

# DATA ANALYTICS USING DEEP LEARNING

GT 8803 // FALL 2019 // JOY ARULRAJ

LECTURE #20: ADVERSARIAL TRAINING

CREATING THE NEXT®

# ADMINISTRIVIA

---

- Reminders
  - Best project prize
  - Quiz cancelled
  - Guest lecture

# CREDITS

---

- Slides based on a lecture by:
  - Ian Goodfellow @ Google Brain



# OVERVIEW

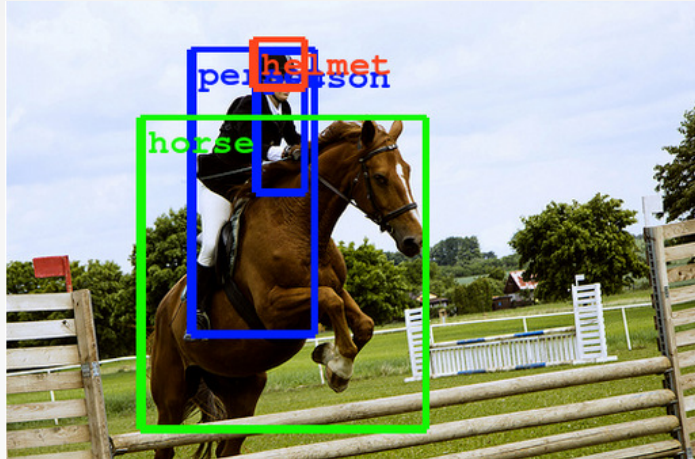
---

- What are adversarial examples?
- Why do they happen?
- How can they be used to compromise machine learning systems?
- What are the defenses?
- How to use adversarial examples to improve machine learning (even without adversary)?



# ADVERSARIAL EXAMPLES

Since 2013, deep neural networks have matched human performance at...



(Szegedy et al, 2014)

...recognizing objects and faces....



(Taigmen et al, 2013)



(Goodfellow et al, 2013)

...solving CAPTCHAS and reading addresses...

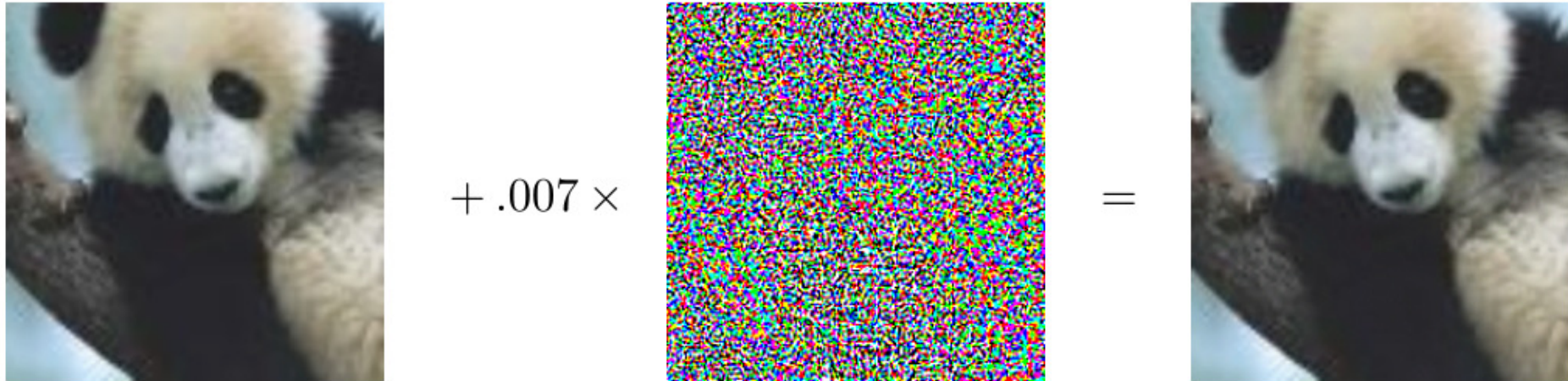


(Goodfellow et al, 2013)

and other tasks...

# ADVERSARIAL EXAMPLES

---



Timeline:

"Adversarial Classification" Dalvi et al 2004: fool spam filter

"Evasion Attacks Against Machine Learning at Test Time" Biggio

2013: fool neural nets

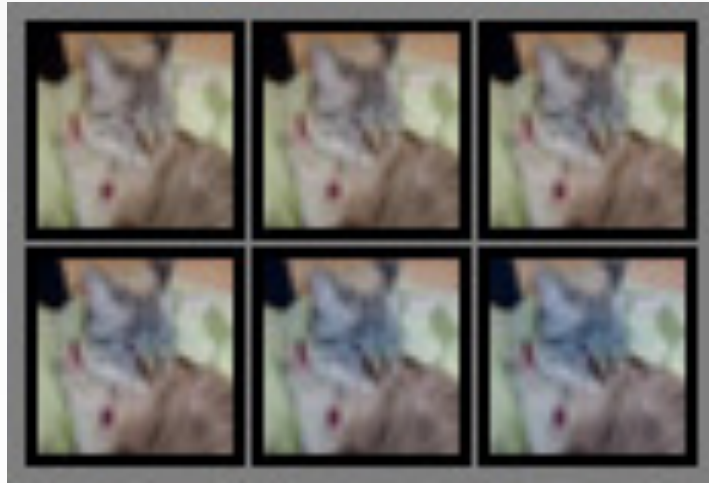
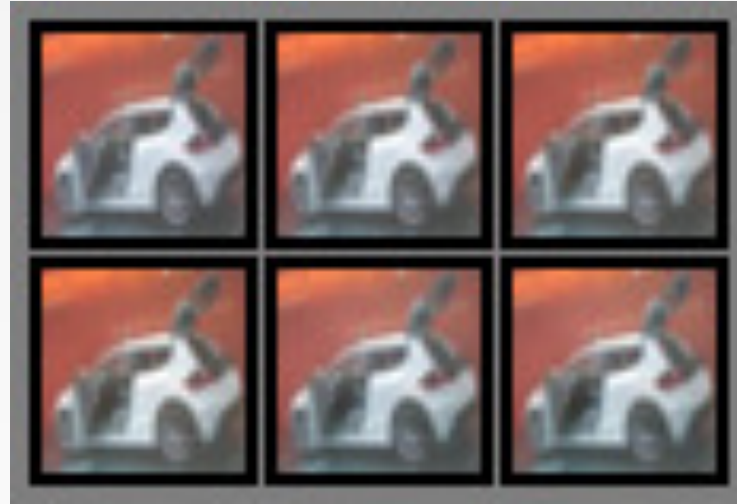
Szegedy et al 2013: fool ImageNet classifiers imperceptibly

