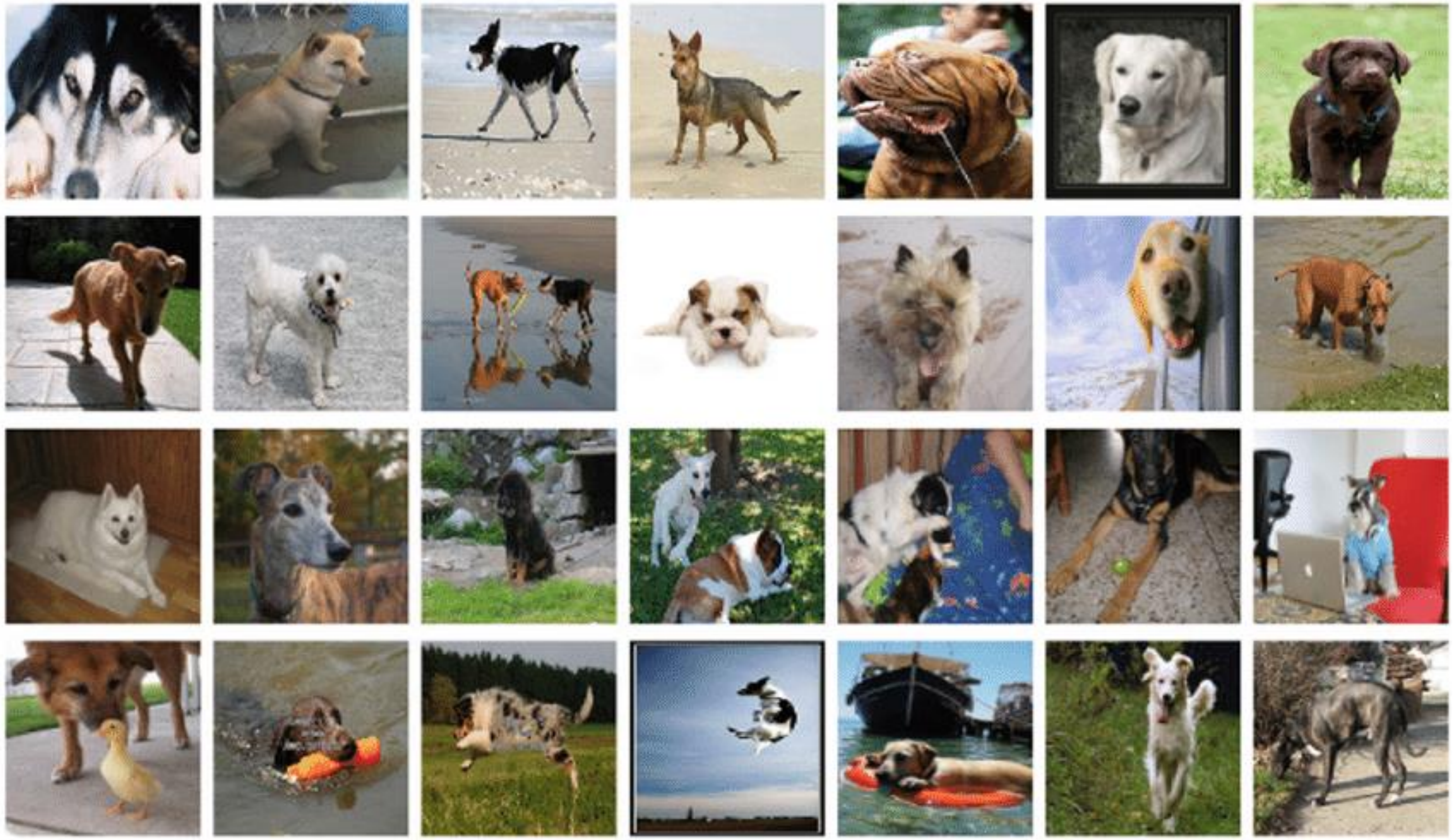
Robustness

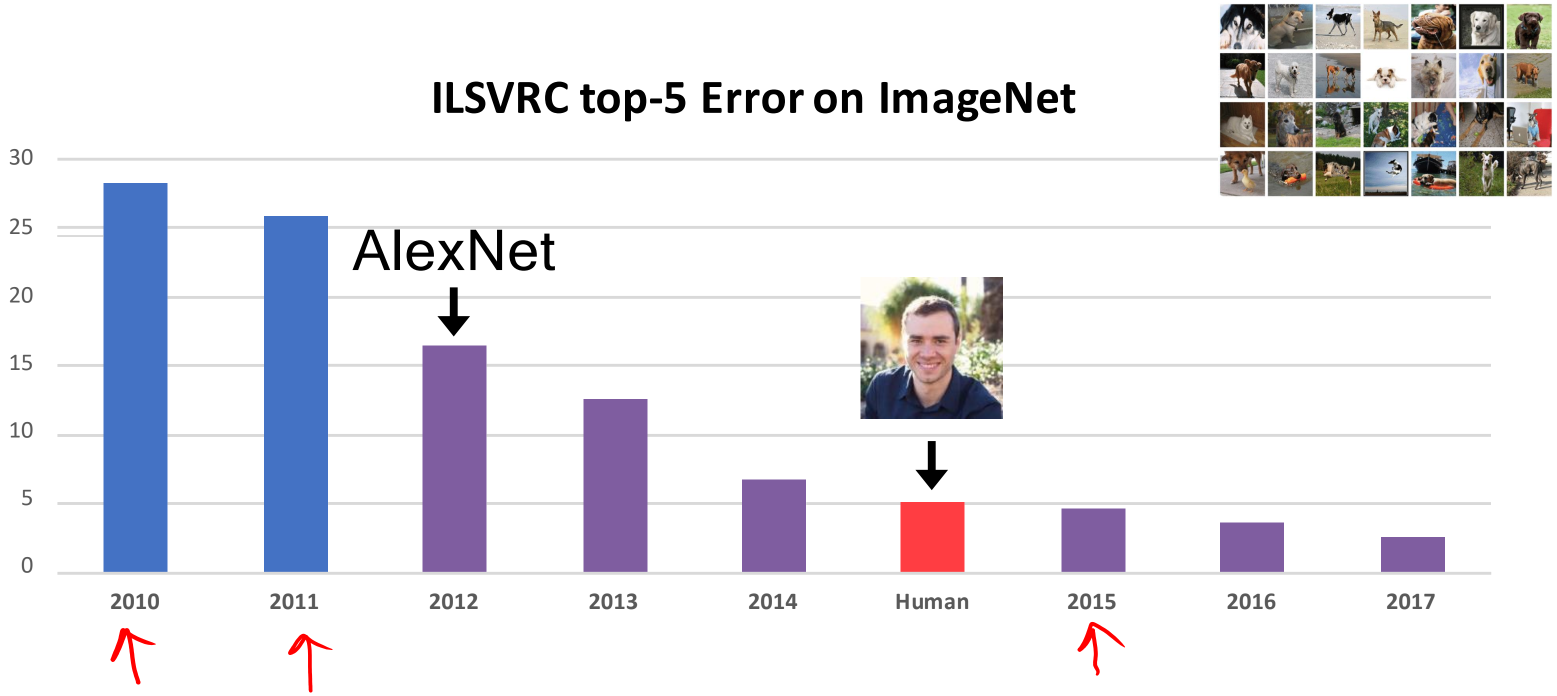
Stanford CS 329T, Spring 2022

Robustness

ImageNet: A success story



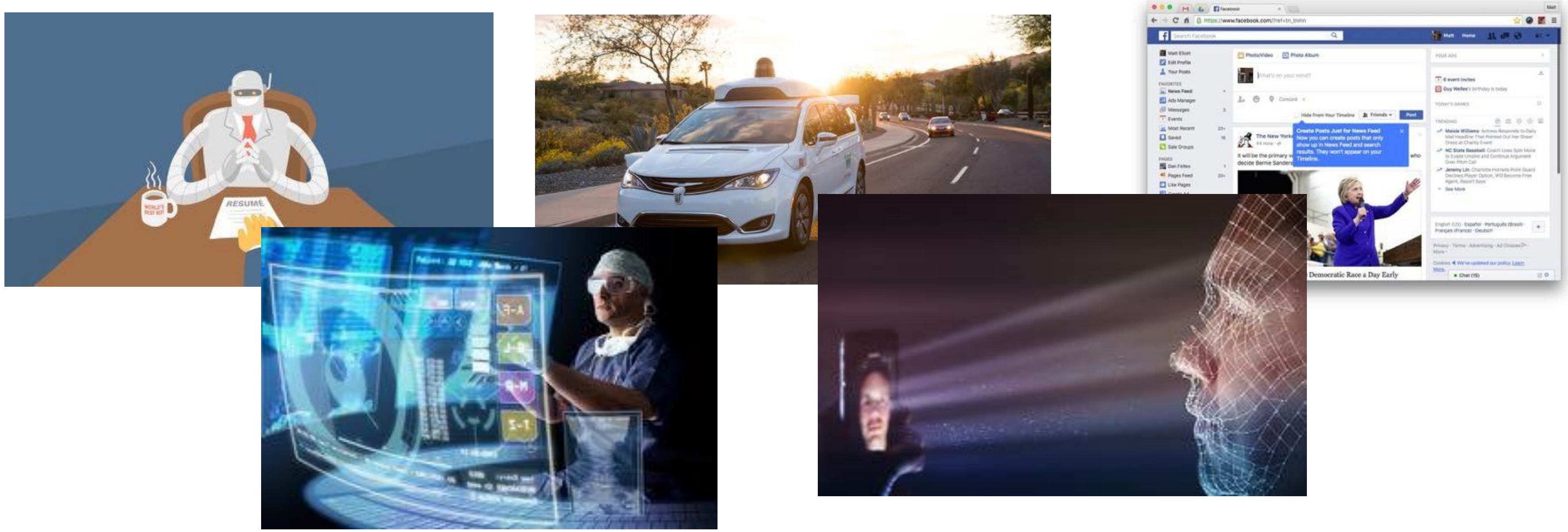
ImageNet: A success story



ILSVRC top-5 Error on ImageNet

