CSCI 618: Trustworthy ML (from an optimization lens)

Vatsal Sharan

Fall 2026

Lecture 2, Aug 26

USC University of Southern California

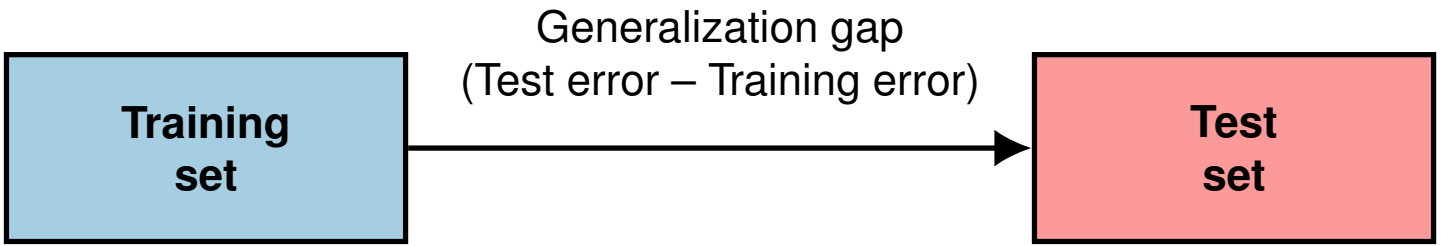

MACHINE LEARNING
Recap
MACHINE LEARNING EVERYWHERE
imgflip.com

Training/Test paradigm

Data Splitting. We randomly divide data into two disjoint subsets:

- **Training set:** subset of data used to train the model.
- **Test set:** subset of data used to evaluate the model.

Generalization gap: Test error – Training error.



We usually add a third split as well.

- **Validation set:** subset of data used to measure generalization, fit hyperparameters

Supervised learning in one slide

Loss function: What is the right loss function for the task?

Representation: What class of functions should we use?

Optimization: How can we efficiently solve the empirical risk minimization problem?

Generalization: Will the predictions of our model transfer gracefully to unseen examples?

All related! And the fuel which powers everything is data.

Gradient descent

GD: keep moving in the *negative gradient direction*

Start from some $w^{(0)}$. For $t = 0, 1, 2, \ldots$

$$w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla F\big(w^{(t)}\big),$$

where $\eta > 0$ is called *step size* or *learning rate*.

- in theory η should be set in terms of some parameters of f
- in practice we just try several small values
- might need to be changing over iterations (think $f(w) = |w|$)
- adaptive and automatic step size tuning is an active research area

Stochastic Gradient descent

GD: keep moving in the *negative gradient direction*

SGD: keep moving in the *noisy negative gradient direction*

$$\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta \tilde{\nabla} F(\boldsymbol{w}^{(t)})$$

where $\tilde{\nabla} F(\boldsymbol{w}^{(t)})$ is a random variable (called **stochastic gradient**) s.t.

$$\mathbb{E}\left[\tilde{\nabla} F(\boldsymbol{w}^{(t)})\right] = \nabla F(\boldsymbol{w}^{(t)}) \qquad \text{(unbiasedness)}$$

- Key point: it could be much faster to obtain a stochastic gradient!
  - If $F(w) = \sum_{i=1}^{n} \ell(x_i, w)$, then how to get stochastic gradient?

- Similar convergence guarantees, usually needs more iterations but each iteration takes less time.

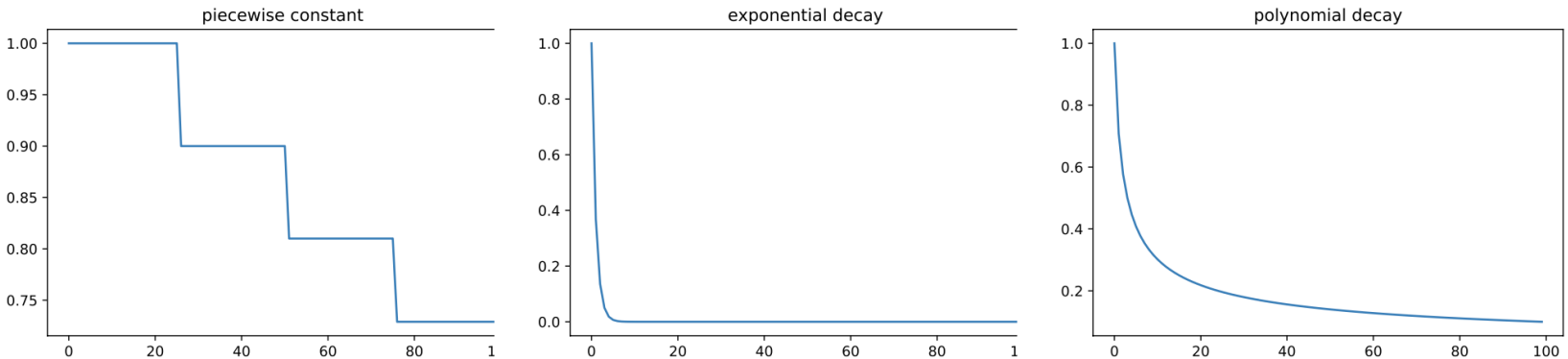
Summary: Gradient descent & Stochastic Gradient descent

- GD/SGD coverages to a stationary point. For convex objectives, this is all we need.
- For nonconvex objectives, can get stuck at local minimizers or “bad” saddle points (random initialization escapes “good” saddle points)
- Recent research shows that *many problems have no “bad” saddle points or even “bad” local minimizers*
- SGD is very popular, another very popular optimization technique is *Adam*
- Adam has two key additional ingredients: adaptive step size & momentum

Adaptive learning rate tuning

``The learning rate is perhaps the most important hyperparameter.
If you have time to tune only one hyperparameter, tune the learning rate."
-Deep learning (Book by Goodfellow, Bengio, Courville)

We often use a **learning rate schedule**.