Goodfellow et al 2014: cheap, closed form attack



# TURNING OBJECTS INTO “AIRPLANES”

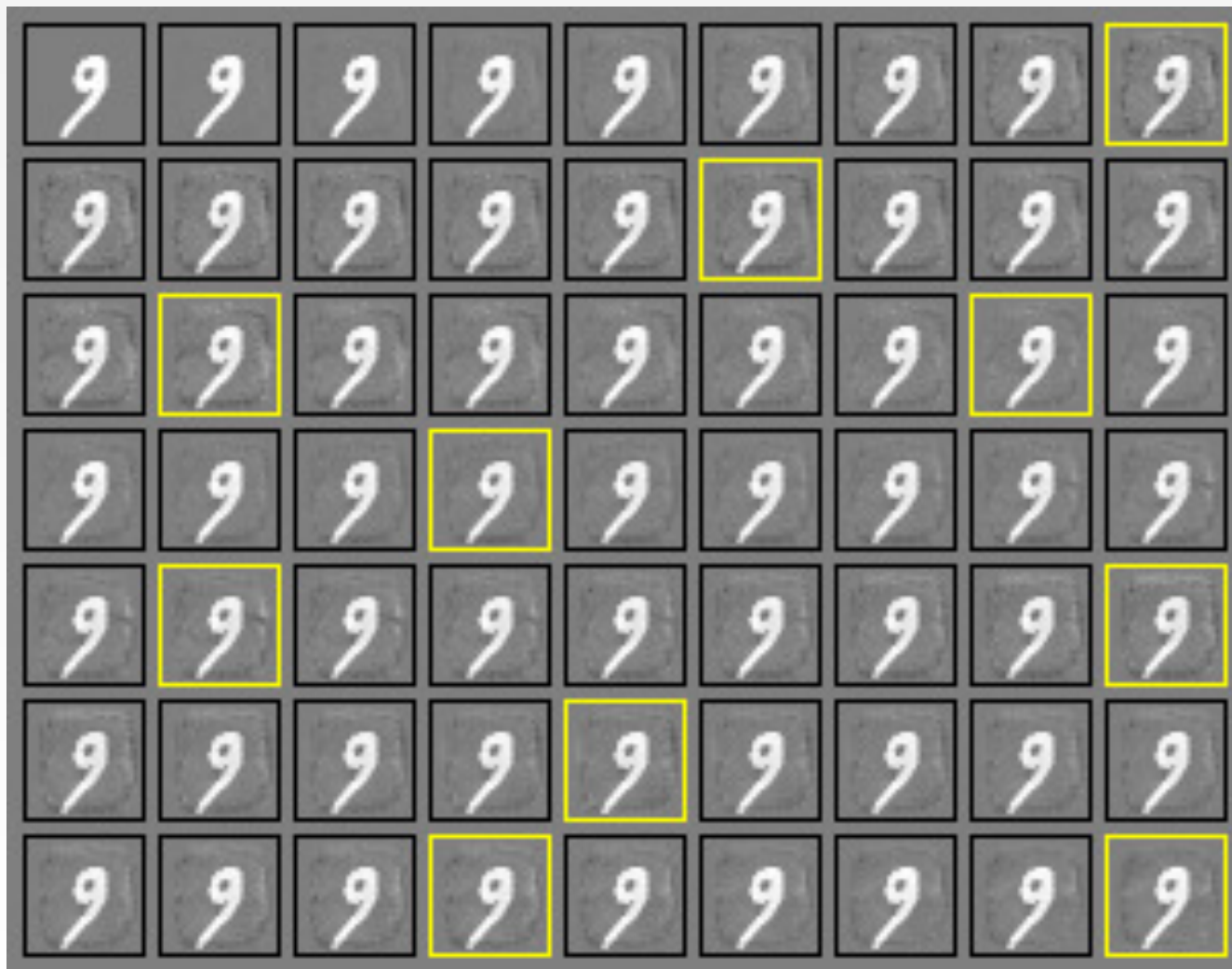
---





# ATTACKING A LINEAR MODEL

---



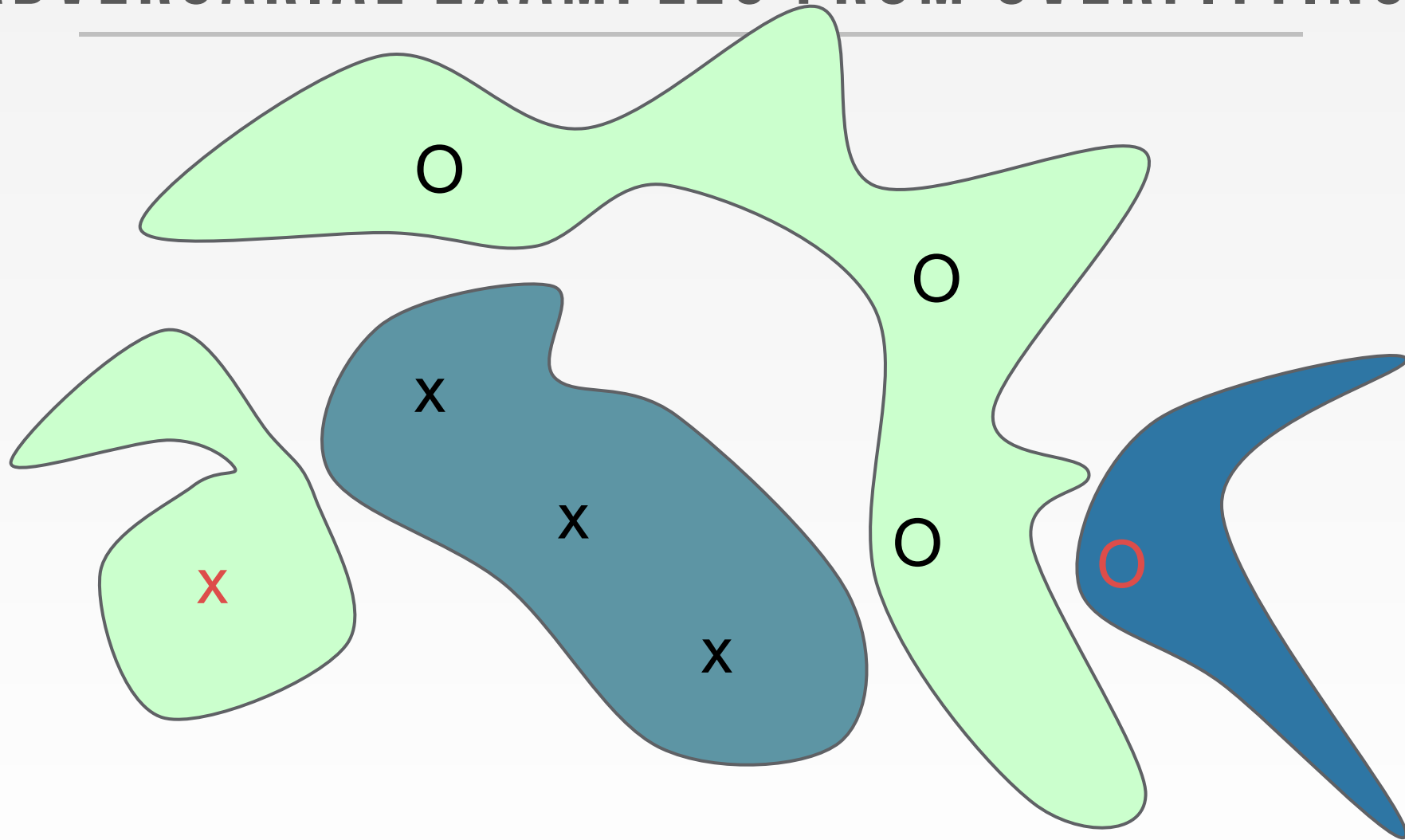
# NOT JUST FOR NEURAL NETS

---

- Linear models
  - Logistic regression
  - Softmax regression
  - SVMs
- Decision trees
- Nearest neighbors

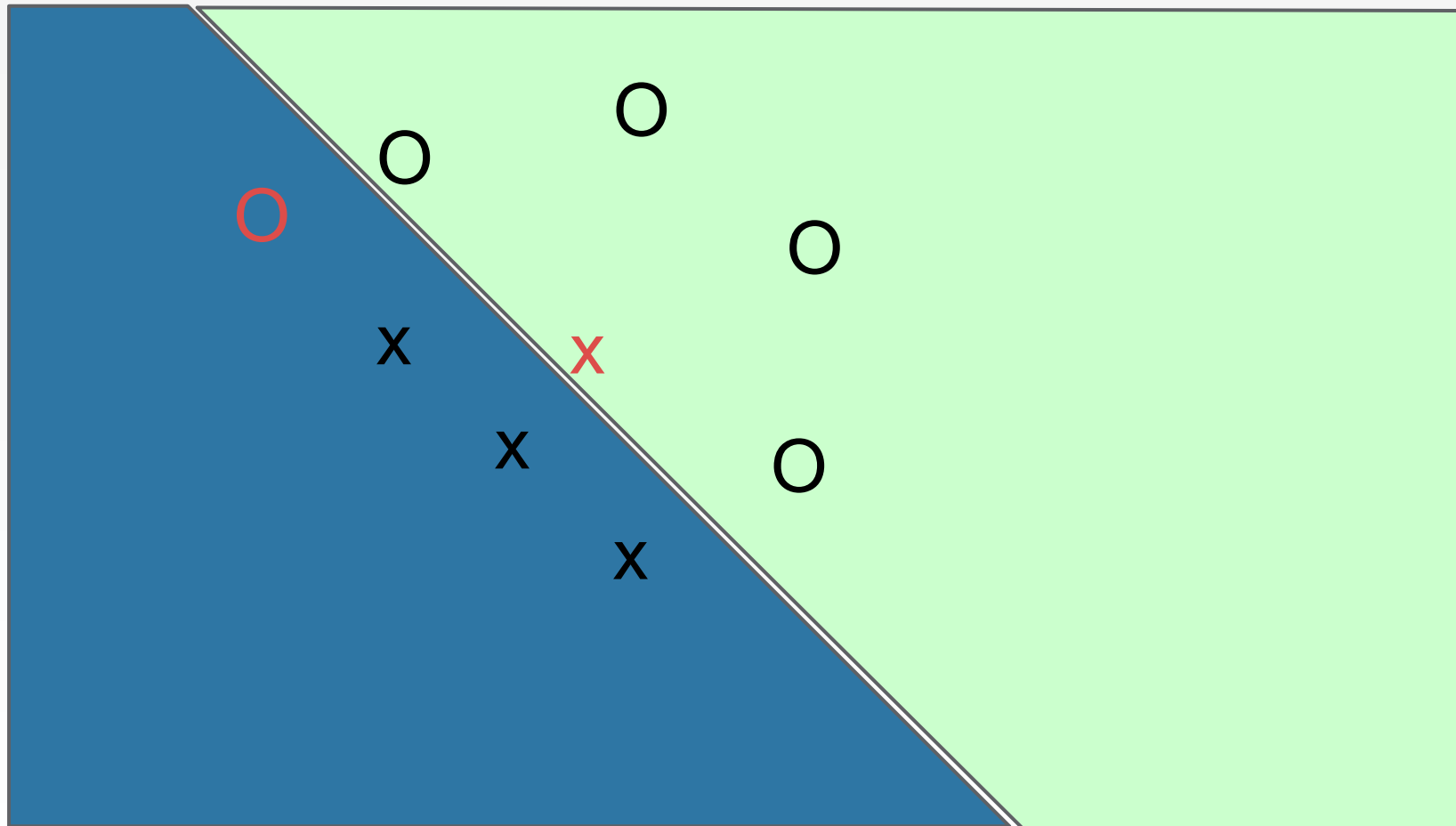
# ADVERSARIAL EXAMPLES FROM OVERFITTING

---



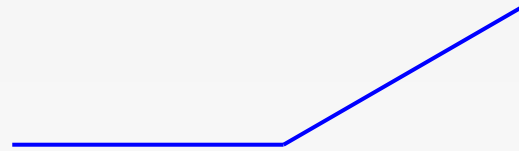
# ADVERSARIAL EXAMPLES FROM OVERFITTING

---

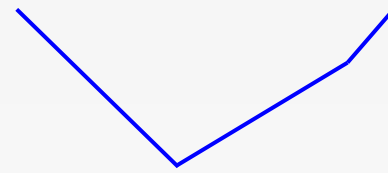


# MODERN DEEP NETS ARE VERY PIECEWISE LINEAR

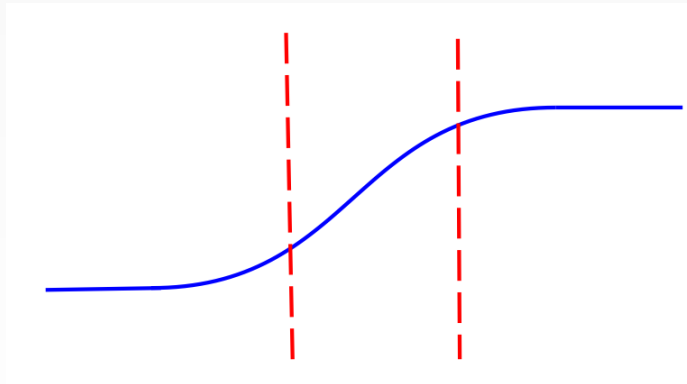
Rectified linear unit



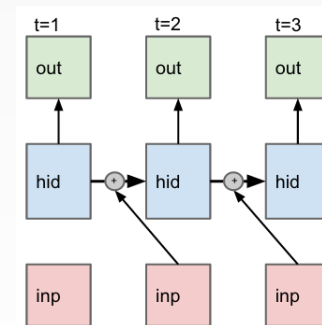
Maxout



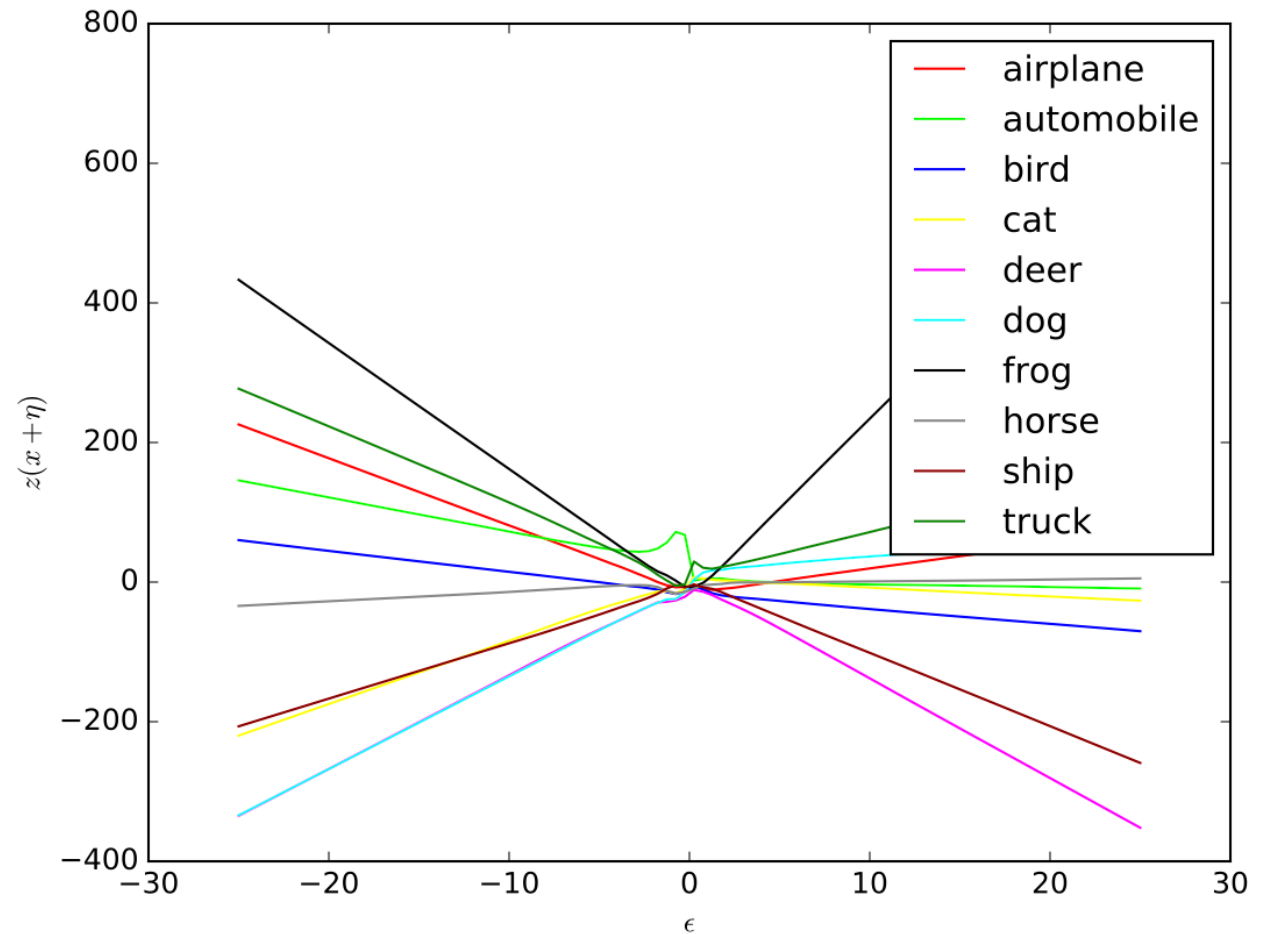
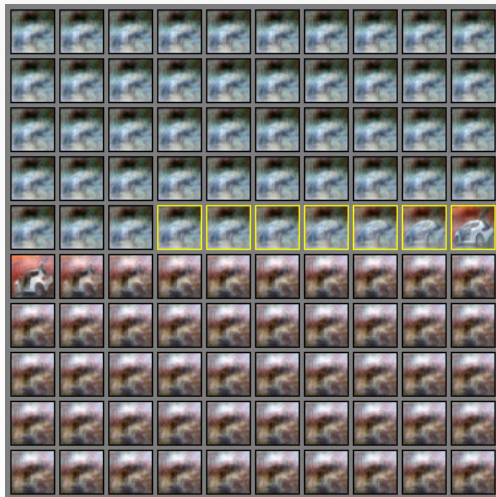
Carefully tuned sigmoid



