Adversarial Examples

presentation by Ian Goodfellow

Deep Learning Summer School
Montreal
August 9, 2015

In this presentation….

- “Intriguing Properties of Neural Networks.” Szegedy et al., ICLR 2014.
- “Explaining and Harnessing Adversarial Examples.” Goodfellow et al., ICLR 2014.
- “Distributional Smoothing by Virtual Adversarial Examples.” Miyato et al ArXiv 2015.

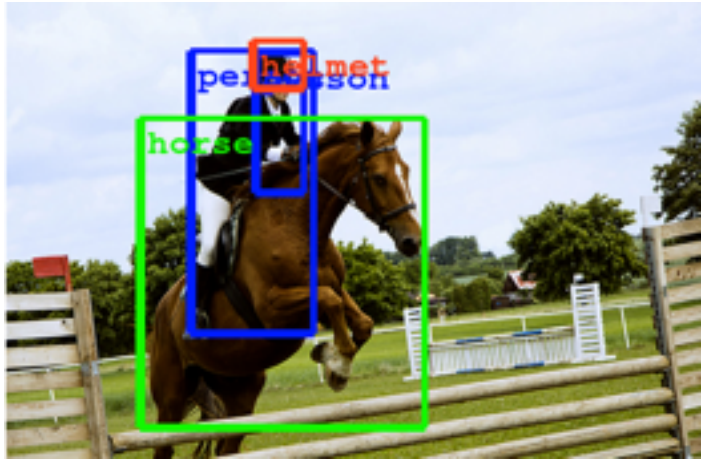
Google

Universal engineering machine (model-based optimization)

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

Training data | Extrapolation

Deep neural networks are as good as humans 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...

Do neural networks “understand” these tasks?

- John Searle’s “Chinese Room” thought experiment

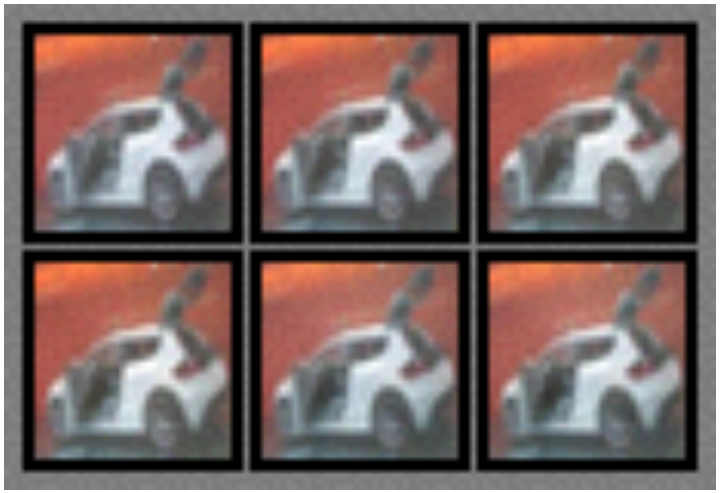
你好嗎? -> 我很好，你呢?

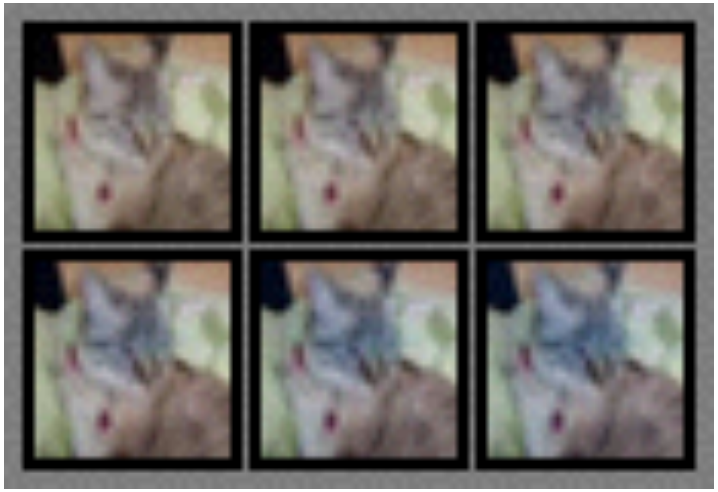
- What happens for a sentence not in the instruction book?

您貴姓大名？ ->

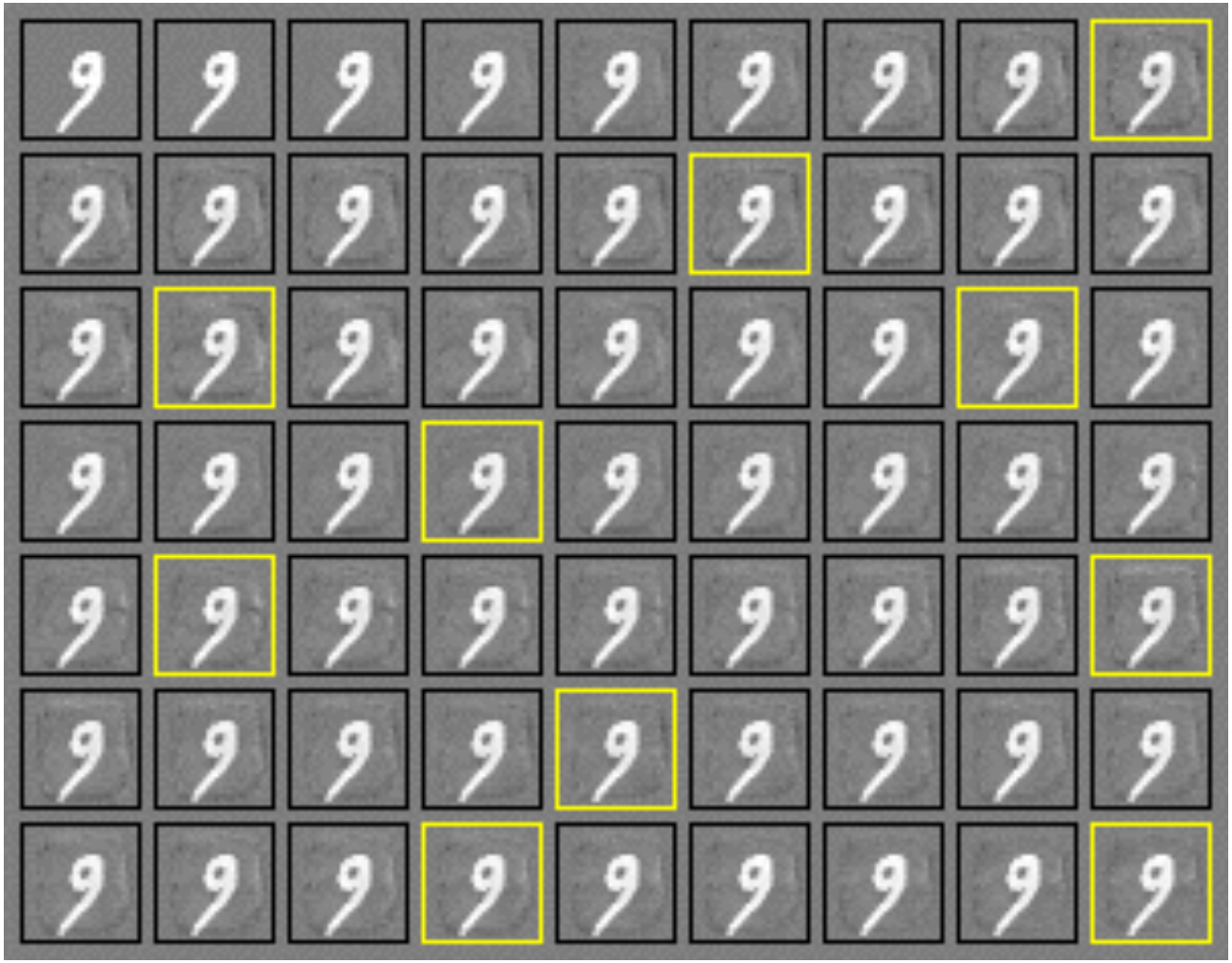
Google

Turning objects into “airplanes”





