Fooling neural networks

$+ .007 \times$

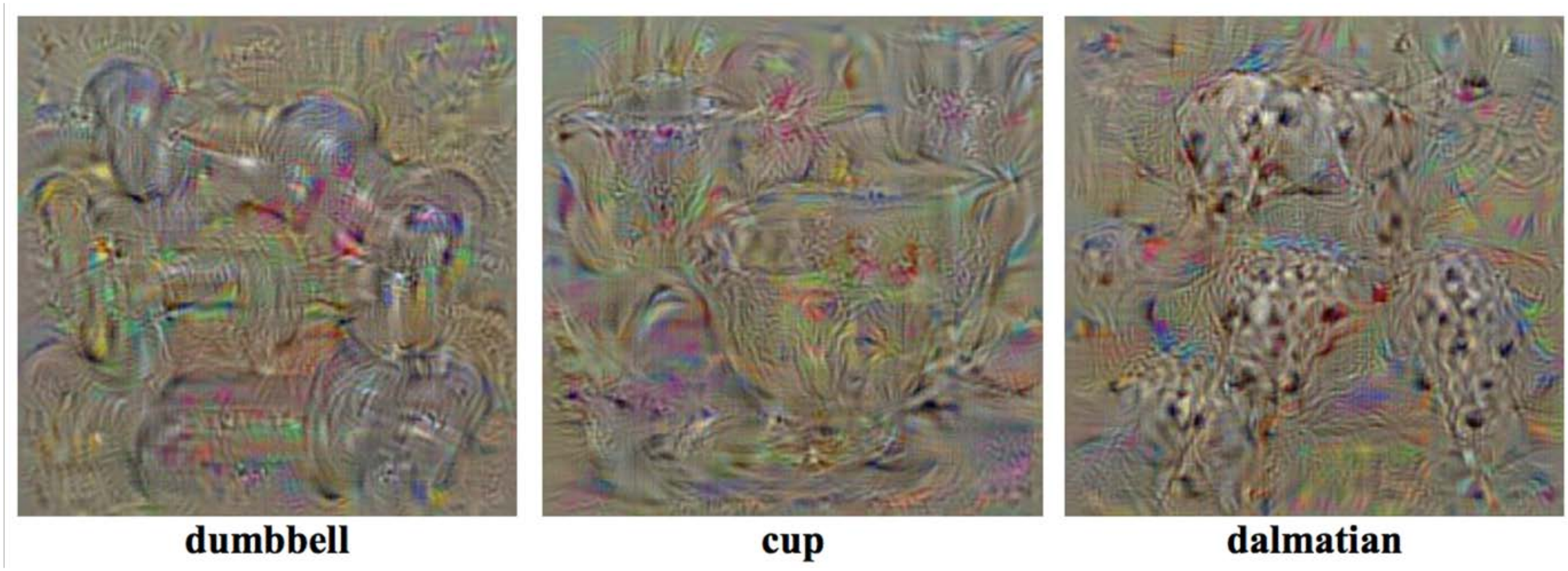
$=$

"panda"
57.7% confidence

"gibbon"
99.3 % confidence

Review: Generating preferred inputs

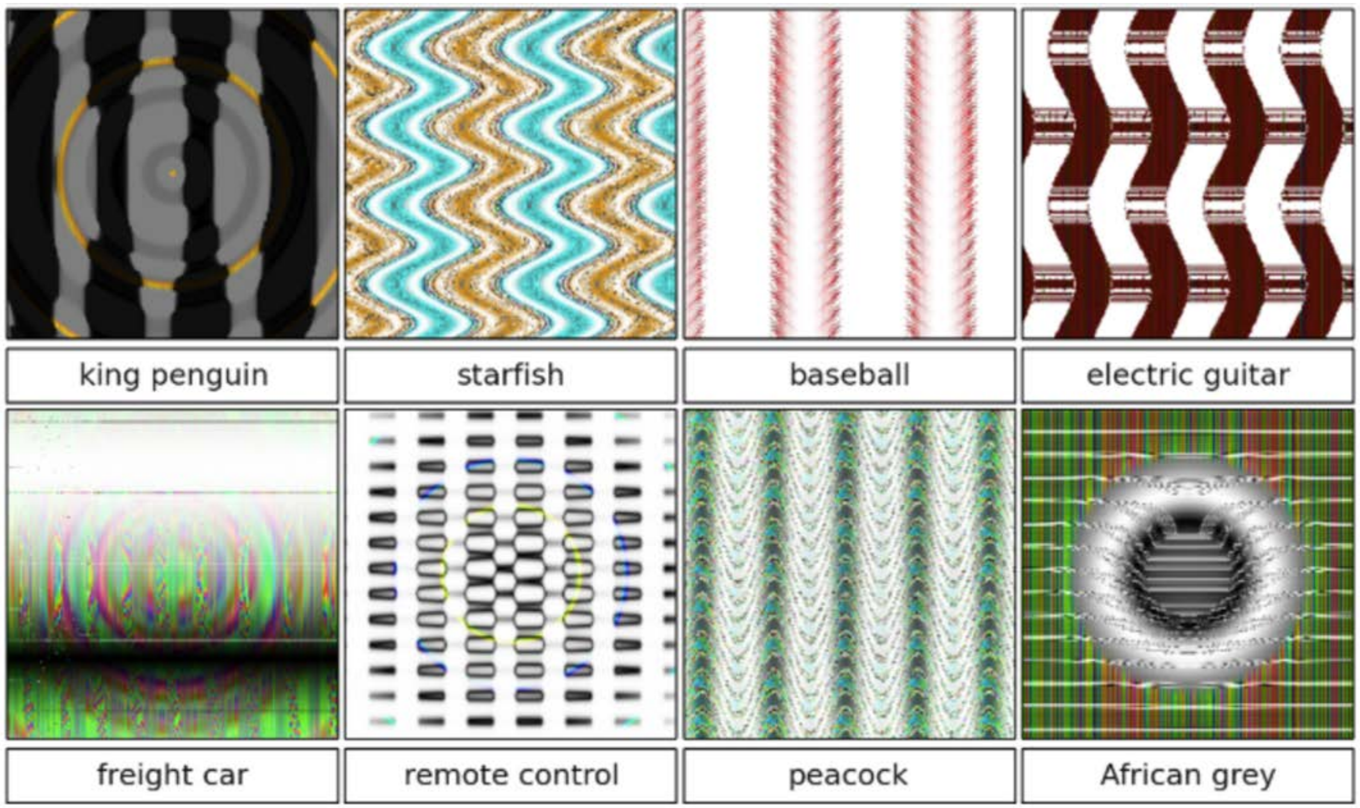
- We can use gradient ascent to generate weird-looking images to maximize activation of a given unit



K. Simonyan, A. Vedaldi, and A. Zisserman, Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, ICLR 2014

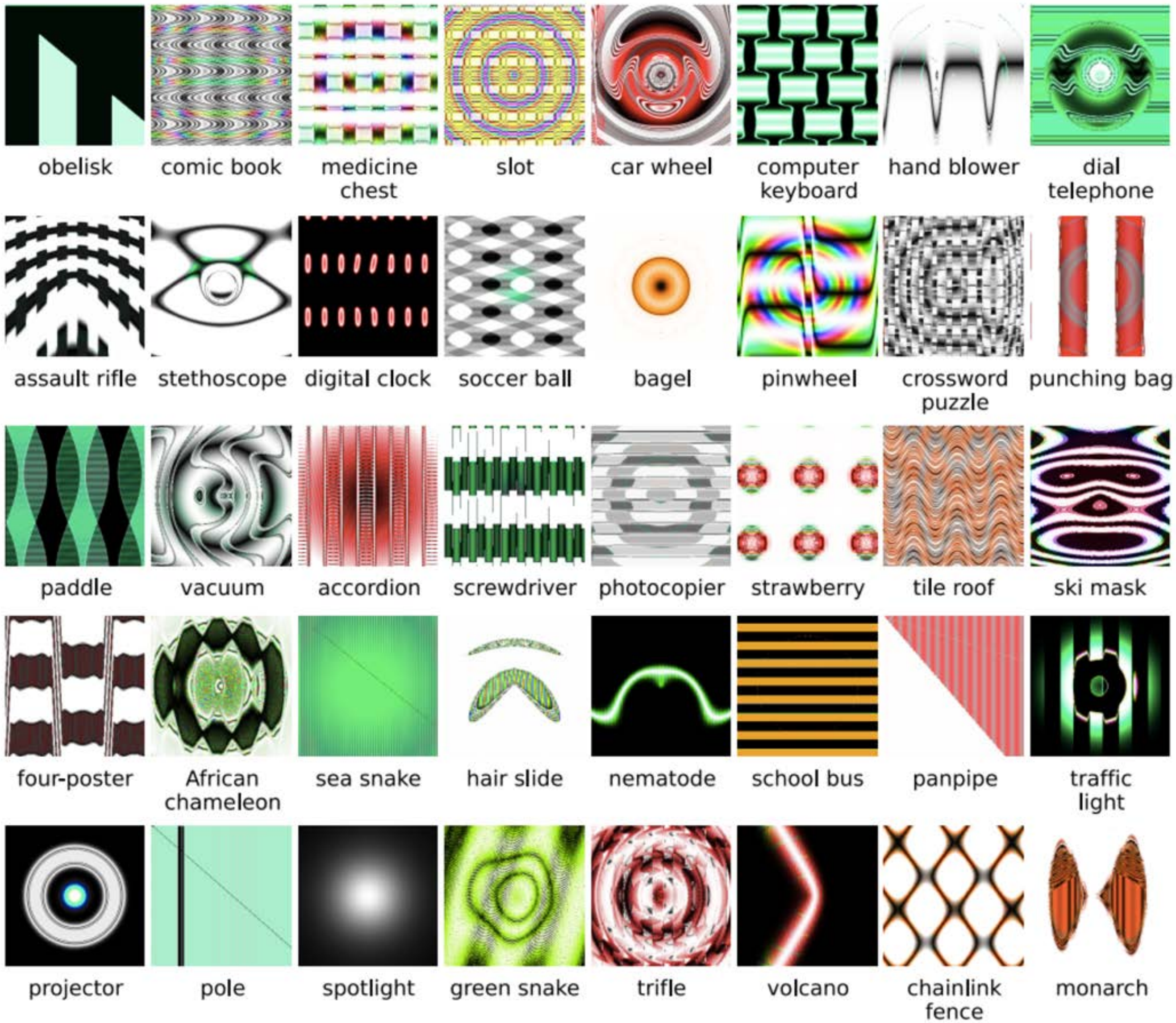
Generating preferred inputs

- Related finding: it is easy to generate perceptually meaningless images that will be classified as any given class with high confidence