Have we achieved truly super-human performance?

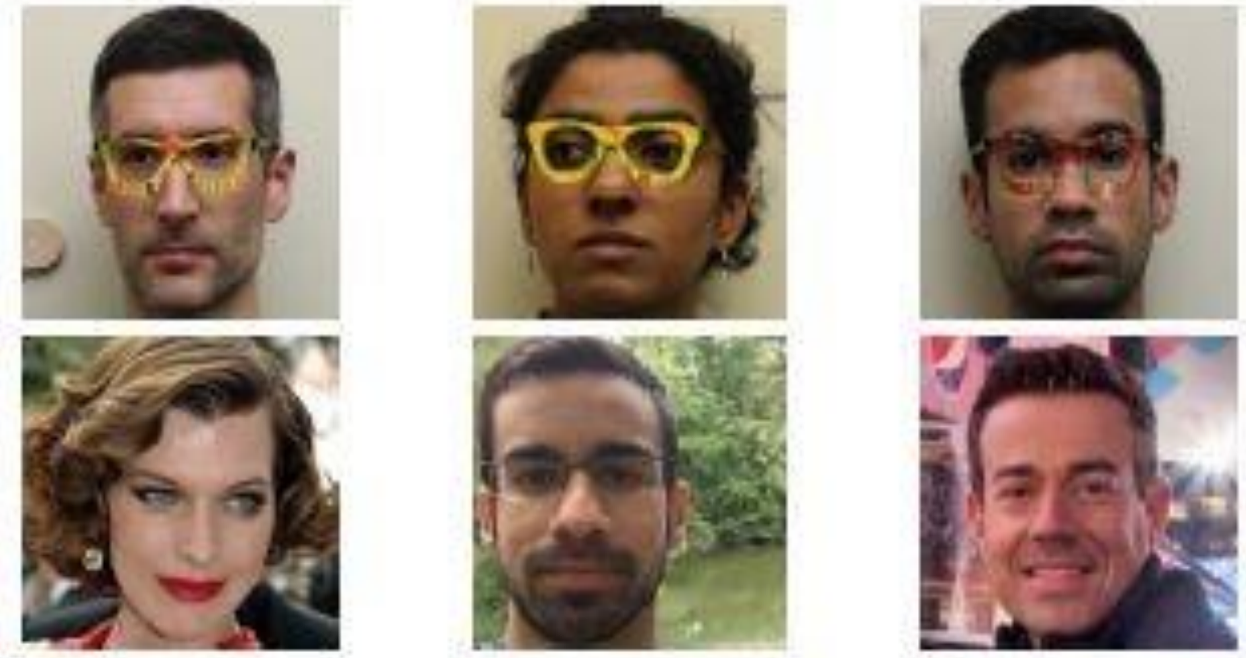
Real-world deployment



Are ML systems ready for the real world?

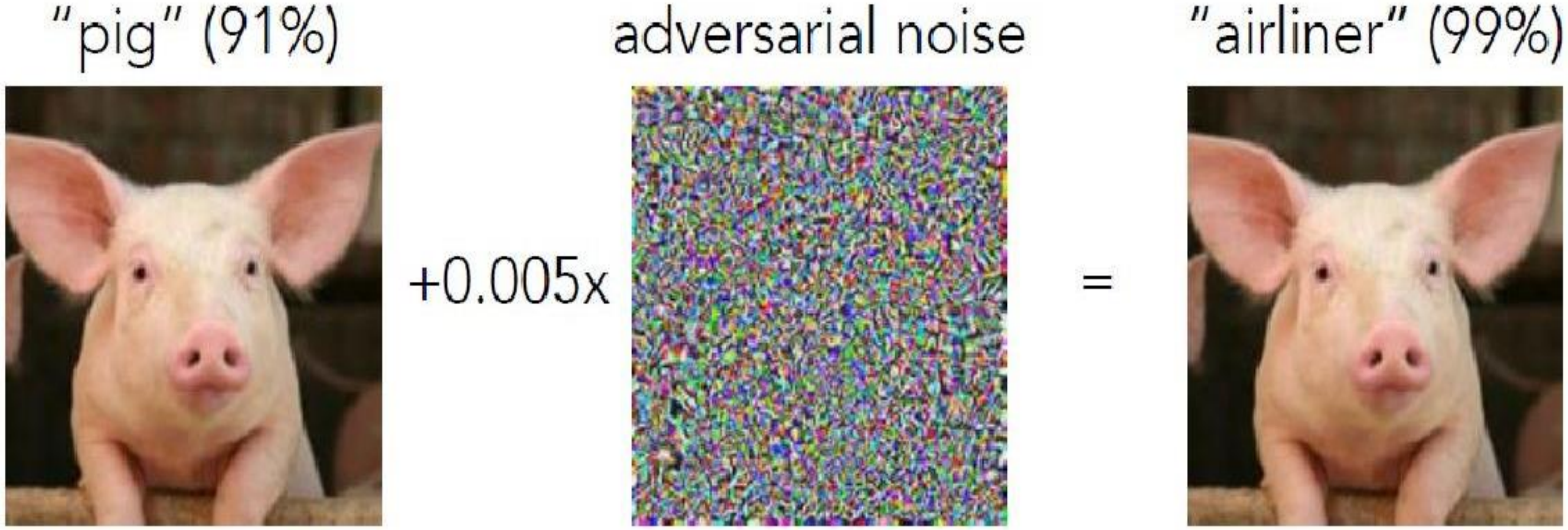
There are surprising attacks on classifiers