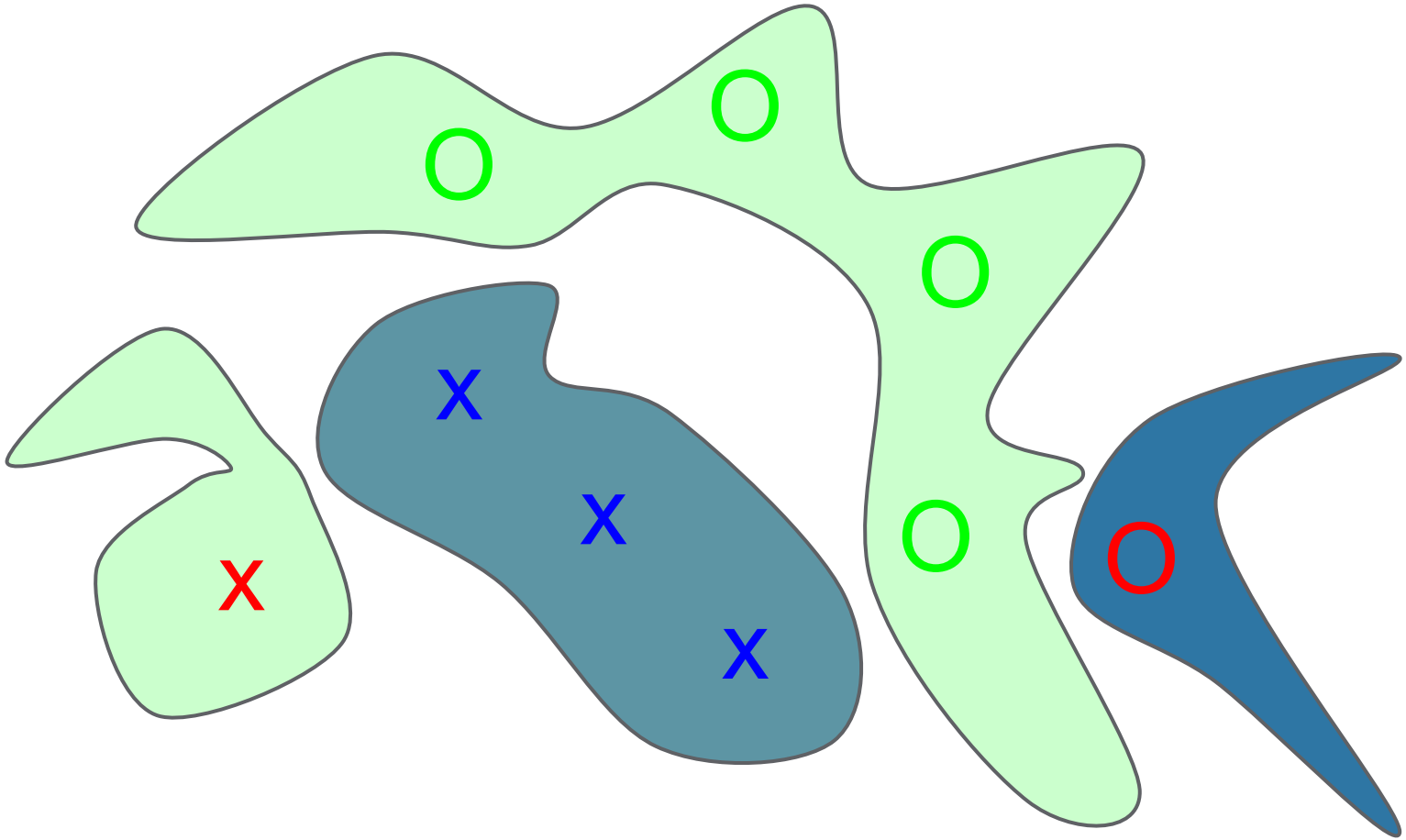
Attacking a linear model



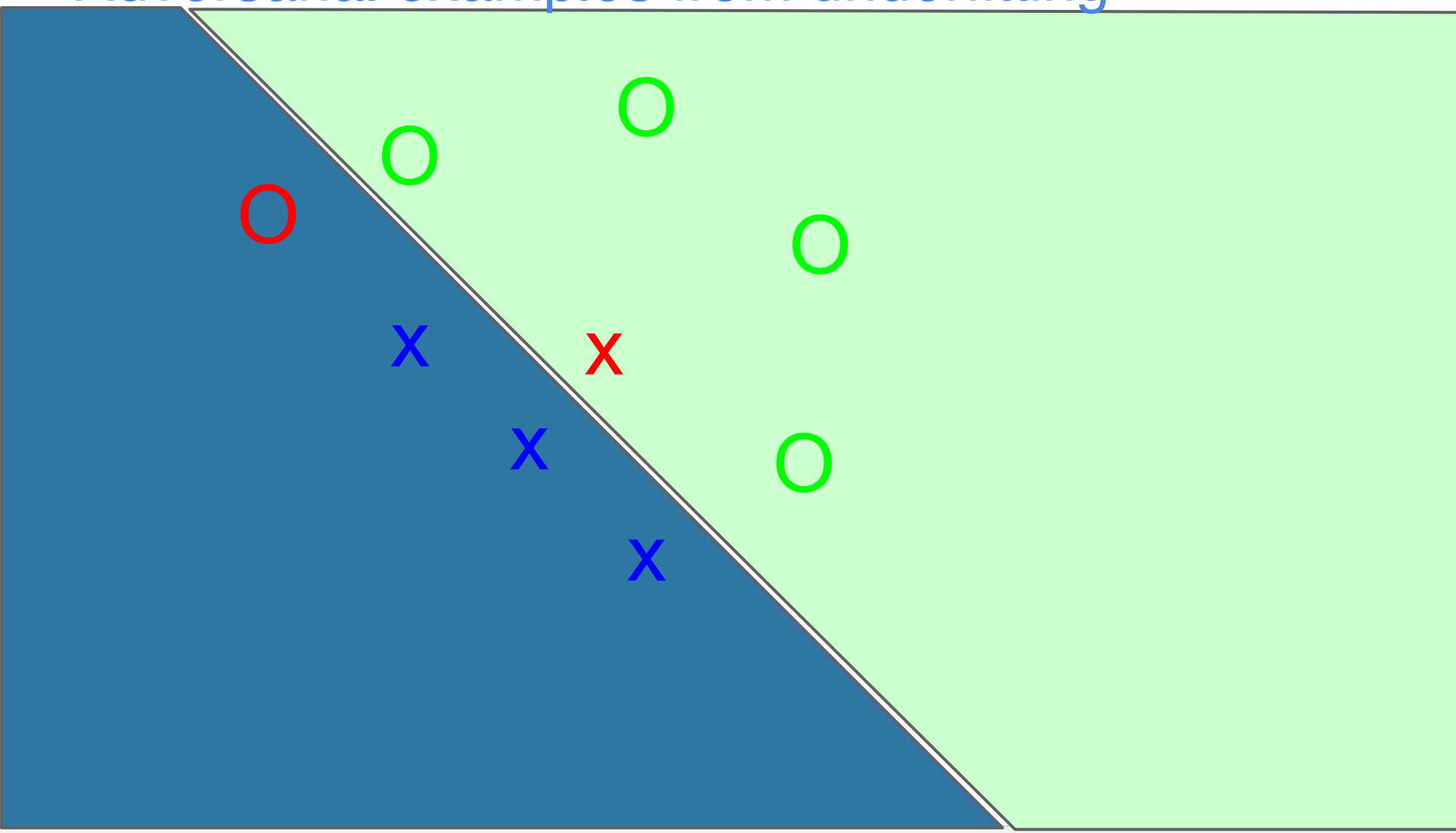
Clever Hans

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

Adversarial examples from overfitting

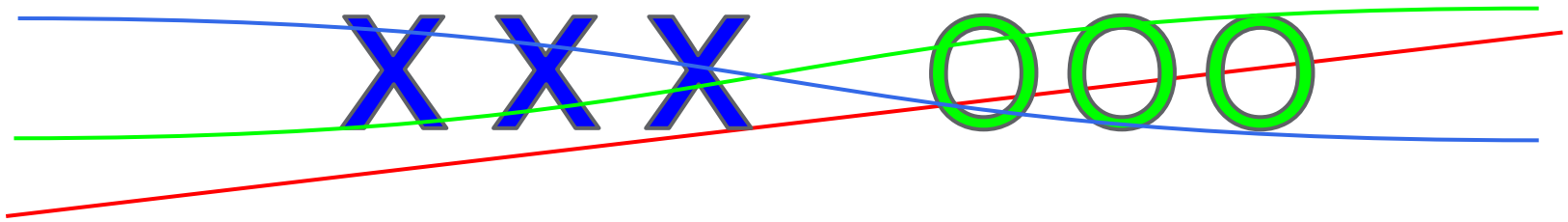


Adversarial examples from underfitting

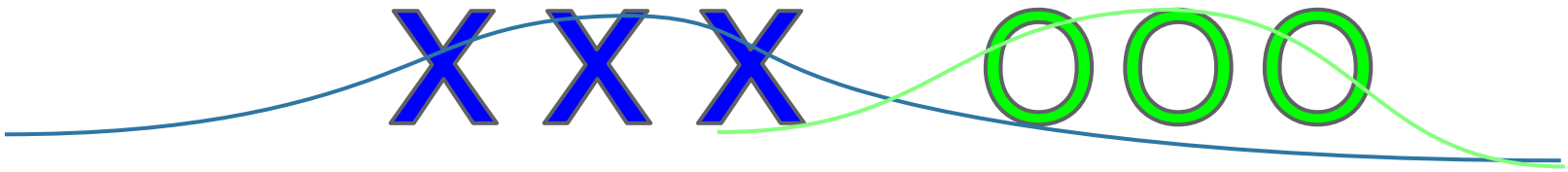


Different kinds of low capacity

Linear model: overconfident when extrapolating

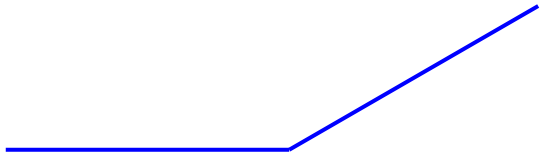


RBF: no opinion in most places

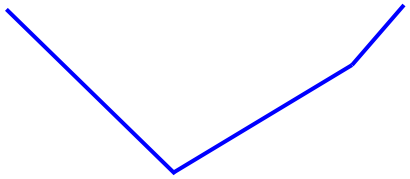


Modern deep nets are very (piecewise) linear

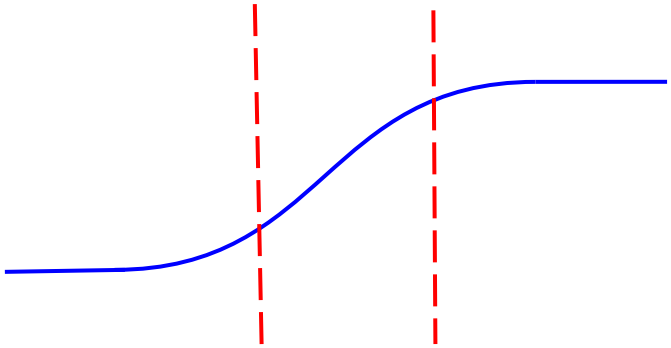
Rectified linear unit



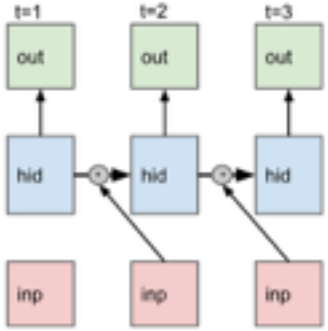