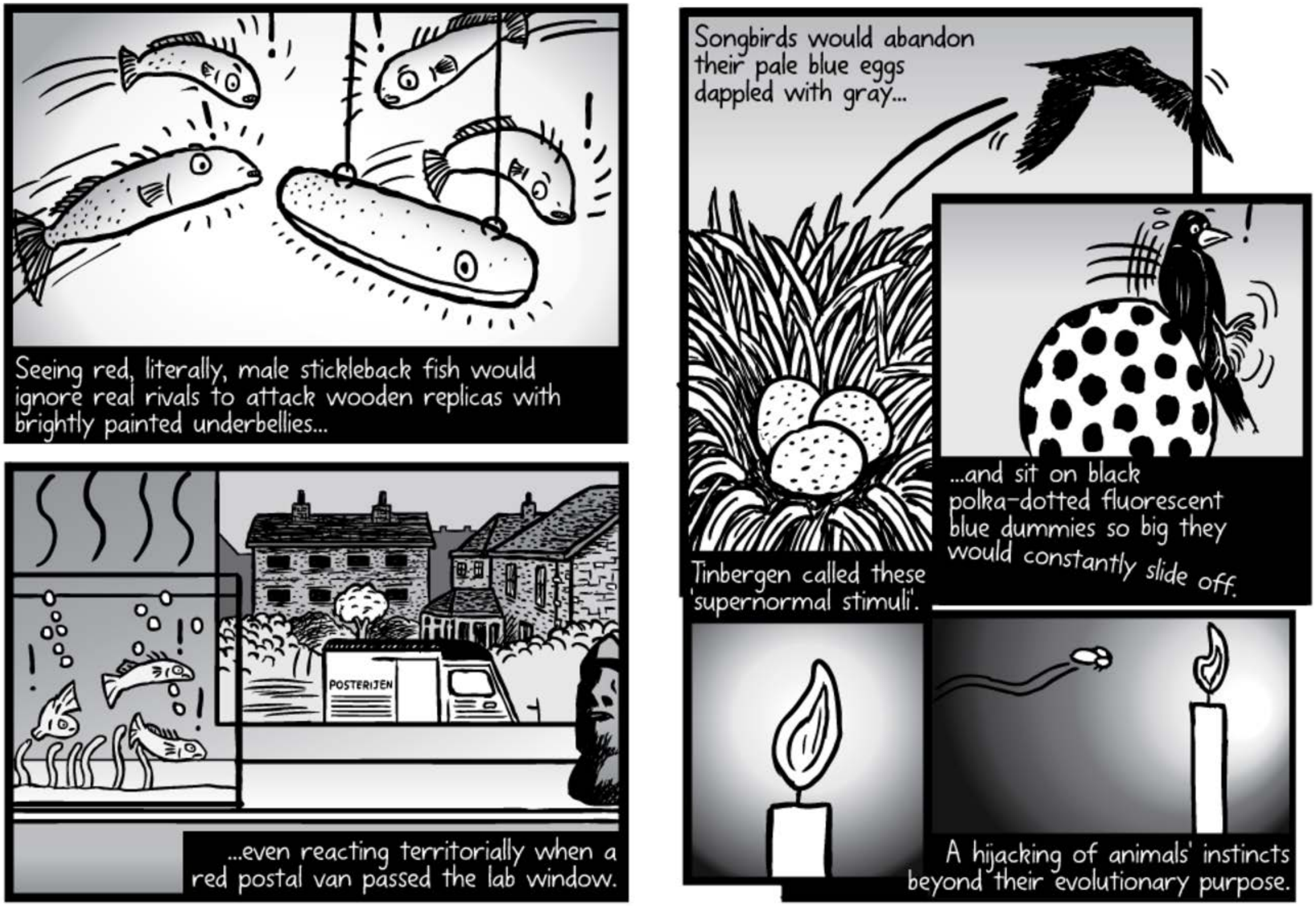
A. Nguyen, J. Yosinski, J. Clune, [Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images](), CVPR 2015

Generating preferred inputs



A. Nguyen, J. Yosinski, J. Clune, [Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images](), CVPR 2015

Biological phenomenon: Supernormal stimuli



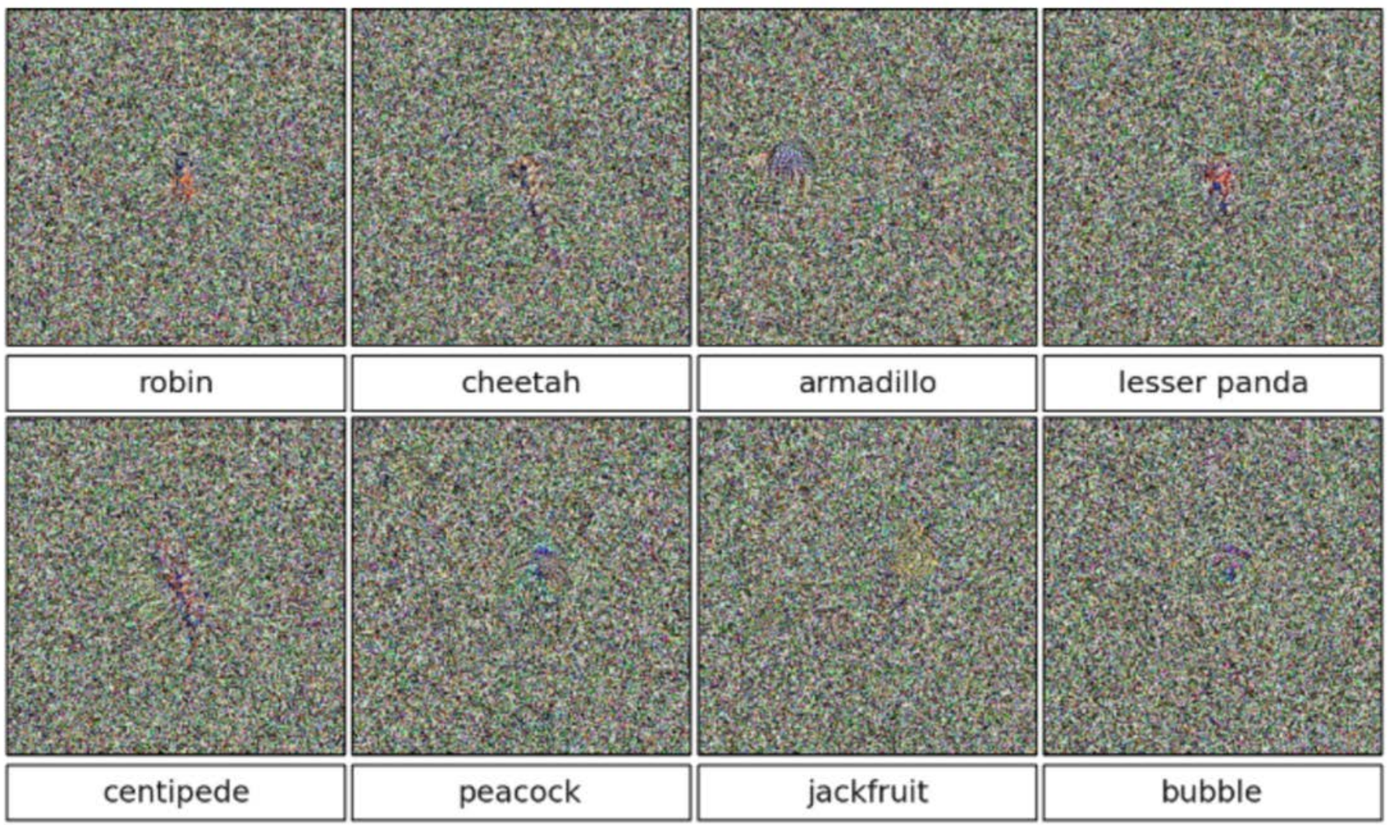
http://www.stuartmcmillen.com/comic/supernormal-stimuli/

https://en.wikipedia.org/wiki/Supernormal_stimulus

Supernormal stimuli for humans?

Generating preferred inputs

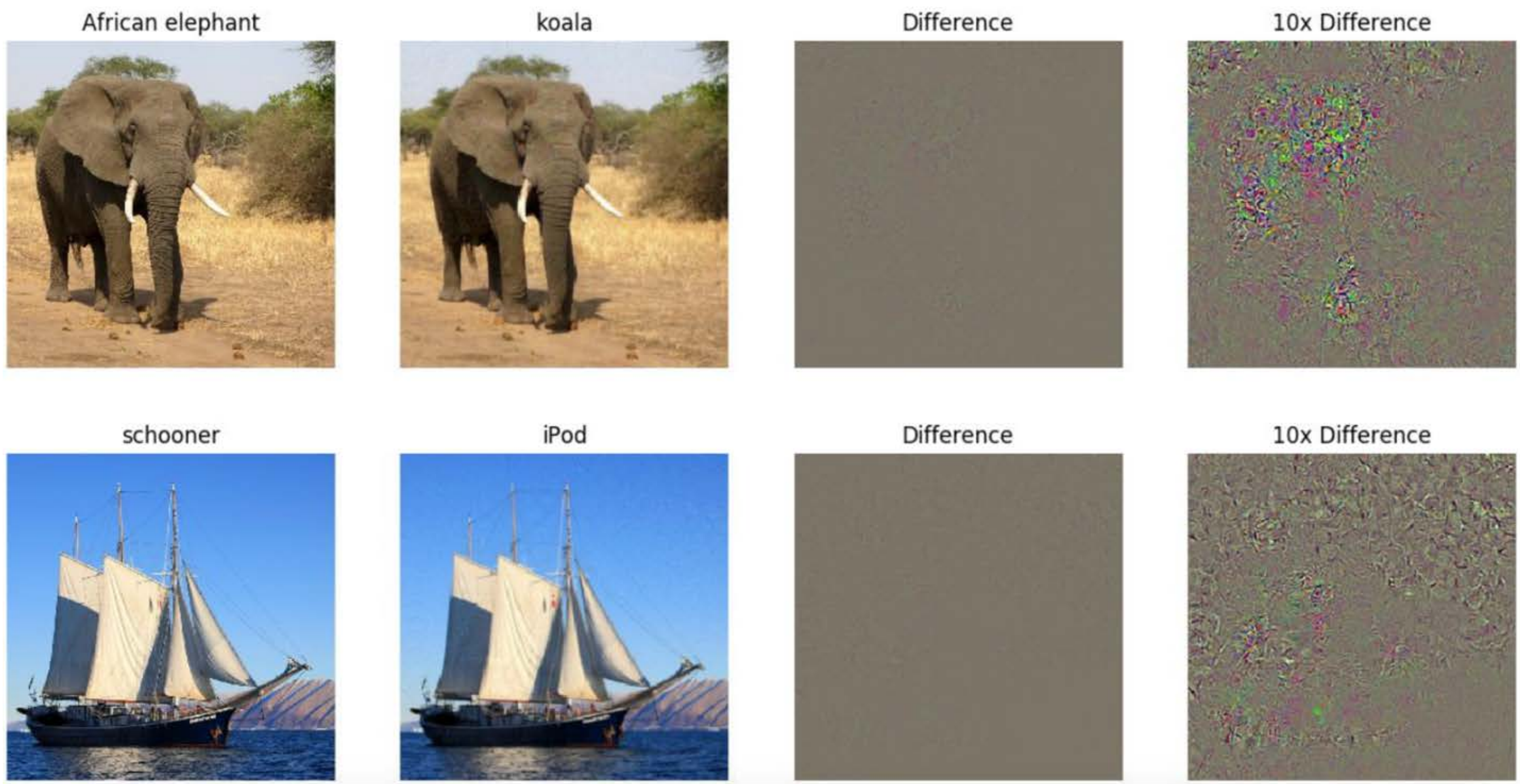
- It is easy to generate perceptually meaningless images that will be classified as any given class with high confidence



