Robust Learning

Adversarial Attacks

- $$\mathbb{E}_{x,y}[L(f_\theta(x).\ y)] \to \mathbb{E}_{x,y}[\max_{\delta\in\Delta} L(f_\theta(x+\delta), y)]$$
- Use **Projected Gradient Descent** to find optimal $\delta \in \Delta$:
- $\delta := \mathcal{P}_\Delta(\delta + \nabla_\delta L(x+\delta.\ y;\theta))$
- **Fast Gradient Sign method**: Let $\Delta = \{\delta : |\delta|_\infty \leq \epsilon\}$
- As $\alpha \to \infty$, we always reach the corner: $\delta = \epsilon \cdot sign(\nabla_\delta L(x_\delta, y;\theta))$

Adversarial training

- Objective:
$$\min_\theta \sum_{x,y\in S} \max_{\delta\in\Delta} L(f_\theta(x+\delta), y)$$
- Repeat:
  - Select minibatch B
  - For each $(x, y) \in B$, compute adversarial example $\delta^*(x)$
  - Update parameters: $\theta := \theta - \frac{\alpha}{|B|}\sum_{x,y\in B}\frac{\partial}{\partial\theta}L(f_\theta(x+\delta^*(x)), y)$
- Evaluate robust model:
- Robust models are not universal: Robust on ℓ_1 does not guarantees robustness on ℓ_2 or ℓ_∞
- Robust Feature dataset:
  - First get a robust model M
  - For each training image x we generate x_r from random initialization and gradient descent s.t.:
  - $g(x) = g(x_r)$, where g is the feature extraction function of M
  - Train from scratch using dataset $\{x_r\}$ yields good robust performance
  - Intuitively, x_r have similar robust features but the non-robust features are distributed independently

Attack techniques

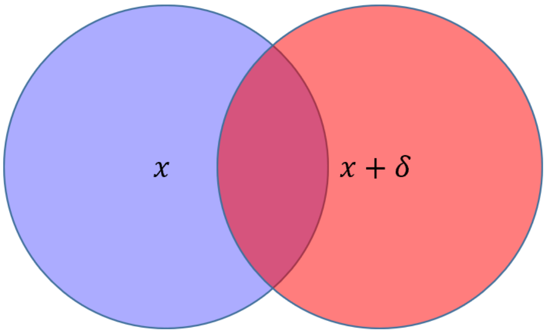
- Attack model with hidden gradient: Backward pass differentiable approximation
- Attack model with randomization: Take expectation of gradient by sampling

Provable robust certificates

- In high dimensional space, adversarial points are easy to find (Spiky)
- Use **histogram** to smooth the prediction: Draw a big ball and predict the label with maximal volume
- Largest perturbation δ: $f(x) \neq f(x+\delta)$. To make small for test set
- Suppose f is the base classifier, we find a robust classifier g:

$$g(x) = \int_{v \in B_r(x)} f(x+v) \, \mathrm{Pr}(v) \mathrm{d}v$$

- We need to make $g(x)$ of the blue area bigger than $1/2$ and at the same time make $g(x)$ of the red area smaller than $1/2$, in order to attack the model. This is a continuous version of knapback problem, so we can greedily fill colors in the purple area to red or blue, to try to satisfy these constraints.



- For each point y, compute the likelihood:

$$\frac{\mathrm{Pr}_{B_r(x)}(y)}{\mathrm{Pr}_{B_r(x+\delta)}(y)}$$

  - Sort the likelihood from big to small,color y to blue or red greedily