Robustness

Gautam Kamath

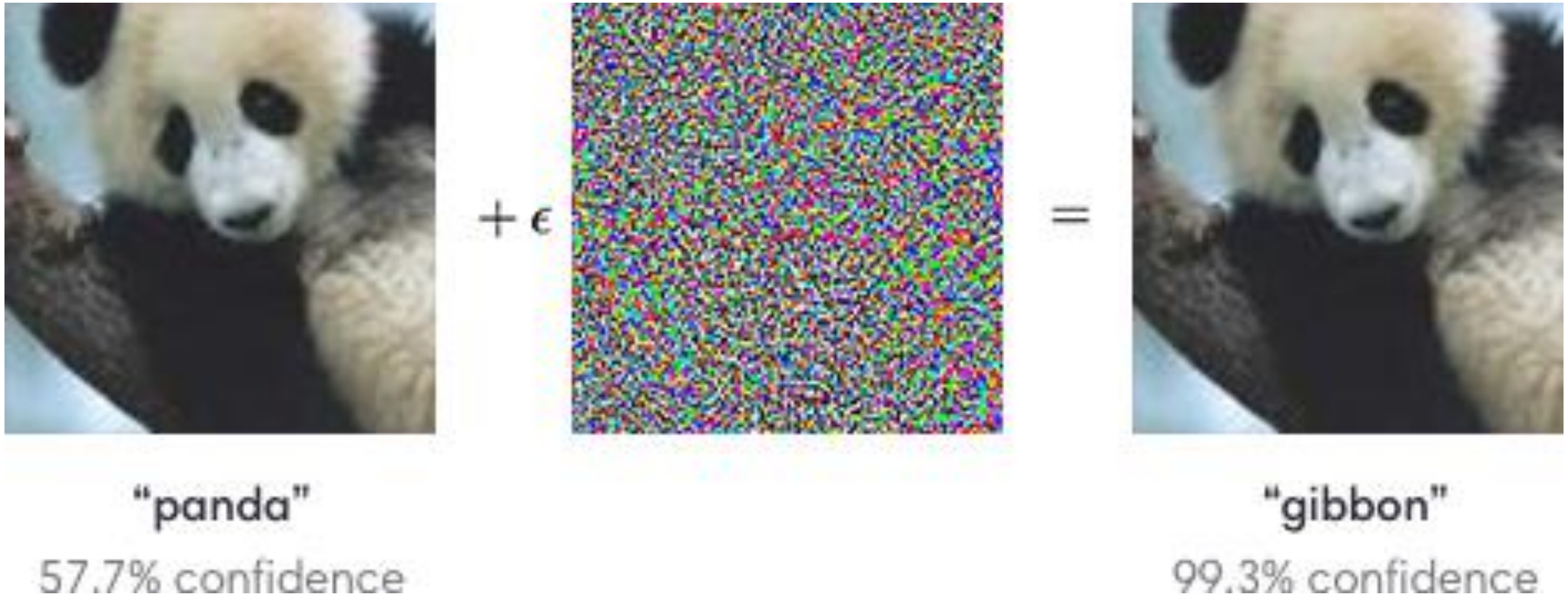
Robustness

- What if the setting deviates from what we assume?
  - E.g., generally assume train and test data generated i.i.d. from a distribution
- Why might this happen?
  - Model misspecification
  - Measurement error
  - Dirty data
  - Adversarial manipulation

Simple example

- Suppose $X_1, \ldots, X_n \sim N(\mu, 1)$ (Draw)
- Goal: estimate μ from $X_1, \ldots, X_n$
- Easy solution: let $\hat{\mu} = \frac{1}{n} \sum X_i$. If n is large, $\hat{\mu} \approx \mu$.
- But what if there are outliers? Even just one outlier (draw)
  - Example of an *attack*
- If outlier is very large ($\gg 100n + \mu$), then $|\mu - \hat{\mu}|$ will be large (> 100)
- How can we *defend*?
  - Prune outliers
  - Use the median instead of mean (example of a robust statistic)

Today: Adversarial Examples



Adversarial Examples Setting

- Train and test data are obtained (as usual)
- Model is trained on the training data (as usual)
- At test time, each feature vector can be modified a “small amount” (arbitrarily/adversarially)
- x' is an adversarial example for x on model f_θ if

1. (Informal) $x \approx x' \leftrightarrow \mathrm{dist}(x, x')$ is small $\leftrightarrow$ x and x' have same label according to human
2. $f_\theta(x) \neq f_\theta(x')$

Distance between points

- "x and x' have same label according to human"
- Can ask a human this question, but not clear how to encode human perception…
- Instead, use distances between x and x' as a proxy
- Most common: ℓ_p-distance $\|x - x'\|_p = \left(\sum (x_j - x_j')^p\right)^{1/p}$
- Given true test point x adversary can replace with point x' in $\{y : \|x - y\|_p \leq \varepsilon\}$, where ε is some small number (problem dependent)
- Common: $p = 0, 2, \infty$ (discuss each, today we focus on ∞)
- Other distances: Wasserstein, translation, rotations, resizing (draw)

Attacker: How to create adversarial examples?