A. Nguyen, J. Yosinski, J. Clune, Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images, CVPR 2015

Adversarial examples

- We can “fool” a neural network by imperceptibly perturbing an input image so it is misclassified



Source: [Stanford CS231n]

Outline

- How to fool neural networks?
- Properties of adversarial examples
- Why are neural networks easy to fool?
- How to defend against being fooled?
- Fooling detectors

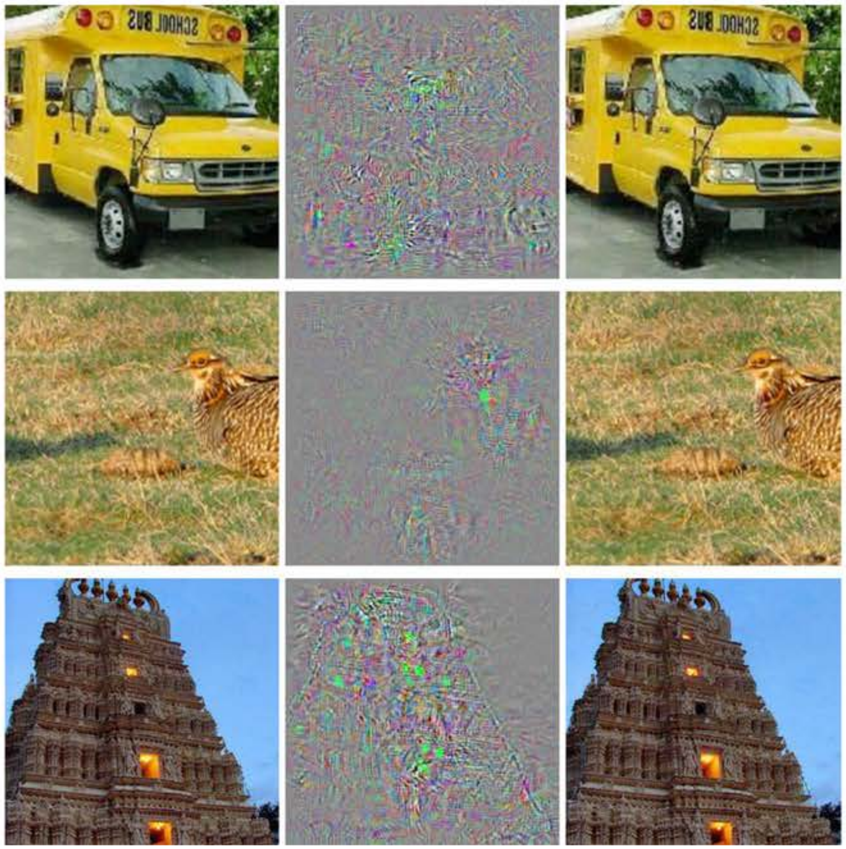
Finding the smallest adversarial perturbation

- Start with correctly classified image x
- Find perturbation r minimizing $\|r\|_2$ such that
  - $x + r$ is misclassified (or classified as specific target class)
  - All values of $x + r$ are in the valid range
- Constrained non-convex optimization, can be done using L-BFGS

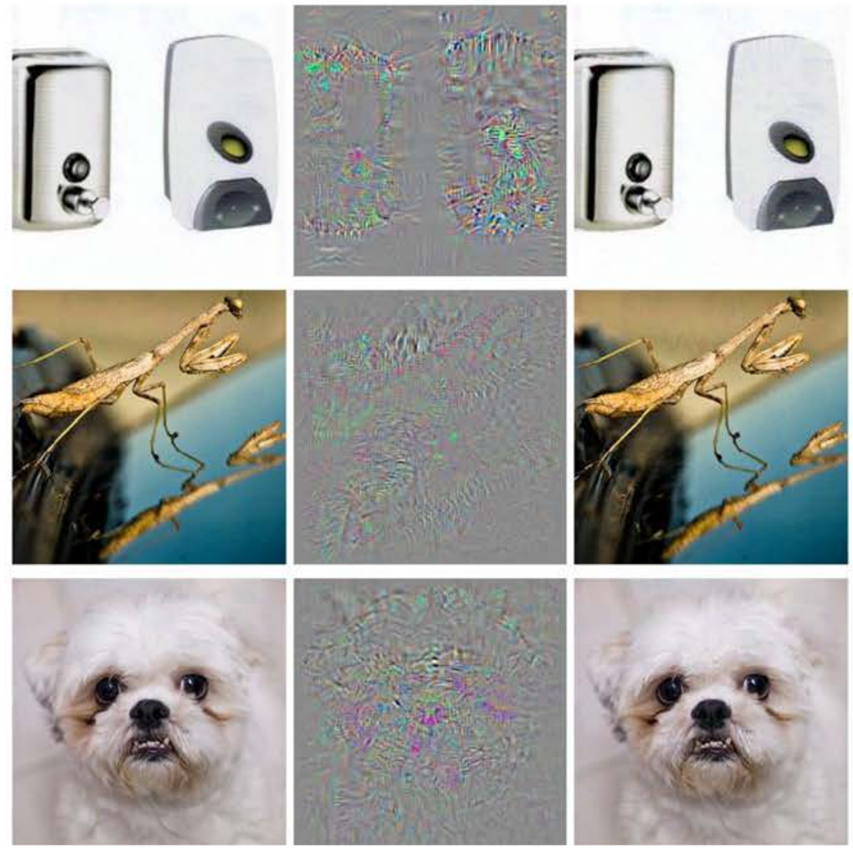
C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, ICLR 2014

Finding the smallest adversarial perturbation

- Sample results:



Input | Perturbation x 10 | "Ostrich"



Input | Perturbation x 10 | "Ostrich"

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, [Intriguing properties of neural networks](), ICLR 2014

Gradient ascent

- Rather than searching for the smallest possible perturbation, it is easier to take small gradient steps in desired direction
- Decrease score (increase loss) of correct class y^*:

$$x \leftarrow x - \eta \frac{\partial f(x,y^*)}{\partial x} \text{ or } x \leftarrow x + \eta \frac{\partial L(x,y^*)}{\partial x}$$

- Increase score (decrease loss) of incorrect target class $\hat{y}$:

$$x \leftarrow x + \eta \frac{\partial f(x,\hat{y})}{\partial x} \text{ or } x \leftarrow x - \eta \frac{\partial L(x,\hat{y})}{\partial x}$$

Fooling a linear classifier

- Increase score of target class $\hat{y}$:

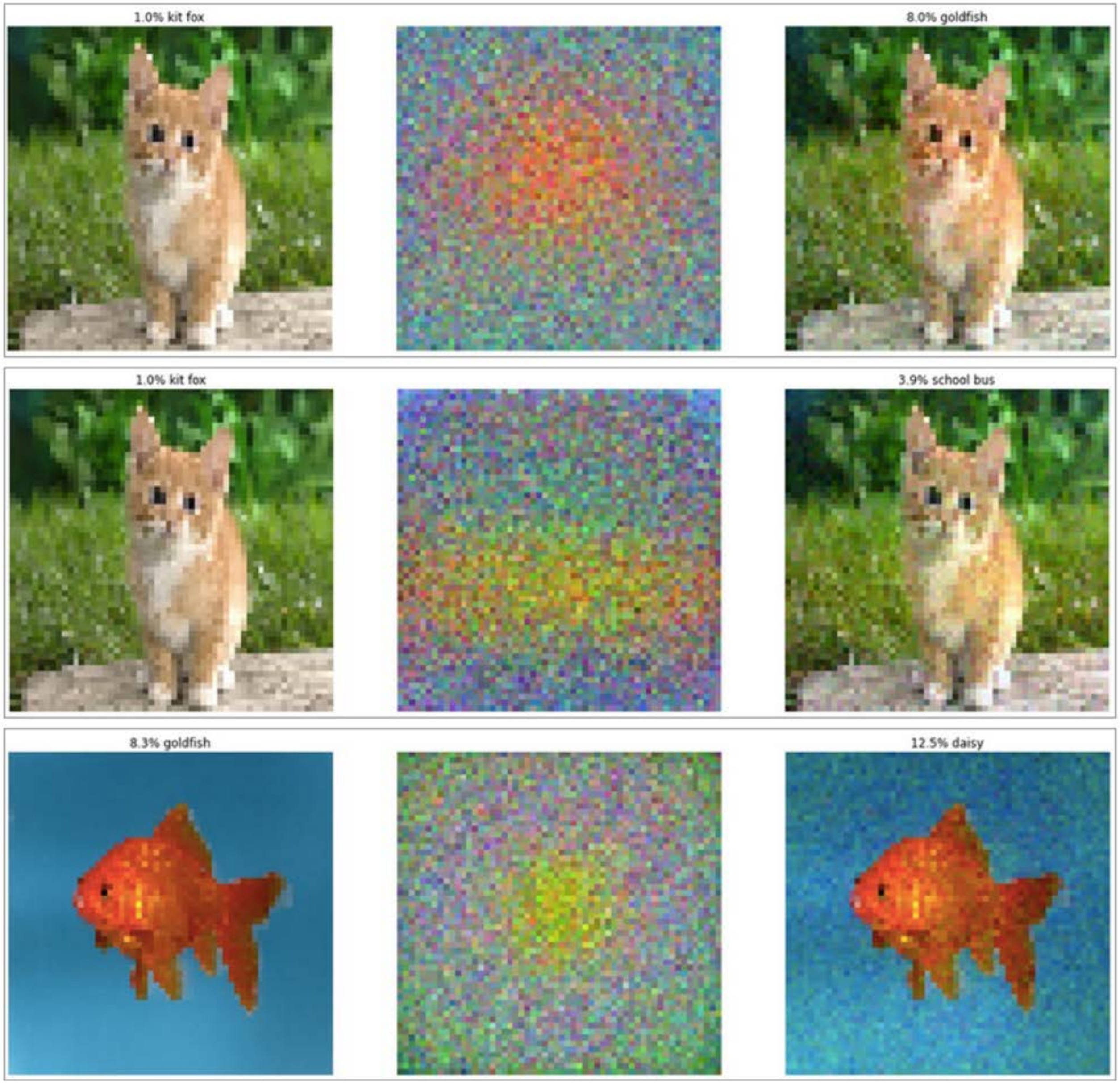
$$x \leftarrow x + \eta \frac{\partial f(x, \hat{y})}{\partial x}$$