LSTM












# NEARLY LINEAR RESPONSES IN PRACTICE





# SMALL INTER-CLASS DISTANCES

Clean example	Perturbation	Corrupted example	
			Perturbation changes the true class
			Random perturbation does not change the class
			Perturbation changes the input to "rubbish class"

All three perturbations have L2 norm 3.96

This is actually small. We typically use 7!

# THE FAST GRADIENT SIGN METHOD

---

$$J(\tilde{\mathbf{x}}, \boldsymbol{\theta}) \approx J(\mathbf{x}, \boldsymbol{\theta}) + (\tilde{\mathbf{x}} - \mathbf{x})^\top \nabla_{\mathbf{x}} J(\mathbf{x}).$$

Maximize

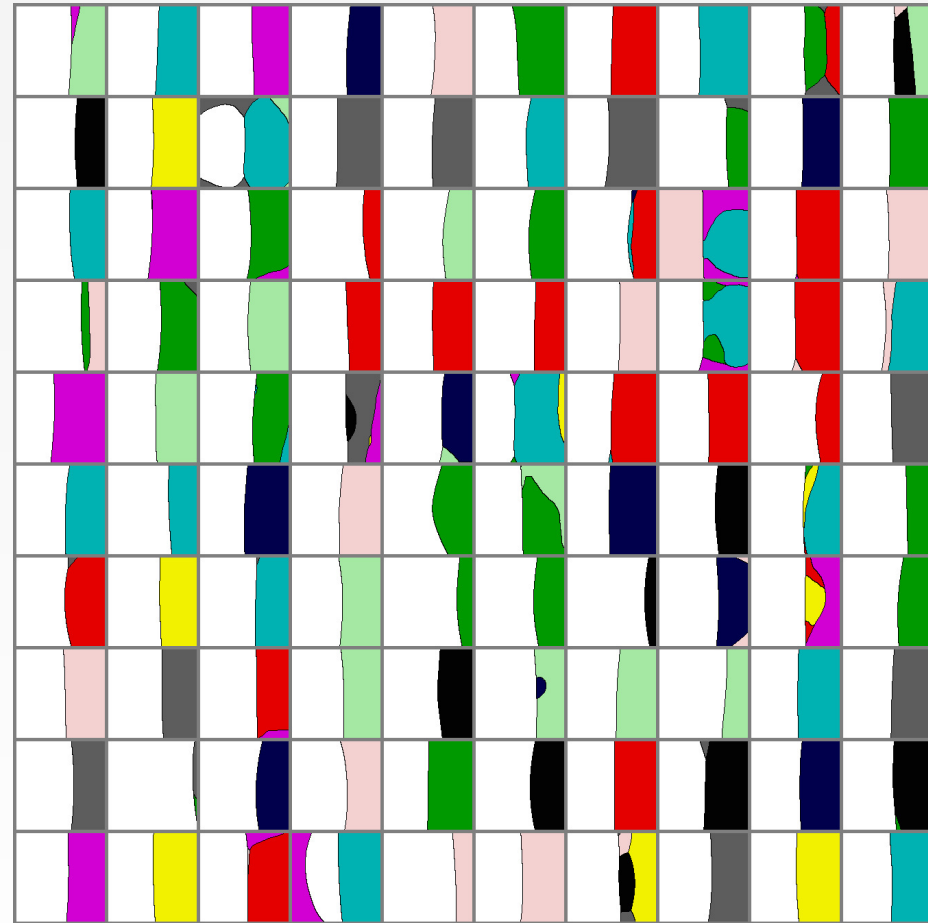
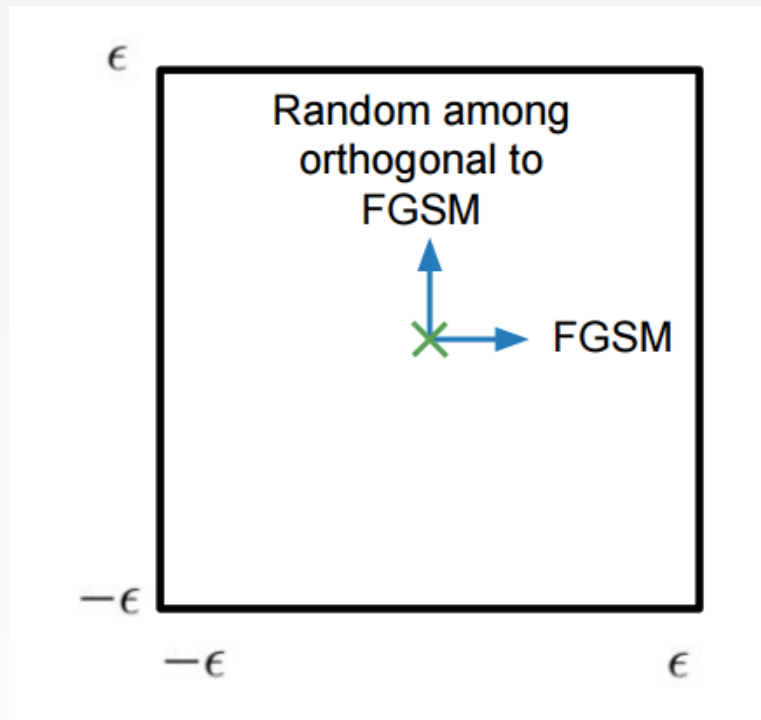
$$J(\mathbf{x}, \boldsymbol{\theta}) + (\tilde{\mathbf{x}} - \mathbf{x})^\top \nabla_{\mathbf{x}} J(\mathbf{x})$$

subject to

$$\|\tilde{\mathbf{x}} - \mathbf{x}\|_\infty \leq \epsilon$$

$$\Rightarrow \tilde{\mathbf{x}} = \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} J(\mathbf{x})).$$