Maxout



Carefully tuned sigmoid

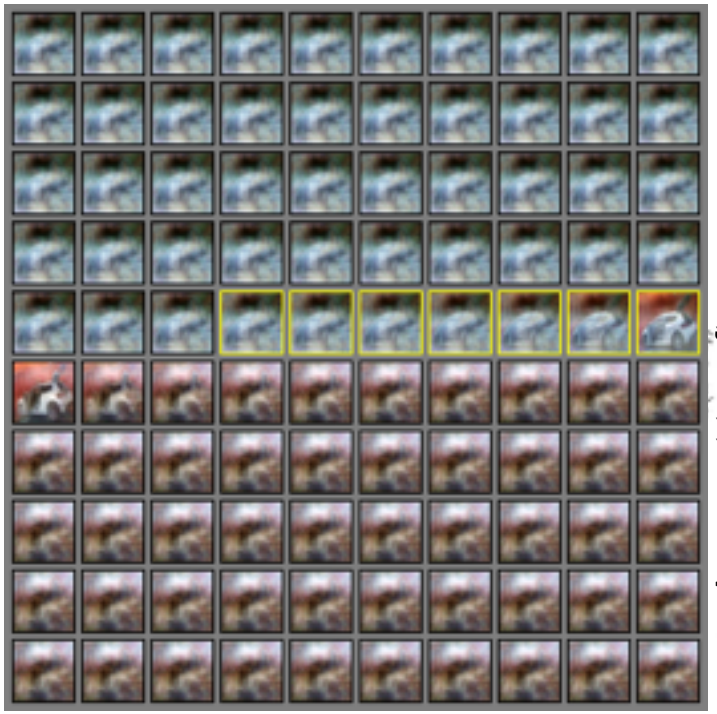


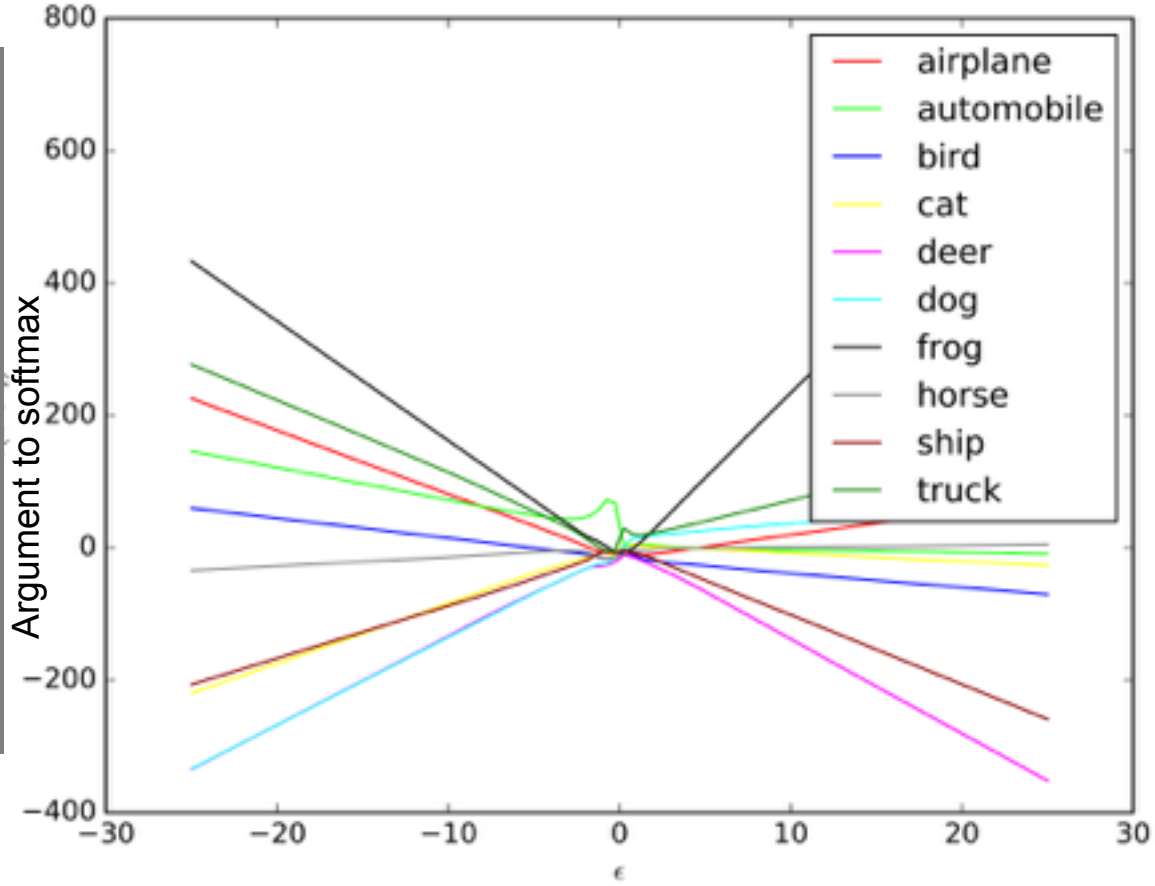
LSTM



Google

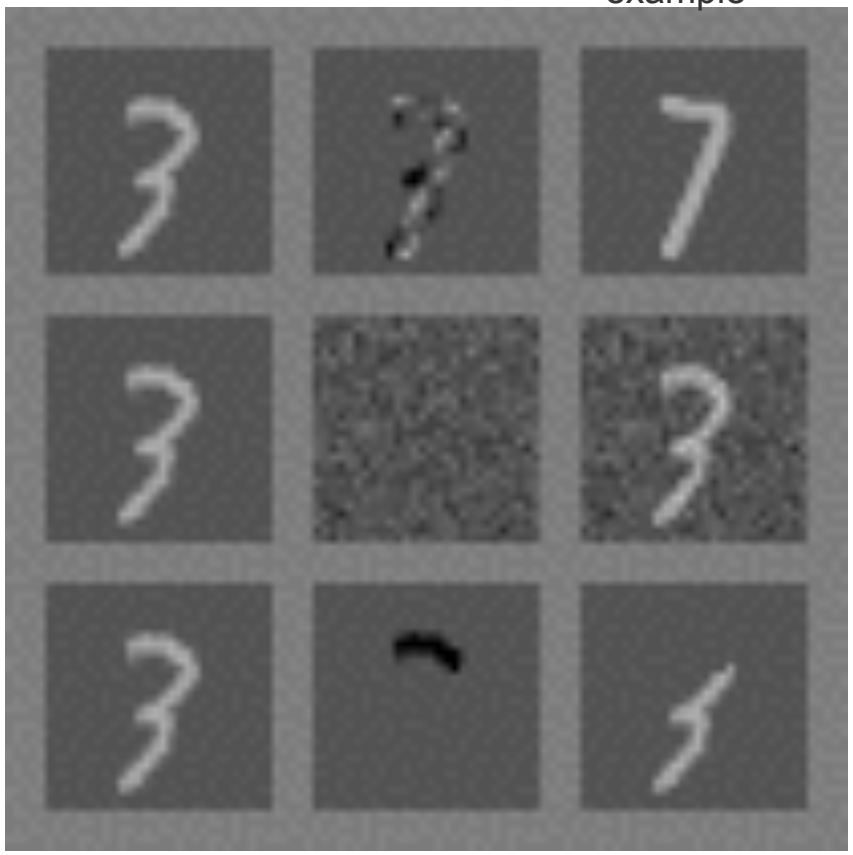
A thin manifold of accuracy





Google

Not every class change is a mistake



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

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

Maximize

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

subject to

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

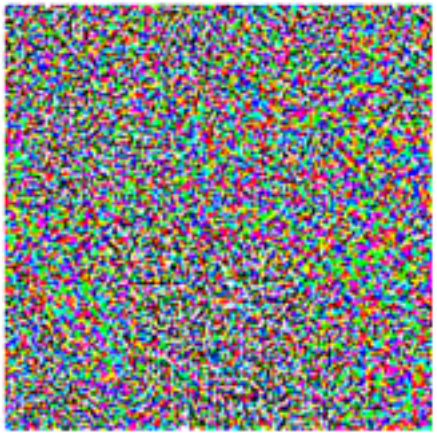
$$\Rightarrow \tilde{\boldsymbol{x}} = \boldsymbol{x} + \epsilon \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{x})\right).$$