- For a linear classifier with $f(x, \hat{y}) = w^T x$:

$$x \leftarrow x + \eta\, w$$

- To fool a linear classifier, add a small multiple of the target class weights to the test example

Fooling a linear classifier



http://karpathy.github.io/2015/03/30/breaking-convnets/

Analysis of the linear case

- Response of classifier with weights w to adversarial example $x + r$:

$$w^T(x + r) = w^T x + w^T r$$

- Suppose the pixel values have precision ϵ, i.e., the classifier is normally expected to predict the same class for x and $x + r$ as long as $\|r\|_\infty \le \epsilon$
- How to choose r to maximize the increase in activation $w^T r$ subject to $\|r\|_\infty \le \epsilon$?

$$r = \epsilon \operatorname{sgn}(w)$$

I. Goodfellow, J. Schlens, C. Szegedy, Explaining and harnessing adversarial examples, ICLR 2015

Analysis of the linear case

- Response of classifier with weights w to adversarial example $x + r$, $r = \epsilon \operatorname{sgn}(w)$:

$$w^T(x + r) = w^T x + \epsilon\, w^T \operatorname{sgn}(w)$$

- If w is d-dimensional and avg. element magnitude is m, how will the activation increase?
  - By $\epsilon d m$, i.e., linearly as a function of d
- The higher the dimensionality, the easier it is to make many small changes to the input that cause a large change in the output

I. Goodfellow, J. Schlens, C. Szegedy, Explaining and harnessing adversarial examples, ICLR 2015

Toy example

x	2	-1	3	-2	2	2	1	-4	5	1
w	-1	-1	1	-1	1	-1	1	1	-1	1

$$w^T x = -2 + 1 + 3 + 2 + 2 - 2 + 1 - 4 - 5 + 1 = -3$$

$$\sigma(w^T x) = \frac{1}{1 + e^{-(-3)}} = 0.047$$

http://karpathy.github.io/2015/03/30/breaking-convnets/

Toy example

x	2	-1	3	-2	2	2	1	-4	5	1
w	-1	-1	1	-1	1	-1	1	1	-1	1
$x + r$	1.5	-1.5	3.5	-2.5	2.5	1.5	1.5	-3.5	4.5	1.5

$$w^T x = -2 + 1 + 3 + 2 + 2 - 2 + 1 - 4 - 5 + 1 = -3$$

$$\sigma(w^T x) = \frac{1}{1 + e^{-(-3)}} = 0.047$$

$$w^T(x + r) = -3 + 10 * 0.5 = 2$$

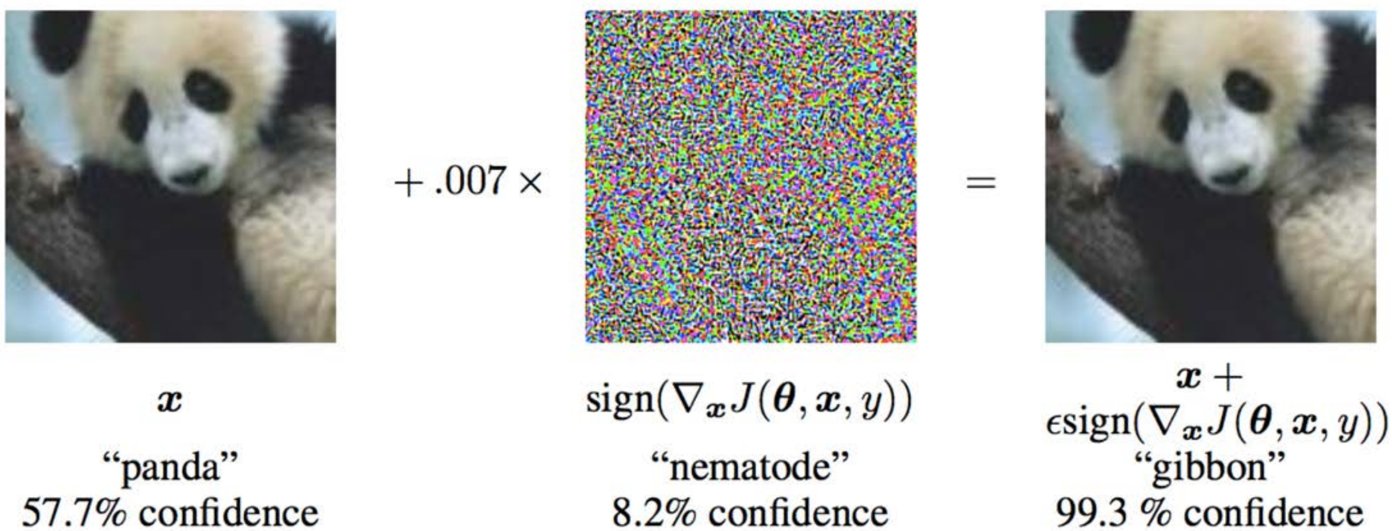
$$\sigma(w^T(x + r)) = \frac{1}{1 + e^{-2}} = 0.88$$

http://karpathy.github.io/2015/03/30/breaking-convnets/

Generating adversarial examples

- **Fast gradient sign method:** Find the gradient of the loss w.r.t. correct class y^*, take element-wise sign, update in resulting direction:

$$x \leftarrow x + \epsilon \,\mathrm{sgn}\left(\frac{\partial L(x, y^*)}{\partial x}\right)$$



I. Goodfellow, J. Schlens, C. Szegedy, [Explaining and harnessing adversarial examples](), ICLR 2015

Generating adversarial examples

- **Fast gradient sign method:**
$$x \leftarrow x + \epsilon \operatorname{sgn}\left(\frac{\partial L(x, y^*)}{\partial x}\right)$$
- **Iterative gradient sign method:** take multiple small steps until misclassified, each time clip result to be within ϵ-neighborhood of original image
- ***Least likely class* method:** try to misclassify image as class $\hat{y}$ with smallest initial score:
$$x \leftarrow x - \epsilon \operatorname{sgn}\left(\frac{\partial L(x, \hat{y})}{\partial x}\right)$$

A. Kurakin, I. Goodfellow, S. Bengio, [Adversarial examples in the real world](), ICLR 2017 workshop

Generating adversarial examples

Comparison of methods for $\epsilon = 32$

Clean image | “Fast”; L_∞ distance to clean image = 32

“Basic iter.”; L_∞ distance to clean image = 32 | “L.l. class”; L_∞ distance to clean image = 28