(a) *(b)* *(c)*

Some common learning rate schedules (figure from PML)

Adaptive learning rate methods (Adagrad, RMSProp) scale the learning rate of each parameter based on some moving average of the magnitude of the gradients.

[PML] *Probabilistic Machine Learning: An Introduction* by Kevin Murphy.

Momentum

"move faster along directions that were previously good, and to slow down along directions where the gradient has suddenly changed, just like a ball rolling downhill." [PML]

Initialize $\boldsymbol{w}_0$ and (velocity) $\boldsymbol{v} = \mathbf{0}$

For $t = 1, 2, \ldots$

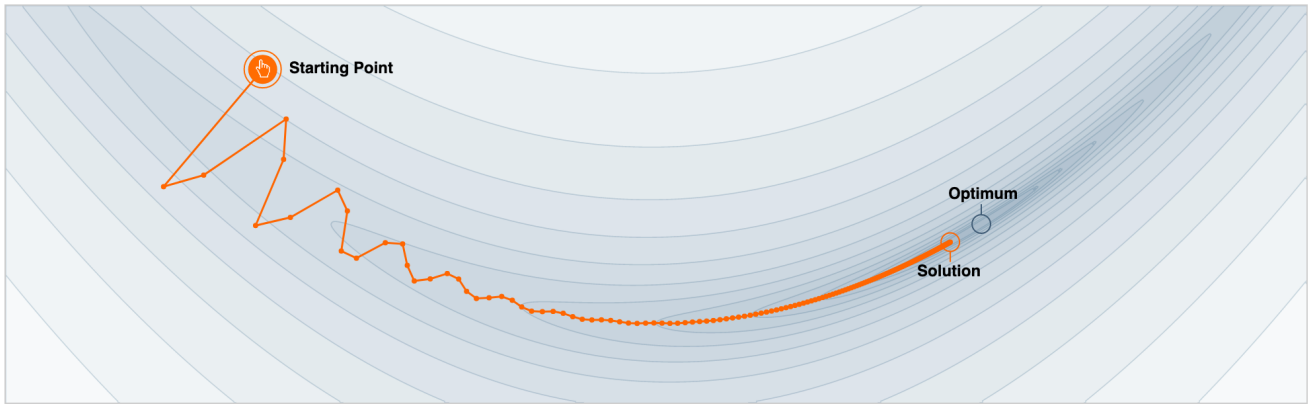
- estimate a stochastic gradient $\boldsymbol{g}_t$
- update $\boldsymbol{v} \leftarrow \alpha \boldsymbol{v} + \boldsymbol{g}_t$ for some discount factor $\alpha \in (0, 1)$
- update weight $\boldsymbol{w}_t \leftarrow \boldsymbol{w}_{t-1} - \eta \boldsymbol{v}$

Updates for first few rounds:

- $\boldsymbol{w}_1 = \boldsymbol{w}_0 - \eta \boldsymbol{g}_1$
- $\boldsymbol{w}_2 = \boldsymbol{w}_1 - \alpha\eta \boldsymbol{g}_1 - \eta \boldsymbol{g}_2$
- $\boldsymbol{w}_3 = \boldsymbol{w}_2 - \alpha^2\eta \boldsymbol{g}_1 - \alpha\eta \boldsymbol{g}_2 - \eta \boldsymbol{g}_3$
- $\ldots$

Momentum

Why Momentum Really Works



Step-size α = 0.02

0 0.003 0.006

Momentum β = 0.99

0.00 0.500 0.990

We often think of Momentum as a means of dampening oscillations and speeding up the iterations, leading to faster convergence. But it has other interesting behavior. It allows a larger range of step-sizes to be used, and creates its own oscillations. What is going on?

