Recap:

Last lecture ⇒ tricks to improve performance

① Ensemble → average over multiple models

② Dropout → can be viewed as stochastic version of ensemble

⇕

cooperate with Neural Networks

③ Batch Normalization → $\boxed{\text{Decouple}}$

④ Data Augmentation → $\boxed{\text{change-invariant prior}}$
→ more data, less generalization gap

⑤ Learning Rate Decay Scheme

⇓

- High learning rate at begining stage
- Then decay over training

⑥ Adversarial Training

Defn: $\boxed{\text{Robust}}$

⇕

$\hat{f}(x') = \hat{f}(x)$ for $\forall \|x' - x\| \ll 1$

small perturbation

DSA5204 Lec 8

$\boxed{\text{Continuous topic}}$: (one last topic)

→ Adversarial Training ⟷ $\boxed{\text{adversarial example}}$

① Definition (adversarial example)

given parameter $\hat{\theta}$, $\boxed{\text{adversarial example}}$ is $\boxed{x' = \underset{z: \|z-x\| \le \delta}{\arg\max} L(\hat{y}(z, \hat{\theta}), y)}$ → the worst example

② Definition (adversarial training)

How can we reduce the effect of "Adversarial example"?

↓

$\boxed{\text{1-sample}}$ $\min_{\theta} \boxed{\max_{z: \|x-z\| \le \delta} L(\hat{y}(z; \theta), y)}$

$\boxed{\text{can be achieved by gradient ascent}}$

$\boxed{\text{multi-sample}}$ $$\min_\theta \underline{\frac{1}{N}\sum_{i=1}^{N} \max_{z_i : \|x_i - z_i\| \le \delta} L(\hat{y}(z_i;\theta), y)}$$

③ Algorithm (Fast Gradient Sign Method [FGSM])

→ For $k = 0, 1, \dots$

$$z_0 = x$$

gradient ascent in input space $\left\{ \begin{array}{l} \text{For } j = 0, 1, \dots, J-1 \\ \boxed{z_{j+1} = z_j + \varepsilon_2 \,\mathrm{sign}(\nabla_z L(\hat{y}(z_j;\theta_k), y))} \end{array} \right.$

find the "Adversarial Example"

$$\theta_{k+1} = \theta_k - \varepsilon_1 \nabla_\theta L(\hat{y}(z_J;\theta_k), y)$$

in argmax

Here, $\delta = J \times \varepsilon_2$ (we want $\|z - x\| \le \delta$)

Rmk: Generally speaking, we do not need to pre-train model first, then find adversarial examples and store them, lastly re-train model.

Instead, we modify the loss function and train model directly

$$\min_\theta \max_{z : \|z - x\| \le \delta} L(\hat{y}(z;\theta), y)$$

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$$

$\boxed{\text{Something like Data Augmentation}}$

④ 1-D Example

Model: $\hat{y}(x;\theta) = \theta x$

Data $\{x = 1, y = 0\} = \mathcal{D}$

Loss function: $L(\hat{y}, y) = \begin{cases} \frac{1}{2}(\hat{y} - y - \mathrm{sign}(\hat{y} - y))^2 & \text{o/w} \\ 0 & |\hat{y} - y| \le 1 \end{cases}$

Then $L(\hat{y}, y) = \begin{cases} \frac{1}{2}(\theta - \text{sign}(\theta))^2 & |\theta| > 1 \\ 0 & |\theta| \leq 1 \end{cases}$

$\Downarrow$

Trivial Loss, via GD, $\boxed{\hat{\theta}^{\infty}_{GD} = 1}$ if we start from $\theta^{(0)} > 1$

$\rightarrow$ Adversarial Loss $\quad L_{adv}(\theta) = \max_{z : \|1 - z\| \leq \delta} L(\hat{y}(z, \theta), y)$

$$= \max_{z : \|1 - z\| \leq \delta} \frac{1}{2}(\theta z - \text{sign}(\theta z))^2 \mathbb{1}\{|\theta z| \geq 1\}$$

$$= \frac{1}{2}(\theta(1+\delta) - 1)^2 \mathbb{1}\{|\theta(1+\delta)| \geq 1\}$$

Model $\boxed{y = \theta x}$

$y = 1$ and $x \approx 0$

$\Rightarrow$ we want $\theta \approx 0$

$\Rightarrow \hat{\theta}^{\infty}_{adv} = \frac{1}{1+\delta}$ is more close to 0

and $L_{adv}(\hat{\theta}^{\infty}_{GD}) = \frac{\delta^2}{2} > 0$

Today's topic: Unsupervised Learning + Semi-supervised Learning

1. Task

a) Supervised Learning $\longrightarrow$ input: $x \xleftrightarrow{GT:\ f^*}$ output: y

b) Unsupervised Learning → no output !

↓

learn some task-agnostic patterns

① Dimensionality Reduction

② Generative Model

③ Clustering

④ Density Estimation

$z \longrightarrow \boxed{f(z)}$

c) Semi-supervised Learning

2. PCA

① Design Matrix $X \in \mathbb{R}^{N \times d}$

② Covariance Matrix $S = \frac{1}{N} X^T X$

③ Eigenvalue Decomp. $S = U \Sigma U^T$

$\Updownarrow$

$SU = U\Sigma$ $\boxed{U = (u_1, \dots, u_d)}$

$u_i \in \mathbb{R}^d$

m Principle component direction

$U_m = (u_1, \dots, u_m)$ $m < d$

↓

$\boxed{Z_m = X U_m \in \mathbb{R}^{N \times m}}$ → Principle Scoring

Reconstruction : $X \approx X U_m U_m^T = Z_m \cdot U_m^T \in \mathbb{R}^{N \times d}$

$= Z_m \begin{pmatrix} u_1^T \\ \vdots \\ u_m^T \end{pmatrix}$

explained variance : $\boxed{\sum_{i=1}^{m} \lambda_i}$

3. Auto-Encoder (AE)

Re-cap : $\boxed{\text{PCA framework}}$

$$x: \text{input} \xrightarrow{U_m^T} z \in \mathbb{R}^m \xrightarrow{U_m} x': \text{compressed}$$

small reconstruction loss

$\Rightarrow$ Generalization (AE): small reconstruction loss

$$x: \text{input} \xrightarrow{f_{enc}(\cdot)} z \in \mathbb{R}^m \xrightarrow{g_{dec}(\cdot)} x': \text{compressed}$$

small reconstruction loss

"Denoising AE"

$$x \xrightarrow{\text{noise}} \tilde{x} \xrightarrow{\text{AE}} x'$$

$\boxed{\text{small reconstruction loss}}$

4. Semi-supervised Learning

a) transductive / inductive task

b) Method to label unlabelled data

① Naive approach (Self-learning)

train model → label unlabelled → re-train

② Label propagation

Rmk: different from some clustering algorithms like

$\boxed{\text{K-means}}$ etc.