A. Kurakin, I. Goodfellow, S. Bengio, [Adversarial examples in the real world](), ICLR 2017 workshop

Generating adversarial examples

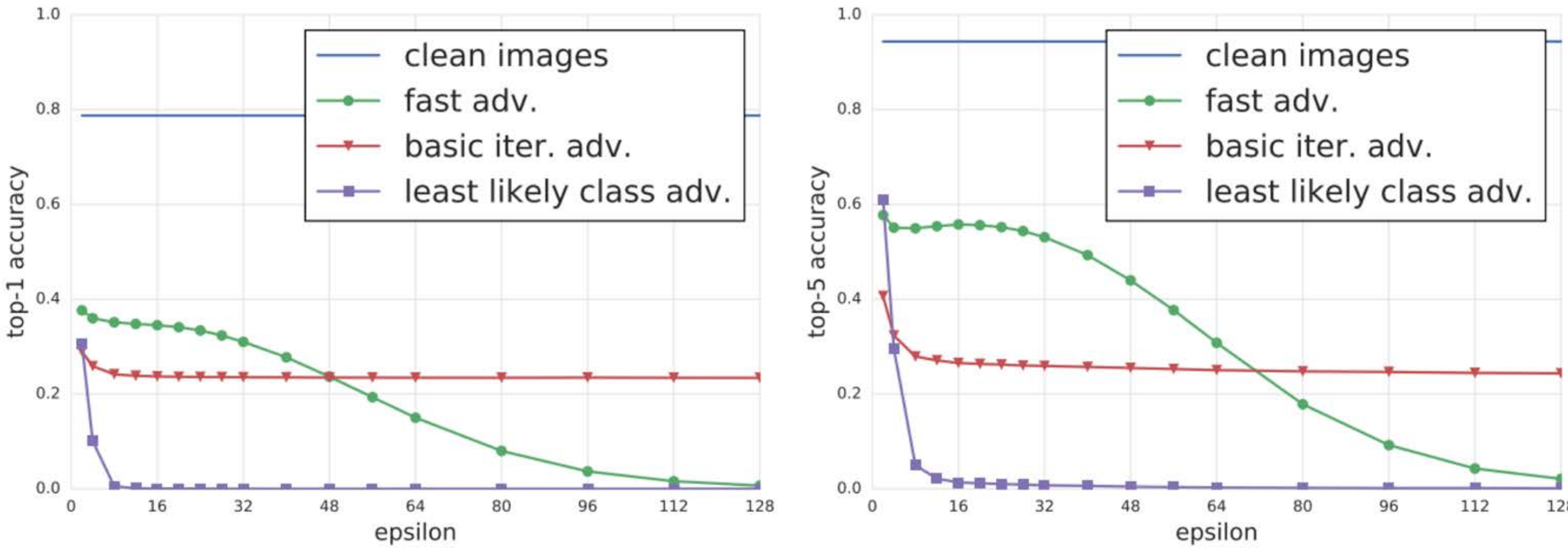
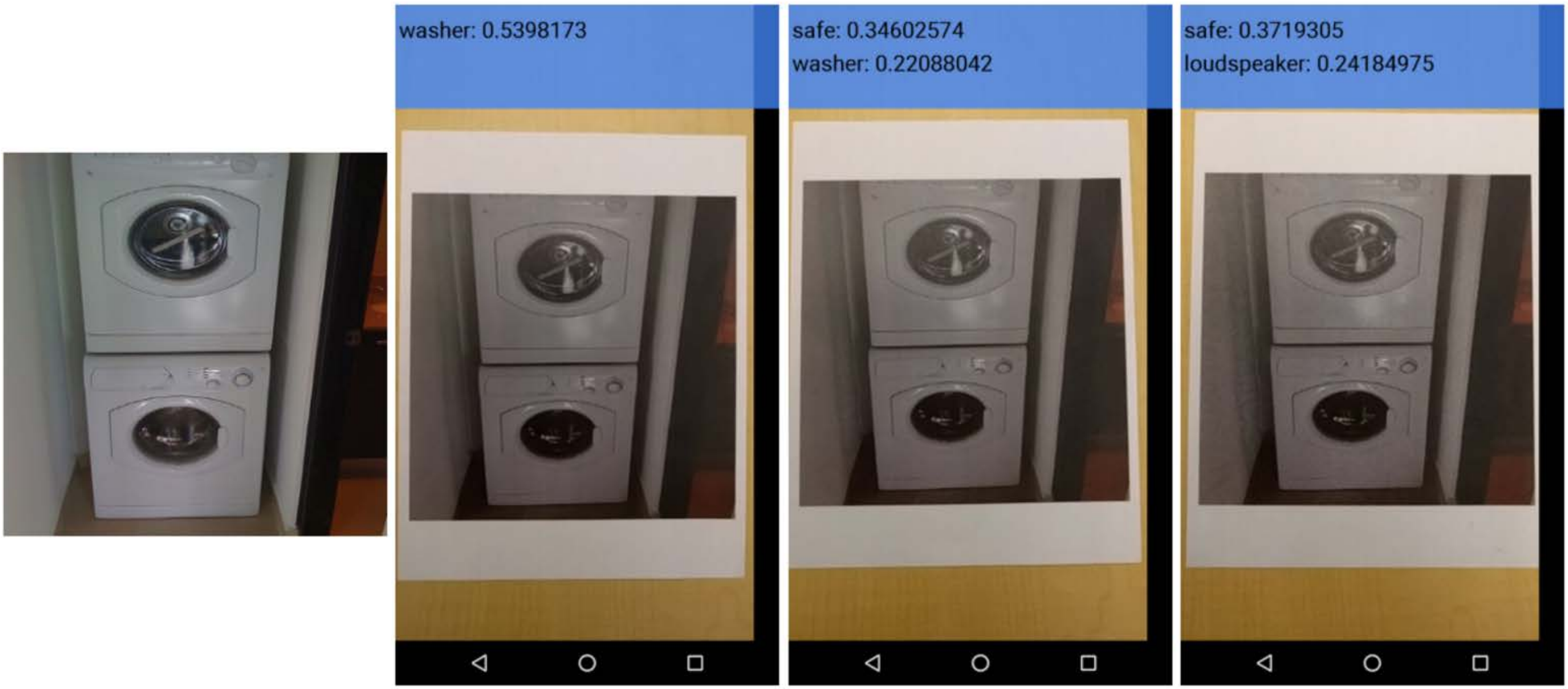


Figure 2: Top-1 and top-5 accuracy of Inception v3 under attack by different adversarial methods and different ϵ compared to "clean images" — unmodified images from the dataset. The accuracy was computed on all $50,000$ validation images from the ImageNet dataset. In these experiments ϵ varies from 2 to 128.

A. Kurakin, I. Goodfellow, S. Bengio, [Adversarial examples in the real world](), ICLR 2017 workshop

Printed adversarial examples

- “Black box” attack on a cell phone app:



(a) Image from dataset (b) Clean image (c) Adv. image, $\epsilon = 4$ (d) Adv. image, $\epsilon = 8$

A. Kurakin, I. Goodfellow, S. Bengio, [Adversarial examples in the real world](), ICLR 2017 workshop

Printed adversarial examples

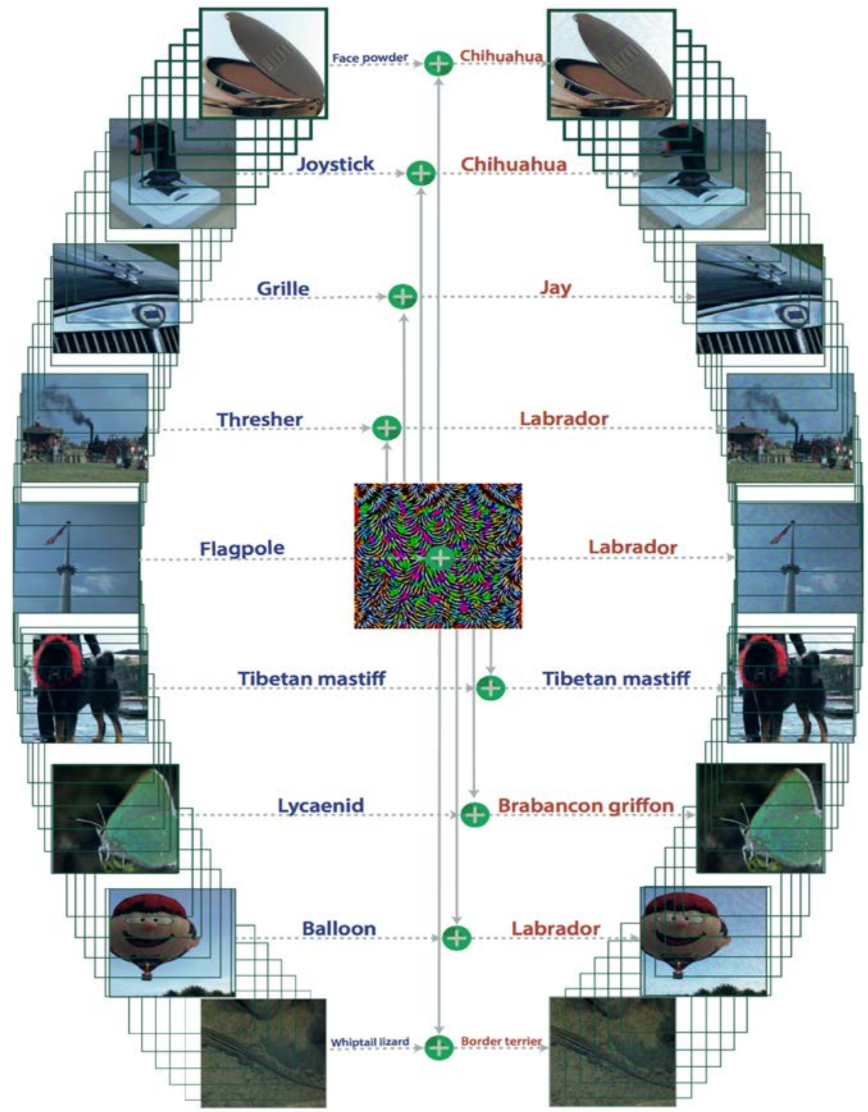
- Accuracies for printed vs. digital images:

Adversarial method	Photos				Source images			
	Clean images		Adv. images		Clean images		Adv. images	
	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5
fast $\epsilon = 16$	81.8%	97.0%	5.1%	39.4%	100.0%	100.0%	0.0%	0.0%
fast $\epsilon = 8$	77.1%	95.8%	14.6%	70.8%	100.0%	100.0%	0.0%	0.0%
fast $\epsilon = 4$	81.4%	100.0%	32.4%	91.2%	100.0%	100.0%	0.0%	0.0%
fast $\epsilon = 2$	88.9%	99.0%	49.5%	91.9%	100.0%	100.0%	0.0%	0.0%
iter. basic $\epsilon = 16$	93.3%	97.8%	60.0%	87.8%	100.0%	100.0%	0.0%	0.0%
iter. basic $\epsilon = 8$	89.2%	98.0%	64.7%	91.2%	100.0%	100.0%	0.0%	0.0%
iter. basic $\epsilon = 4$	92.2%	97.1%	77.5%	94.1%	100.0%	100.0%	0.0%	0.0%
iter. basic $\epsilon = 2$	93.9%	97.0%	80.8%	97.0%	100.0%	100.0%	0.0%	1.0%
l.l. class $\epsilon = 16$	95.8%	100.0%	87.5%	97.9%	100.0%	100.0%	0.0%	0.0%
l.l. class $\epsilon = 8$	96.0%	100.0%	88.9%	97.0%	100.0%	100.0%	0.0%	0.0%
l.l. class $\epsilon = 4$	93.9%	100.0%	91.9%	98.0%	100.0%	100.0%	0.0%	0.0%
l.l. class $\epsilon = 2$	92.2%	99.0%	93.1%	98.0%	100.0%	100.0%	0.0%	0.0%

A. Kurakin, I. Goodfellow, S. Bengio, Adversarial examples in the real world, ICLR 2017 workshop

Universal adversarial perturbations

- Goal: for a given network, find an *image-independent* perturbation vector that causes *all images* to be misclassified with high probability



S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, [Universal adversarial perturbations](), CVPR 2017