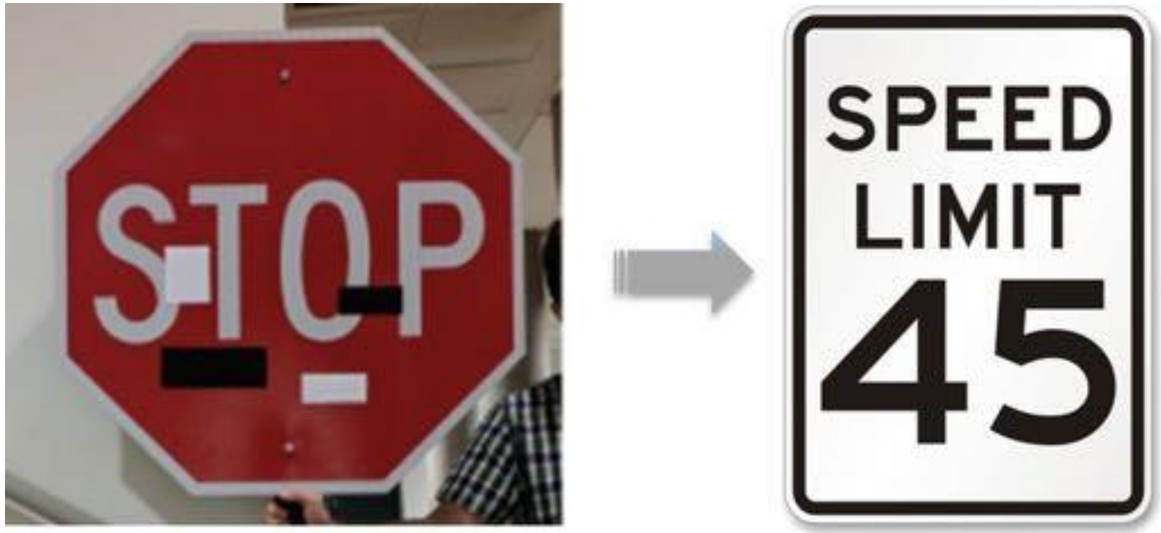
- Wearing crazy glasses can cause the model to name a celebrity



Systematic methods can increase robustness

- Robust training
  - Train so that small changes in input do not change the classifier output





[Evtimov et al. 2018]

Robustness

- Machine learning models can be very brittle
  - A small change in the input can produce a very different model output
  - Adversarial examples can be produced systematically
- There are systematic methods to improve robustness
  - Adversarial learning takes small changes in input into account
- There are limitations in the current state of the art/science
  - More robust models may be less accurate, depending on circumstances
  - Intuitive measures of “small change in input” are tricky - no clear metric for human perception

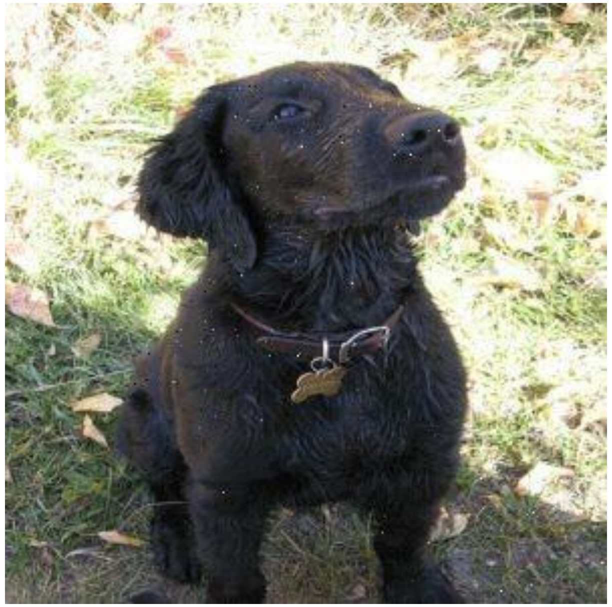
Adversarial attacks

Slide image: Carlini

Adversarial attack

- Given an input image X and a label T, find X’ ≈ X with F(X’)=T

Dog



Hummingbird

This is a strong form of attack: for *any* X and *any* T, find X’ with predicted class T

Adversarial Attack

- Basic idea
  - Given input X and desired label T, find adversarial input X'
    - Minimize d(X,X') – adversarial input should be small perturbation of given input X
    - Satisfying F(X')=T – adversarial input will have desired label
    - And such that X' is valid (has the right structure to be an image)
- Problem
  - Non-linear constraints as in this objective function do not work well

Adversarial Attack: Optimization problem

$$\min_{x'} \; d(x, x')$$

s.t. $\quad F(x') = T$

Targeted

Non_Targeted

$$\min_{x'} \; d(x, x') + \lambda \left(F(x') - T \right)^2$$

5

4

1

$$\boxed{\min_{x'} \; d(x, x') + \lambda \, CE\Big(\underset{\substack{\swarrow \\ \text{one-hot vector}}}{T}, F(x')\Big)}$$

Non-Target

$$\boxed{F(x) = q}$$

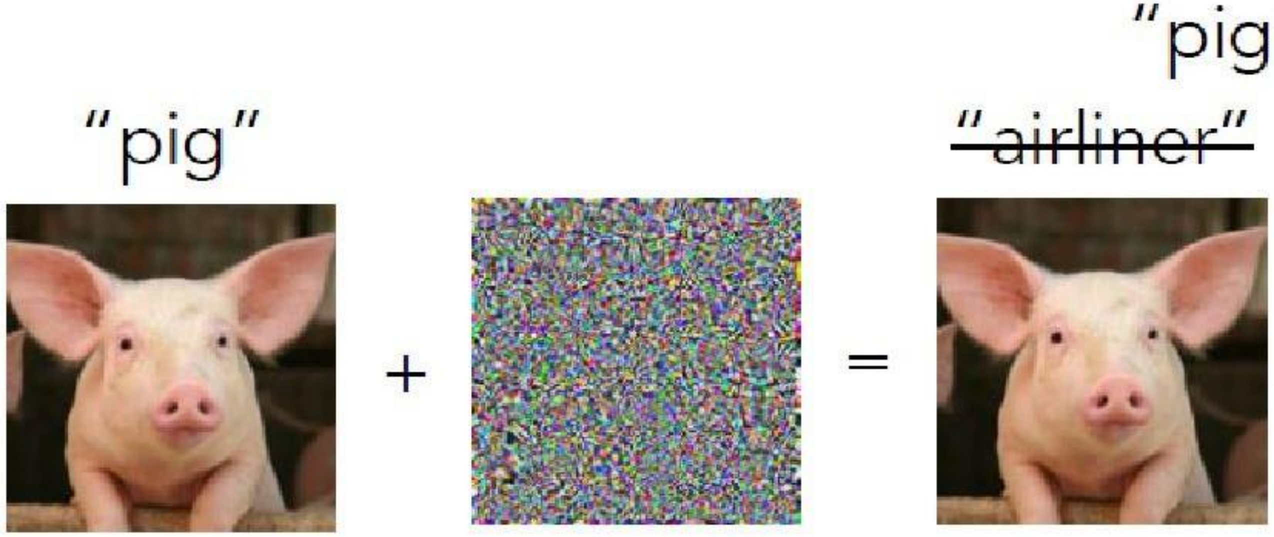
$$\min_{x'} \underbrace{d(x, x')}_{\downarrow} - \lambda \underbrace{CE(q, F(x'))}_{\uparrow}$$

Robust training

How do we train robust models?

Specifically, how can we build models to similar inputs produce similar results?

Our focus:



Recall: How to find adversarial examples

Standard training

model parameters input label

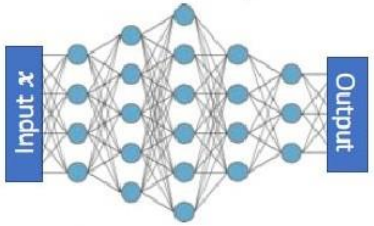
$$\min_{\theta} \mathbb{E}_{x,y\sim D}\left[loss(\theta, x, y)\right]$$

Adversarial attacks

$$\max_{\delta\in\Delta} loss(\theta, x + \boldsymbol{\delta}, y)$$

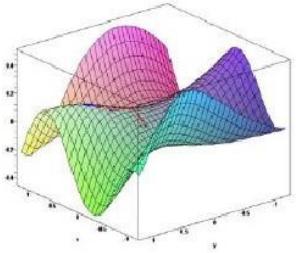
Allowed perturbations: pixel-wise, rotations, …

differentiable



Parameters θ

Gradient Descent to find θ



Use similar framework to train robustly

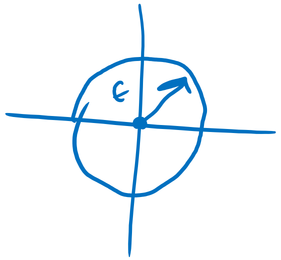
$$\min_{\theta} \mathbb{E}_{x,y\sim D,\ \delta\sim N(0,\sigma^2 I)} \left[loss(\theta, x+\delta, y) \right] \qquad \delta \in \Delta$$