Linear Adversarial examples

$\boldsymbol{x}$

"panda"
57.7% confidence

$+ .007 \times$



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

"nematode"
8.2% confidence

$=$

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

"gibbon"
99.3 % confidence

Google

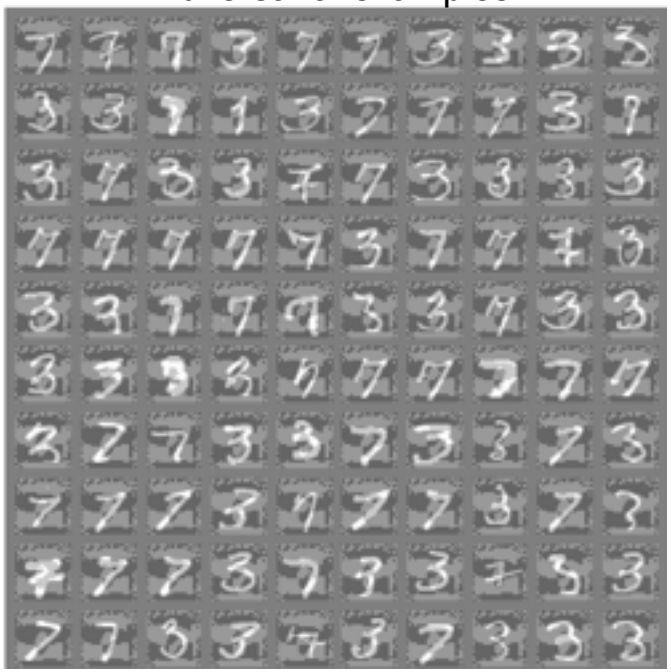
High-dimensional linear models

Weights

Signs of weights

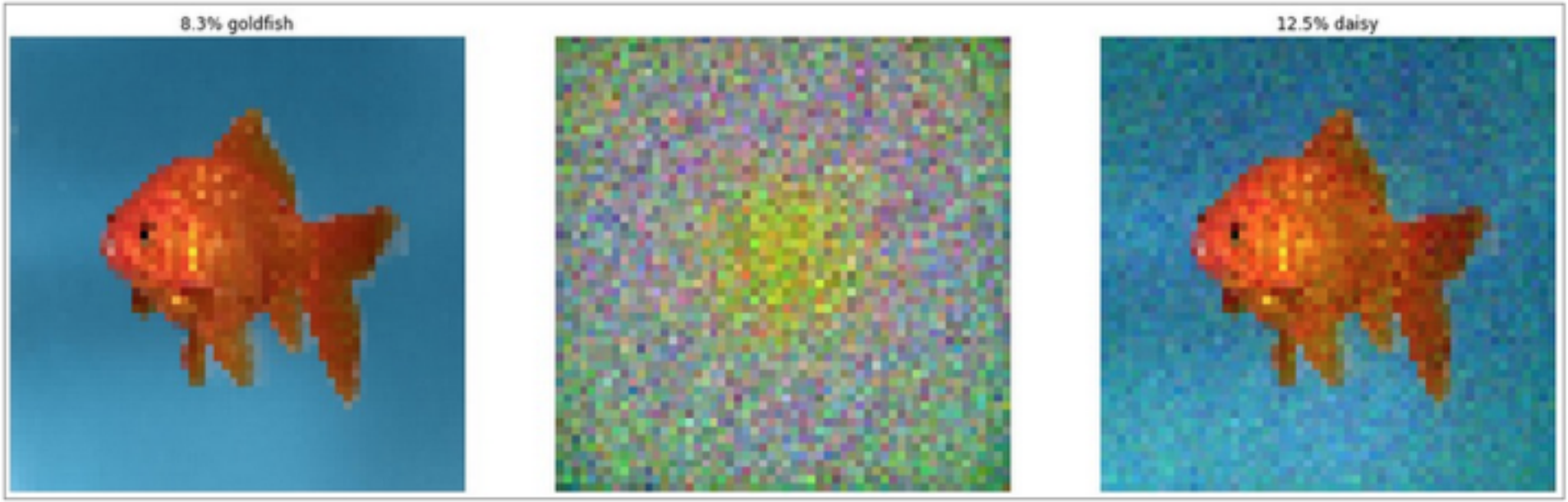
Clean examples

Adversarial examples



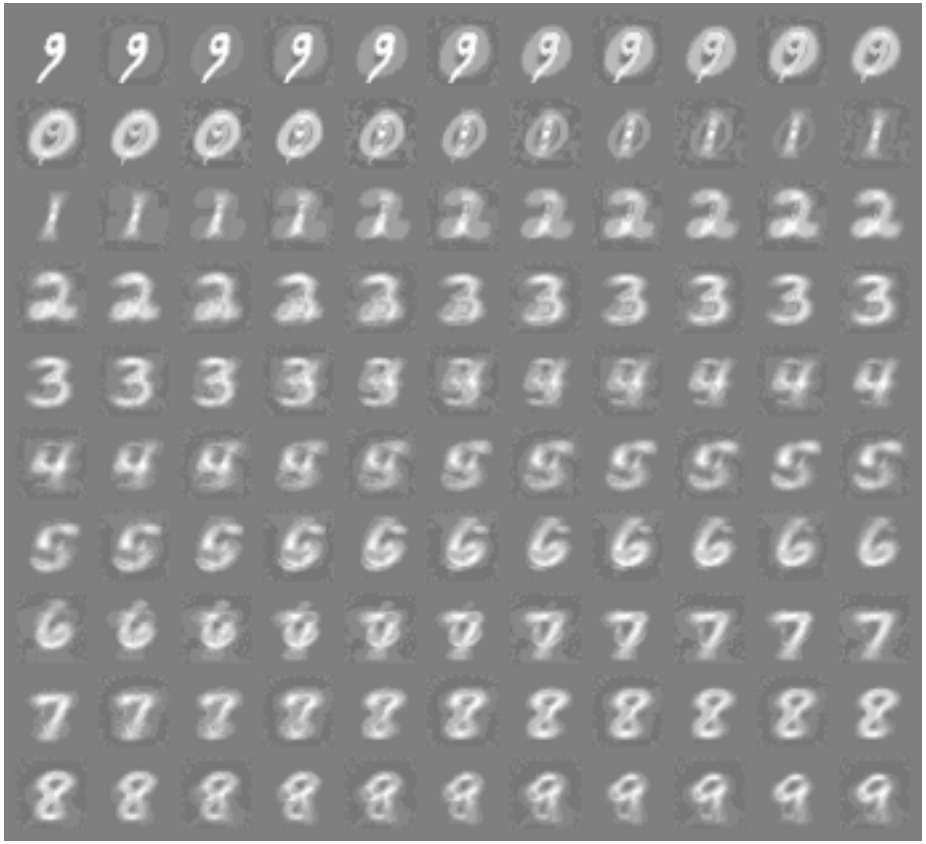
Google

Higher-dimensional linear models

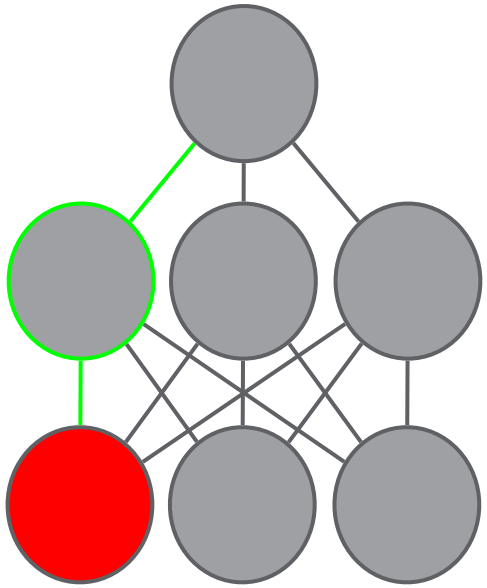


(Andrej Karpathy, “Breaking Linear Classifiers on ImageNet”)

