Adversarial Training

AmirHossein Bagheri
Radmehr Karimian
Zahra Sodagar

Sharif University of Technology
Department of Electrical Engineering

High Dimensional Probability
Prof. Mohammad Hossein Yassaee

July 9, 2023

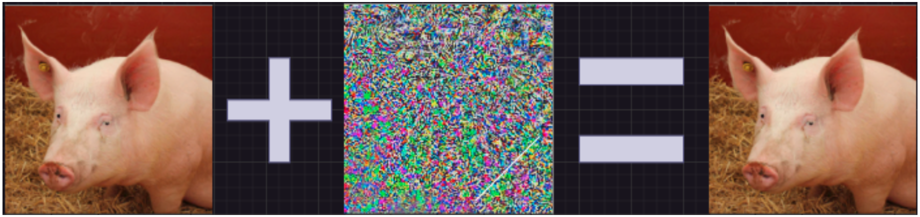
Table of Contents

Contents

Targeted attacks

Targeted attacks

1. Adding adversarial noise to input changes model output from pig to airline
2. It is done by targeted attach means we can lead the noise to any arbitrary class
3. So reliability of model is questioned



Definition

Problem Definition

Adversarial training is a technique that trains models on both clean and adversarial examples to enhance their robustness against adversarial attacks. By exposing the model to perturbed data during training, it learns to better handle and detect such attacks, ultimately improving its overall performance and reliability. [1] [2] [3]

Importance & App

Importance & Applications

1. Safe AI and Trust
2. Law and principal appliance
3. Avoid race , gender (etc) discrimination

Notations

Mathematical Notation

$$\ell(h_\theta(x), y) \tag{1}$$

$$\underset{\theta}{\text{minimize}} \frac{1}{m} \sum_{i=1}^{m} \ell\left(h_\theta\left(x_i\right), y_i\right) \tag{2}$$

$$\theta := \theta - \frac{\alpha}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla_\theta \ell\left(h_\theta\left(x_i\right), y_i\right) \tag{3}$$

Mathematical Notation Adversarial Enters

$$\underset{\hat{x}}{\text{maximize}}\, \ell\left(h_\theta(\hat{x}), y\right) \tag{4}$$

$$\underset{\delta \in \Delta}{\text{maximize}}\, \ell\left(h_\theta(x+\delta), y\right) \tag{5}$$

Notations Cont.

Adversarial

$$\Delta = \{\delta : \|\delta\|_p \leq \epsilon\} \tag{6}$$

$$\underset{\delta \in \Delta}{\text{maximize}}\, \ell\left(h_\theta(x+\delta), y\right) \tag{7}$$

Targeted Adversarial

$$\underset{\delta \in \Delta}{\text{maximize}}\left(\ell\left(h_\theta(x+\delta), y\right) - \ell\left(h_\theta(x+\delta), y_{\text{target}}\right)\right) \tag{8}$$

If we use Cross Entropy loss function

$$\underset{\delta \in \Delta}{\text{maximize}}\left(h_\theta(x+\delta)_{y_{\text{target}}} - h_\theta(x+\delta)_y\right) \tag{9}$$

Risk

Risk

Traditional notion of risk as it is used in machine learning

$$R\left(h_{\theta}\right)=\mathbf{E}_{(x,y)\sim\mathcal{D}}\left[\ell\left(h_{\theta}(x)\right),y)\right] \tag{10}$$

$$D=\left\{(x_i,y_i)\sim\mathcal{D}\right\},i=1,\ldots,m \tag{11}$$

Empirical Risk

Consider the empirical risk

$$\hat{R}\left(h_{\theta},D\right)=\frac{1}{|D|}\sum_{(x,y)\in D}\ell(h_{\theta}(x),y) \tag{12}$$

Our objection would be

$$\underset{\theta}{\operatorname{minimize}}\,\hat{R}\left(h_{\theta},D_{\text{train}}\right) \tag{13}$$

Adversarial Risk

Adversarial Risk

Alternative to the traditional risk, we can also consider an adversarial risk

$$R_{\mathrm{adv}}\left(h_\theta\right)=\mathbf{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\Delta(x)}\ell\left(h_\theta(x+\delta)\right),y\Bigg)\right] \tag{14}$$

$$\hat{R}_{\mathsf{adv}}\left(h_\theta,D\right)=\frac{1}{|D|}\sum_{(x,y)\in D}\max_{\delta\in\Delta(x)}\ell\left(h_\theta(x+\delta)\right),y) \tag{15}$$

Our new Objective function is

$$\underset{\theta}{\operatorname{minimize}}\,\hat{R}_{\mathrm{adv}}\left(h_\theta,D_{\mathsf{train}}\right)\equiv\underset{\theta}{\operatorname{minimize}}\,\frac{1}{\left|D_{\mathsf{train}}\right|}\sum_{(x,y)\in D_{\mathsf{train}}}\max_{\delta\in\Delta(x)}\ell(h_\theta(x+\delta),y) \tag{16}$$