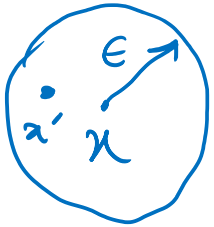
$$\|\delta\|_2 \leq \epsilon$$

$$\min_{\theta} \mathbb{E}_{x,y\sim D} \left[\max_{\boxed{\delta\in\Delta}} loss(\theta, \underbrace{x + \delta}_{x'}, y) \right]$$

finding a robust model

finding a worst-case perturbation





Improve robustness: Train on perturbed inputs

(aka "adversarial training" [Goodfellow et al. 2015])

Actually leads to **robust models** (with some care)

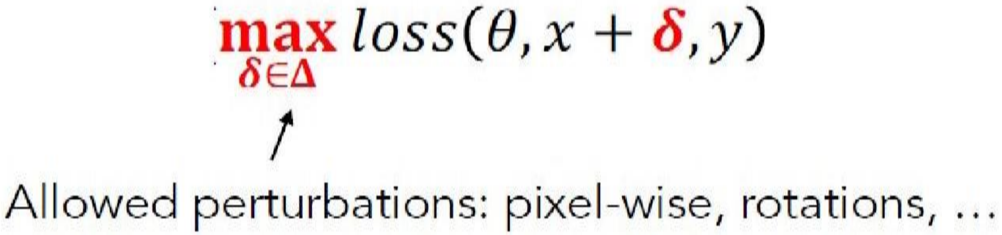
Questions for discussion

- Do you think this goal captures “robustness” correctly?
    - A small change in input should not produce arbitrary large changes in output

- Are there other goals you associate with “robustness”?

- Does you think it is always better to build modules by perturbing input slightly, as shown in last few slides?

- What kind of input changes qualify as “slight perturbation”?

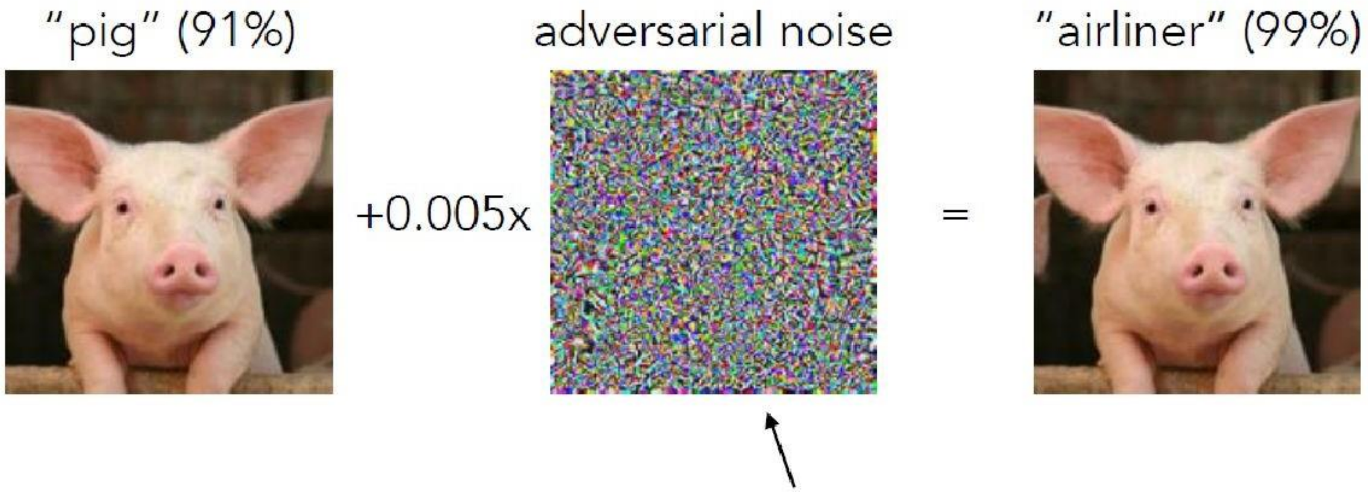


$$\max_{\delta \in \Delta} loss(\theta, x + \boldsymbol{\delta}, y)$$

Allowed perturbations: pixel-wise, rotations, ...

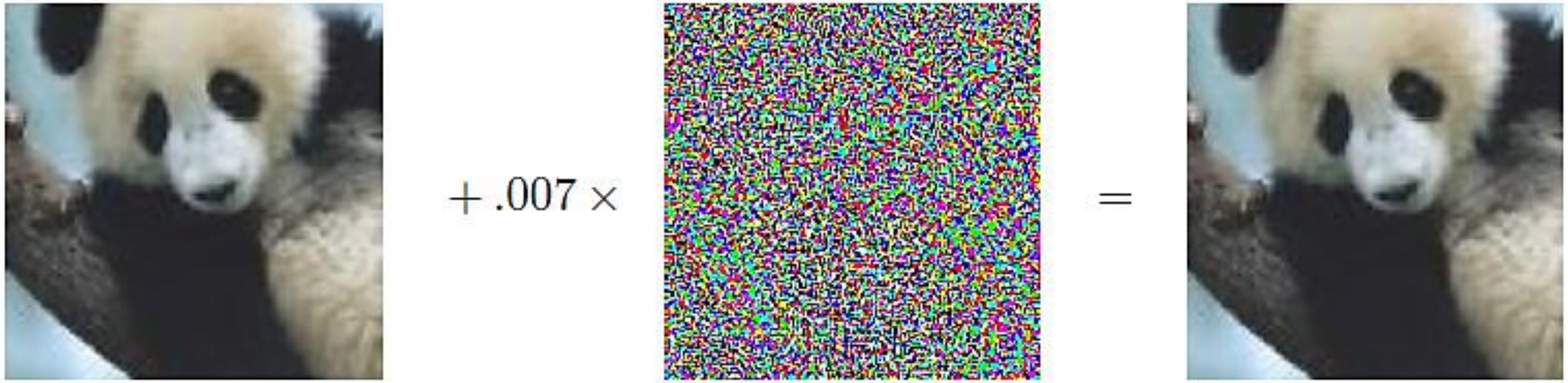
Human Alignment: Robustness & Explainability

How are DL models making predictions?



Why is this important to the model?

Adversary Example: FGSM Attack



$\boldsymbol{x}$

"panda"

57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"

8.2% confidence

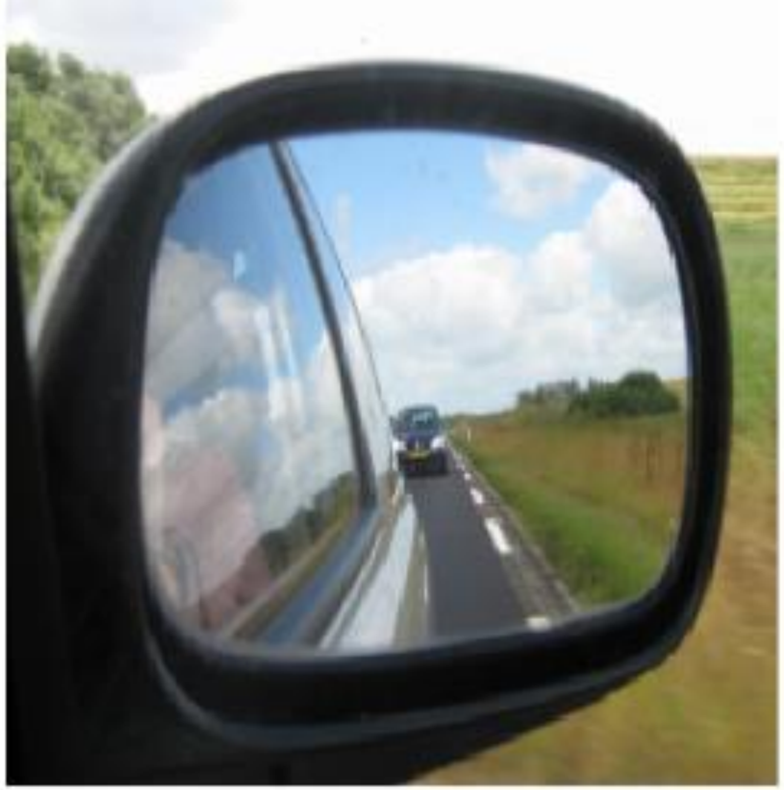
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"