RBFs behave more intuitively far from the data

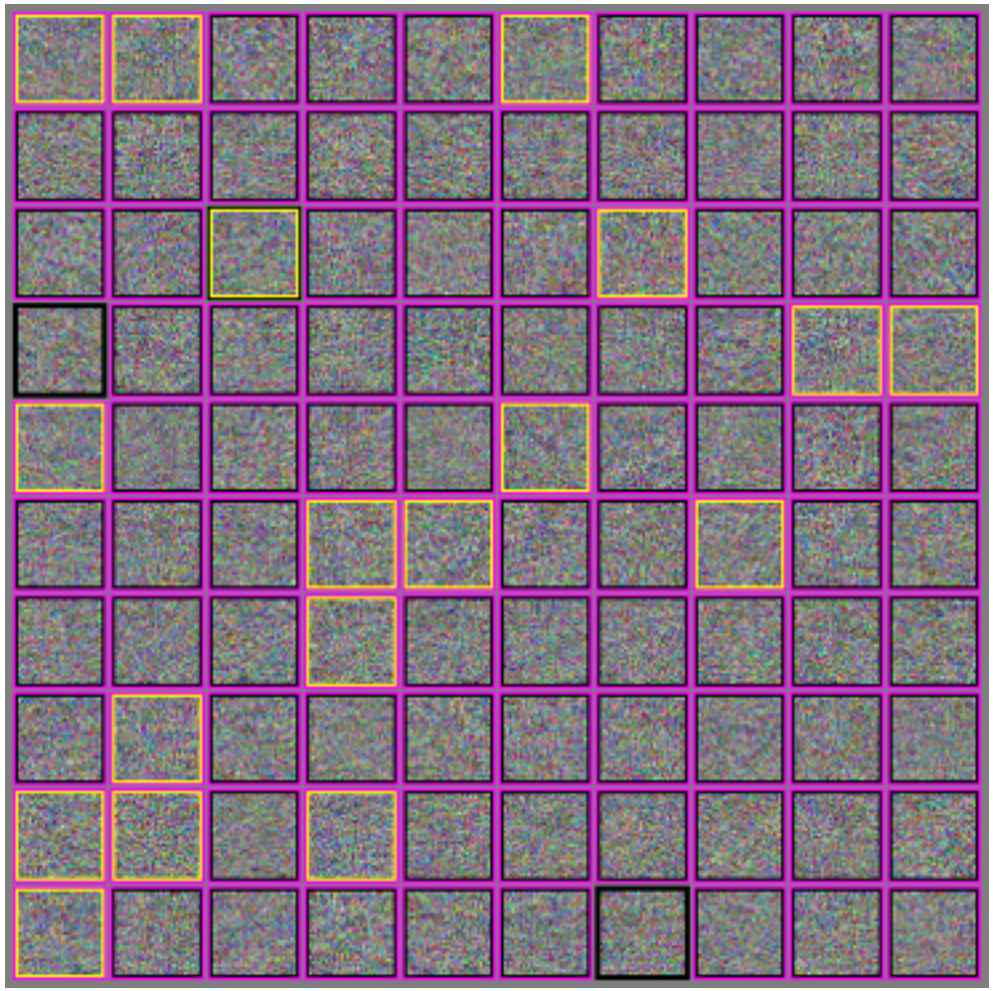


Easy to optimize = easy to perturb

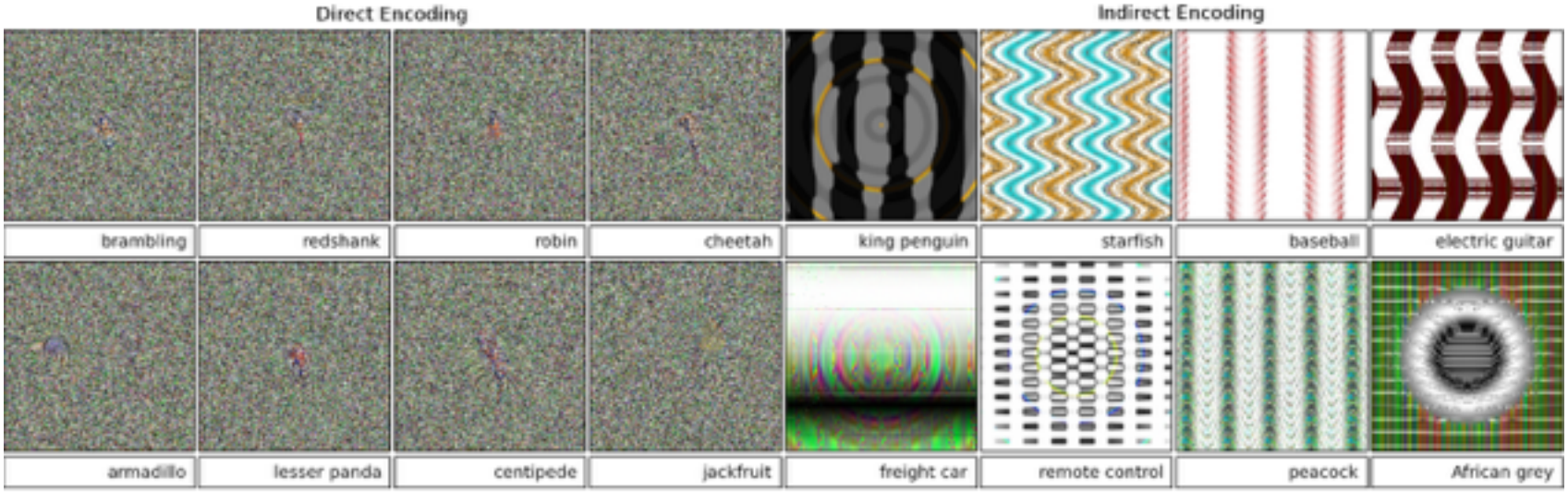


Do we need to move past gradient-based optimization to overcome adversarial examples?

Ubiquitous hallucinations



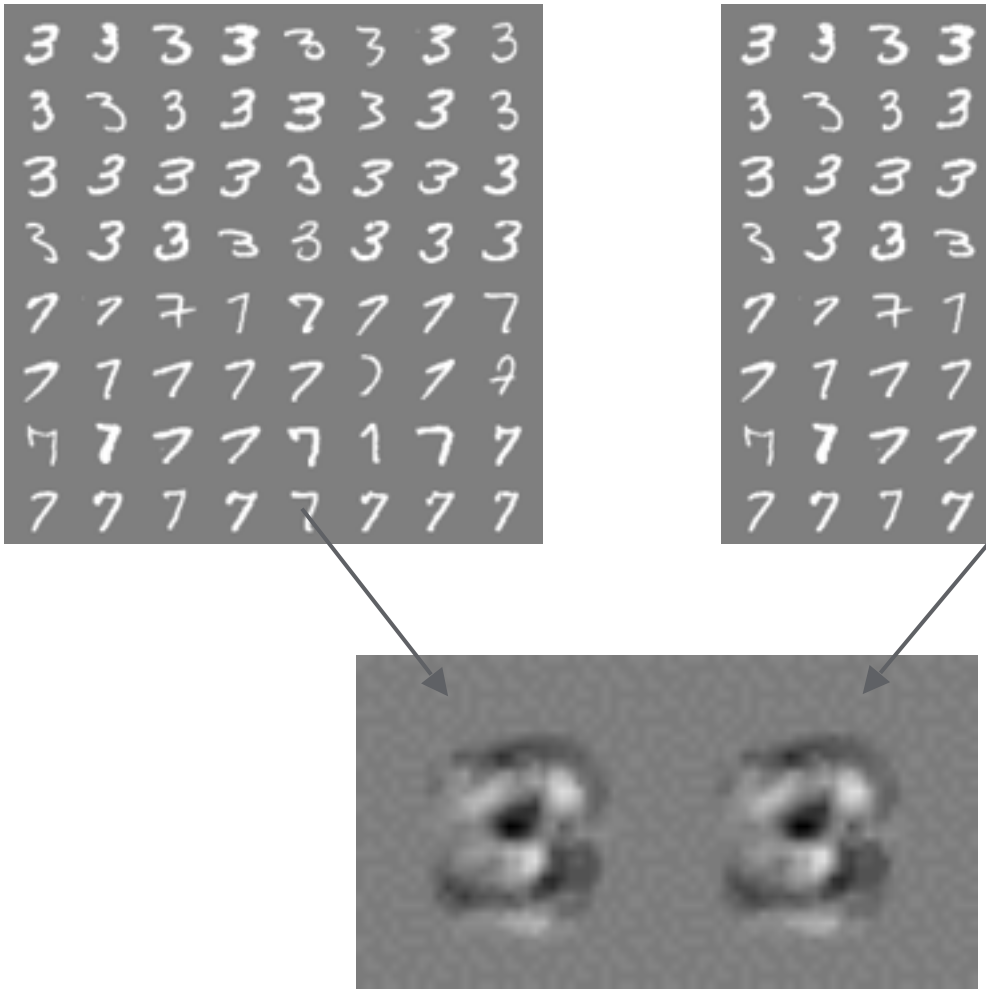
Methods based on expensive search, strong hand-designed priors



(Nguyen et al 2015)

(Olah 2015)

