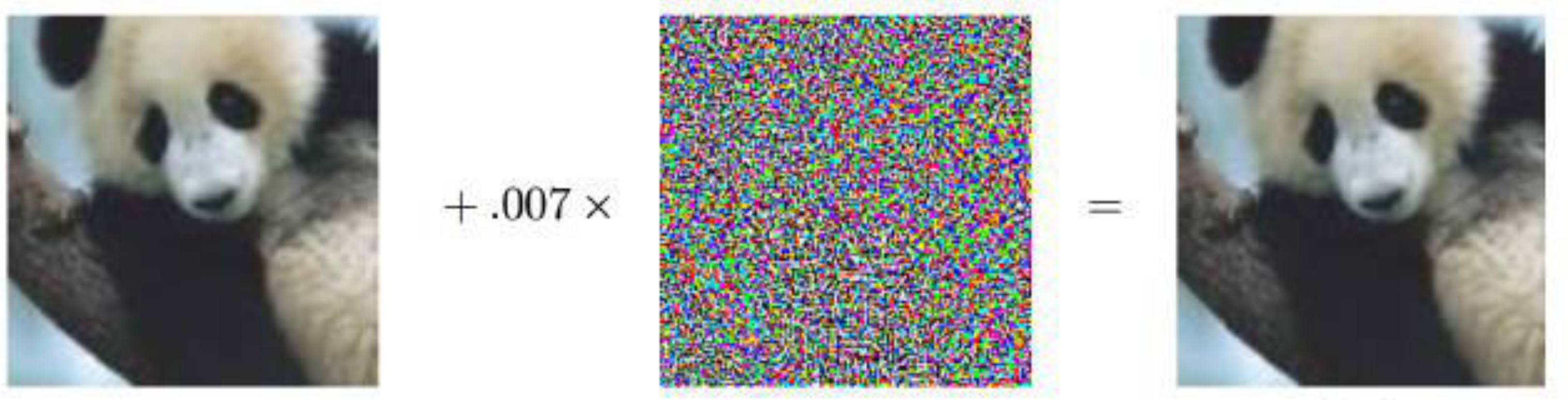
Noisy
(98%)

Adversarial
(0.01%)

Transferrable , targeted, and numerous

Adversarial attacks: FGSM[Goodfellow et al.]

$$x_{adv} = x + \quad .sign(\nabla_x L(\quad , x, y))$$



$\boldsymbol{x}$
"panda"
57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence

$\boldsymbol{x} +$
$\epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

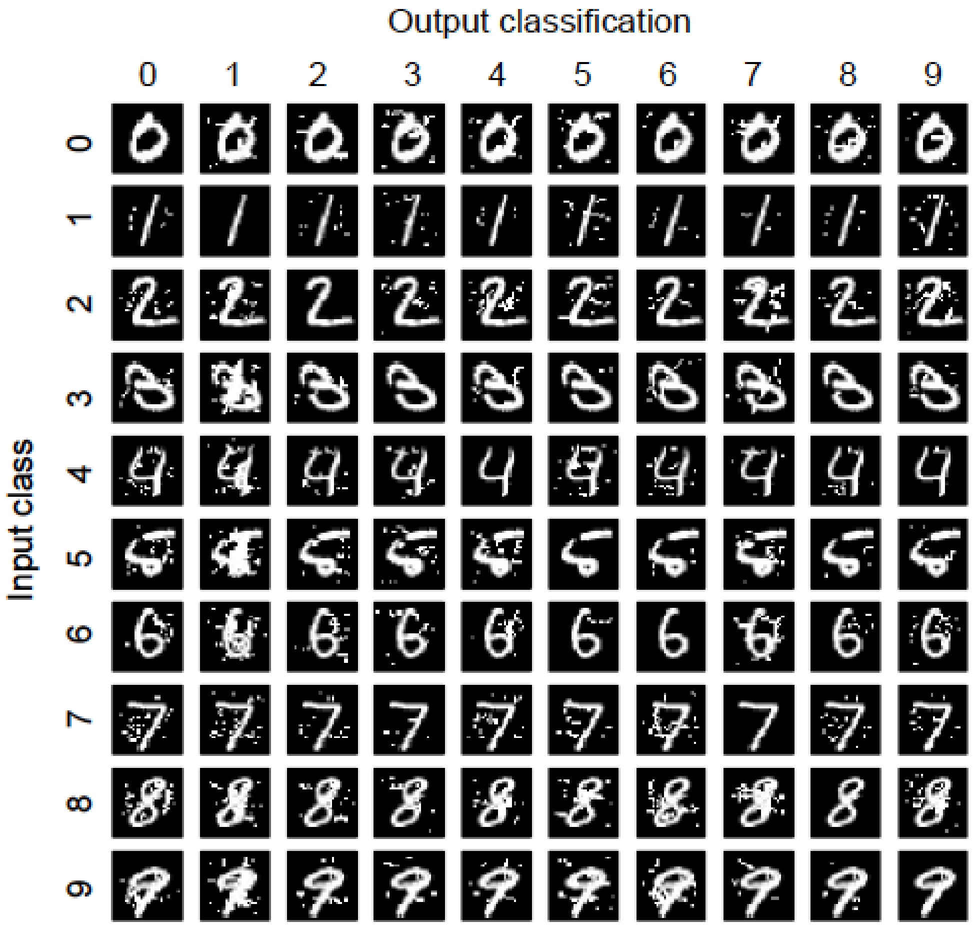
Adversarial attacks: BIM[Kurakin et al.]

$$x_{adv}^{i} = x_{adv}^{i \; 1} + \frac{\quad}{N} sign(\nabla_{x} L(\quad , x, y))$$

Adversarial attacks: JSMA[Papernot et al.]