GABRIEL GOH
UC Davis

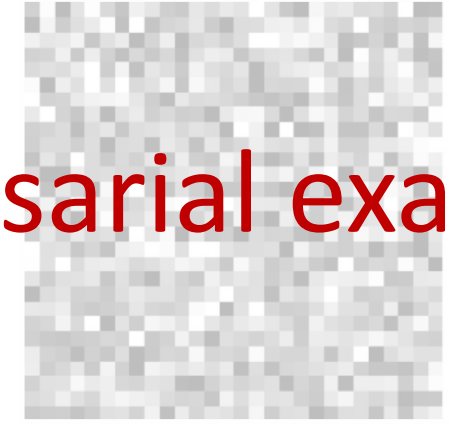
April. 4
2017

Citation:
Goh, 2017

https://distill.pub/2017/momentum/

Adversarial examples

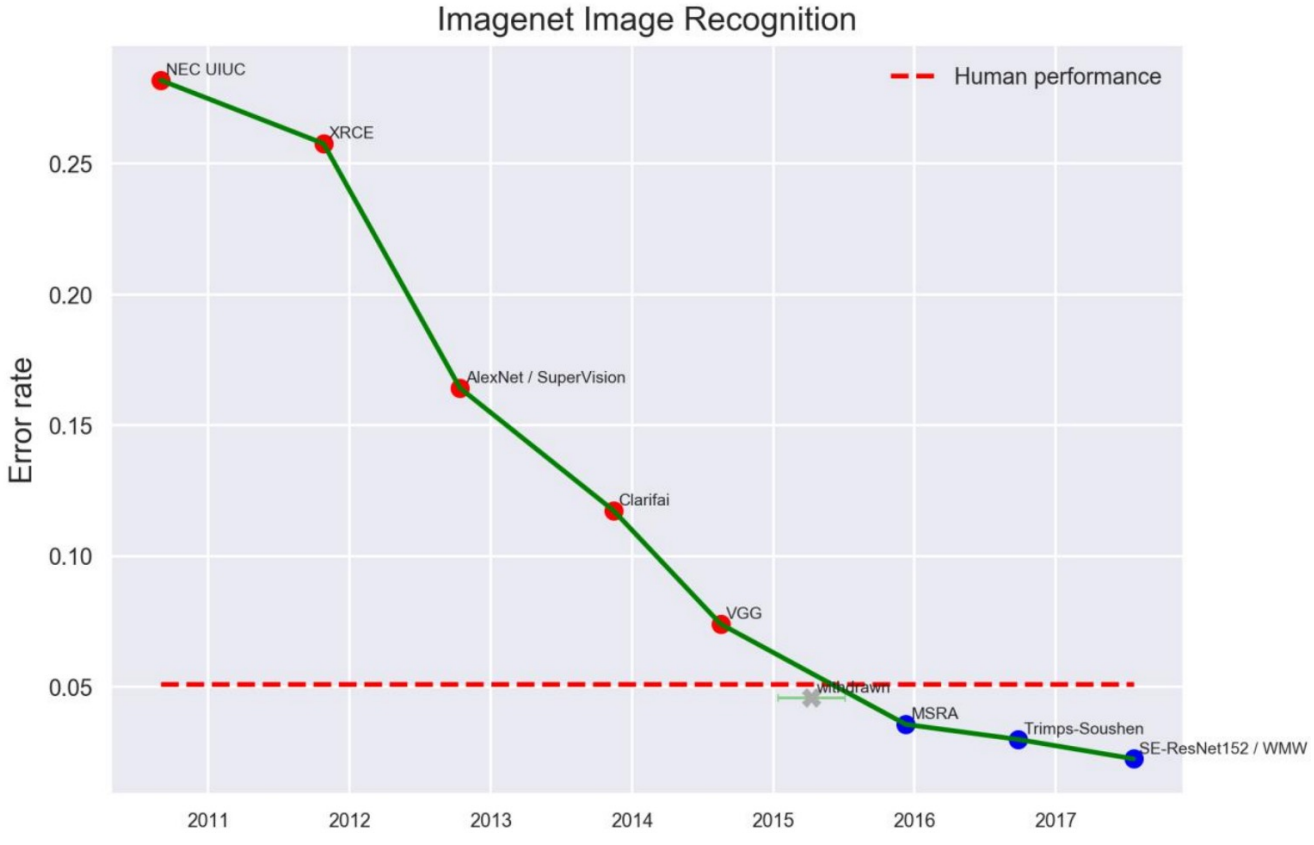
Pig
(90% confidence)



Small amount of
adversarial noise

Airplane!
(99.9% confidence)

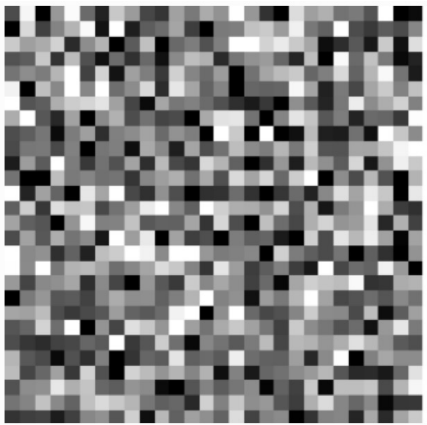
CNNs are great at image classification



However, models can also be very sensitive to small variations in the input

Pig
(90% confidence)

+



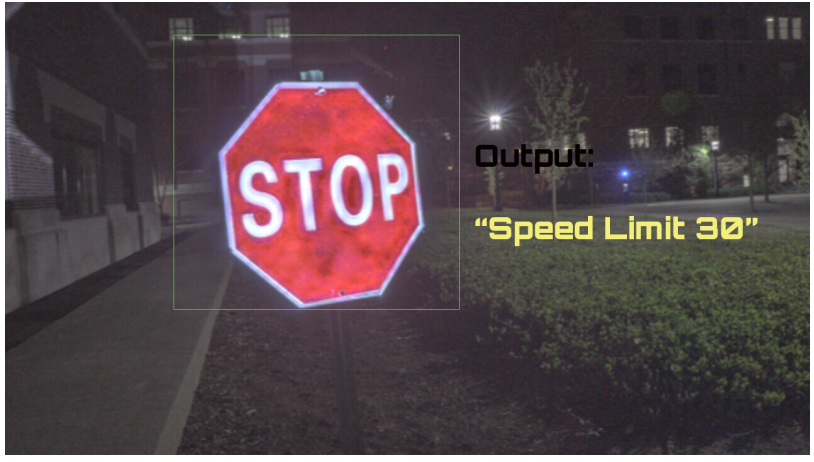
Small amount of
adversarial noise

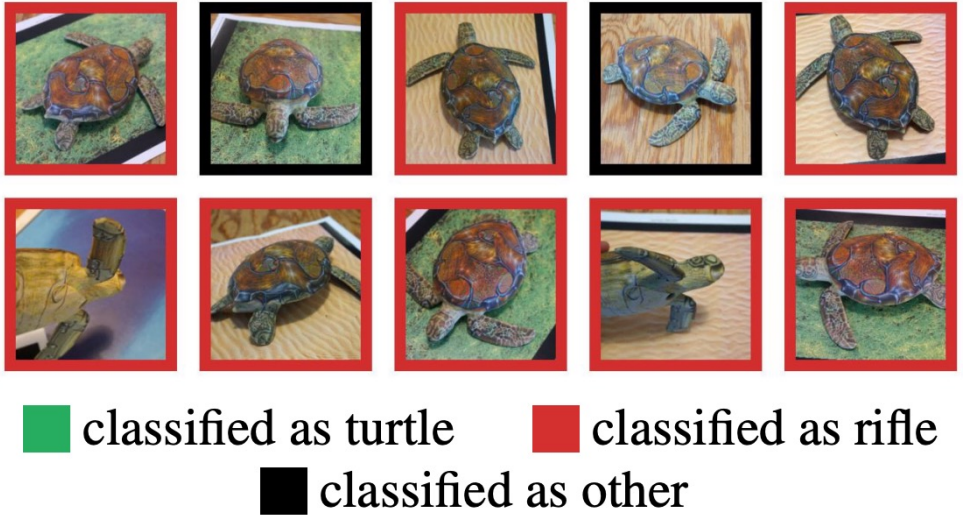
=

Airplane!
(99.9% confidence)