Cross-model, cross-dataset generalization

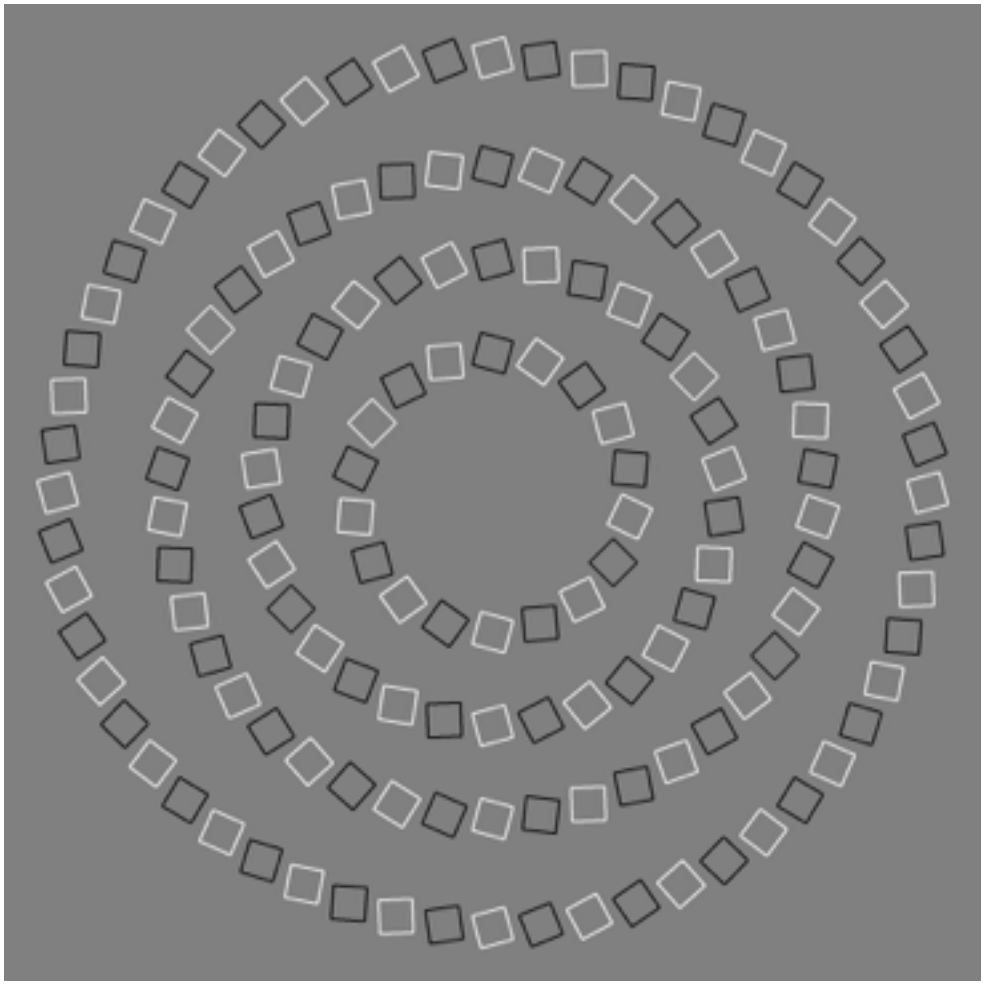


Cross-model, cross-dataset generalization

Neural net -> nearest neighbor: 25.3% error rate
Smoothed nearest neighbor -> nearest neighbor: 47.2% error rate
(a non-differentiable model doesn't provide much protection, it just requires the attacker to work indirectly)
Adversarially trained neural net -> nearest neighbor: 22.15% error rate
(Adversarially trained neural net -> self: 18% error rate)
Maxout net -> relu net: 99.4% error rate
agree on wrong class 85% of the time
Maxout net -> tanh net: 99.3% error rate
Maxout net -> softmax regression: 88.9% error rate
agree on wrong class 67% of the time
Maxout net -> shallow RBF: 36.8% error rate
agree on class 43% of the time

Google

Adversarial examples in the human visual system

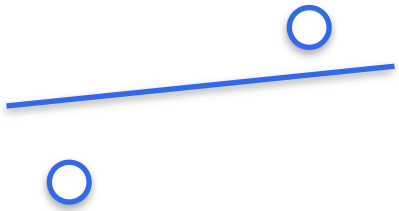


(Circles are concentric but appear intertwining)

(Pinna and Gregory, 2002)

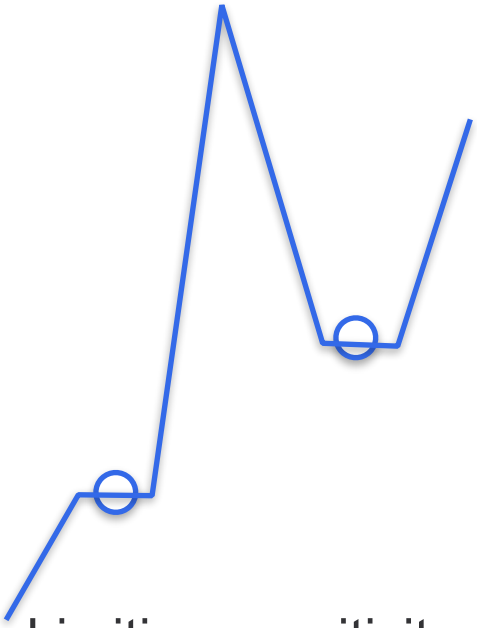
Failed defenses

- Defenses that fail due to cross-model transfer:
  - Ensembles
  - Voting after multiple saccades
- Other failed defenses:
  - Noise resistance
  - Generative modeling / unsupervised pretraining
  - Denoise the input with an autoencoder (Gu and Rigazio, 2014)
- Defenses that solve the adversarial task only if they break the clean task performance:
  - Weight decay (L1 or L2)
  - Jacobian regularization (like double backprop)
  - Deep RBF network



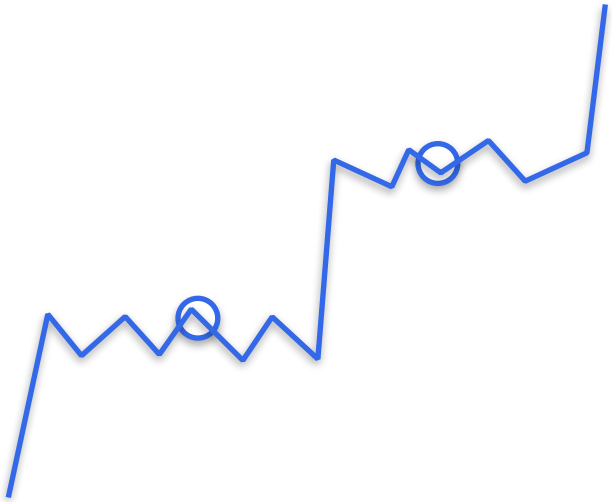
Limiting total variation (weight constraints)

Usually underfits before it solves the adversarial example problem.



Limiting sensitivity to infinitesimal perturbation (double backprop, CAE)

-Very hard to make the derivative close to 0
-Only provides constraint very near training examples, so does not solve adversarial examples.