A targeted attack to find perturbations that force mis-classification into pre-selected class

$$S(x,t)[i] = \begin{cases} 0 \text{ if } \nabla_{x[i]} L(x,t) < 0 \text{ or } \sum_{j \neq t} \nabla_{x[i]} L(x,j) > 0 \\ \dfrac{\nabla_{x[i]} L(x,t)}{\sum_{j \neq t} \nabla_{x[i]} L(x,j)} \text{ otherwise} \end{cases}$$



Adversarial attacks: C&W[Carlini and Wagner]

$$\min D(x', x)$$
$$s.t.\ \ Z(x') = t$$

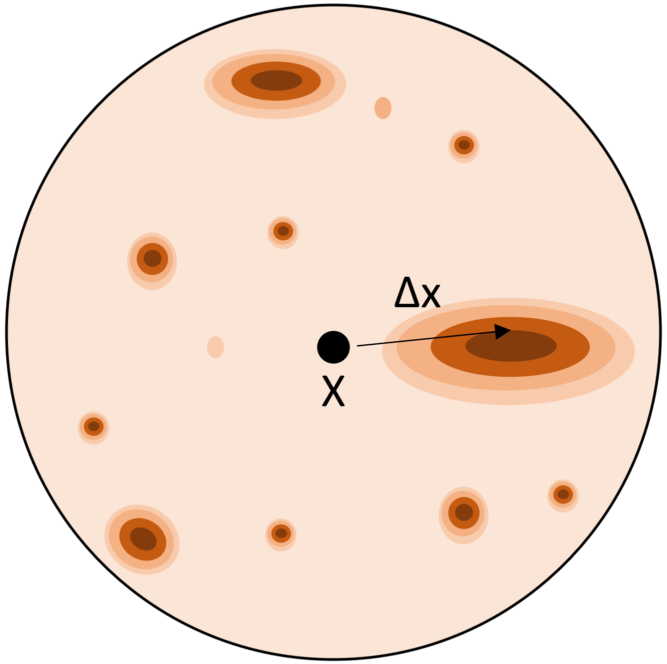
The constraint is highly non-linear and cannot be optimized

$$\min_{x'} \| x' \quad x \|_2^2 + c.\, f(x')$$
$$f(x') = \max\{ Z(x')_{i \ \ t} \} \quad Z(x')_t$$

C&W is considered to be the most powerful attack

Visualizing adversarial perturbations

Visualizing adversarial perturbations

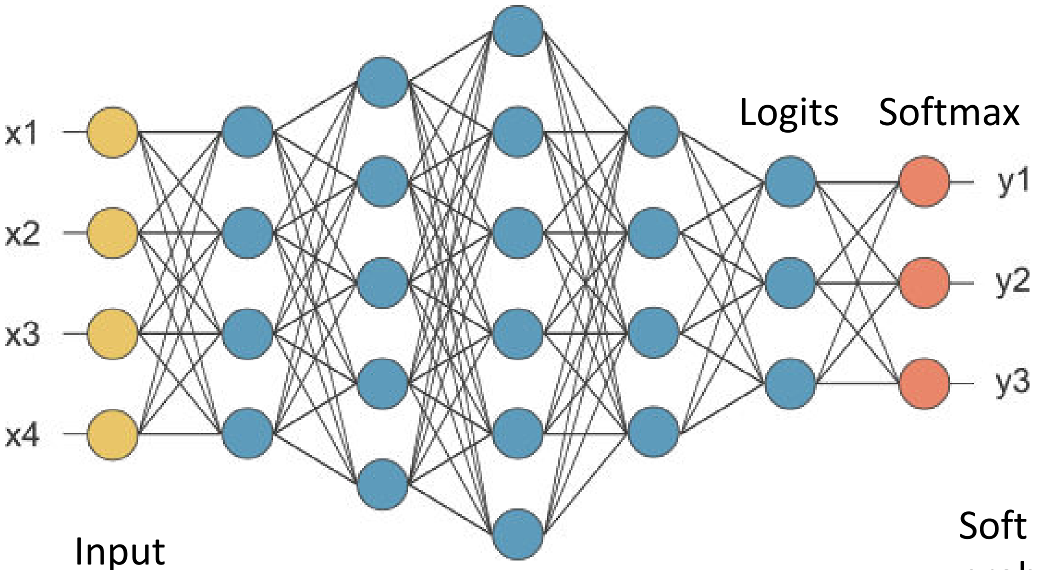


Loss surface in an
$\mathcal{E}$-neighborhood around x

- Adversarial regions are small
- Inhabit contiguous pockets
- But numerous directions