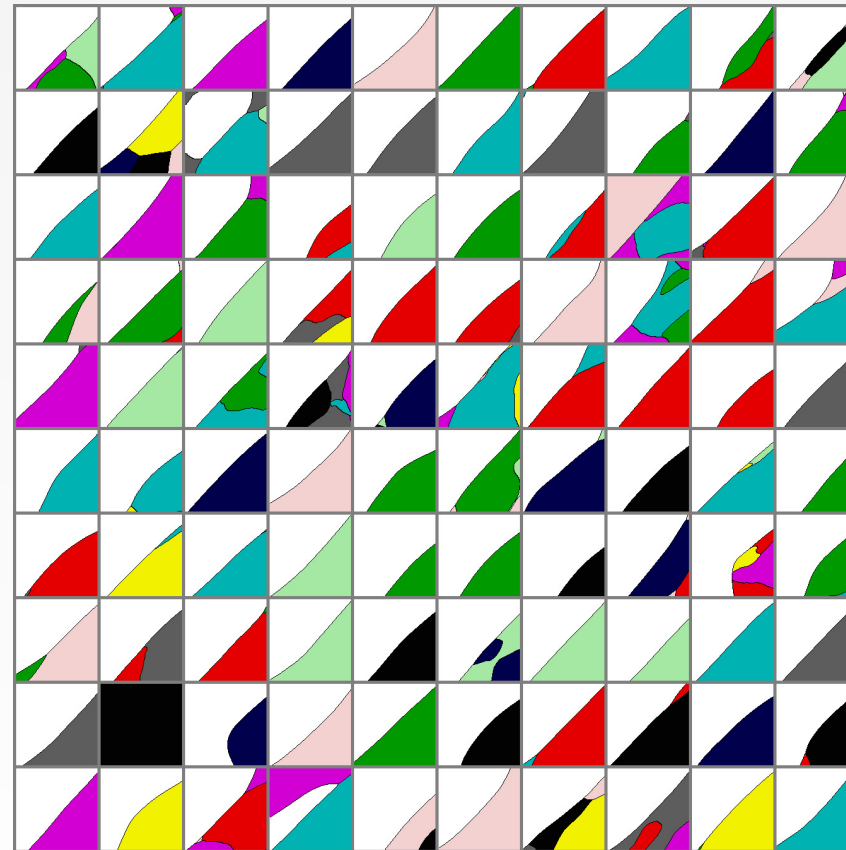
# MAPS OF ADVERSARIAL AND RANDOM CROSS-SECTIONS



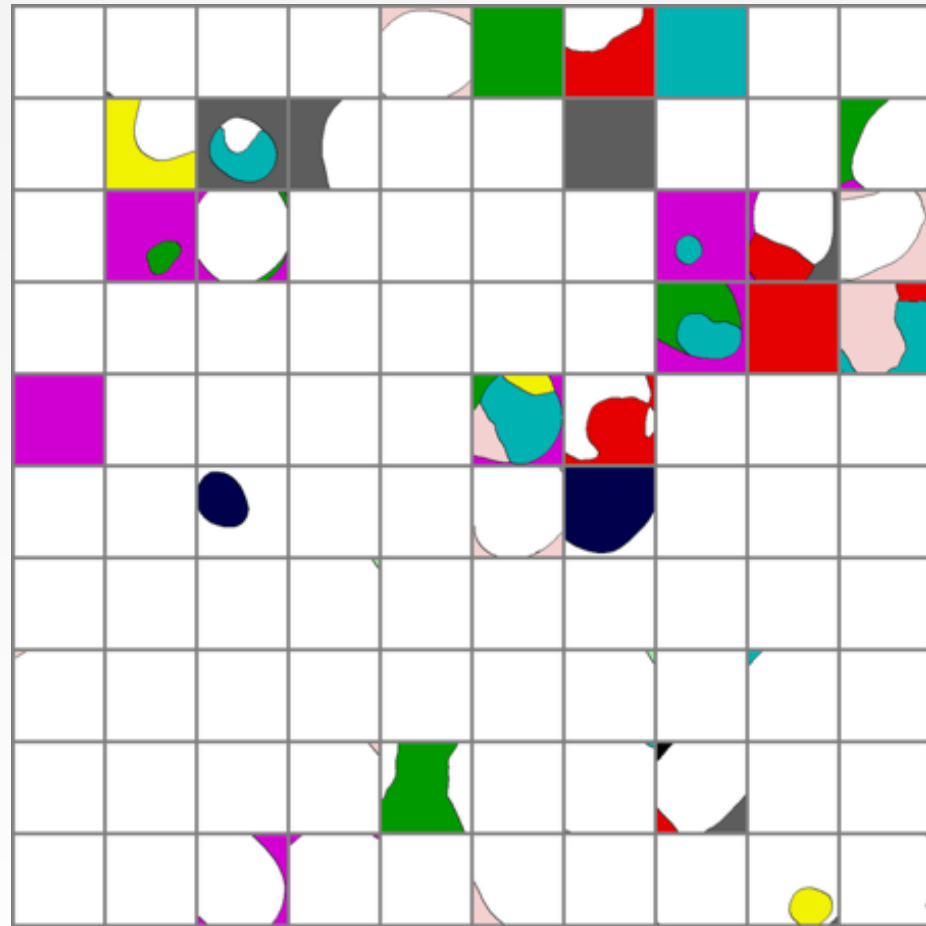
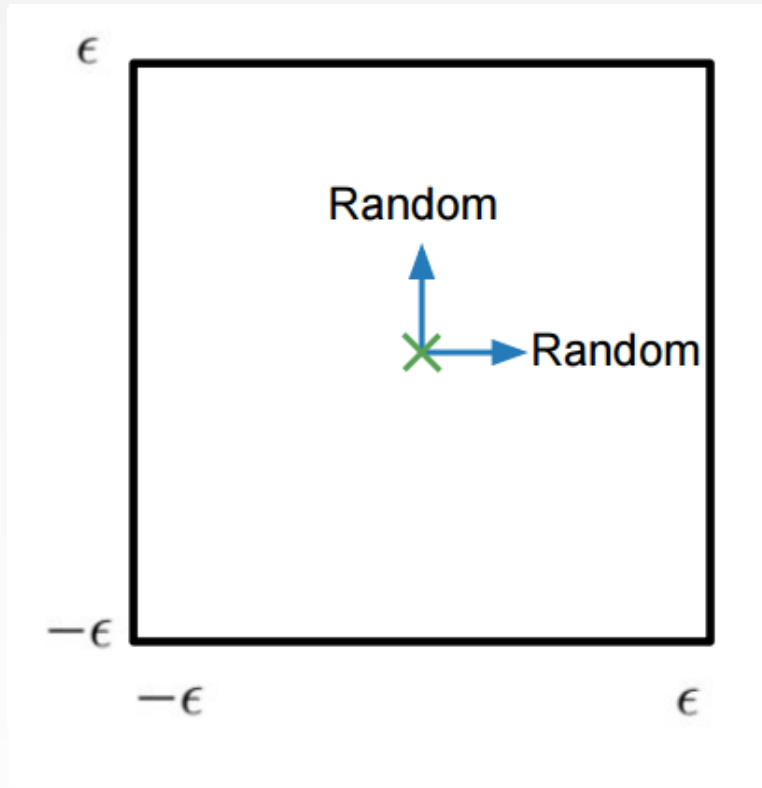
(collaboration with David Warde-Farley and Nicolas Papernot)

# MAPS OF ADVERSARIAL CROSS-SECTIONS

---

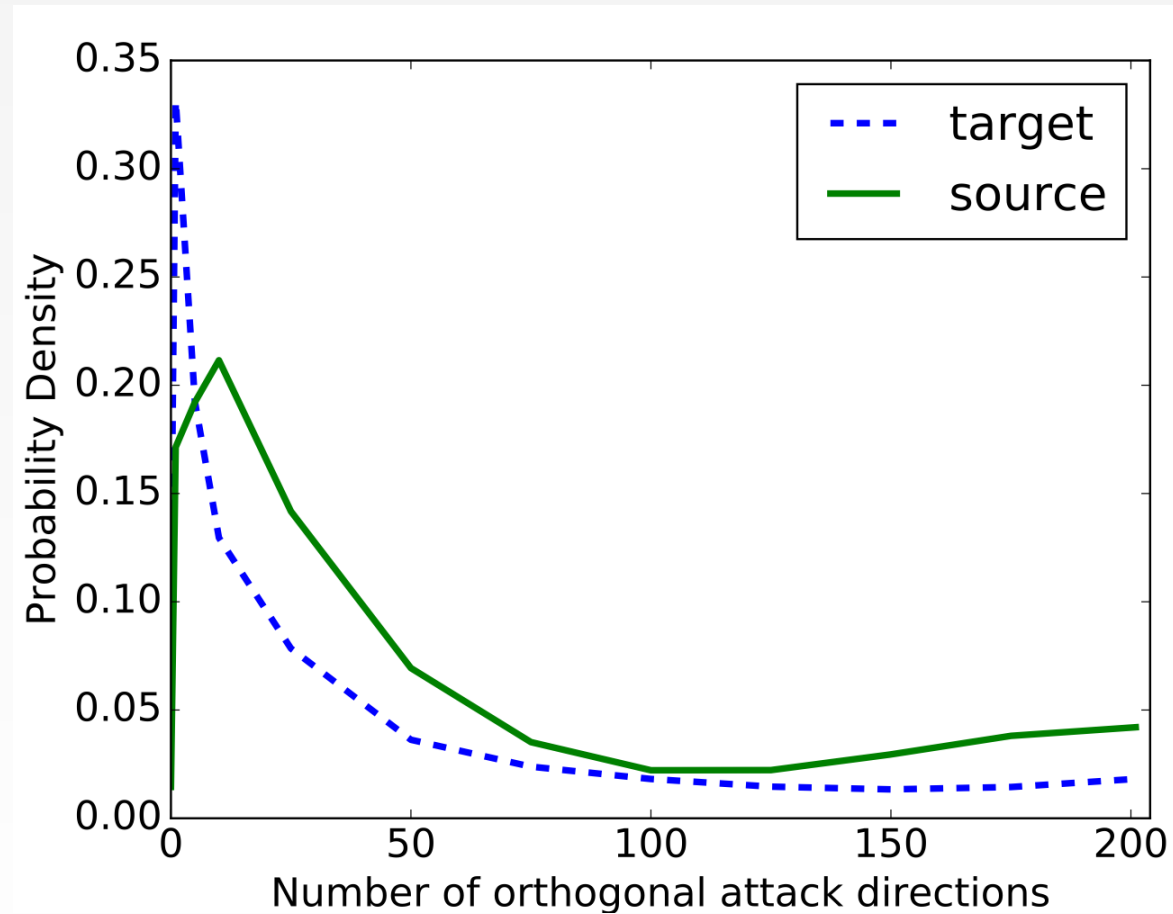


# MAPS OF RANDOM CROSS-SECTIONS



(collaboration with David Warde-Farley and Nicolas Papernot)

# ESTIMATING THE SUBSPACE DIMENSIONALITY





# CLEVER HANS

---

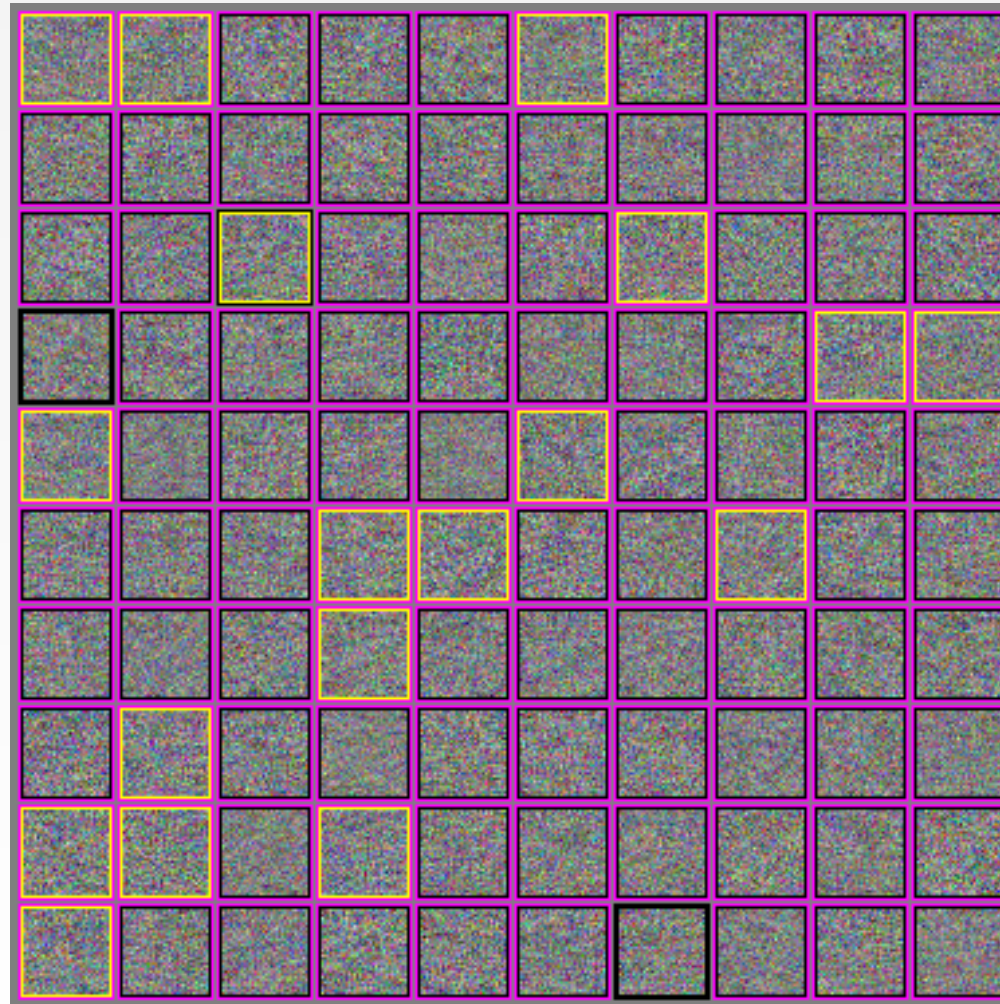


("Clever Hans,  
Clever Algorithms,"  
Bob Sturm)