99.3 % confidence

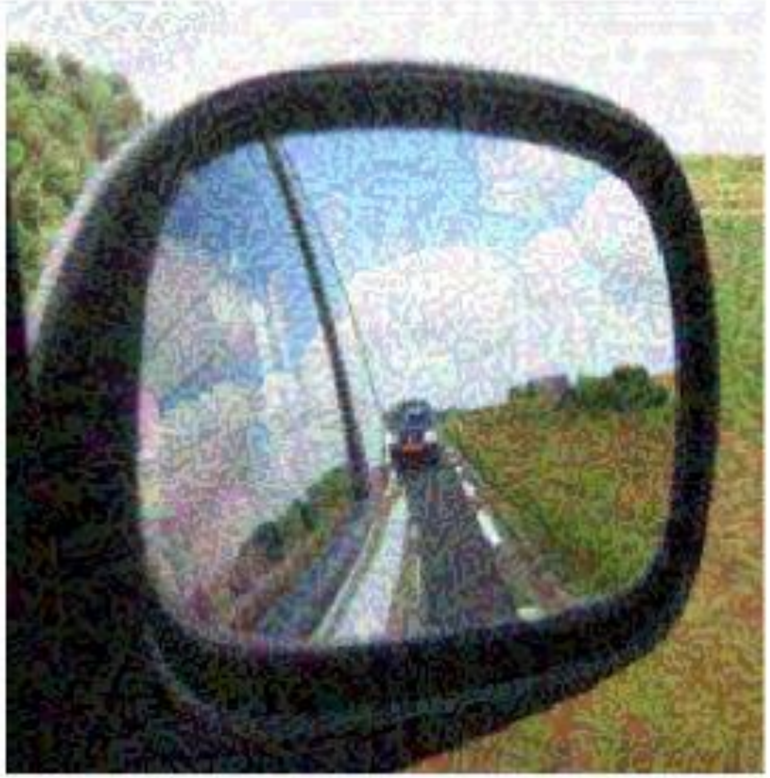
Adversary Example: FGSM Attack

Original image



Prediction: car mirror

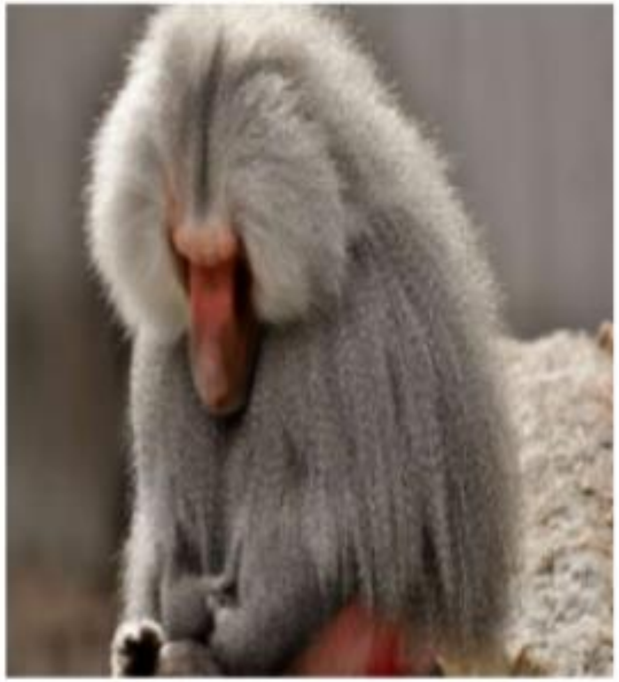
Adversarial image



Prediction: sunglasses

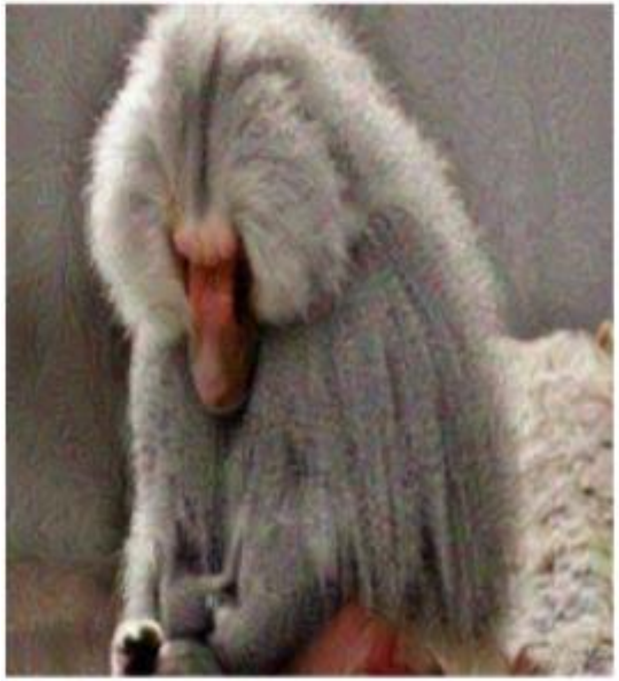
Adversary Attack: PGD Attack

Original image



Prediction: baboon

Adversarial image

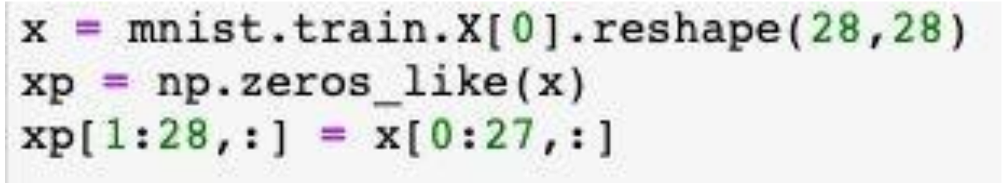


Prediction: Egyptian cat

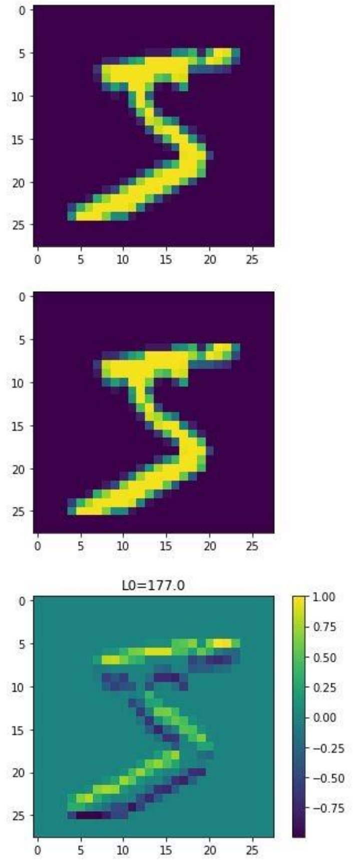
Egyptian cat

Human Alignment: Metrics used for robustness

- Do metrics match human perceptibility?
  - L_0 – how many pixels changed AT ALL $\|x-x'\|_0$
    - ?
  - L_1 – total absolute pixel change $\|x-x'\|_1$
    - ?
  - L_2 – euclidean distance $\|x-x'\|_2$
    - ?
  - L_{inf} – max change of any single pixel $\|x-x'\|_\infty$
    - ?



```
x = mnist.train.X[0].reshape(28,28)
xp = np.zeros_like(x)
xp[1:28,:] = x[0:27,:]
```