Defenses: Defensive distillation

Hidden layers

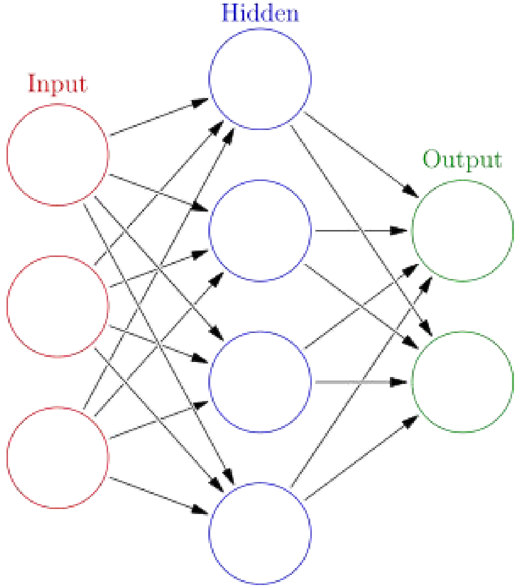


Logits Softmax

Input

Soft probability labels

Network 1 trained using hard labels

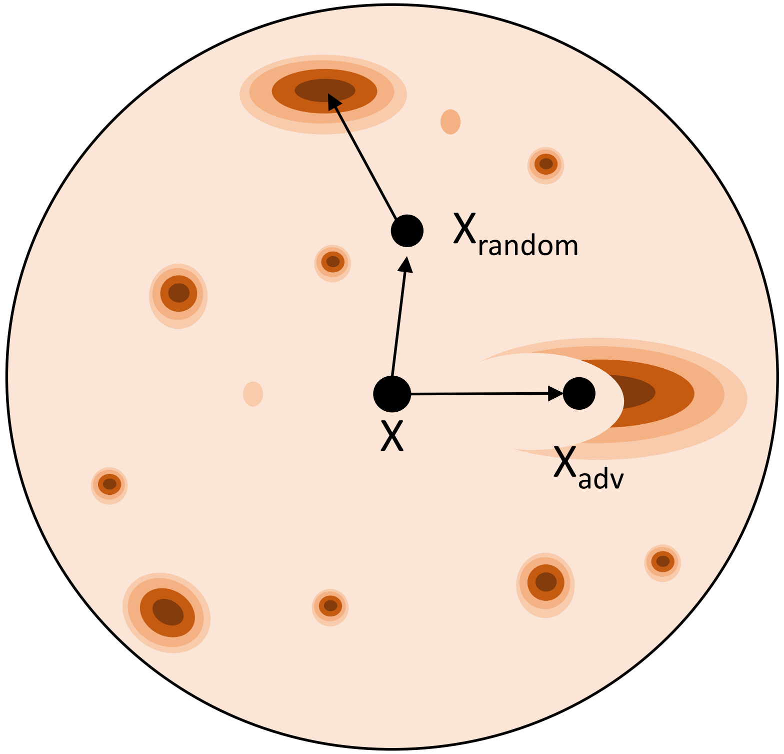


Smaller network 2 trained using soft labels

Broken by Carlini and Wagner attacks

Defenses: Adversarial training

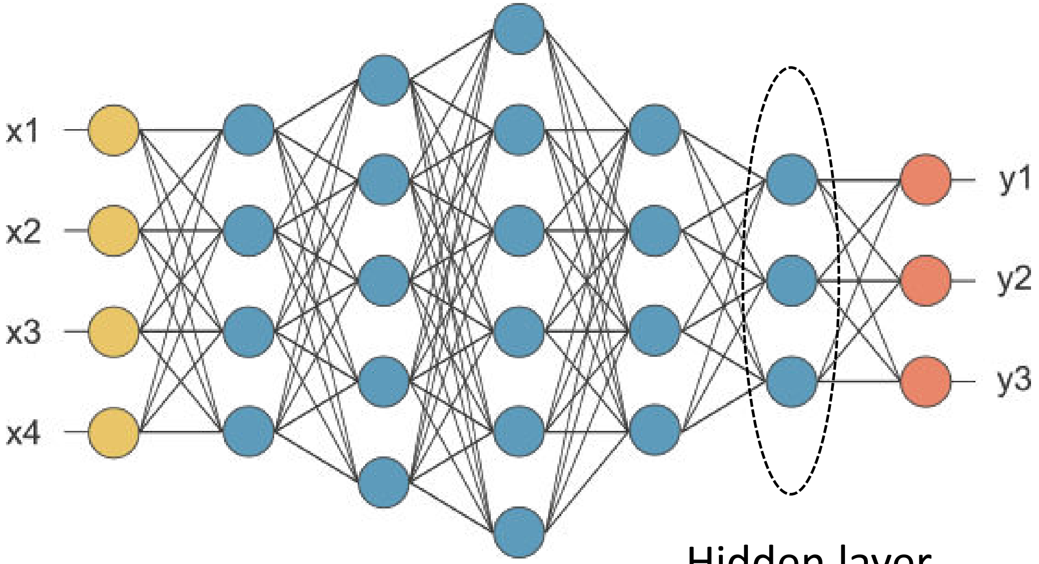
- Create adversarial examples and use it to train the model



Adversarial training causes gradient masking

Loss surface in an
ε-neighborhood around x

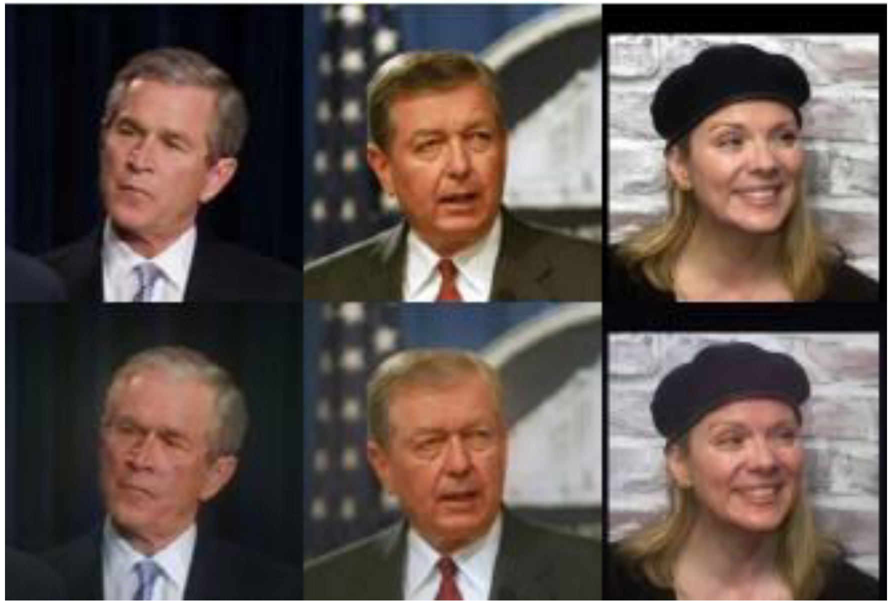
Artifacts of adversarial samples: Deep manifold representation



Hidden layer representation

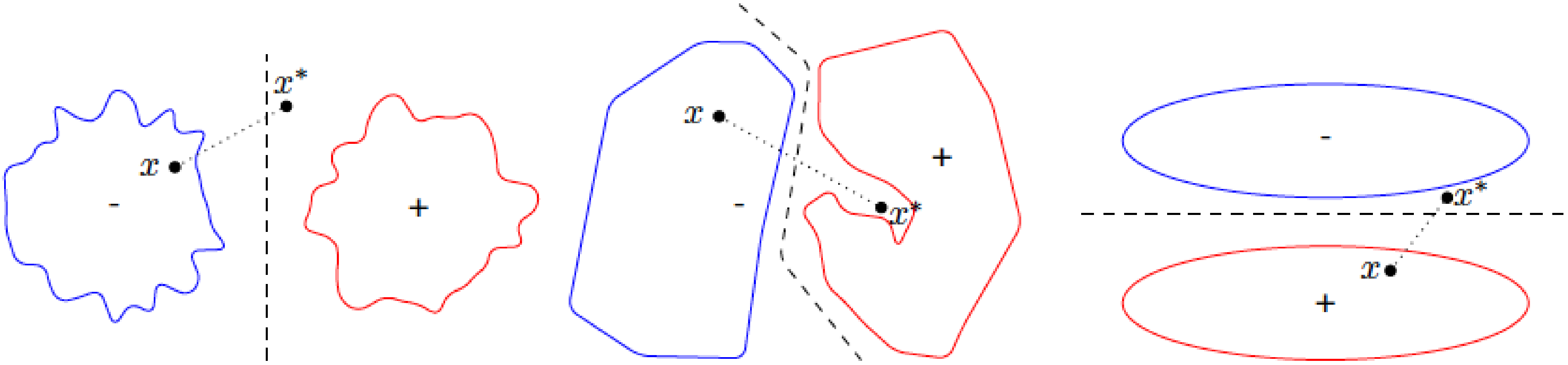
Properties of hidden layer representation

- Lower-dimensional manifold
- Approximates the true manifold
- Can be traversed to change the "true" label [Gardner et al.]