- Given trained model f_θ, test example x, construct x'
- Need: $\|x - x'\|_\infty \leq \varepsilon$ and $f_\theta(x) \neq f_\theta(x')$
- **White-box** vs black-box?
- **Untargeted** vs targeted attacks?
  - Targeted attacks: $f_\theta(x') = c \neq f_\theta(x)$, where c is a target label
- How do we optimize ML models normally? Gradient descent
- $\arg\min_\theta \frac{1}{n} \sum \ell(x_i, y_i, \theta)$
- Update steps $\theta \leftarrow \theta - \frac{\eta}{n} \sum \nabla_\theta \ell(x_i, y_i, \theta)$

Adversarial Example Formulation

- $\delta' = \arg\max_{\delta} \ell(x + \delta, y, \theta)$ s.t. $||\delta||_\infty \leq \varepsilon$, where $\delta \in \mathbf{R}^d$
  - $x' = x + \delta'$
- Gradient-based optimization
- Simple: Fast Gradient Sign Method
- To maximize, step in direction $\nabla_\delta \ell(x + \delta, y, \theta)$, but note constraint
  - Take biggest step allowed (Draw FGSM, small or large gradient but fit to box)
  - $\varepsilon^* = \varepsilon \cdot \operatorname{sign}\left(\nabla_\delta \ell(x + \delta, y, \theta)\right) \in \{\pm\varepsilon\}^d$

Better: Projected Gradient Descent (PGD)

- Multi-step version of FGSM
- $\delta^{(t+1)} = \mathrm{Proj}\left(\delta^{(t)} + \eta \cdot \mathrm{sign}\left(\nabla_\delta \ell(x+\delta, y, \theta)\right)\right)$
  - (Draw idea of gradient descent without projection, add projection)
  - η is a hyperparameter
  - Project into $[-\varepsilon, \varepsilon]^d$ if necessary
    - E.g. $\mathrm{Proj}([3\varepsilon, -2\varepsilon, 0.5\varepsilon]) = [\varepsilon, -\varepsilon, 0.5\varepsilon]$
  - Technical note: this is actually projected steepest gradient ascent, to deal with issues of gradients being small

Untargeted vs. Targeted

- Untargeted: $\max_{\delta} \ell(x+\delta, y, \theta)$ s.t. $\|\delta\|_\infty \leq \varepsilon$
- Targeted to c: $\max_{\delta} \ell(x+\delta, y, \theta) - \ell(x+\delta, c, \theta)$ s.t. $\|\delta\|_\infty \leq \varepsilon$

Defenses? Adversarial Training

- Usual goal: $\min_{\theta} E_{(x,y)\sim p}[\ell(x, y, \theta)]$
- Robust setting: $\min_{\theta} E_{(x,y)\sim p}\left[\max_{\delta:\|\delta\|_\infty \leq \varepsilon} \ell(x + \delta, y, \theta)\right]$
  - Train a network to anticipate attacks
    - To be a good defender have to be a good attacker
  - On an actual dataset, $\min_{\theta} \frac{1}{n} \sum \max_{\delta_i:\|\delta_i\|_\infty \leq \varepsilon} \ell(x_i + \delta_i, y_i, \theta)$

1. Draw a minibatch B
2. For each (x_i, y_i) in B, compute $\delta_i^* = \arg \max_{\delta_i:\|\delta_i\|_\infty \leq \varepsilon} \ell(x_i + \delta_i, y_i, \theta)$
3. $\theta \leftarrow \theta - \frac{\eta}{|B|} \sum_{i \in B} \nabla_\theta \ell(x_i + \delta_i^*, y_i, \theta)$
4. Repeat

Attacks are more effective than defenses

- Broke 7/9 defenses submitted to ICLR 2018 the day after they were accepted

Defense	Dataset	Distance	Accuracy
Buckman et al. (2018)	CIFAR	0.031 (ℓ_∞)	0%*
Ma et al. (2018)	CIFAR	0.031 (ℓ_∞)	5%
Guo et al. (2018)	ImageNet	0.005 (ℓ_2)	0%*
Dhillon et al. (2018)	CIFAR	0.031 (ℓ_∞)	0%
Xie et al. (2018)	ImageNet	0.031 (ℓ_∞)	0%*
Song et al. (2018)	CIFAR	0.031 (ℓ_∞)	9%*
Samangouei et al. (2018)	MNIST	0.005 (ℓ_2)	55%**
Madry et al. (2018)	CIFAR	0.031 (ℓ_∞)	47%
Na et al. (2018)	CIFAR	0.015 (ℓ_∞)	15%

Backdoor attacks

- Modify training and test data