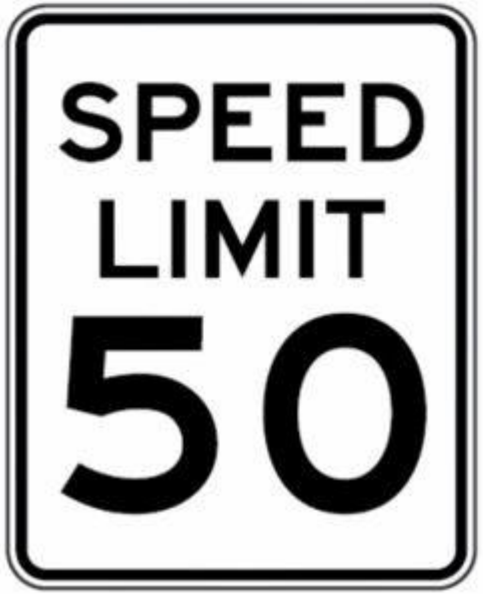
$d(x, x')$

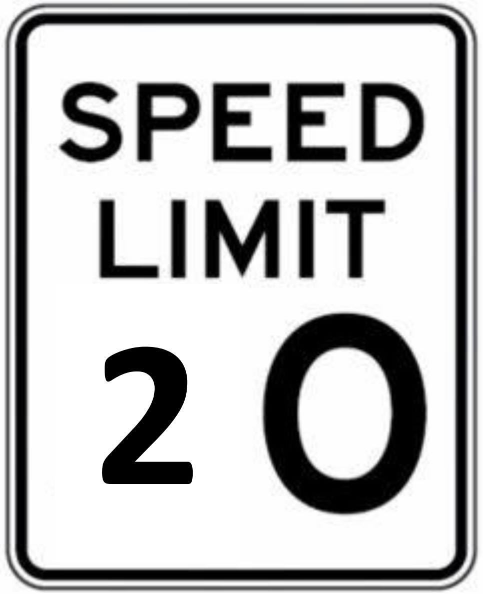
$\downarrow$

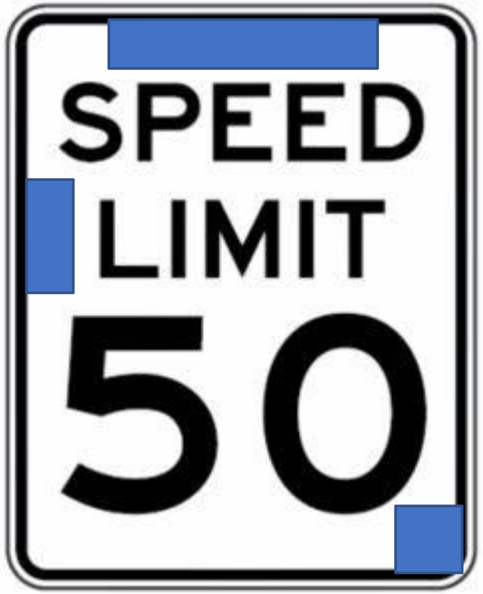
$d = ?$

Human Alignment: Metrics used for robustness

- Do metrics match semantic difference in the application?
  - One goal: d(X,X') = "a human's measure of how different* X and X' are"
  - What if "different" includes semantics of what is pictured?







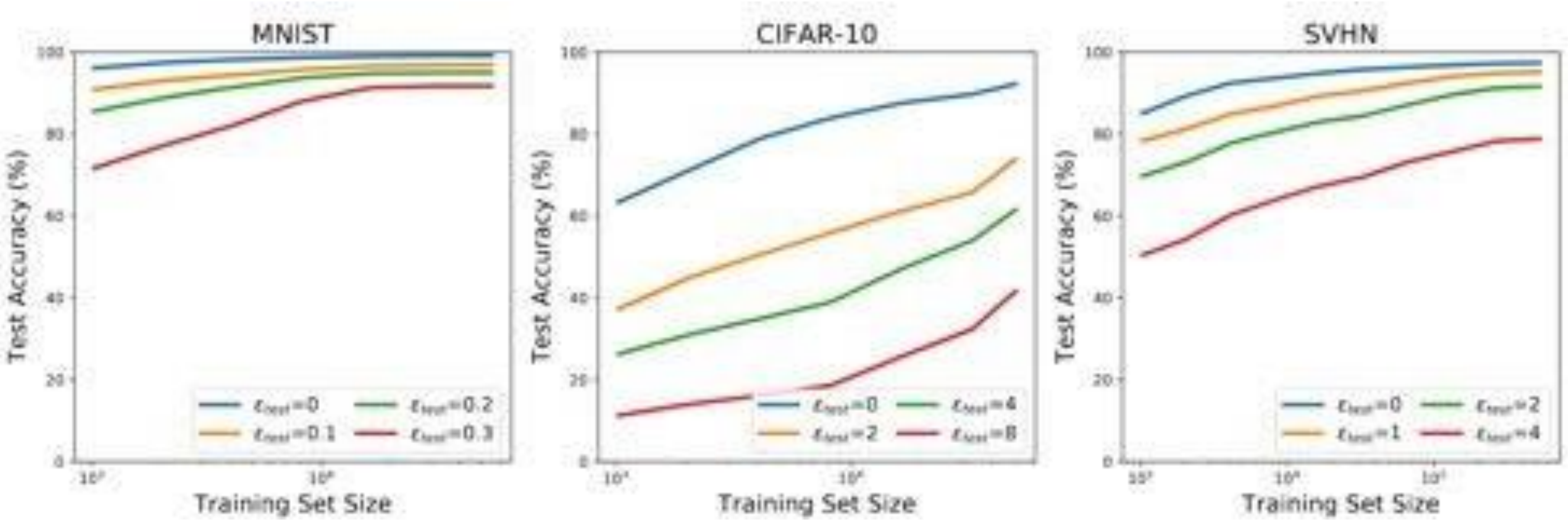
Questions for discussion

- Do you think robustness should correlate with explainability?
- Should “small change in input” mean “imperceptible to human”?
  - What other factors might be important?
- How important is robustness overall, compared to precision, accuracy and other measure?

Robust generalization is hard

Theorem: The sample complexity of robust generalization can be significantly larger than that of “standard” generalization.

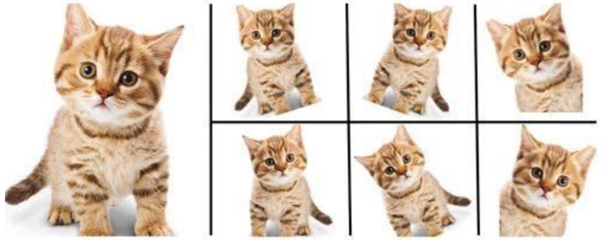
Empirically:



Does robustness improve accuracy?

Data augmentation: Train on random transformations of the input

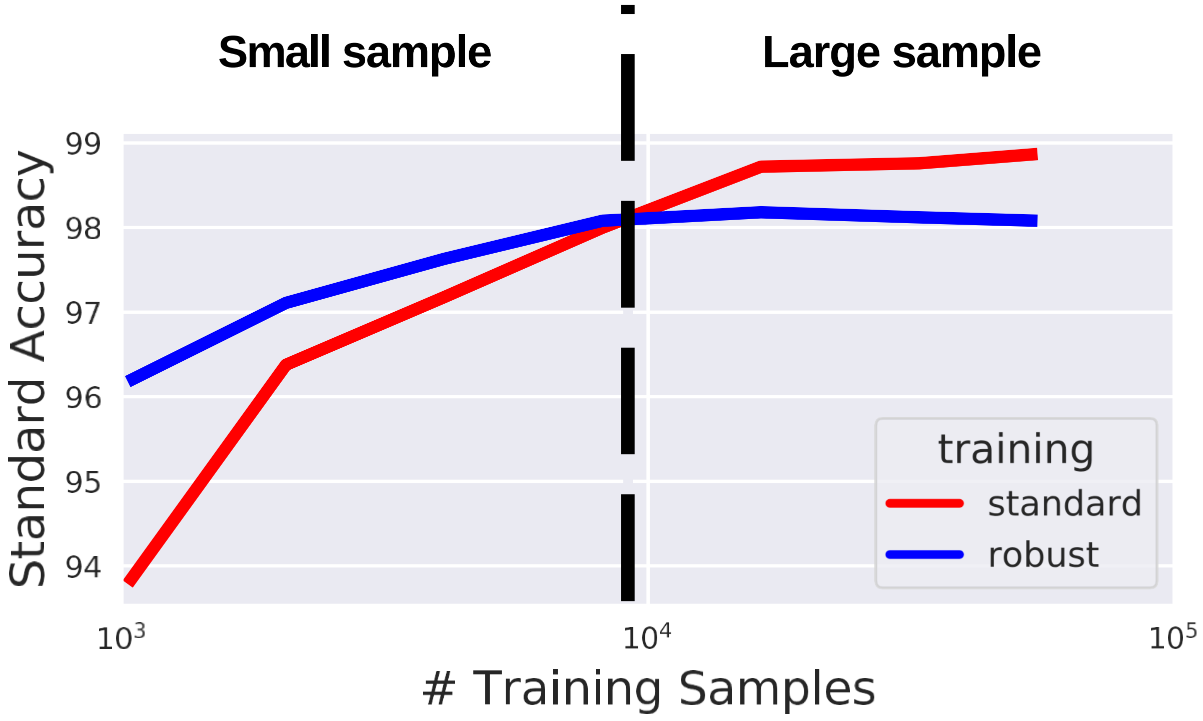
→ Significantly improves test accuracy.