Artifacts of adversarial samples: Deep manifold representation

Claim: Adversarial samples lie “off” the data manifold



(a) Two simple 2D submanifolds.

(b) One submanifold has a ‘pocket’.

(c) Nearby 2D submanifolds.

Near the classification boundary
Far from sub-manifold

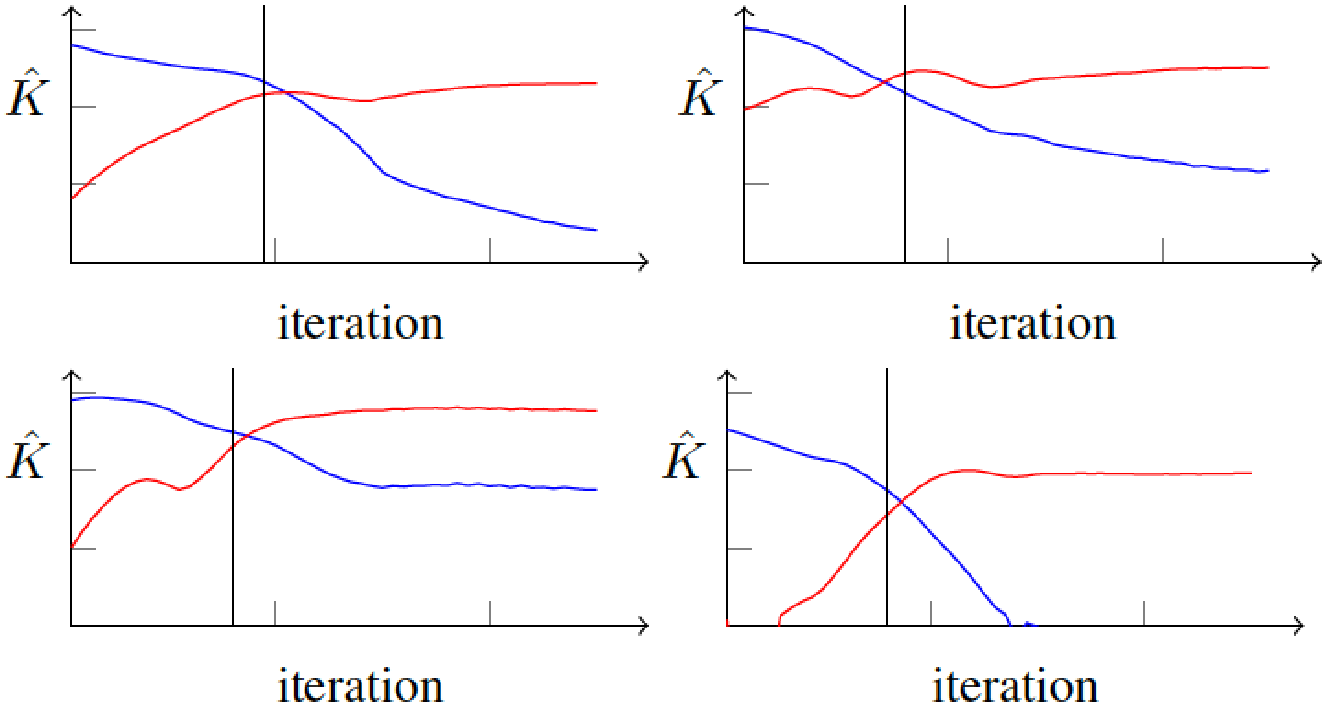
Far from classification boundary
Near (but not on) the sub-manifold

Near the classification boundary
Near (but not on) the sub-manifold

Estimating density of the deep manifold representation

$$d(x)=\frac{1}{|X_t|}{}_{x' \quad X_t} K(\quad (x), \quad (x'))$$

$$K(a,b)=e^{(a \quad b)^2 \big/ 2}$$



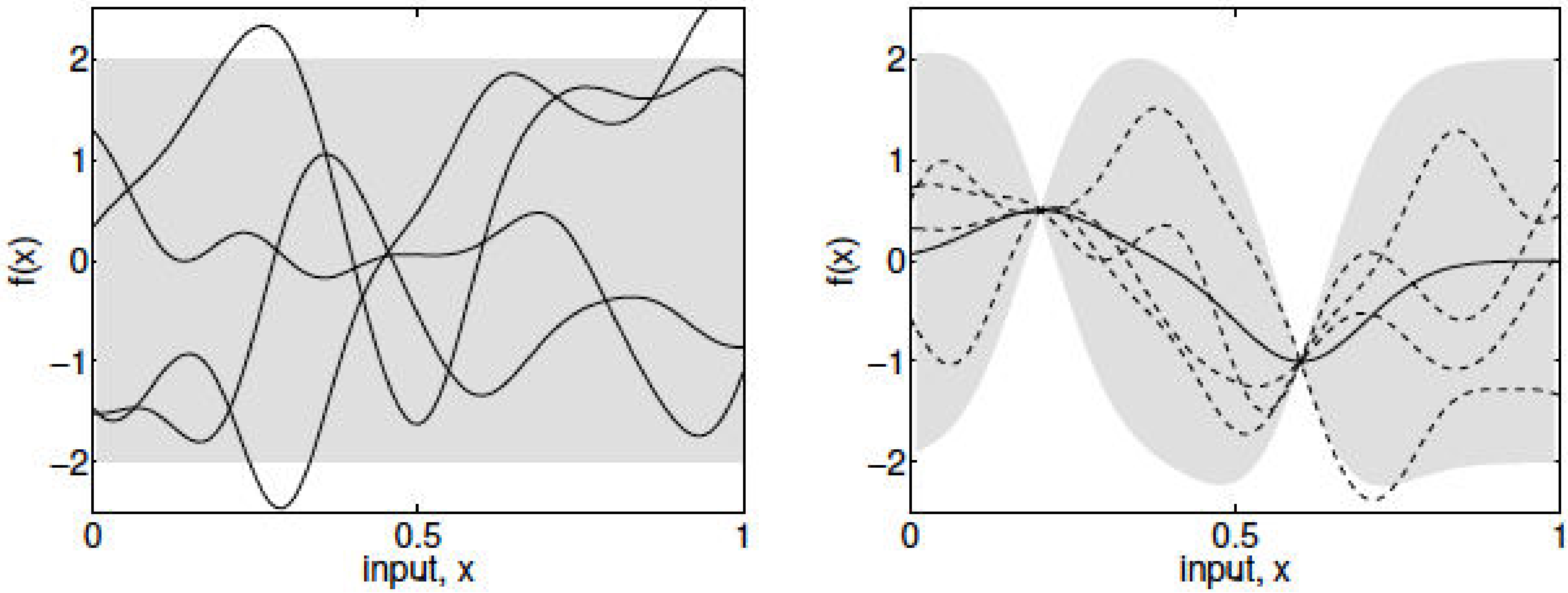
Adversarial point leaving the source class and moving towards the target class