# WRONG ALMOST EVERYWHERE

---

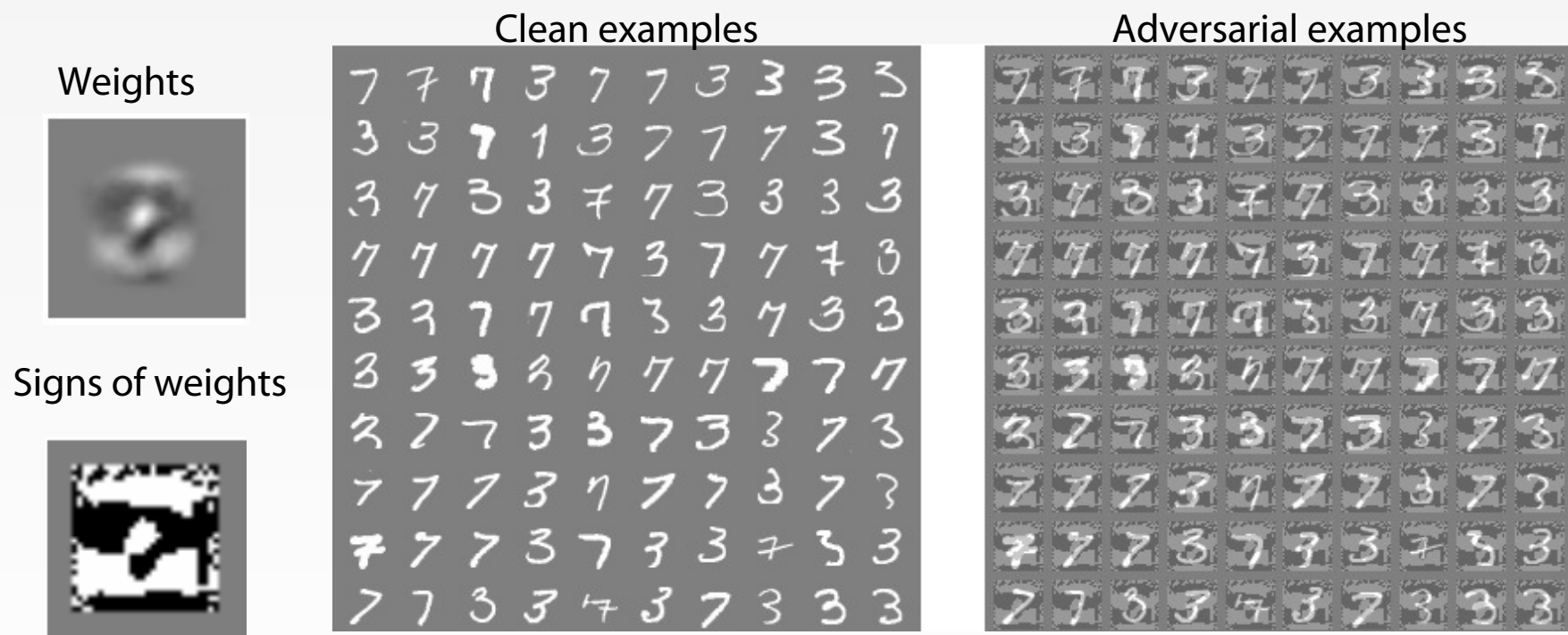


# ADVERSARIAL EXAMPLES FOR RL

The video player displays a comparison of test-time execution in the game Seaquest. The left side shows 'Test-Time Execution' with 'raw input' and 'output action distribution'. The right side shows 'Test-Time Execution with  $\ell_2$ -norm FGSM Adversary' with 'raw input', 'adversarial perturbation (unscaled)', 'adversarial input', and 'output action distribution'. The adversarial perturbation is a grayscale image with the mathematical expression  $\frac{\nabla_x J(\theta, x, y)}{\|\nabla_x J(\theta, x, y)\|_2}$  below it. The video player interface includes a progress bar at 0:05 / 5:40, a volume icon, and a red 'HD' logo. Below the video, the title is 'Adversarial Attacks: Seaquest, A3C, L2-Norm', the channel is 'Sandy Huang', and there is a 'Subscribe' button with a '7' notification. The view count is '6,295 views'.

([Huang et al.](#), 2017)

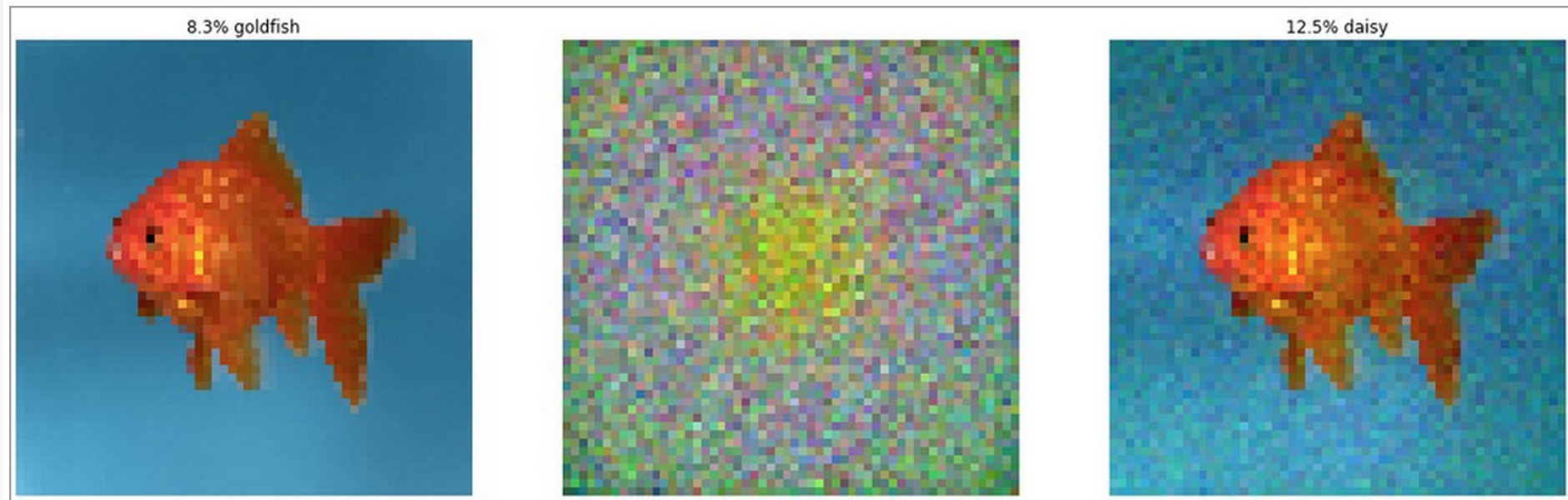
# HIGH-DIMENSIONAL LINEAR MODELS





# LINEAR MODELS OF IMAGENET

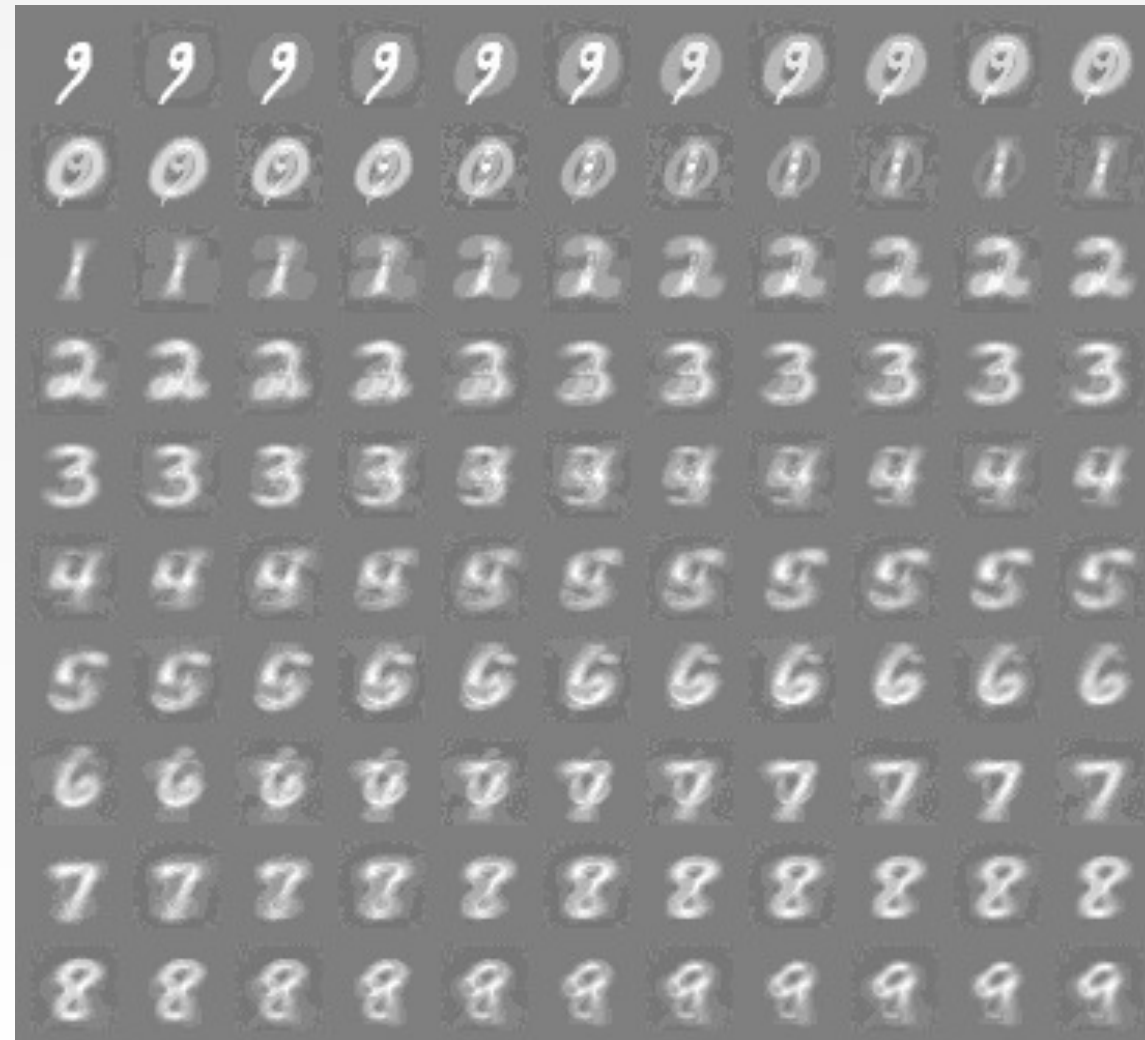
---



(Andrej Karpathy, "Breaking Linear Classifiers on ImageNet")