Adversarial training – Augment with the "most helpful" example

Does adversarial training improve **standard accuracy**?

Does robustness improve accuracy?



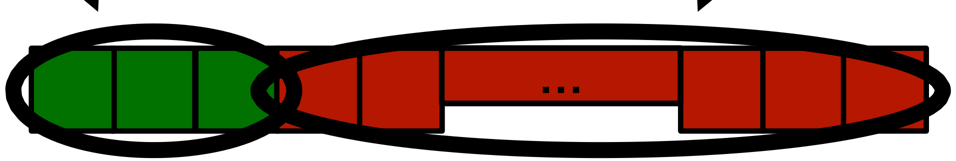
Why are robust models **less accurate**?

Does robustness improve accuracy?

Theorem: There can exist an inherent trade-off between accuracy and robustness (no “free lunch”).

Strong correlation with label

Weak correlation with label



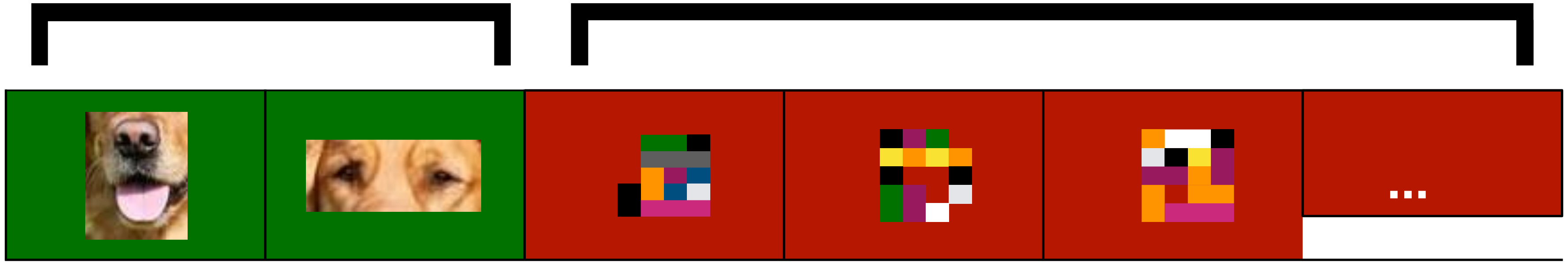
Standard Training: use all the features to maximize accuracy

Adversarial Training: use **only** strong features **(lower accuracy)**

ML security exploits

Robust features
Correlated with label
even with adversary

Non-robust features
Correlated with label on average,
but can be manipulated



Adversary manipulates input
features used for classification

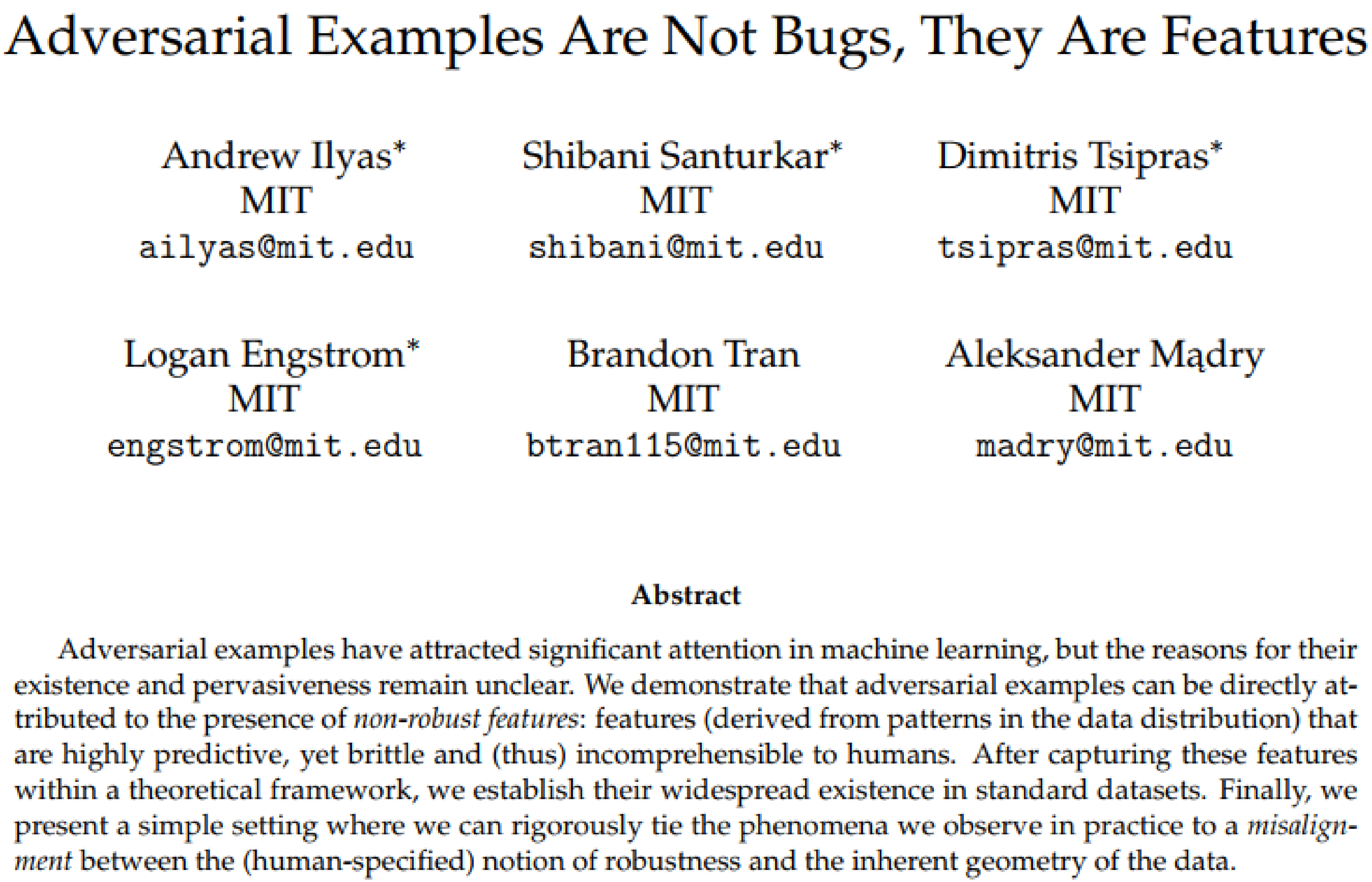
Back to adversarial examples

Non-robust features can be **quite predictive**

We train classifiers to **maximize accuracy**:
No wonder they utilize non-robust features

Relying on non-robust features **directly leads** to adversarial vulnerability

Thus: Adversarial examples are not bugs, they are features



Adversarial Examples Are Not Bugs, They Are Features

Andrew Ilyas*
MIT
ailyas@mit.edu

Shibani Santurkar*
MIT
shibani@mit.edu

Dimitris Tsipras*
MIT
tsipras@mit.edu

Logan Engstrom*
MIT
engstrom@mit.edu

Brandon Tran
MIT
btran115@mit.edu

Aleksander Mądry
MIT
madry@mit.edu

Abstract

Adversarial examples have attracted significant attention in machine learning, but the reasons for their existence and pervasiveness remain unclear. We demonstrate that adversarial examples can be directly attributed to the presence of *non-robust features*: features (derived from patterns in the data distribution) that are highly predictive, yet brittle and (thus) incomprehensible to humans. After capturing these features within a theoretical framework, we establish their widespread existence in standard datasets. Finally, we present a simple setting where we can rigorously tie the phenomena we observe in practice to a *misalignment* between the (human-specified) notion of robustness and the inherent geometry of the data.