Image adapted from https://adversarial-ml-tutorial.org/

These are known as ***adversarial examples***





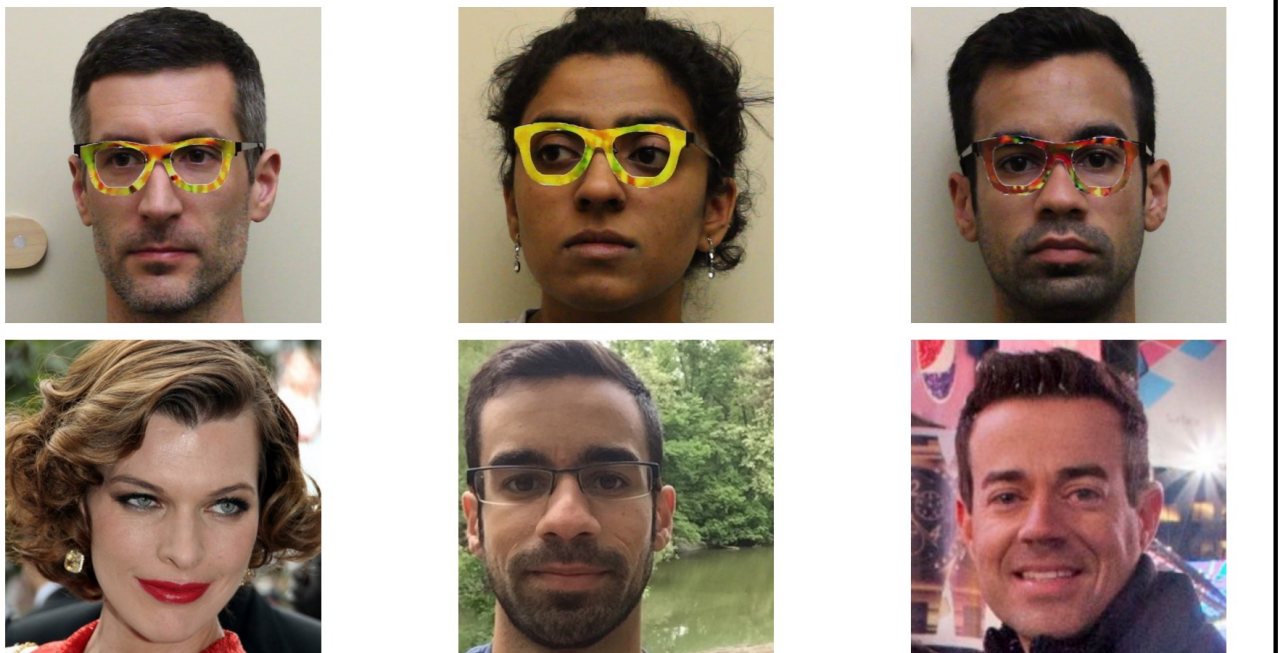
Adversarial examples have been shown to also hold for real-world tasks.

They are an issue because

1. Can pose potential security risks
2. Indicate that even though models are good, they don't quite work the same way as we do

More studies on adversarial examples

Dodging detection from face detection using glasses



Person in top row impersonating person in bottom row using glasses

From *Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition, Sharif et al. '16*

Examples from other modalities

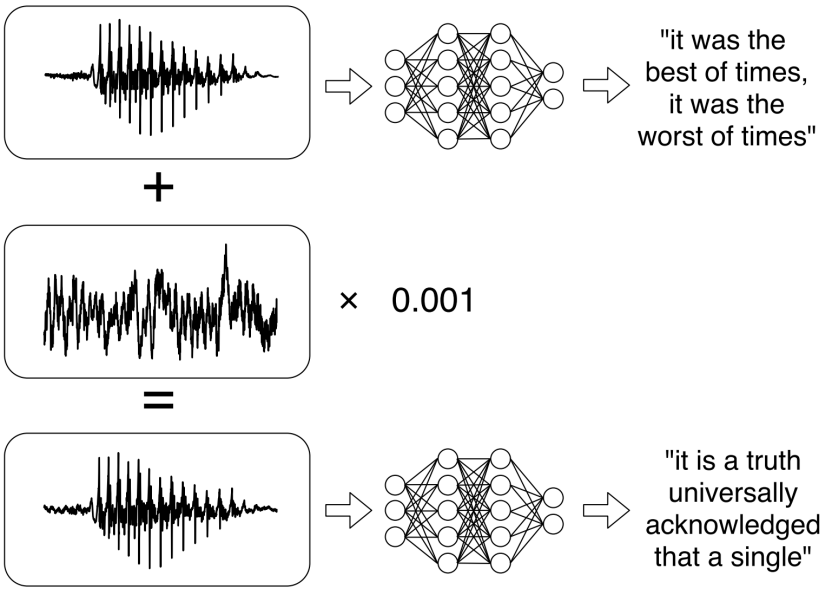


Figure 1. Illustration of our attack: given any waveform, adding a small perturbation makes the result transcribe as any desired target phrase.