Universal adversarial perturbations

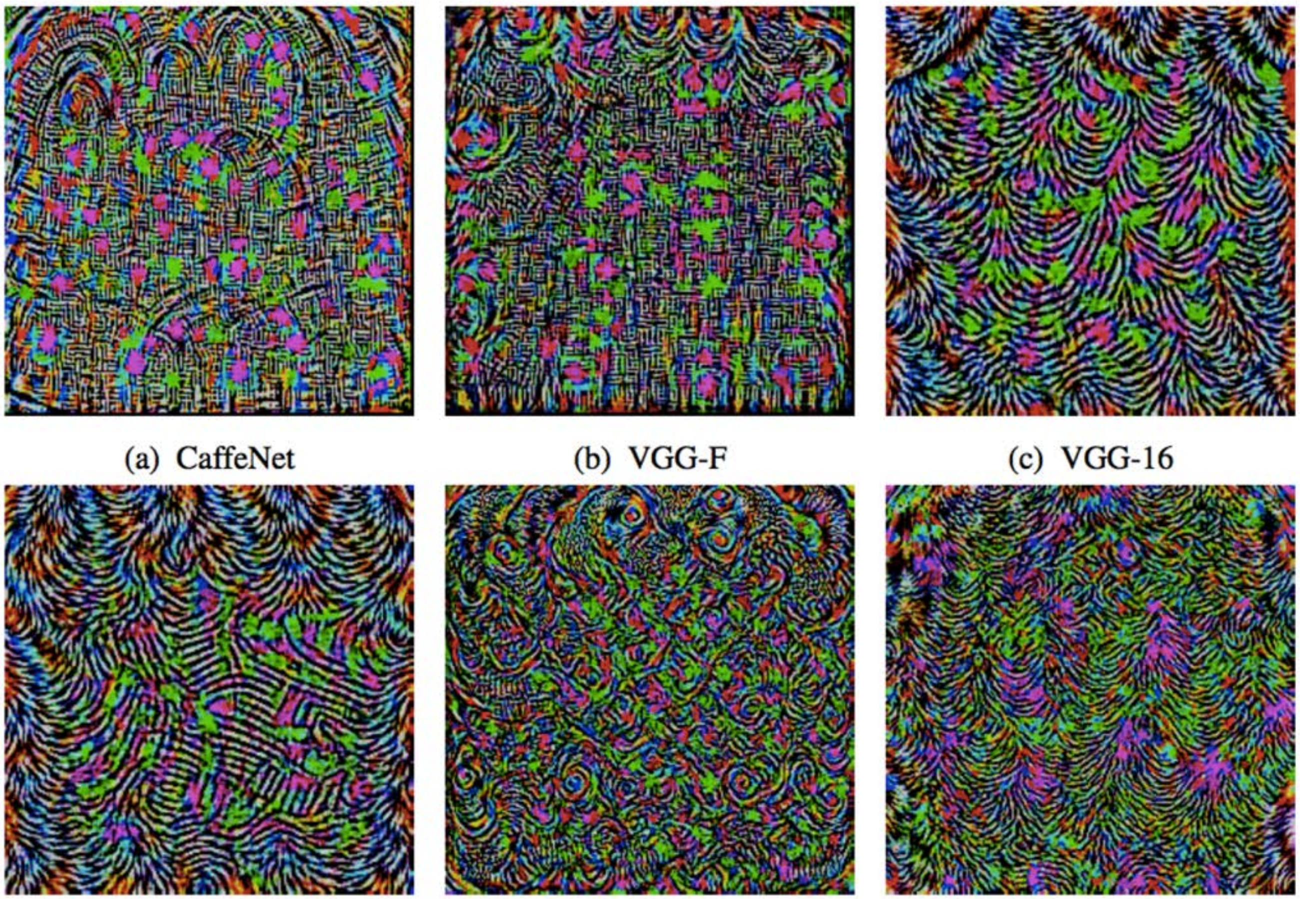
Approach:

- Start with $r = 0$
- Cycle through training examples x_i (in multiple passes)
  - If $x_i + r$ is misclassified, skip to x_{i+1}
  - Find minimum perturbation Δr that takes $x_i + r + \Delta r$ to another class
  - Update $r \leftarrow r + \Delta r$, enforce $\|r\| \leq \epsilon$
- Terminate when fooling rate on training examples reaches target value

S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, [Universal adversarial perturbations](), CVPR 2017

Universal adversarial perturbations

- Perturbation vectors computed from different architectures:

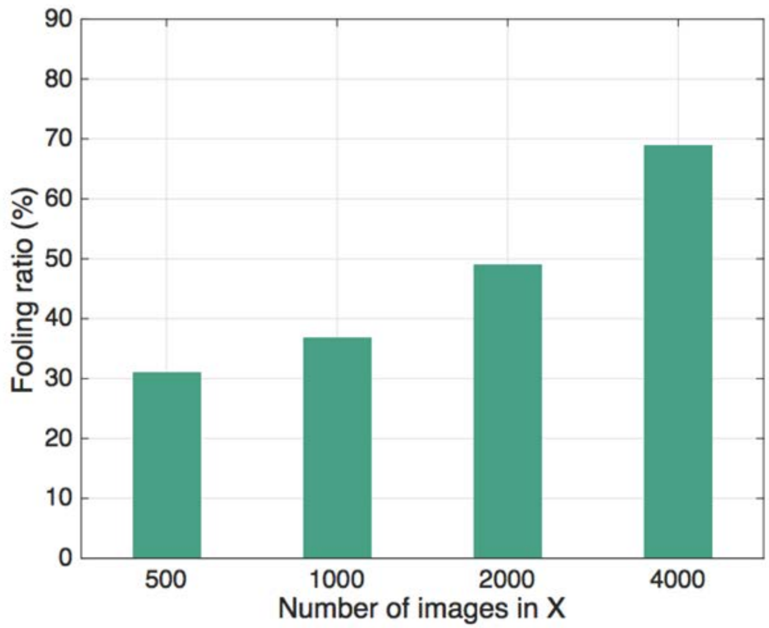


(a) CaffeNet (b) VGG-F (c) VGG-16

(d) VGG-19 (e) GoogLeNet (f) ResNet-152

S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, [Universal adversarial perturbations](), CVPR 2017

Universal adversarial perturbations



Fooling ratio on validation set vs. training set size for GoogLeNet

Fooling rates on different models after training on 10,000 images

		CaffeNet [8]	VGG-F [2]	VGG-16 [17]	VGG-19 [17]	GoogLeNet [18]	ResNet-152 [6]
ℓ_2	X	85.4%	85.9%	90.7%	86.9%	82.9%	89.7%
	Val.	85.6	87.0%	90.3%	84.5%	82.0%	88.5%
ℓ_∞	X	93.1%	93.8%	78.5%	77.8%	80.8%	85.4%
	Val.	93.3%	93.7%	78.3%	77.8%	78.9%	84.0%

S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, [Universal adversarial perturbations](), CVPR 2017

Universal adversarial perturbations

- Universal perturbations turn out to generalize well across models!

Fooling rate when computing a perturbation for one model (rows) and testing it on others (columns)

	VGG-F	CaffeNet	GoogLeNet	VGG-16	VGG-19	ResNet-152
VGG-F	**93.7%**	71.8%	48.4%	42.1%	42.1%	47.4 %
CaffeNet	74.0%	**93.3%**	47.7%	39.9%	39.9%	48.0%
GoogLeNet	46.2%	43.8%	**78.9%**	39.2%	39.8%	45.5%
VGG-16	63.4%	55.8%	56.5%	**78.3%**	73.1%	63.4%
VGG-19	64.0%	57.2%	53.6%	73.5%	**77.8%**	58.0%
ResNet-152	46.3%	46.3%	50.5%	47.0%	45.5%	**84.0%**

S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, Universal adversarial perturbations, CVPR 2017

Black-box adversarial examples

- Suppose the adversary can only query a target network with chosen inputs and observe the outputs
- Key idea: learn substitute for target network using synthetic input data, use substitute network to craft adversarial examples
- Successfully attacked third-party APIs from MetaMind, Amazon, and Google, but only on low-res digit and street sign images

N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, A. Swami, [Practical Black-Box Attacks against Machine Learning](), ACM Asia Conference on Computer and Communications Security, 2017

Properties of adversarial examples

- For any input image, it is usually easy to generate a very similar image that gets misclassified by the same network
- To obtain an adversarial example, one does not need to do precise gradient ascent
- Adversarial images can (sometimes) survive transformations like being printed and photographed
- It is possible to attack many images with the same perturbation
- Adversarial examples that can fool one network have a high chance of fooling a network with different parameters and even architecture

Why are deep networks easy to fool?

- Networks are “too linear”: it is easy to manipulate output in a predictable way given the input
- The input dimensionality is high, so one can get a large change in the output by changing individual inputs by small amounts
- Neural networks can fit anything, but nothing prevents them from behaving erratically between training samples
  - Counter-intuitively, a network can both generalize well on natural images and be susceptible to adversarial examples
- Adversarial examples generalize well because different models learn similar functions when trained to perform the same task?