Gaussian Processes

Consider an appropriate model, e.g linear

Draw weights from a prior Gaussian distribution

Consider only those functions that satisfy training constraints



(a), prior

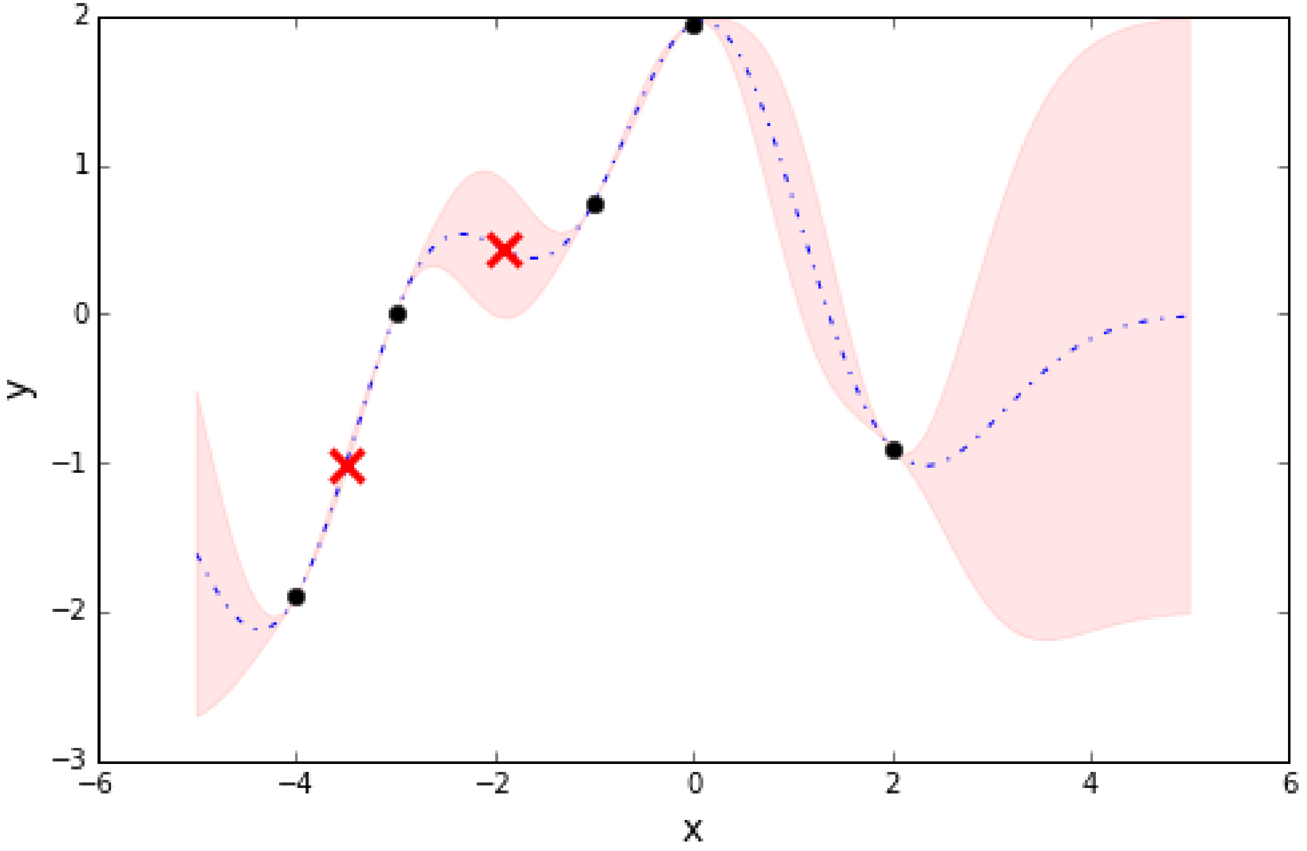
(b), posterior

Dropout as a Gaussian process

Dropout is an approximate Gaussian process

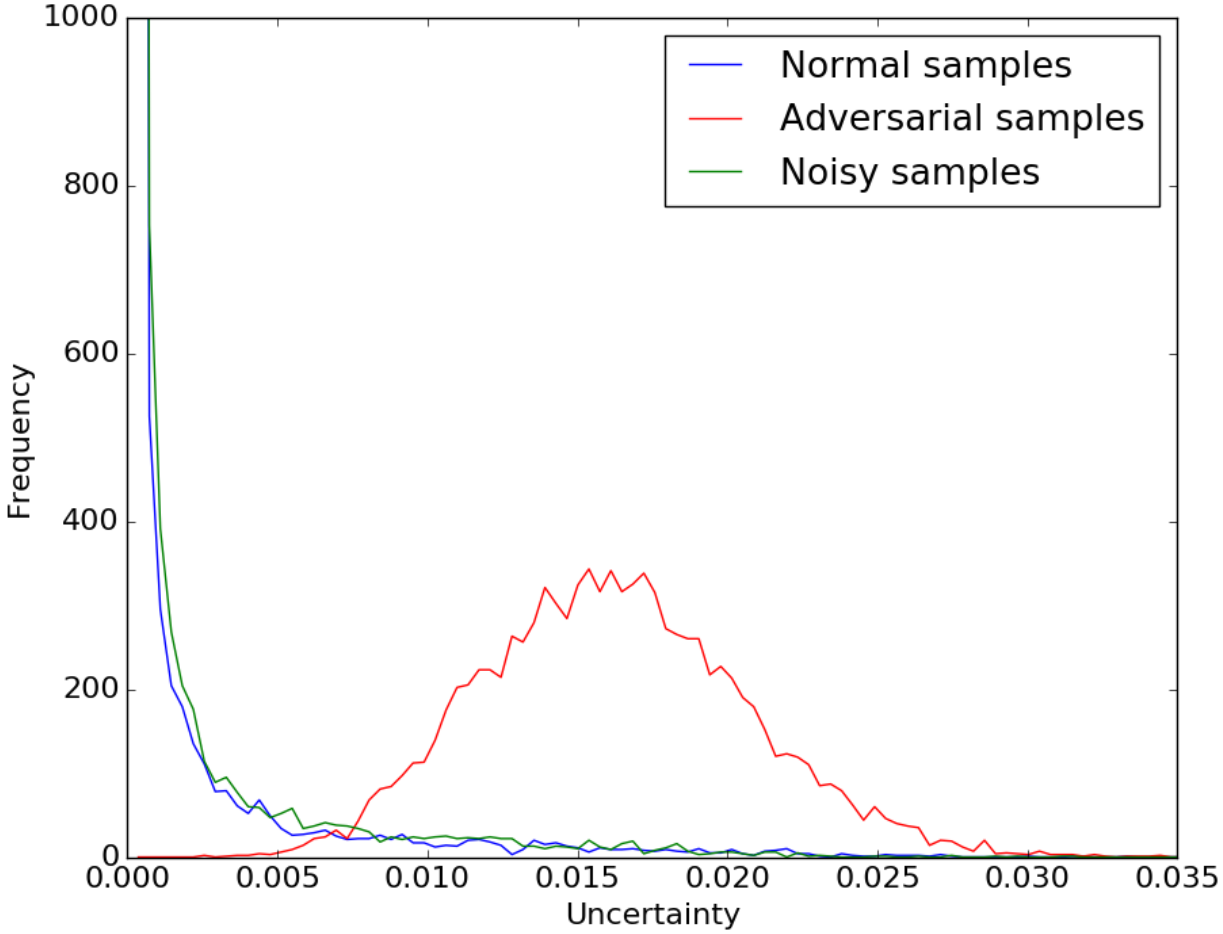