# RBFS BEHAVE MORE INTUITIVELY

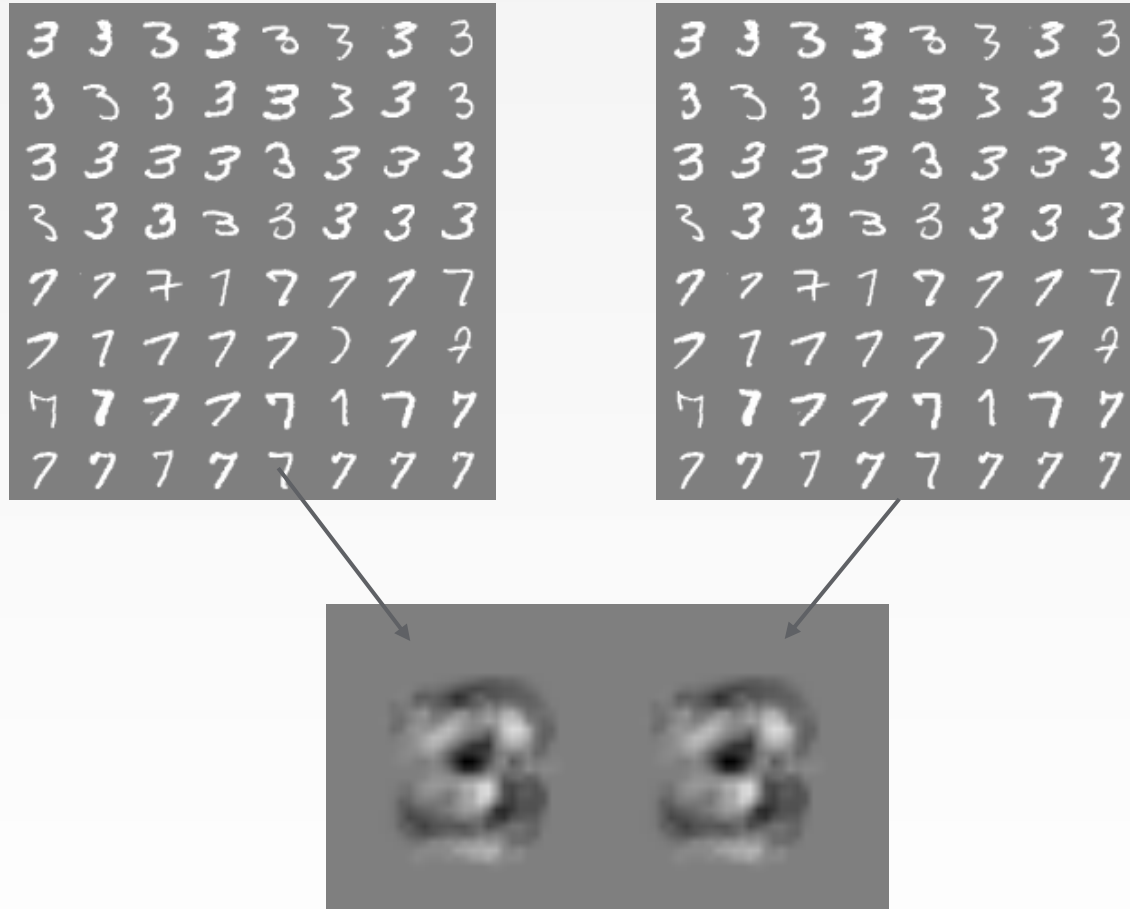
---





# CROSS-MODEL, CROSS-DATASET GENERALIZATION

---



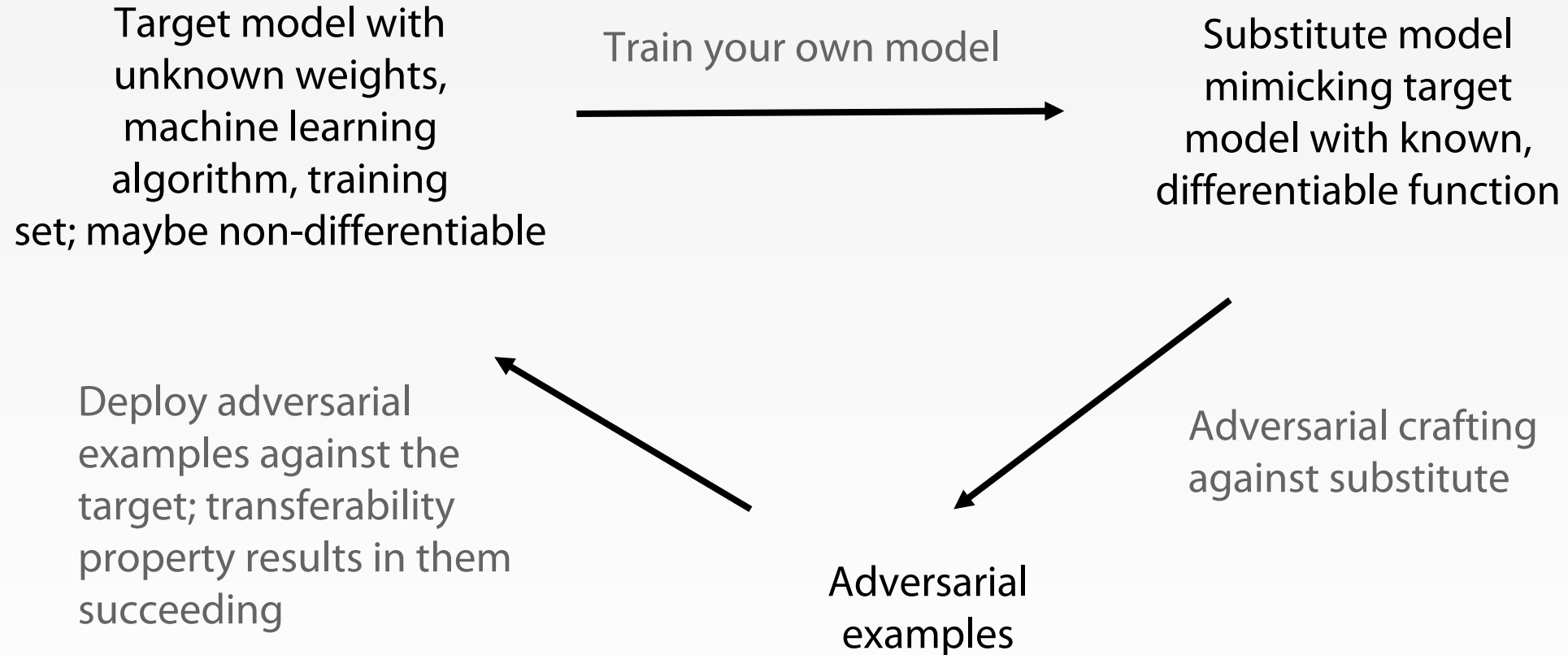
# CROSS-TECHNIQUE TRANSFERABILITY

Source Machine Learning Technique	DNN	LR	SVM	DT	kNN	Ens.
DNN	38.27	23.02	64.32	79.31	8.36	20.72
LR	6.31	91.64	91.43	87.42	11.29	44.14
SVM	2.51	36.56	100.0	80.03	5.19	15.67
DT	0.82	12.22	8.85	89.29	3.31	5.11
kNN	11.75	42.89	82.16	82.95	41.65	31.92

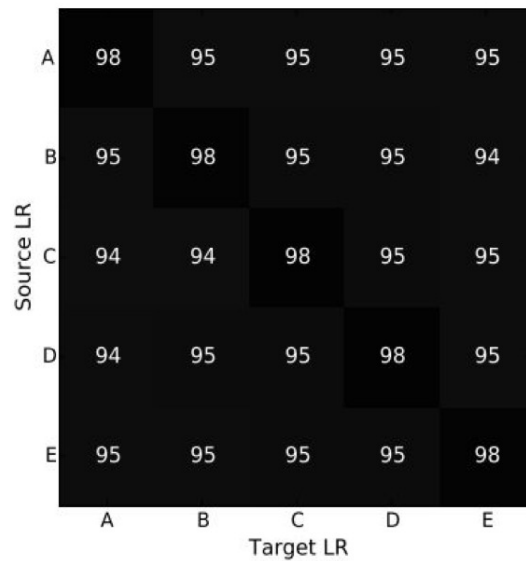
(Papernot 2016)

# TRANSFERABILITY ATTACK

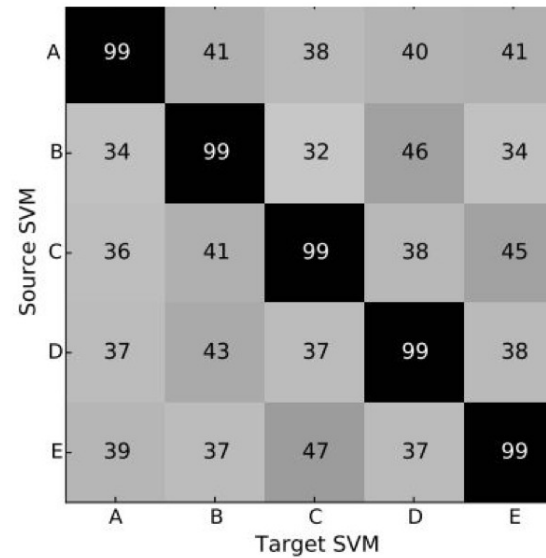
---



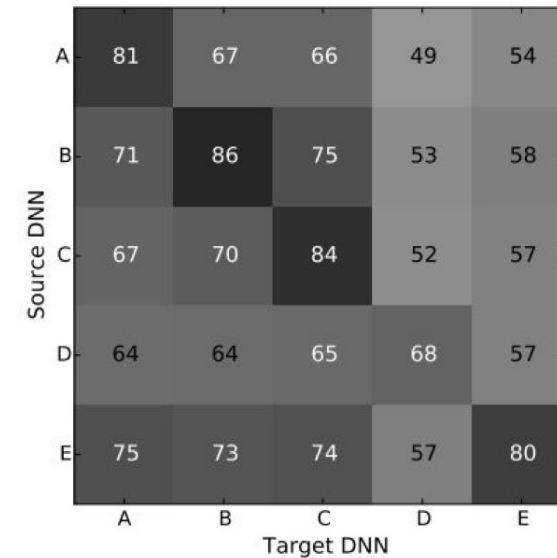
# Cross-Training Data Transferability



Strong



Weak



Intermediate

(Papernot 2016)

# ENHANCING TRANSFER WITH ENSEMBLES

---

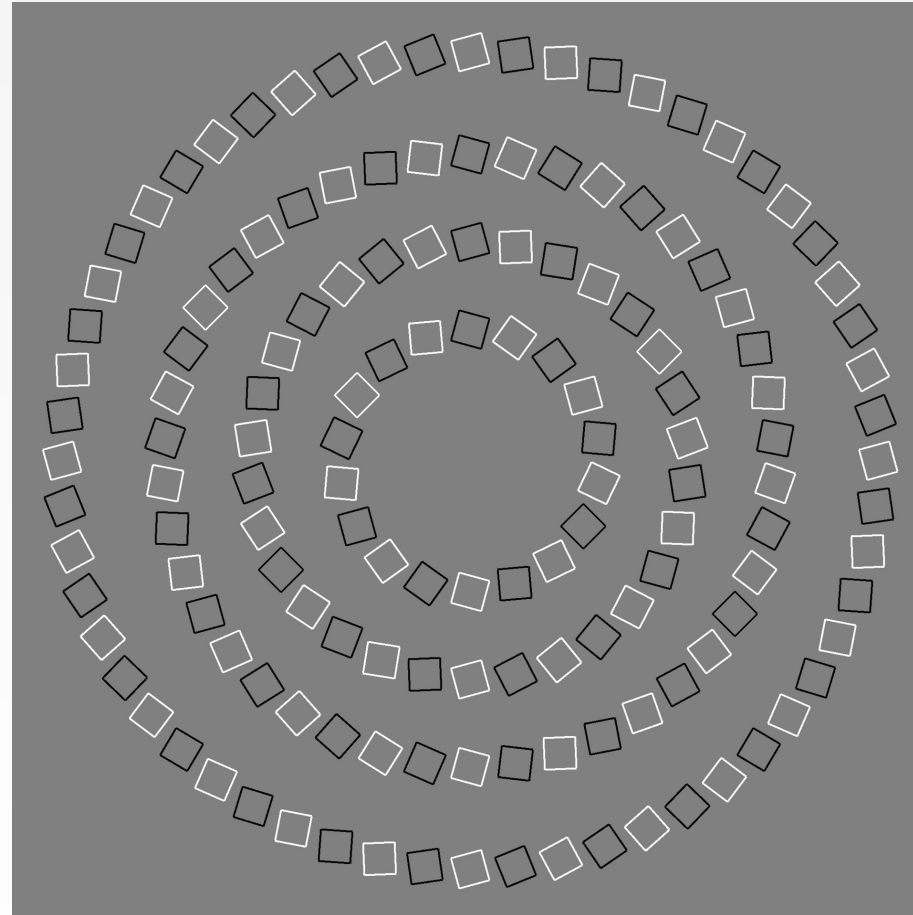
	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
-ResNet-152	17.17	0%	0%	0%	0%	0%
-ResNet-101	17.25	0%	1%	0%	0%	0%
-ResNet-50	17.25	0%	0%	2%	0%	0%
-VGG-16	17.80	0%	0%	0%	6%	0%
-GoogLeNet	17.41	0%	0%	0%	0%	5%

Table 4: Accuracy of non-targeted adversarial images generated using the optimization-based approach. The first column indicates the average RMSD of the generated adversarial images. Cell  $(i, j)$  corresponds to the accuracy of the attack generated using four models except model  $i$  (row) when evaluated over model  $j$  (column). In each row, the minus sign “-” indicates that the model of the row is not used when generating the attacks. Results of top-5 accuracy can be found in the appendix (Table 14).

(Liu et al, 2016)

# ADVERSARIAL EXAMPLES IN THE HUMAN BRAIN

---



These are concentric circles,  
not intertwined  
spirals.