Defending against adversarial examples

Defending against adversarial examples

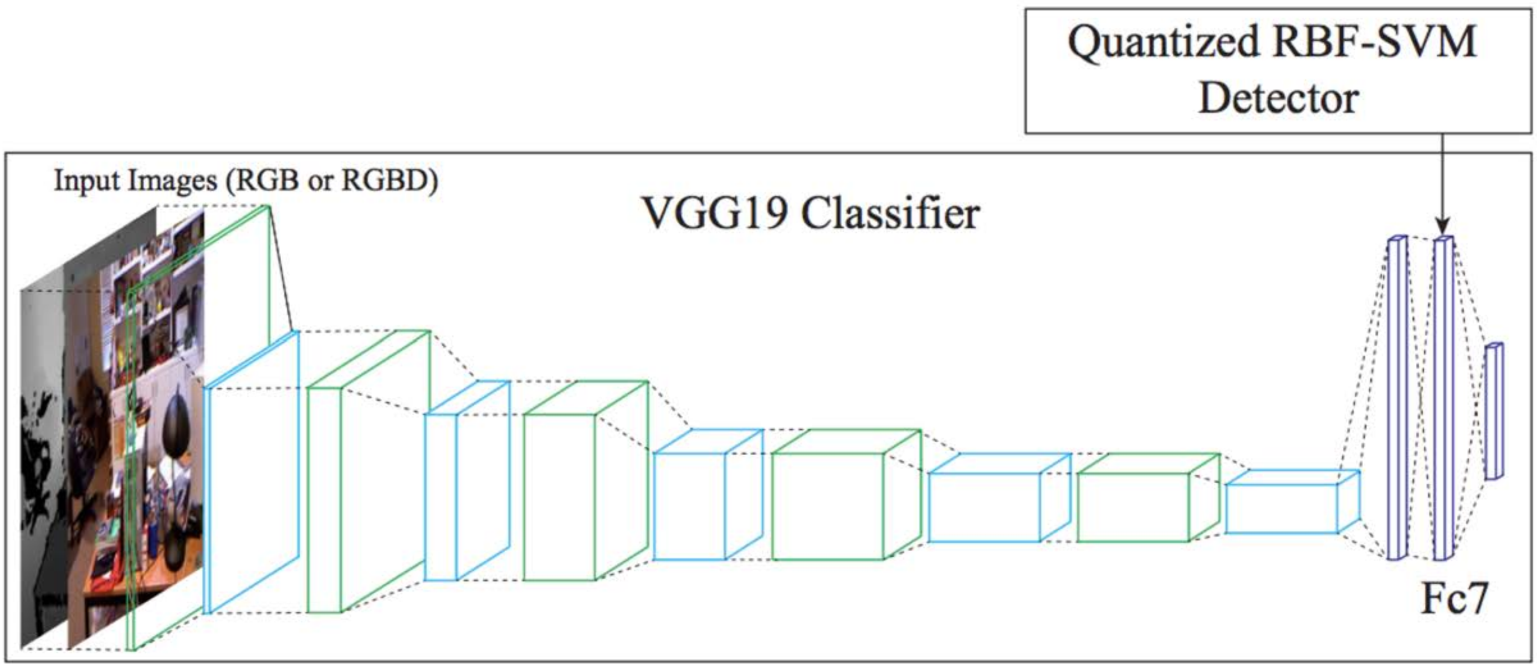
- Adversarial training: networks can be made somewhat resistant by augmenting or regularizing training with adversarial examples

I. Goodfellow, J. Schlens, C. Szegedy, [Explaining and harnessing adversarial examples](), ICLR 2015

F. Tramer, A. Kurakin, N. Papernot, D. Boneh, P. McDaniel, [Ensemble adversarial training: Attacks and defenses](), ICLR 2018

Defending against adversarial examples

- Train a separate model to reject adversarial examples: SafetyNet



J. Lu, T. Issaranon, D. Forsyth, [SafetyNet: Detecting and Rejecting Adversarial Examples Robustly](), CVPR 2017

Defending against adversarial examples

- Design highly nonlinear architectures robust to adversarial perturbations

D. Krotov, J. Hopfield, [Dense Associative Memory is Robust to Adversarial Inputs](), arXiv 2017

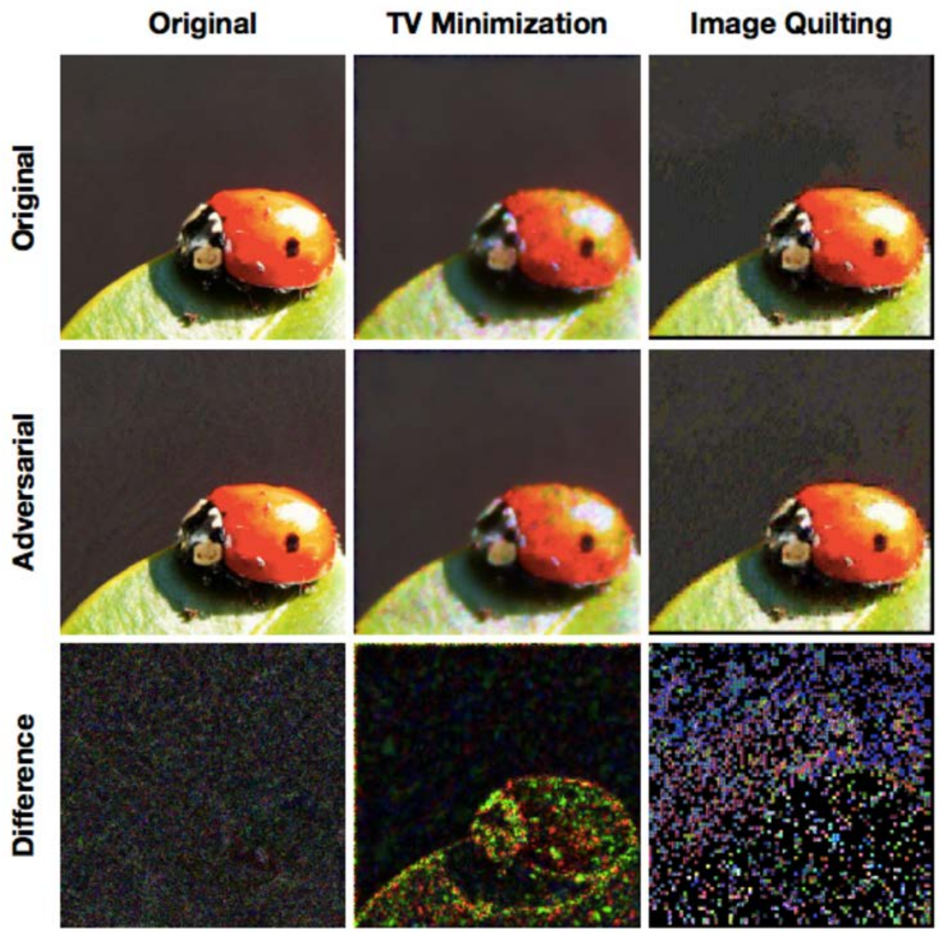
Adversarial examples: Summary

Adversarial examples: Summary

- Generating adversarial examples
  - Finding smallest “fooling” transformation
  - Gradient ascent
  - Fast gradient sign, iterative variants
  - Universal adversarial perturbations
- Generalizability of adversarial examples
- Why are neural networks easy to fool?
- Defending against adversarial examples
  - Adversarial training
  - Learning to reject adversarial examples
  - Robust architectures
  - Image pre-processing