Explains why it prevents over-fitting

Variance of predictions is high when the model is extrapolating

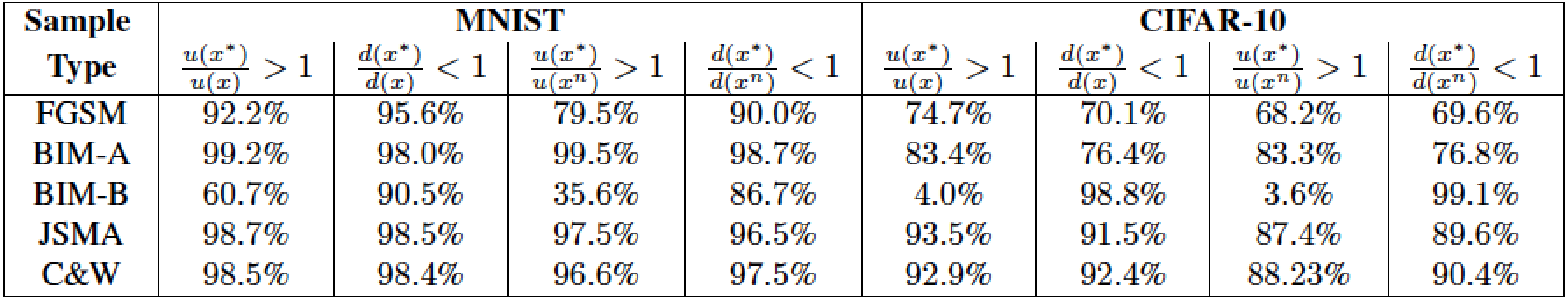


Artifacts of adversarial samples: Bayesian Uncertainty

- Run dropout during test time with *T=50* iterations
- Predicted value == mean prediction
- Bayesian uncertainty == variance of predictions