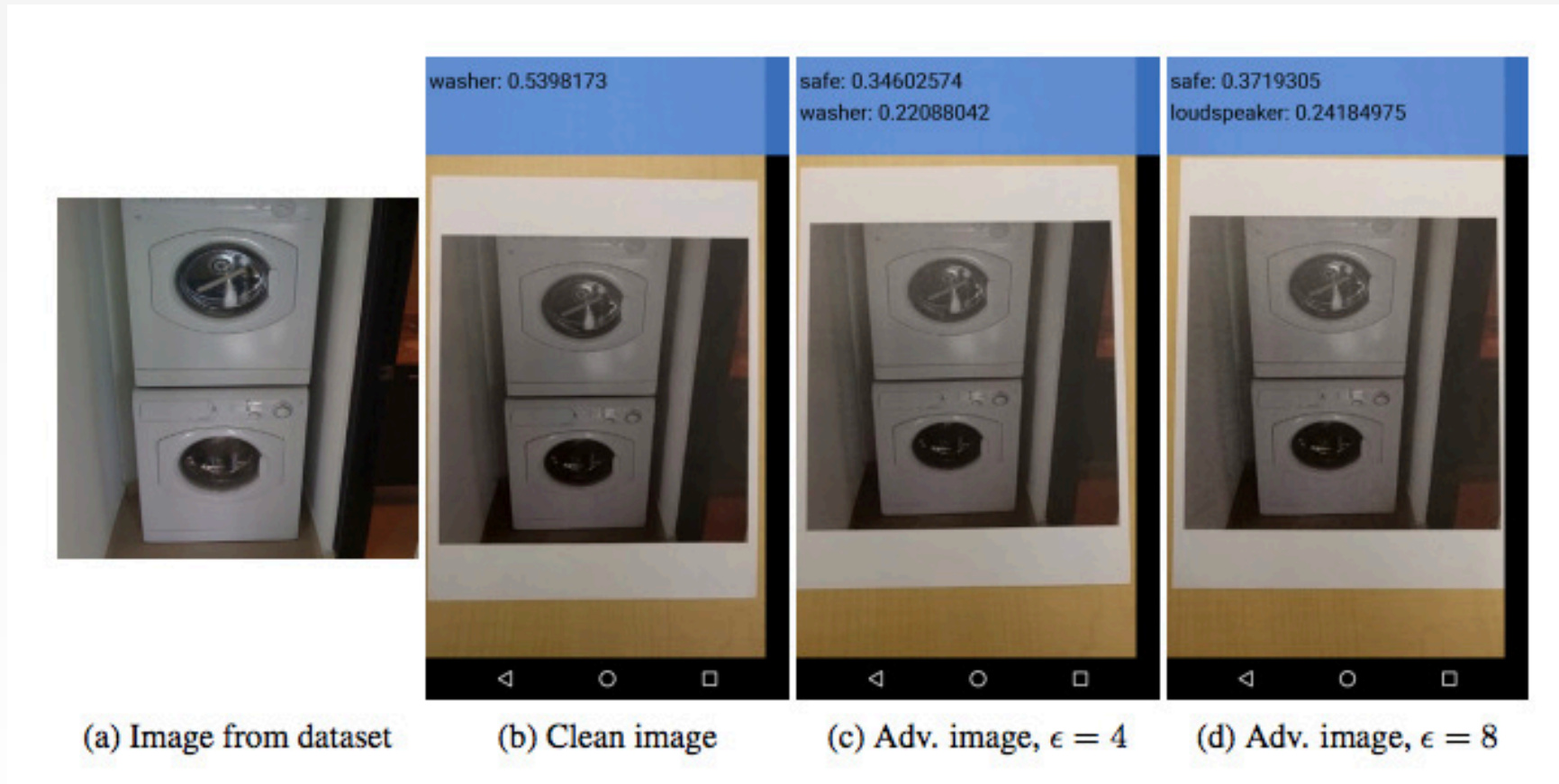
(Pinna and Gregory, 2002)

# PRACTICAL ATTACKS

---

- Fool real classifiers trained by remotely hosted API (MetaMind, Amazon, Google)
- Fool malware detector networks
- Display adversarial examples in the physical world and fool machine learning systems that perceive them through a camera

# ADVERSARIAL EXAMPLES IN THE PHYSICAL WORLD





# FAILED DEFENSES

---

GENERATIVE  
PRETRAINING

REMOVING PERTURBATION  
WITH AN AUTOENCODER

ADDING NOISE  
AT TEST TIME

ENSEMBLES

CONFIDENCE-REDUCING  
PERTURBATION AT TEST TIME

ERROR CORRECTING  
CODES

MULTIPLE GLIMPSES

WEIGHT DECAY

DOUBLE BACKPROP

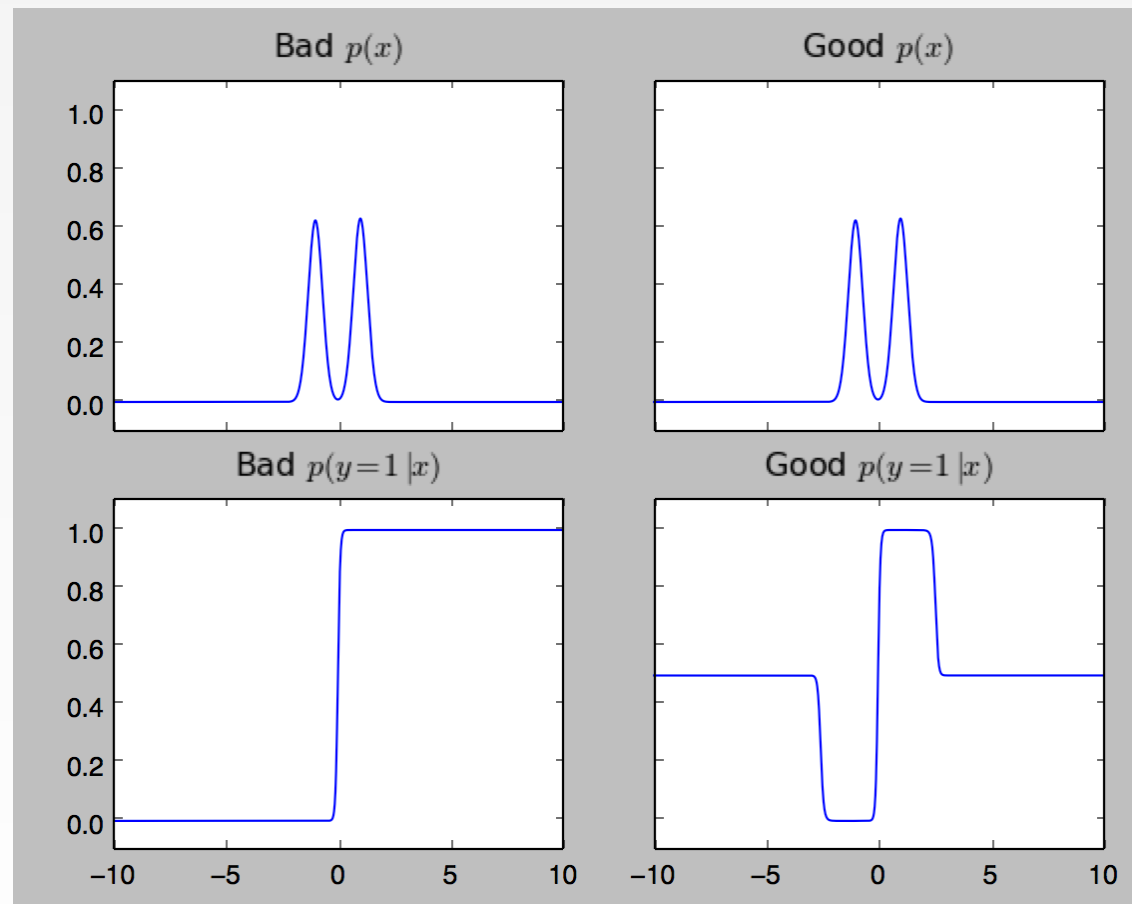
ADDING NOISE  
AT TRAIN TIME

VARIOUS  
NON-LINEAR UNITS

DROPOUT

# GENERATIVE MODELING IS NOT SUFFICIENT

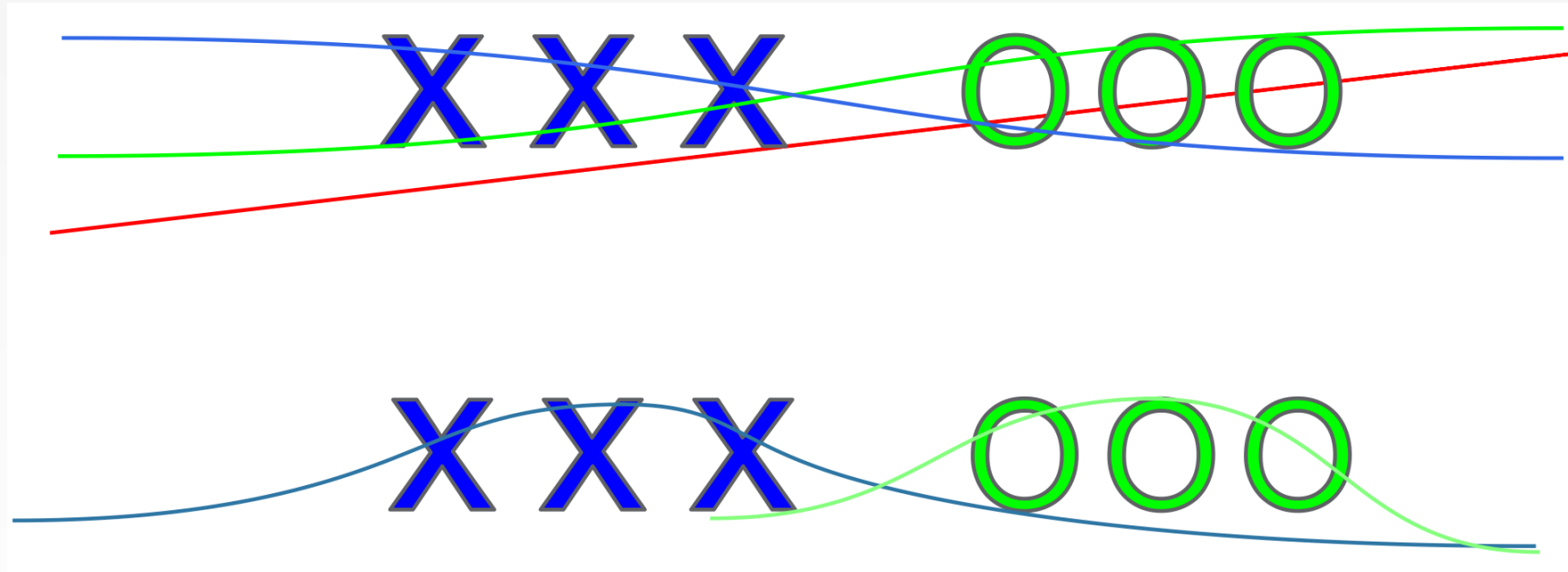
---



# UNIVERSAL APPROXIMATOR THEOREM

---

Neural nets can represent either function:

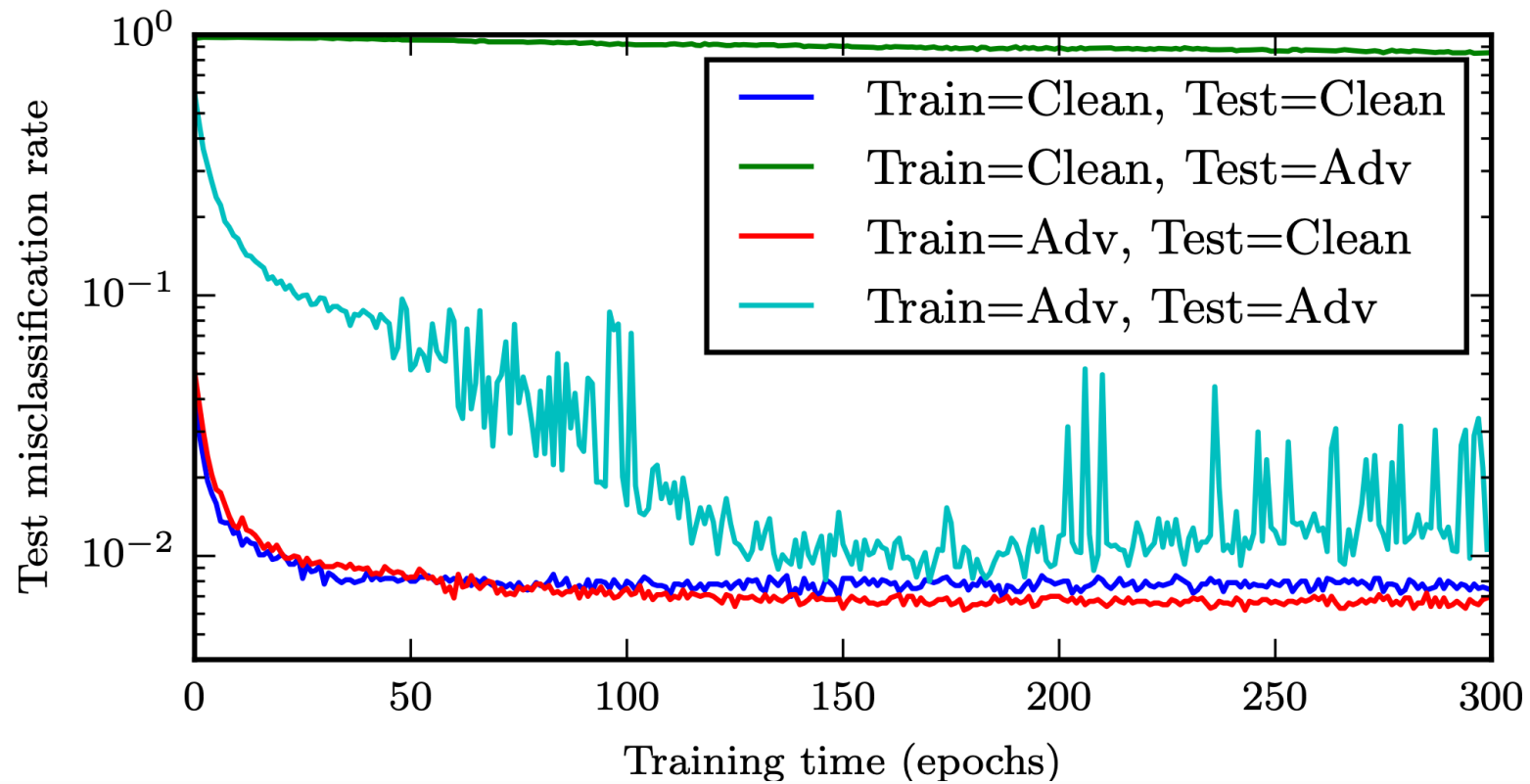


Maximum likelihood doesn't cause them to learn the right function. But we can fix that...



# ADVERSARIAL TRAINING

# TRAINING ON ADVERSARIAL EXAMPLES



# ADVERSARIAL TRAINING OF OTHER MODELS

---

- Linear models: SVM / linear regression cannot learn a step function, so adversarial training is less useful, very similar to weight decay
- *k-NN: adversarial training is prone to overfitting.*
- *Takeway: neural nets can actually become more secure than other models. Adversarially trained neural nets have the best empirical success rate on adversarial examples of any machine learning model.*

# WEAKNESSES PERSIST

---





# ADVERSARIAL TRAINING

---

Labeled as bird



Still has same label (bird)



Decrease  
probability  
of bird class



# VIRTUAL ADVERSARIAL TRAINING

Unlabeled; model  
guesses it's probably  
a bird, maybe a plane



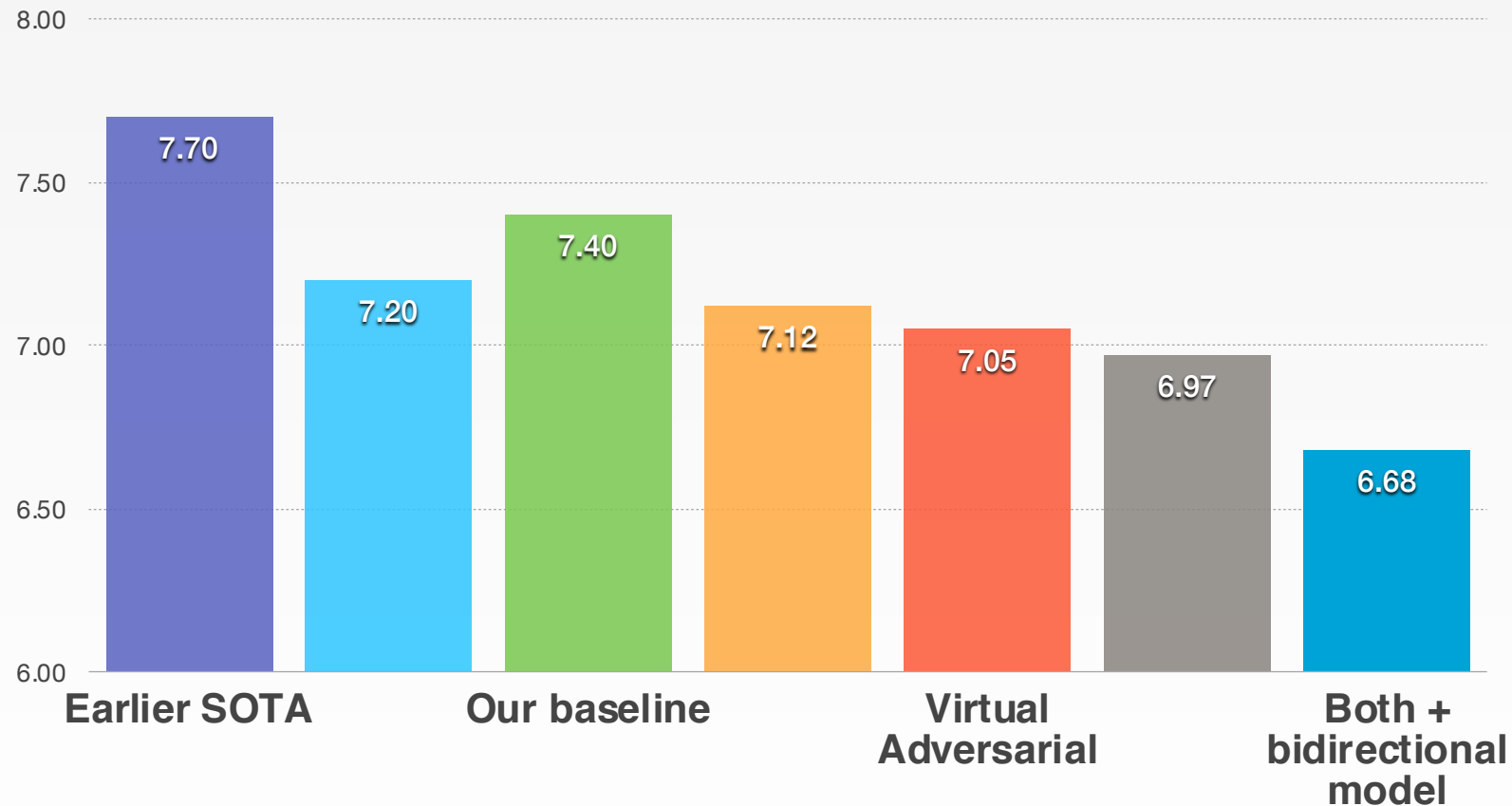
New guess should  
match old guess  
(probably bird, maybe plane)



Adversarial  
perturbation  
intended to  
change the guess

# TEXT CLASSIFICATION WITH VAT

## RCV1 Misclassification Rate

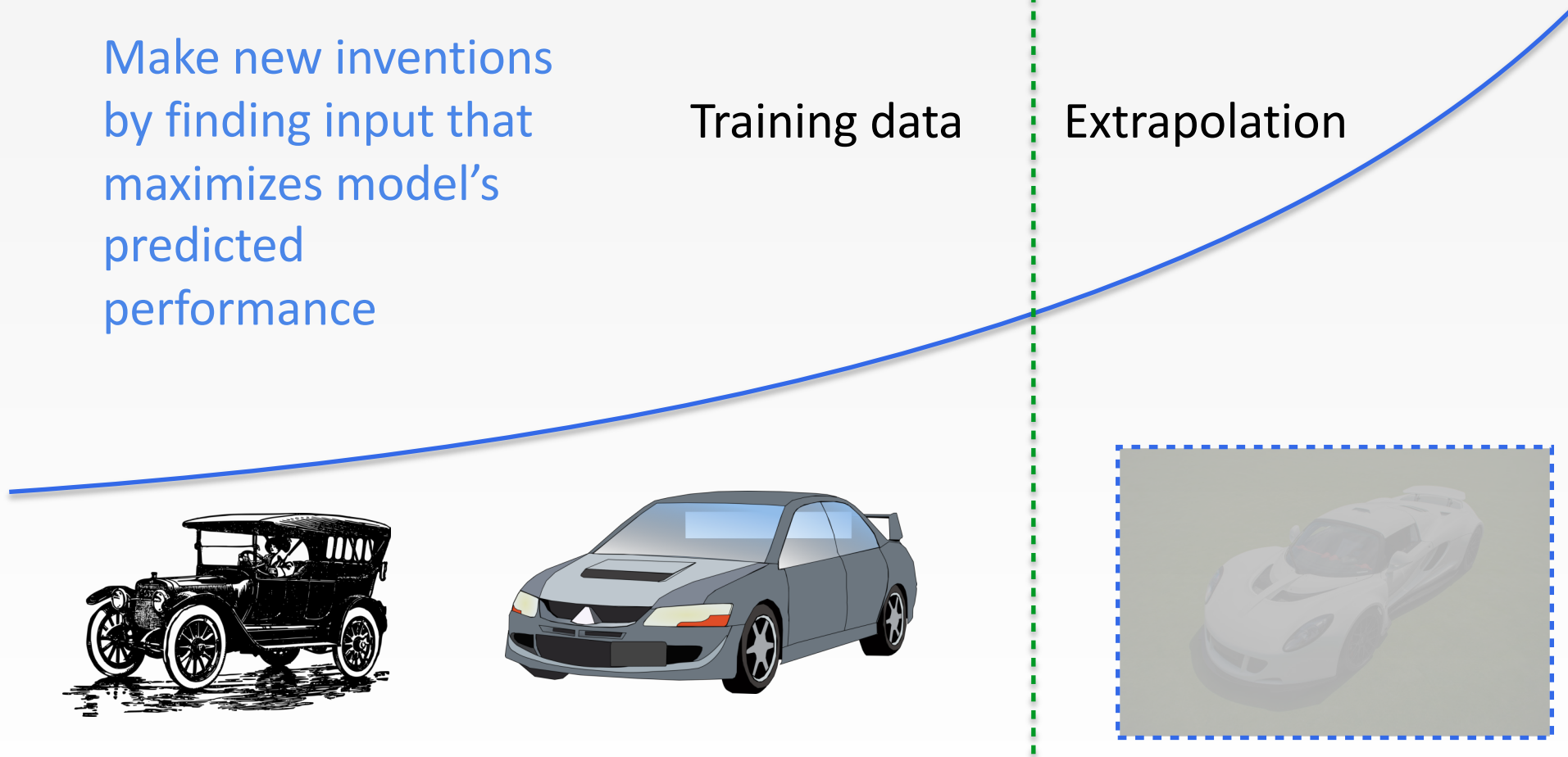


# UNIVERSAL ENGINEERING MACHINE (MODEL-BASED OPTIMIZATION)

Make new inventions  
by finding input that  
maximizes model's  
predicted  
performance

Training data

Extrapolation



# cleverhans

Open-source library available at:

<https://github.com/openai/cleverhans>

Built on top of TensorFlow (Theano support anticipated)  
Standard implementation of attacks, for adversarial training  
and reproducible benchmarks



# CONCLUSION

---

- Attacking is easy
- Defending is difficult
- Adversarial training provides regularization and semi-supervised learning
- The out-of-domain input problem is a bottleneck for model-based optimization generally