Defending against adversarial examples

- Pre-process input images to disrupt adversarial perturbations

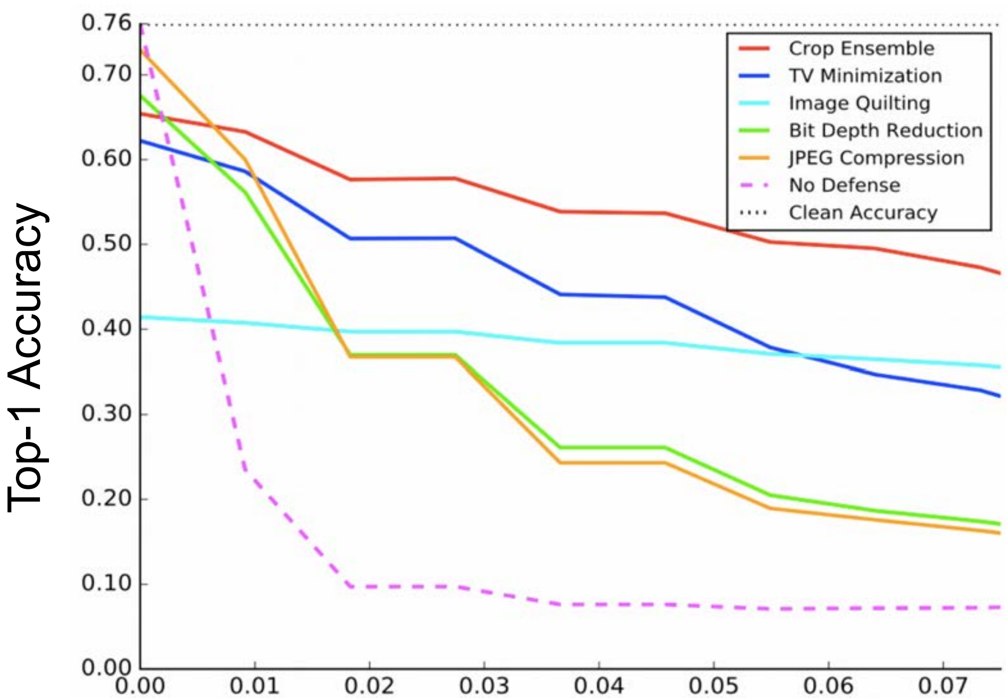


C. Guo, M. Rana, M. Cisse, L. van der Maaten, [Countering Adversarial Images Using Input Transformations](), ICLR 2018

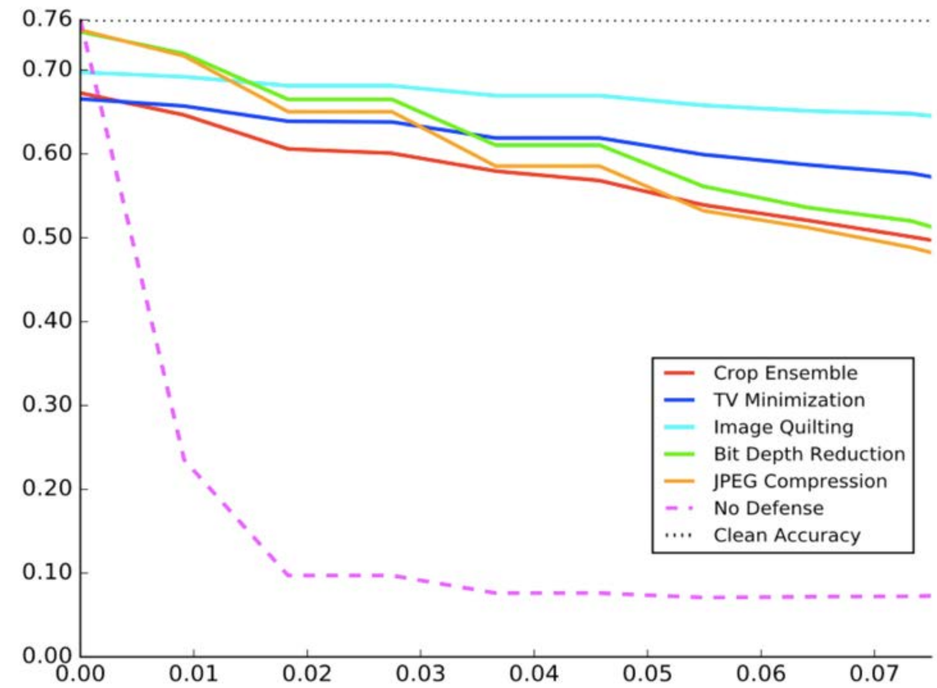
Defending against adversarial examples

- Pre-process input images to disrupt adversarial perturbations

ResNet-50 *tested* on transformed images



ResNet-50 *trained and tested* on transformed images



C. Guo, M. Rana, M. Cisse, L. van der Maaten, [Countering Adversarial Images Using Input Transformations](), ICLR 2018

Adversarial examples for detection

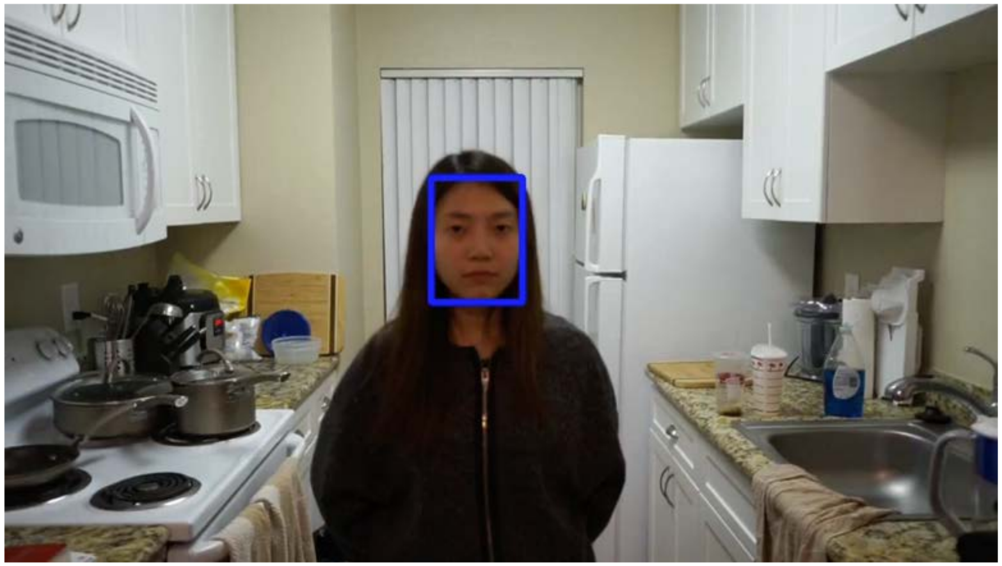
- TL;DR: It is much harder to fool a detector like Faster R-CNN or YOLO than a classifier; large perturbations are currently required

J. Lu, H. Sibai, E. Fabry, [Adversarial examples that fool detectors](), arXiv 2018

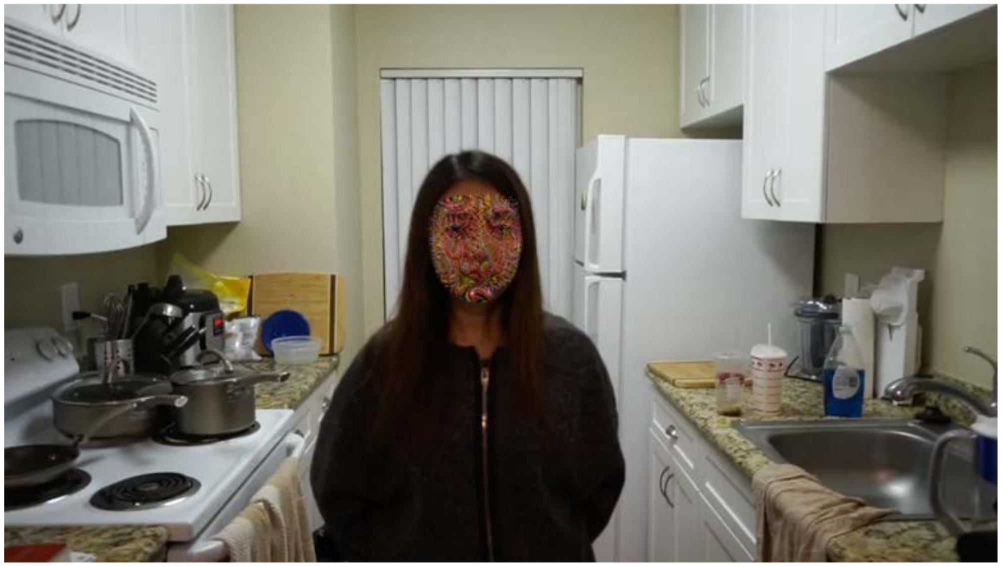
Adversarial examples for detection

- TL;DR: It is much harder to fool a detector like Faster R-CNN or YOLO than a classifier; large perturbations are currently required

Original w/ Faster R-CNN detections



Attacked



J. Lu, H. Sibai, E. Fabry, [Adversarial examples that fool detectors](), arXiv 2018

Adversarial examples for detection

- TL;DR: It is much harder to fool a detector like Faster R-CNN or YOLO than a classifier; large perturbations are currently required
- It is even harder to fool a detector with physical objects

”All three patterns reliably fool detectors when mapped into videos. However, physical instances of these patterns are not equally successful. The first two stop signs, as physical objects, only occasionally fool Faster RCNN; the third one, which has a much more extreme pattern, is more effective.”

J. Lu, H. Sibai, E. Fabry, [Adversarial examples that fool detectors](), arXiv 2018

Adversarial examples for detection

- TL;DR: It is much harder to fool a detector like Faster R-CNN or YOLO than a classifier; large perturbations are currently required
- It is even harder to fool a detector with physical objects

Original w/ Faster R-CNN detections

Digitally attacked

J. Lu, H. Sibai, E. Fabry, [Adversarial examples that fool detectors](), arXiv 2018

Adversarial examples for detection

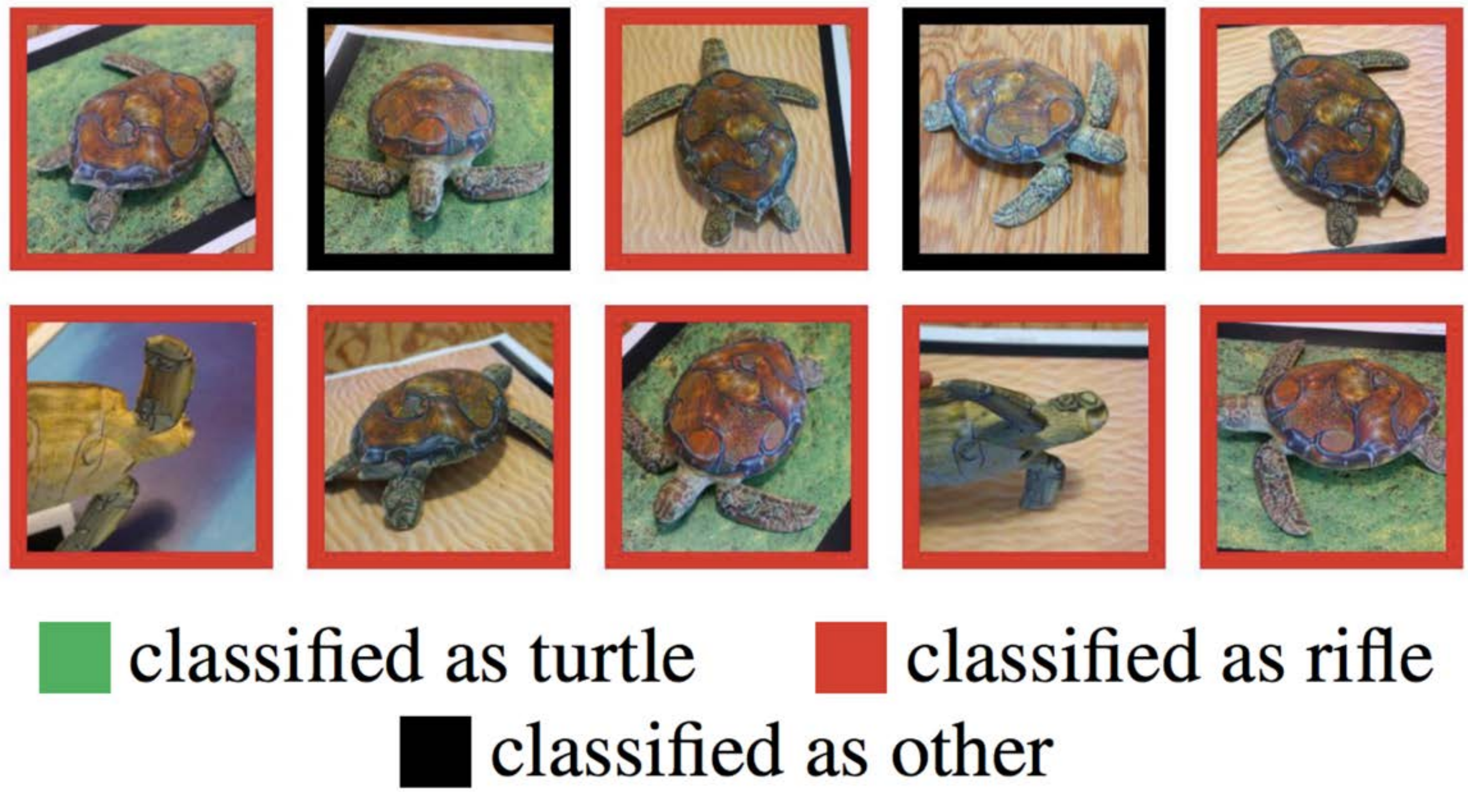
- TL;DR: It is much harder to fool a detector like Faster R-CNN or YOLO than a classifier; large perturbations are currently required
- It is even harder to fool a detector with physical objects

Physical adversarial stop sign

J. Lu, H. Sibai, E. Fabry, [Adversarial examples that fool detectors](), arXiv 2018

Robust adversarial examples

3D printed adversarial object (YouTube video)



A. Athalye, L. Engstrom, A. Ilyas, K. Kwok, Synthesizing Robust Adversarial Examples, arXiv 2018

https://blog.openai.com/robust-adversarial-inputs/

Adversarial examples and humans

- Adversarial examples that are designed to transfer across multiple architectures can also be shown to confuse the human visual system in rapid presentation settings

G. Elsayed et al., Adversarial Examples that Fool both Computer Vision and Time-Limited Humans, arXiv 2018