From *Audio Adversarial Examples: Targeted Attacks on Speech-to-Text, Carlini & Wagner '18*

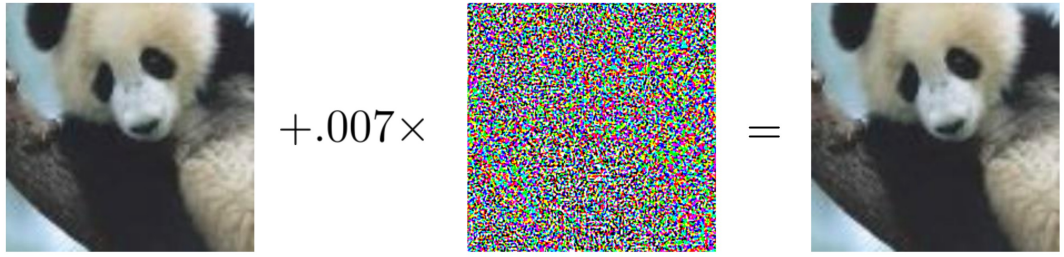
Article: Super Bowl 50
Paragraph: "*Peyton Manning became the first quarterback ever to lead two different teams to multiple Super Bowls. He is also the oldest quarterback ever to play in a Super Bowl at age 39. The past record was held by John Elway, who led the Broncos to victory in Super Bowl XXXIII at age 38 and is currently Denver's Executive Vice President of Football Operations and General Manager. Quarterback Jeff Dean had jersey number 37 in Champ Bowl XXXIV.*"
Question: "*What is the name of the quarterback who was 38 in Super Bowl XXXIII?*"
Original Prediction: John Elway
Prediction under adversary: Jeff Dean

From *Adversarial Examples for Evaluating Reading Comprehension Systems, Jia & Liang '17*

Adversarial examples

Part 1: Basics of finding adversarial examples

Adversarial examples: Setup



Panda
58% confidence

Nematode
8% confidence

Gibbon
99% confidence

Adversary: Given an image x and classifier $f(x)$, comes up with some other image x' which is "similar" to x, such that $f(x) \neq f(x')$.

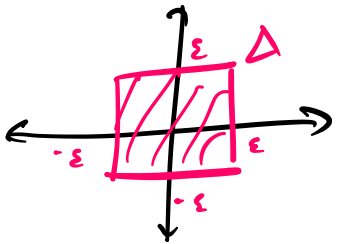
How to define similarity? One notion is small perturbations based on some norm. We typically consider the ℓ_∞ norm: $\| x - x' \|_\infty \leq \epsilon$, where ϵ is the allowed perturbation level.

This means: can perturb every pixel by a perturbation in $[-\epsilon, \epsilon]$.

Finding Adversarial examples: Optimization problem

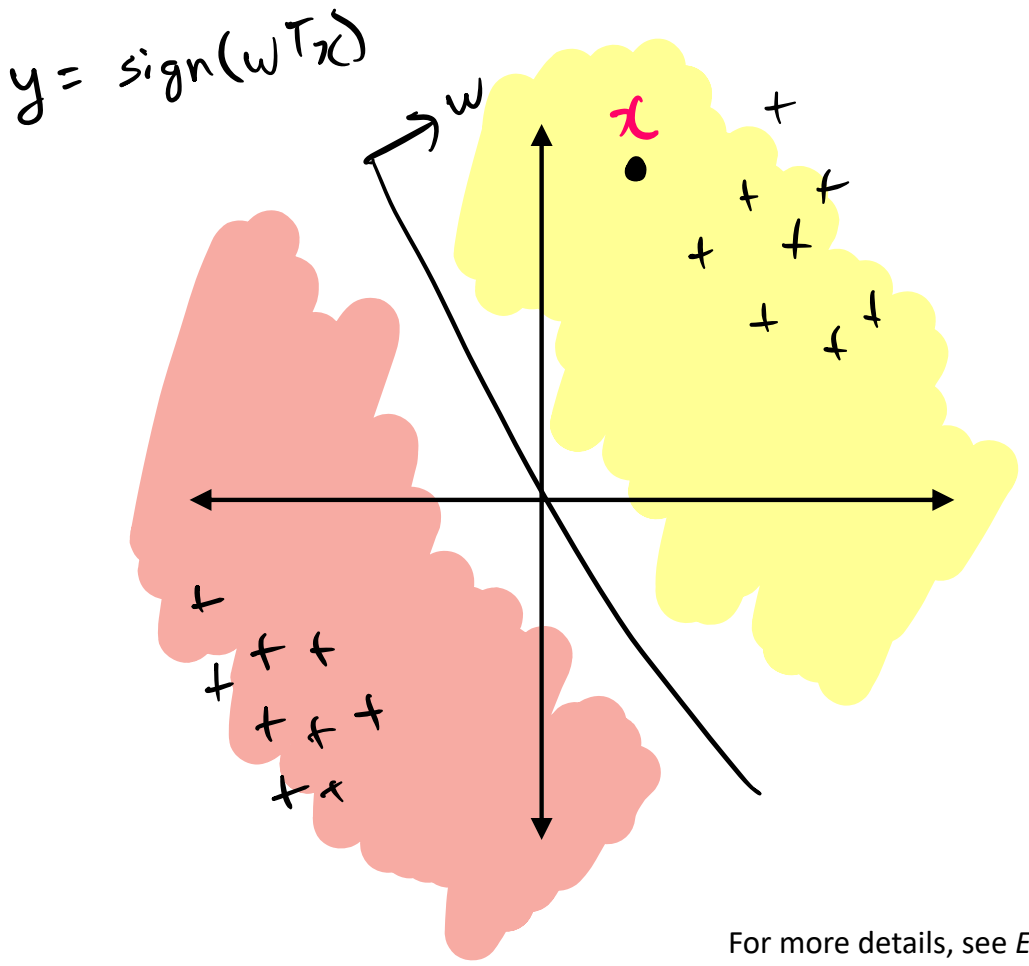
Adversary: Given an image x and classifier $f(x)$, comes up with some other image x' which is "similar" to x, such that $f(x) \neq f(x')$.

$$\Delta = \{\delta \in \mathbb{R}^d : \|\delta\|_\infty \leq \varepsilon\}.$$



$$\ell_{\text{adv}}(f; x, y) := \max_{\delta \in \Delta} \ell\big(f(x+\delta),\, y\big).$$