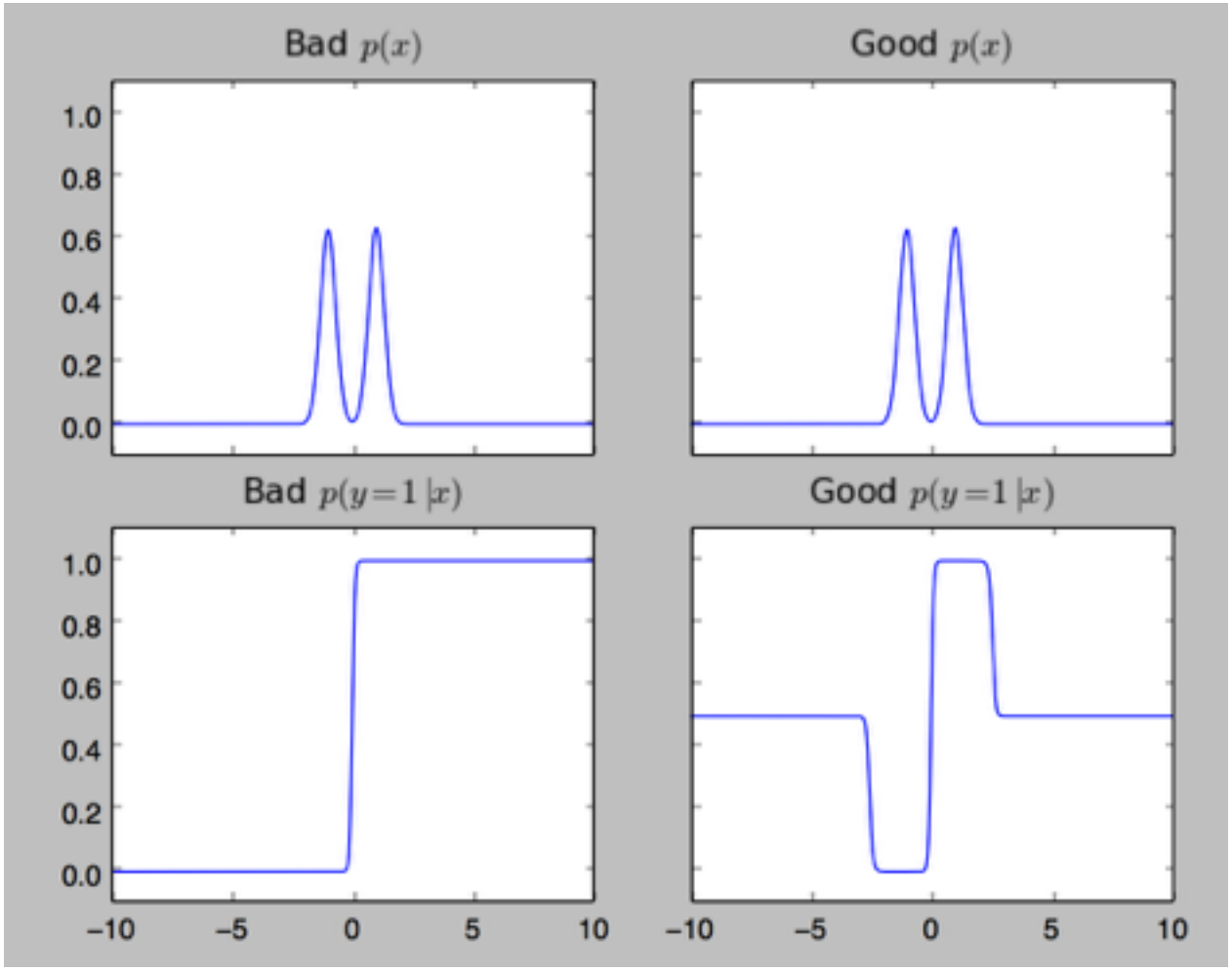


Limiting sensitivity to finite perturbation (adversarial training)

-Easy to fit because slope is not constrained
-Constrains function over a wide area

Generative modeling cannot solve the problem

Both these two class mixture models implement the same marginal over x, with totally different posteriors over the classes. The likelihood criterion can't prefer one to the other, and in many cases will prefer the bad one.

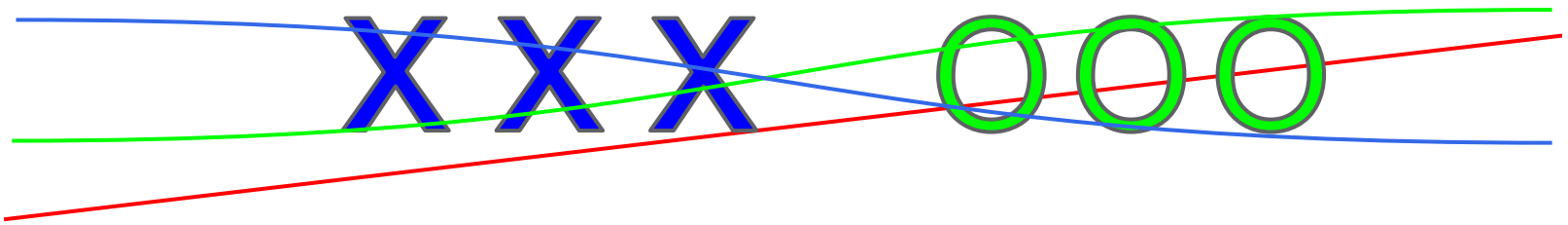


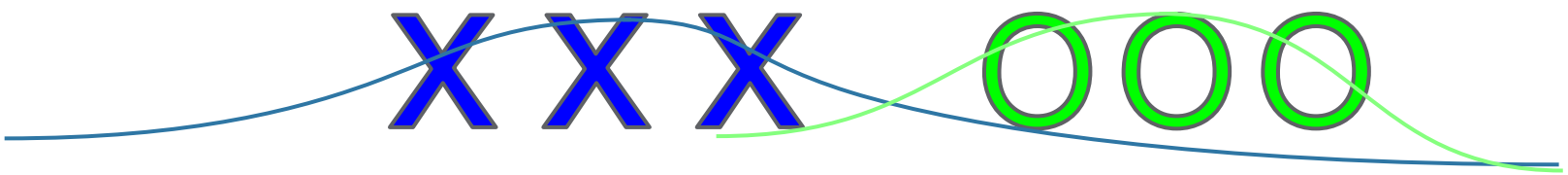
Security implications

- Must consider existence of adversarial examples when deciding whether to use machine learning
- Attackers can shut down a system that detects and refuses to process adversarial examples
- Attackers can control the output of a naive system
- Attacks can resemble regular data, or can appear to be unstructured noise, or can be structured but unusual
- Attacker does not need access to your model, parameters, or training set

Universal approximator theorem

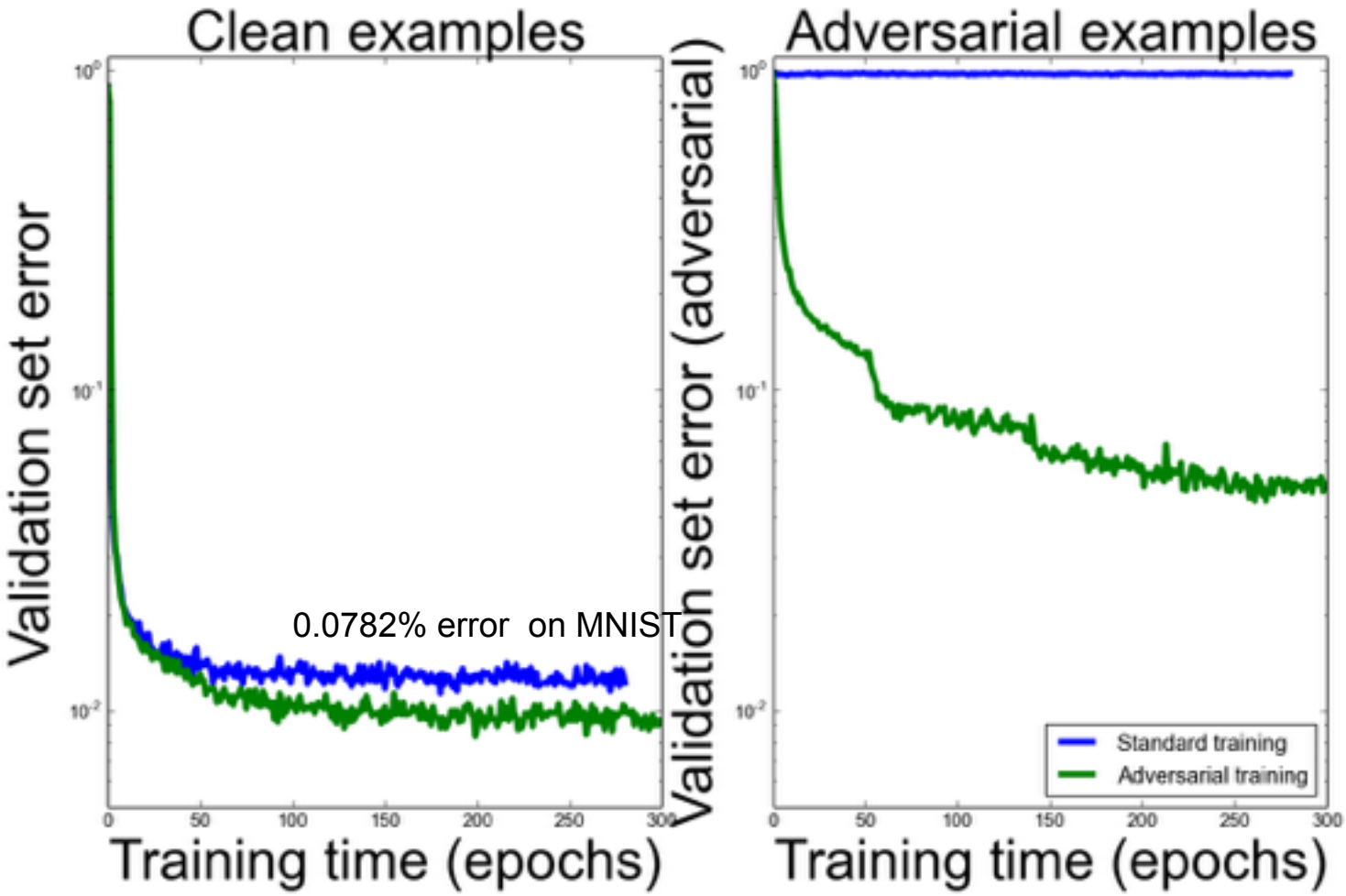
Neural nets can *represent* either function:





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

Training on adversarial examples

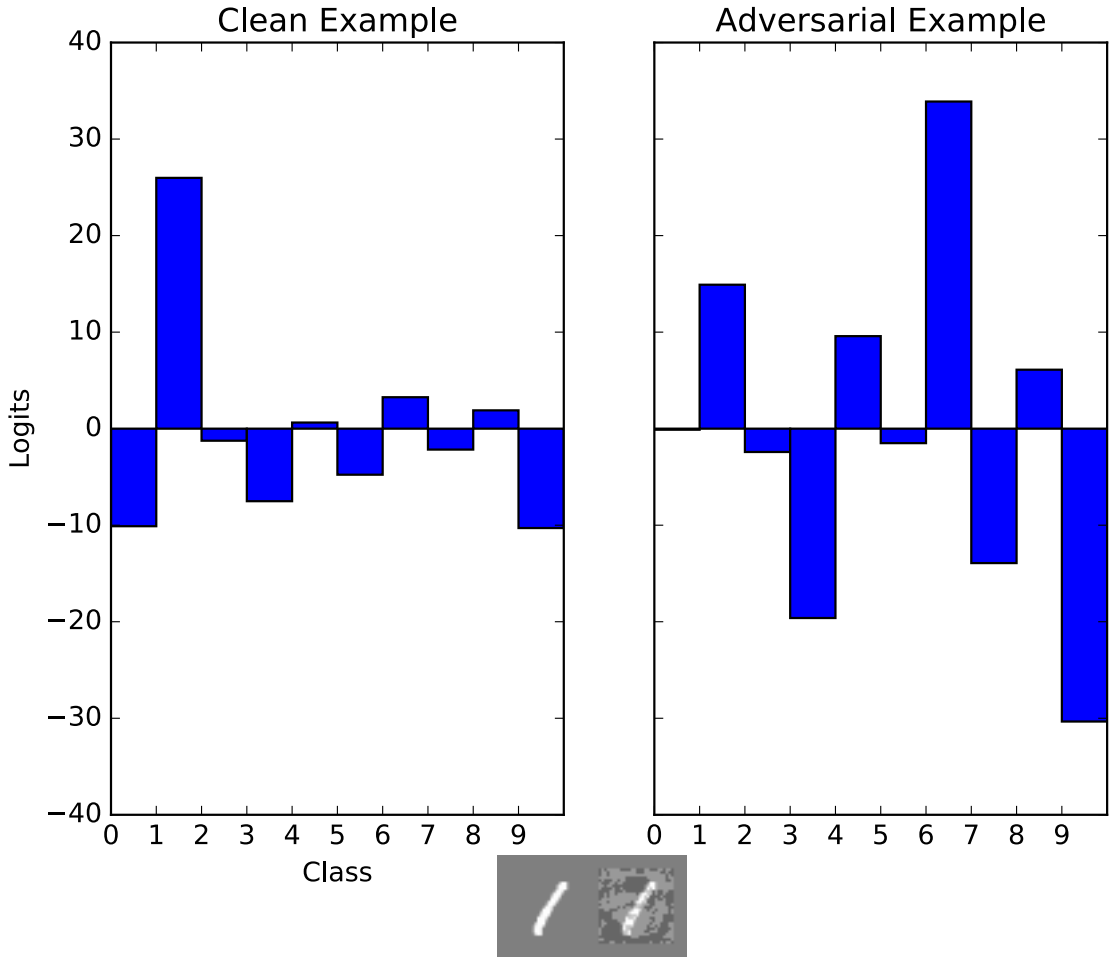


Google

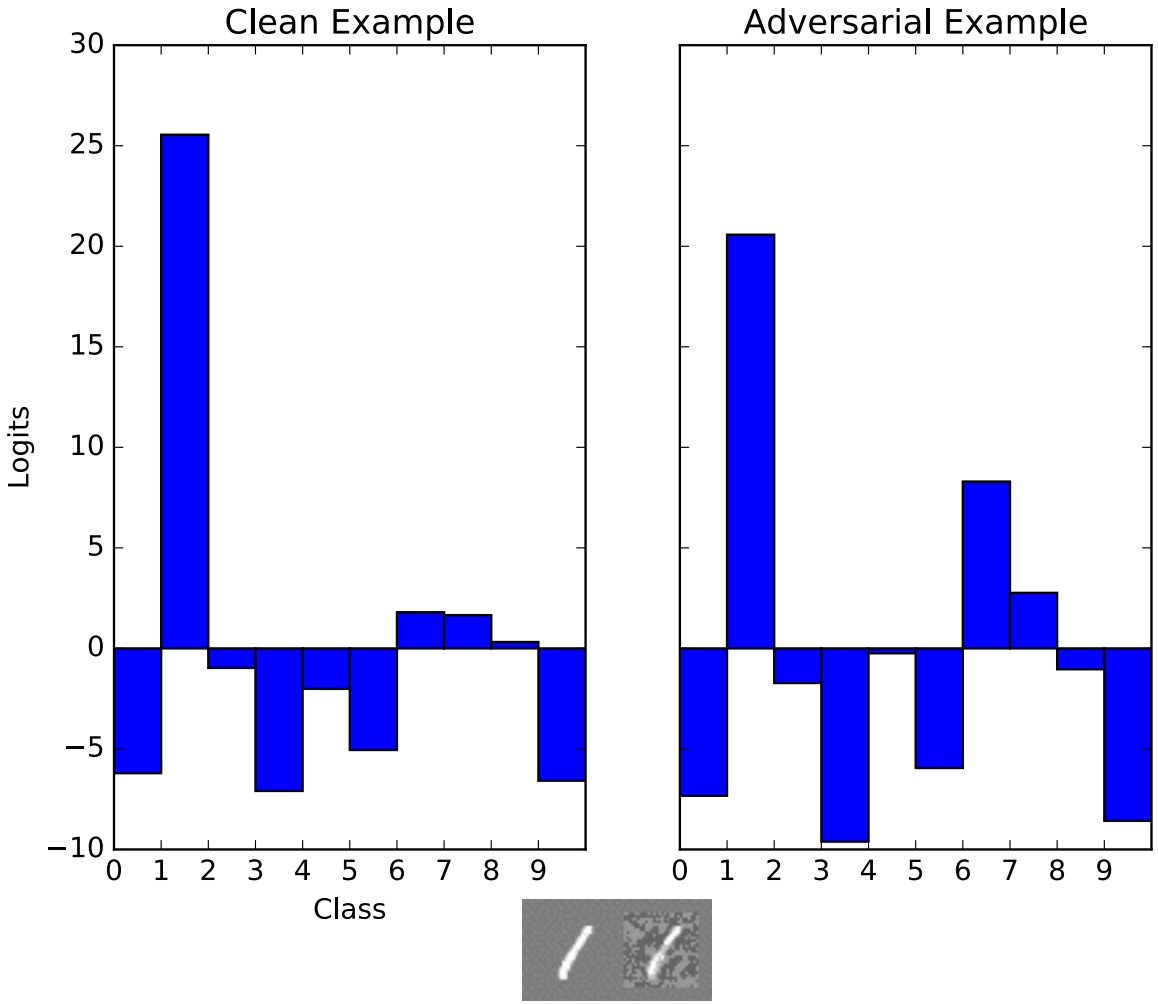
Weaknesses persist

More weaknesses

Pertubation's effect on class distributions



Pertubation's effect after adversarial training



Virtual adversarial training

- Penalize full KL divergence between predictions on clean and adversarial point
- Does not need y
- Semi-supervised learning
- MNIST results:

0.64% test error (statistically tied with state of the art)

100 examples:
VAE -> 3.33% error
Virtual Adversarial -> 2.12%
Ladder network -> 1.13%

Clearing up common misconceptions

- Inputs that the model processes incorrectly are *ubiquitous*, not rare, and occur most often in *half-spaces* rather than *pockets*
- Adversarial examples are *not* specific to deep learning
- Deep learning is uniquely able to overcome adversarial examples, due to the *universal approximator theorem*
- An attacker *does not* need access to a model or its training set
- Common off-the-shelf regularization techniques like model averaging and unsupervised learning *do not* automatically solve the problem

Please use evidence, not speculation

- It's common to say that obviously some technique will fix adversarial examples, and then just assume it will work without testing it
- It's common to say in the introduction to some new paper on regularizing neural nets that this regularization research is justifie because of adversarial examples
- Usually this is wrong
- Please actually test your method on adversarial examples and report the results
- Consider doing this even if you're not primarily concerned with adversarial examples

Recommended adversarial example benchmark

- Fix epsilon
- Compute the error rate on test data perturbed by the fast gradi sign method
- Report the error rate, epsilon, and the version of the model use for both forward and back-prop
- Alternative variant: design your own fixed-size perturbation scheme and report the error rate and size. For example, rotatio by some angle.

Alternative adversarial example benchmark

- Use L-BFGS or other optimizer
- Search for minimum size misclassified perturbation
- Report average size
- Report exhaustive detail to make the optimizer reproducible
- Downsides: computation cost, difficulty of reproducing, hard to guarantee the perturbations will really be mistakes

Recommended fooling image / rubbish class benchmark

- Fix epsilon
- Fit a Gaussian to the training inputs
- Draw samples from the Gaussian
- Perturb them toward a specific positive class with the fast grad sign method
- Report the rate at which you achieved this positive class
- Report the rate at which the model believed any specific non-rubbish class was present (probability of that class being prese exceeds 0.5)
- Report epsilon

Conclusion

- Many modern ML algorithms
  - get the right answer for the wrong reason on naturally occurring inputs
  - get very wrong answers *on almost all inputs*
- Adversarial training can expand the very narrow *manifold of accuracy*
- Deep learning is on course to overcome adversarial examples, but maybe only by expanding our “instruction book”
- Measure your model’s error rate on fast gradient sign method adversarial examples and report it!