Predictive non-robust features

Accuracy	CIFAR10	R. ImageNet
Standard	95%	97%
Non-robust features	44%	64%

Consequences

Dataset robustification: Removing non-robust features can improve **standard** classifiers

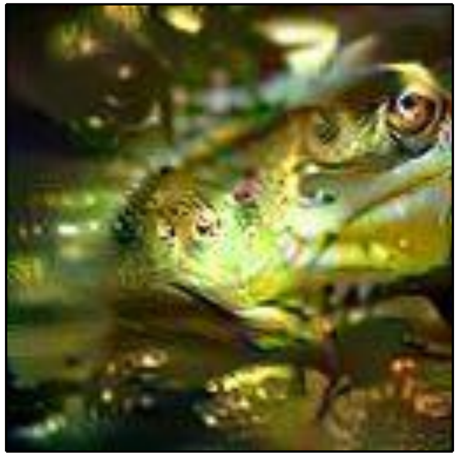
Training set

New training set

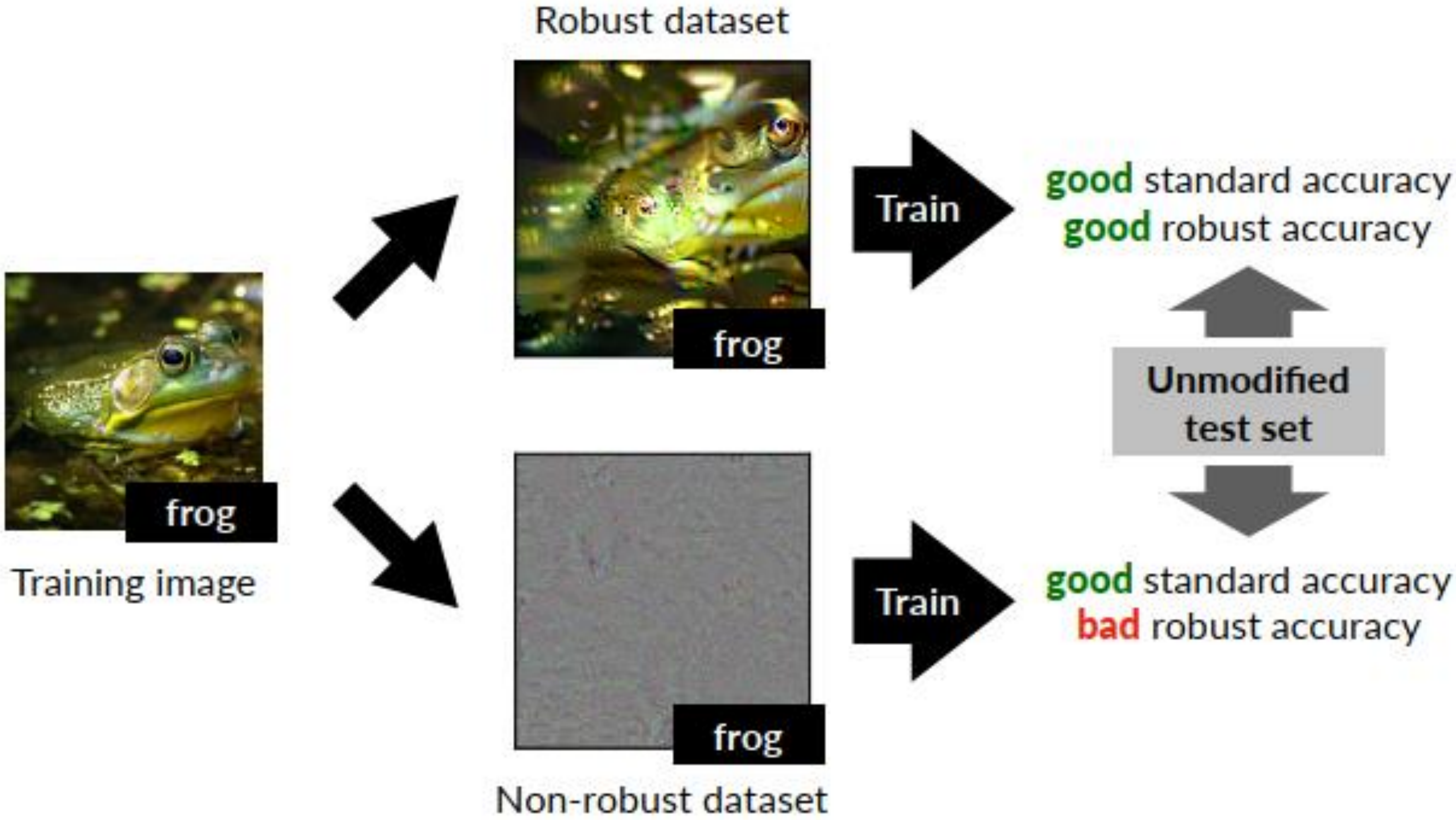
frog

Restrict to features of robust model

Standard training yield **robust classifiers**



“Robustified” frog



Robust dataset

frog

Train

good standard accuracy
good robust accuracy

frog

Training image

Unmodified test set

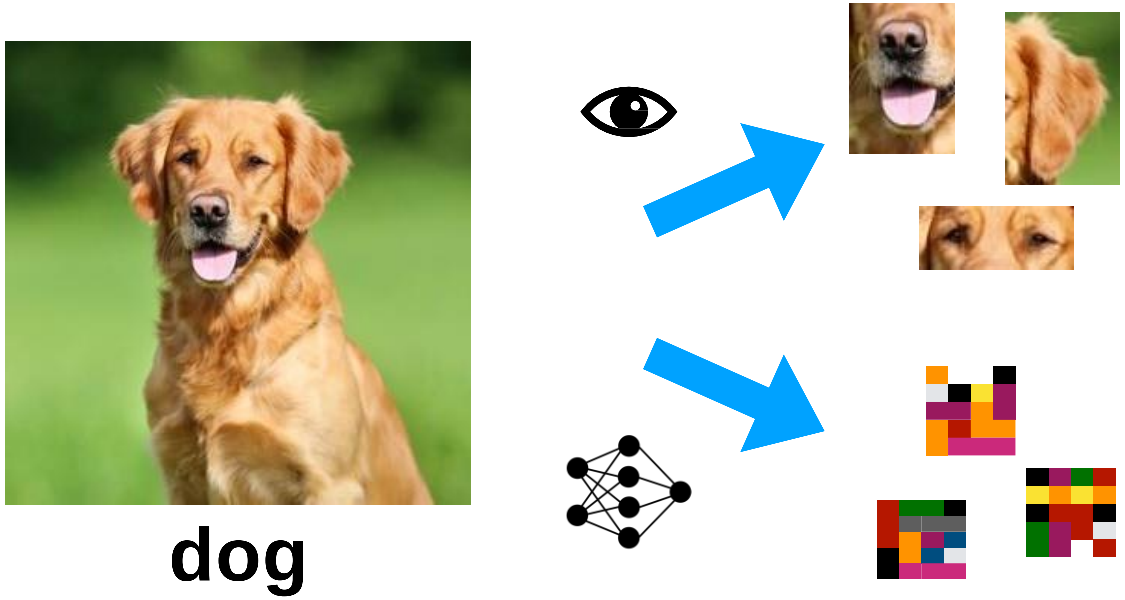
frog

Train

good standard accuracy
bad robust accuracy

Non-robust dataset

Humans vs ML Models



Equally valid classification methods

We need to **explicitly enforce robustness**

Robustness beyond security:

Robust models are more human-aligned

Conclusion

ML models are really **brittle**

Brittleness can arise from **non-robust features**

Robust optimization **can lead to robust models**

Robustness as a tool for **human-aligned** models