Detour: Linearity as a source of brittleness



$$x' = x - \text{sign}(w)\,\varepsilon$$

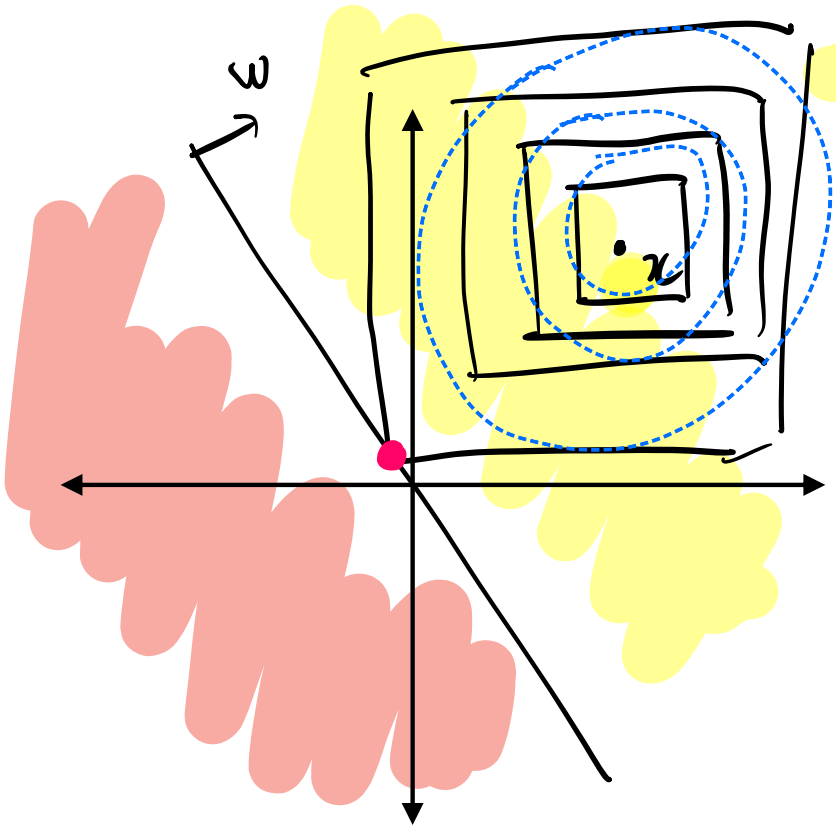
$$\|x - x'\|_\infty = \varepsilon$$

$$w^T x' = w^T x - \varepsilon w^T \text{sign}(w)$$

$$= w^T x - \varepsilon \|w\|_1$$

For more details, see *Explaining and Harnessing Adversarial Examples, Goodfellow et al. '15*

Finding adversarial perturbation for linear models



Claim: $x_{adv} = x - \text{sign}(w) \cdot \varepsilon$

changes the predictions the most among all x' s.t. $\|x' - x\|_\infty \leq \varepsilon$.

For more details, see *Explaining and Harnessing Adversarial Examples, Goodfellow et al. '15*

Back to our optimization problem

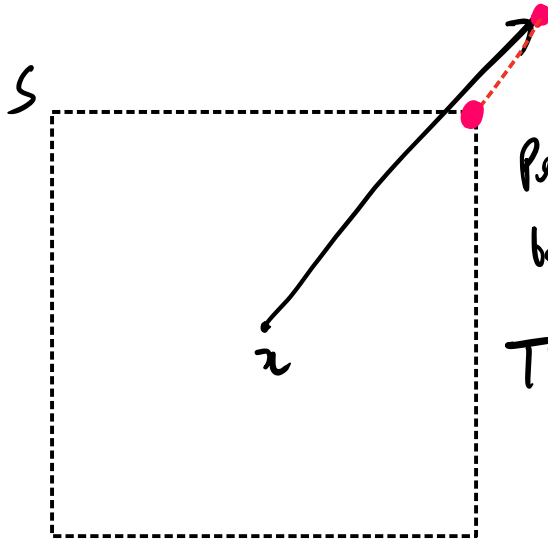
Adversary: Given an image x and classifier $f(x)$, comes up with some other image x' which is "similar" to x, such that $f(x) \neq f(x')$.

$$\Delta = \{\delta \in \mathbb{R}^d : \|\delta\|_\infty \leq \varepsilon\}$$

$$\ell_{\text{adv}}(f; x, y) := \max_{\delta \in \Delta} \ell\big(f(x+\delta), y\big)$$ → This is an optimization problem

Fast gradient sign method (FGSM):

$$\delta_{\text{FGSM}} = \epsilon \cdot \text{sign}(\nabla_x \ell(f(x), y)),$$
$$x^{\text{adv}} = x + \delta_{\text{FGSM}}.$$



Projection onto the l_∞ ball:

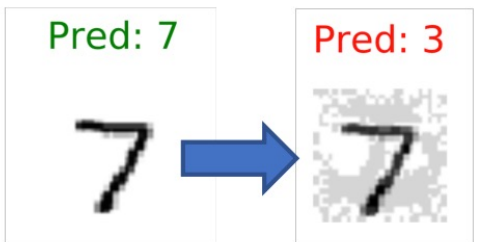
$$\Pi_S(v) = \underset{v' \in S}{\text{argmin}} \|v - v'\|_2$$

FGSM: Results

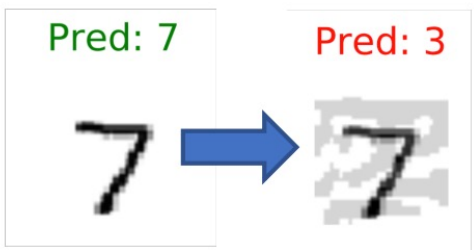
Fast gradient sign method (FGSM):

$$\delta_{\text{FGSM}} = \epsilon \cdot \text{sign}(\nabla_x \, \ell(f(x), y)) \,,$$
$$x^{\text{adv}} = x + \delta_{\text{FGSM}}.$$

MLP:



ConvNet:



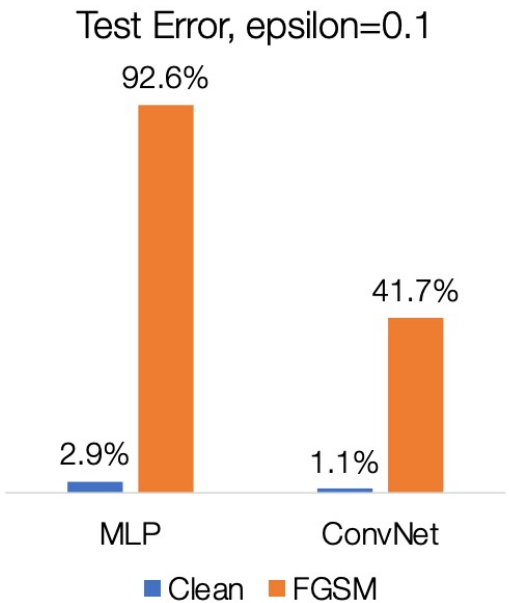


Image from https://adversarial-ml-tutorial.org/ (part 2)

Adversarial examples

Part 2: Basics of defending against adversarial examples

Adversarial robustness: The optimization problem

Adversary: Given an image x and classifier $f(x)$, comes up with some other image x' which is "similar" to x, such that $f(x) \neq f(x')$.

$$\Delta = \{\delta \in \mathbb{R}^d : \|\delta\|_\infty \leq \varepsilon\}$$

$$\ell_{\text{adv}}(f; x, y) := \max_{\delta \in \Delta} \ell\big(f(x+\delta),\, y\big)$$

$$\widehat{R}_S^{\text{adv}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell_{\text{adv}}(f; x_i, y_i),$$

Defender: Train a model such that the adversary is not effective at finding adversarial examples

$$\min_{f \in \mathcal{F}} \widehat{R}_S^{\text{adv}}(f) = \min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \max_{\delta \in \Delta} \ell\big(f(x_i+\delta),\, y_i\big).$$

Min-max optimization: Danskin's Theorem

Defender: Train a model such that the adversary is not effective at finding adversarial examples

$$\min_{f \in \mathcal{F}} \widehat{R}_S^{\mathrm{adv}}(f) = \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \max_{\delta \in \Delta} \ell\big(f(x_i + \delta),\, y_i\big).$$

Danskin's theorem: The gradient of the inner maximization objective with respect to f is given by the gradient at the maximizer of the inner objective.

$$\nabla_f \max_{\delta \in \Delta} \ell\big(f(x+\delta), y\big) = \nabla_f \ell\big(f(x+\delta^\star), y\big),$$

$$\text{where } \delta^\star = \arg\max_{\delta \in \Delta} \ell\big(f(x+\delta), y\big).$$

Adversarial training, inspired by Danskin's Theorem

Danskin's theorem:

$$\nabla_f \max_{\delta \in \Delta} \ell\big(f(x+\delta), y\big) = \nabla_f \, \ell\big(f(x+\delta^\star), y\big),$$

$$\text{where } \delta^\star = \arg\max_{\delta \in \Delta} \ell\big(f(x+\delta), y\big).$$

Repeat:

1. Select a minibatch B of b examples $\{(x_i, y_i)\}_{i=1}^b$ from the training set.

2. For each $(x_i, y_i) \in B$, compute adversarial perturbation using FGSM

$$\delta_i^\star = \arg\max_{\delta \in \Delta} \ell\big(f(x_i+\delta), y_i\big).$$

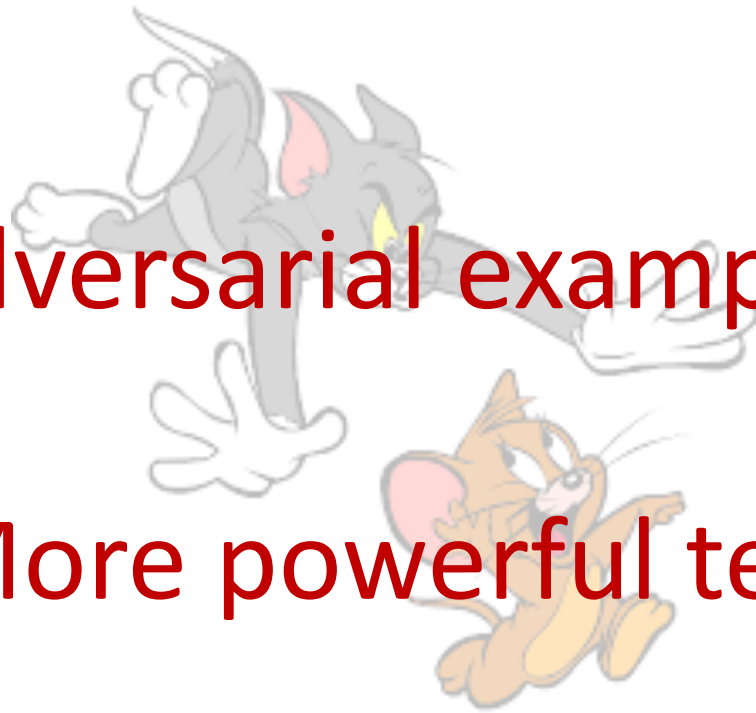
3. Update parameters (for some learning rate α):

$$f := f - \alpha \sum_{i=1}^{b} \nabla_f \, \ell\big(f(x_i+\delta_i^\star), y\big).$$

pretend that $\delta_i^\star$ is the most adversarial perturbation

	Clean Error	FGSM Error
ConvNet	1.1%	41.7%
Robust ConvNet	0.9%	2.6%

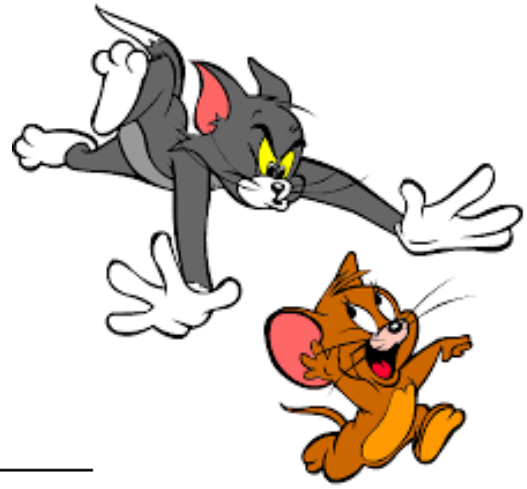
FGSM results on MNIST ($\epsilon = 0.1$)



Adversarial examples

Part 3: More powerful techniques

A cautionary tale



Adversarial examples research has had the nature of a cat-and-mouse game.