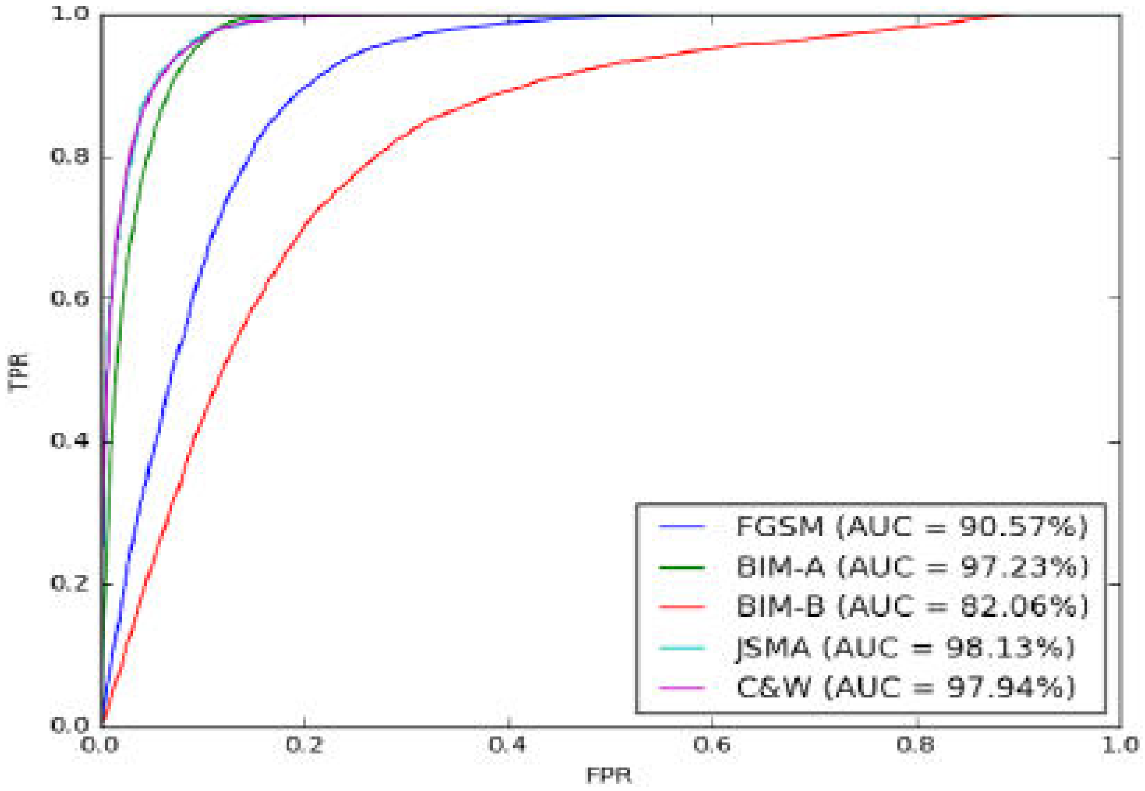
Artifacts of adversarial samples



Sample	MNIST				CIFAR-10			
Type	$\frac{u(x^*)}{u(x)} > 1$	$\frac{d(x^*)}{d(x)} < 1$	$\frac{u(x^*)}{u(x^n)} > 1$	$\frac{d(x^*)}{d(x^n)} < 1$	$\frac{u(x^*)}{u(x)} > 1$	$\frac{d(x^*)}{d(x)} < 1$	$\frac{u(x^*)}{u(x^n)} > 1$	$\frac{d(x^*)}{d(x^n)} < 1$
FGSM	92.2%	95.6%	79.5%	90.0%	74.7%	70.1%	68.2%	69.6%
BIM-A	99.2%	98.0%	99.5%	98.7%	83.4%	76.4%	83.3%	76.8%
BIM-B	60.7%	90.5%	35.6%	86.7%	4.0%	98.8%	3.6%	99.1%
JSMA	98.7%	98.5%	97.5%	96.5%	93.5%	91.5%	87.4%	89.6%
C&W	98.5%	98.4%	96.6%	97.5%	92.9%	92.4%	88.23%	90.4%

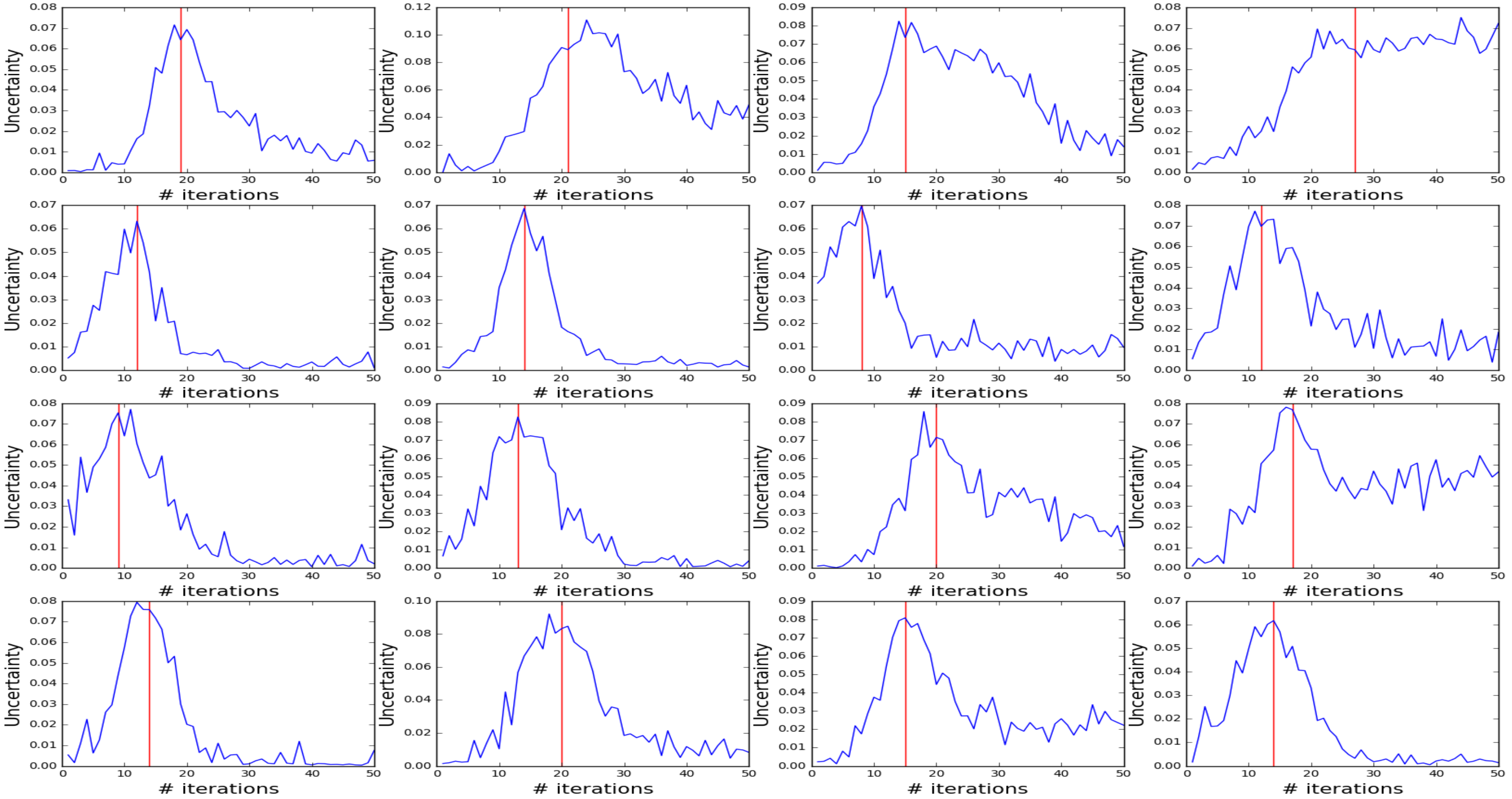
Building a detector

- Compute uncertainty and density estimates
- Build a two-feature logistic regression model