Specific defenses often work against the specific attack they were designed for, but fail more generally.

Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples

---

Anish Athalye [*1] **Nicholas Carlini** [*2] **David Wagner** [2]

This ICML 2018 paper broke 7 out of 9 defense methods appearing a few months ago at ICLR 2018!
The methods were shown to rely on “obfuscated gradients”.

Therefore, important to consider strong attack models.

Projected gradient descent (PGD) for finding adversarial examples

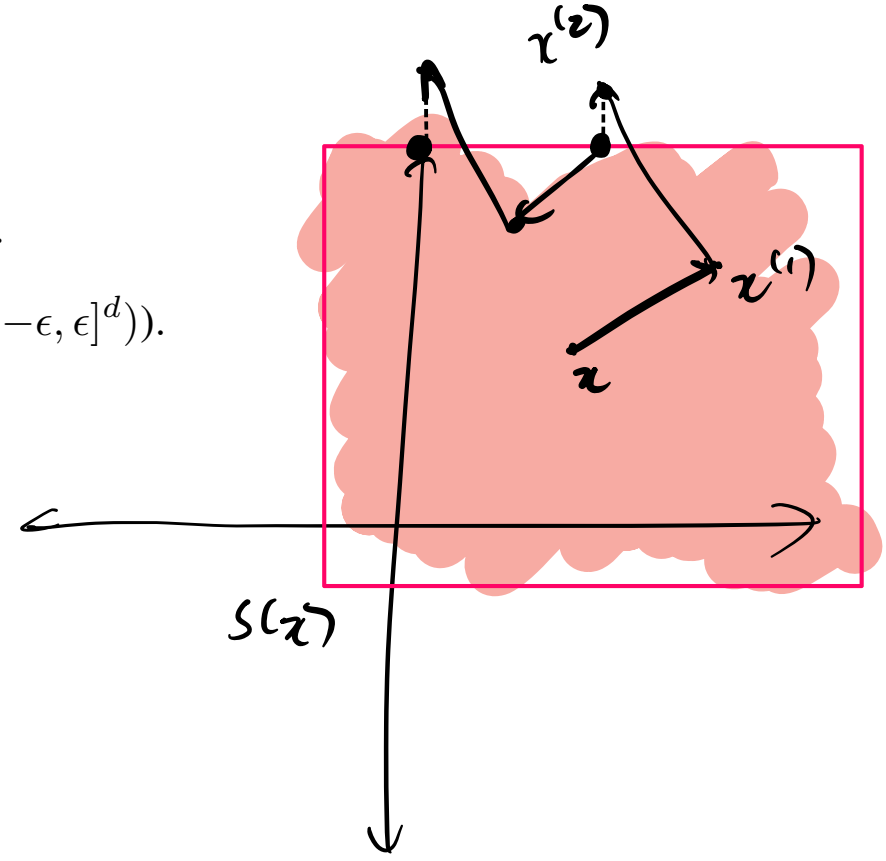
Feasible set: $\Delta = \{\delta \in \mathbb{R}^d : \|\delta\|_\infty \leq \epsilon\}, \quad \mathcal{S}(x) = x + \Delta.$

Initialize: $x^{(0)} = x \quad$ (optionally $x^{(0)} = x + \eta,\ \eta \sim \mathrm{Unif}([-\epsilon, \epsilon]^d)$).

For $t = 0, \ldots, T-1$:

$$g^{(t)} = \nabla_x \, \ell\big(f(x^{(t)}), y\big),$$
$$x^{(t+1)} = \Pi_{\mathcal{S}(x)}\big(x^{(t)} + \alpha \cdot \mathrm{sign}(g^{(t)})\big).$$

Output: $x^{\mathrm{adv}} = x^{(T)}$.



From *Towards Deep Learning Models Resistant to Adversarial Attacks, Madry et al. '18*

Adversarial training using PGD

Repeat:

1. Select a minibatch B of b examples $\{(x_i, y_i)\}_{i=1}^{b}$ from the training set.
2. For each $(x_i, y_i) \in B$, compute adversarial perturbation using PGD

$$\delta_i^\star = \arg\max_{\delta \in \Delta} \ \ell\big(f(x_i + \delta), y_i\big).$$

3. Update parameters (for some learning rate α):

$$f \ := \ f - \alpha \sum_{i=1}^{b} \nabla_f \ \ell\big(f(x_i + \delta_i^\star),\, y\big).$$

Adversarial training using PGD & FGSM, Results

Repeat:

1. Select a minibatch B of b examples $\{(x_i, y_i)\}_{i=1}^{b}$ from training set.
2. For each $(x_i, y_i) \in B$, compute adversarial perturbation

$$\delta_i^\star = \arg\max_{\delta \in \Delta} \ell\big(f(x_i + \delta), y_i\big).$$

3. Update parameters (for some learning rate α):

$$f \ := \ f - \alpha \sum_{i=1}^{b} \nabla_f \, \ell\big(f(x_i + \delta_i^\star), \, y\big).$$

① Natural accuracy goes down

② FGSM gets better accuracy on FGSM than PGD, bad on PGD.

③ FGSM is better on adversarial examples of FGSM than clean examples.

Training type →

Evaluation type ↓

	(a) Standard	(b) FGSM	(c) PGD
Natural	95.2%	90.3%	87.3%
FGSM	32.7%	95.1%	56.1%
PGD	3.5%	0.0%	45.8%

Accuracy on CIFAR10 ($\epsilon = 8$)

From *Towards Deep Learning Models Resistant to Adversarial Attacks, Madry et al. '18*