Breaking uncertainty

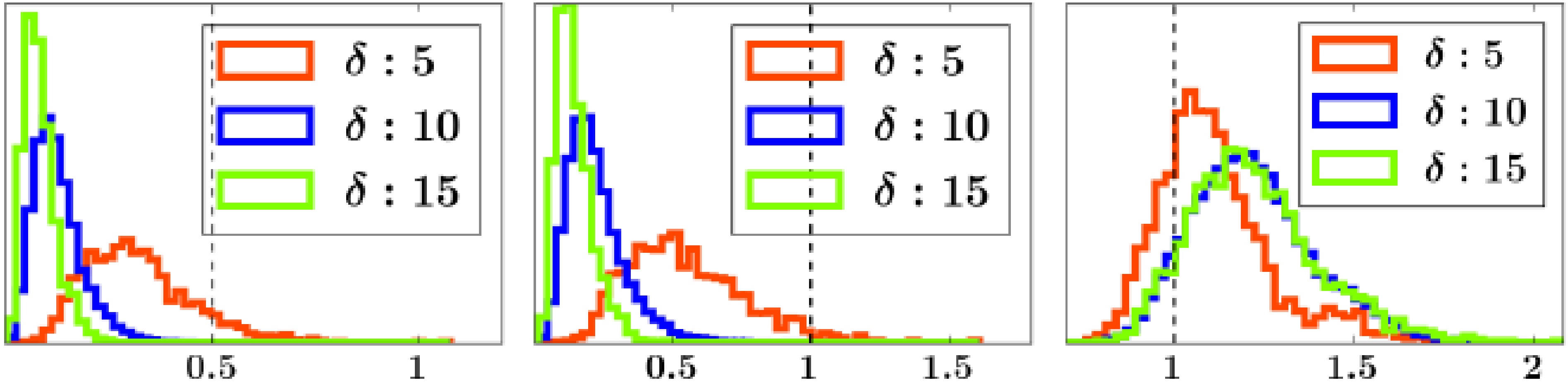
Uncertainty vs. # of FGSM iterations



Breaking density of deep manifold representations [Sabour et al.]

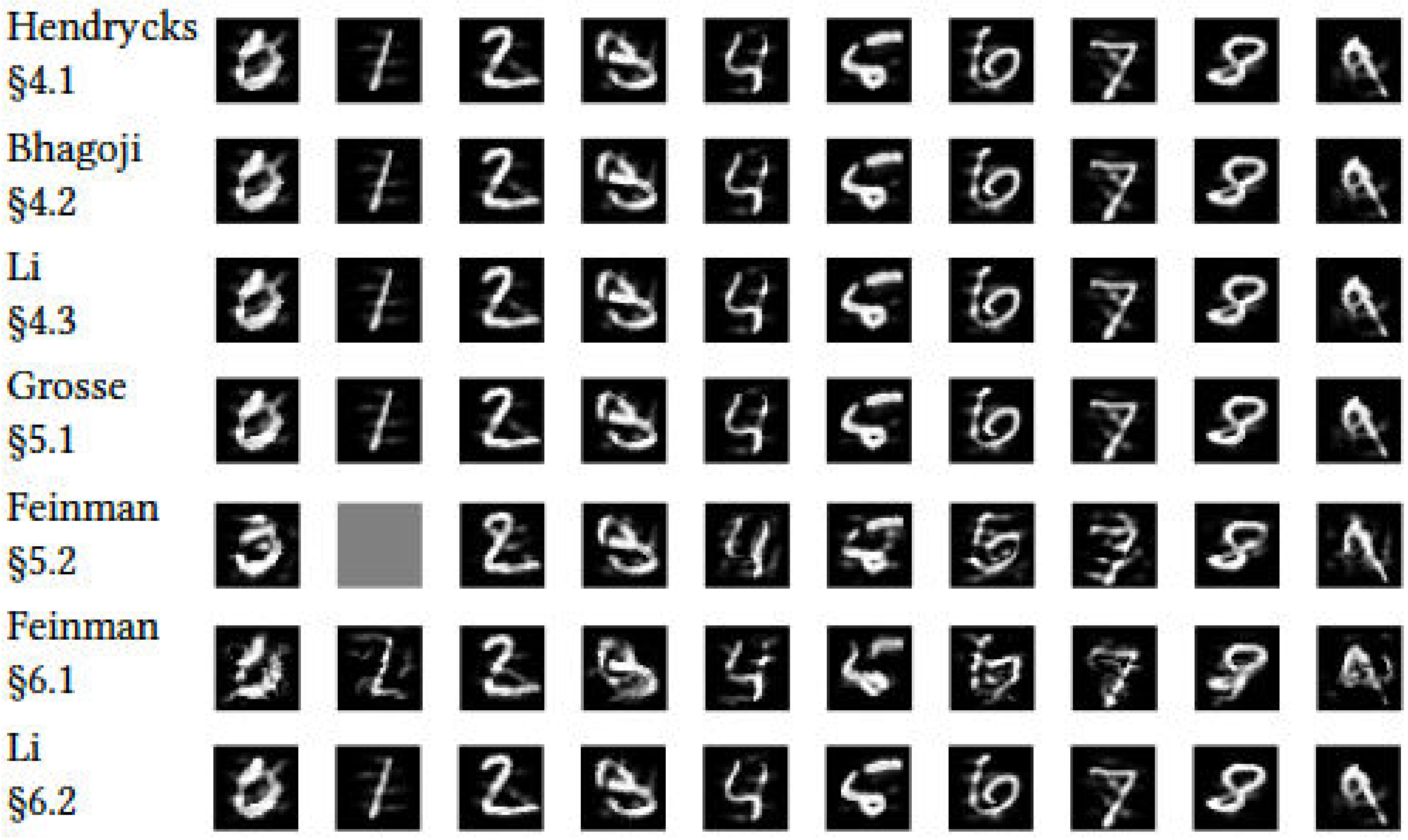
$$\min \| \ (I) \qquad (I_g) \|_2$$

$$\text{such that } \| I \quad I_s \| <$$



(a) $d(\alpha,\mathrm{g})/d(\mathrm{s},\mathrm{g})$ (b) $d(\alpha,\mathrm{g})/\overline{d_1}(\mathrm{g})$ (c) $d(\alpha,\mathrm{s})/\overline{d}(\mathrm{s})$

Comparing defenses



Future work

- Currently, no perfect defenses
- Robust optimization has been proposed as a provable defense
- We are currently working on an approach based on influence functions