Update rule would be

$$\theta:=\theta-\frac{\alpha}{|B|}\sum_{(x,y)\in B}\nabla_\theta\max_{\delta\in\Delta(x)}\ell\left(h_\theta(x+\delta)\right),y) \tag{17}$$

Solving Inner Maximization

Solving Inner Maximization

$$L(z) = \log(1 + \exp(-z)) \tag{18}$$

$$\underset{\|\delta\| \le \epsilon}{\text{maximize}}\, \ell\left(w^T(x+\delta), y\right) \equiv \underset{\|\delta\| \le \epsilon}{\text{maximize}}\, L\left(y \cdot \left(w^T(x+\delta) + b\right)\right) \tag{19}$$

$$\max_{\|\delta\| \le \epsilon} L\left(y \cdot \left(w^T(x+\delta) + b\right)\right) = L\left(y \cdot \left(w^T x + b\right) + \min_{\|\delta\| \le \epsilon} y \cdot w^T \delta\right) \tag{20}$$

Solving Inner Maximization

Solving ℓ_∞

1. Consider $y = +1$, an ℓ_∞ norm constraint. $\|\delta\|_\infty \leq \epsilon$
2. minimize this quantity when we set $\delta_i = -\epsilon$ for $w_i \geq 0$ and $\delta_i = \epsilon$ for $w_i < 0$
3. For $y = -1$, we would just flip these quantities

$$\delta^\star = -y\epsilon \cdot \operatorname{sign}(w) \tag{21}$$

$$y \cdot w^T \delta^\star = y \cdot \sum_{i=1} -y\epsilon \cdot \operatorname{sign}(w_i)\, w_i = -y^2\epsilon \sum_i |w_i| = -\epsilon\|w\|_1 \tag{22}$$

$$\max_{\|\delta\|_\infty \leq \epsilon} L\left(y \cdot \left(w^T(x+\delta) + b\right)\right) = L\left(y \cdot \left(w^T x + b\right) - \epsilon\|w\|_1\right) \tag{23}$$

Solving Inner Maximization

Solve robust optimization for Linear case

$$\min_{\|\delta\|\leq\epsilon} y \cdot w^T\delta = -\epsilon\|w\|_* \tag{24}$$

$$\underset{w,b}{\text{minimize}} \frac{1}{D} \sum_{(x,y)\in D} L\left(y \cdot \left(w^T x + b\right) - \epsilon\|w\|_*\right) \tag{25}$$

1. where $\|\cdot\|_*$ denotes the the dual norm of our original norm bound on $\theta\,(\|\cdot\|_p$ and $\|\cdot\|_q$ are dual norms for $1/p + 1/q = 1)$
2. It is also in contradiction with usual form

$$\text{minimize}_{w,b} \frac{1}{D} \sum_{(x,y)\in D} L\left(y \cdot \left(w^T x + b\right)\right) + \epsilon\|w\|_*$$

Contents

Rademacher Complexity for Adversarially Robust Generalization [4]

Complexity Bound

Theorem 1. Suppose that the range of $\ell(f(\mathrm{x}), y)$ is $[0, B]$. Then, for any $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds for all $f \in \mathcal{F}$,

$$R(f) \leq R_n(f) + 2B\mathfrak{R}_{\mathcal{S}}\left(\ell_{\mathcal{F}}\right) + 3B\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Define the adversarial loss $\widetilde{\ell}(f(\mathbf{x}), y) := \max_{\mathbb{B}_{\mathbf{x}}^{\infty}(\boldsymbol{\epsilon})} \ell\left(f\left(\mathbf{x}'\right), y\right)$ and the function class $\widetilde{\ell}_{\mathcal{F}} \subseteq [0, B]^{\mathcal{X}\times\mathcal{Y}}$ as $\widetilde{\ell}_{\mathcal{F}} := \{\widetilde{\ell}(f(\mathbf{x}), y) : f \in \mathcal{F}\}$.
The following direct corollary 1 of Theorem 1 .

$$\widetilde{R}(f) \leq \widetilde{R}_n(f) + 2B\mathfrak{R}_{\mathcal{S}}\left(\widetilde{\ell}_{\mathcal{F}}\right) + 3B\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Rademacher complexity of the adversarial loss function

generalization ability

1. Rademacher complexity of the adversarial loss function class is the Key quantity for the generalization ability of the learning problem
2. Linear Classifiers
    1. Binary Classification
    2. Multi-class Classification
3. Neural Networks

Rademacher complexity of Binary Linear classifier

Theorem 2

Let $\mathcal{F} := \{f_{\mathbf{w}}(\mathbf{x}) : \|\mathbf{w}\|_p \le W\}$ and $\widetilde{\mathcal{F}} := \left\{\min_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} y \langle \mathbf{w}, \mathbf{x}' \rangle : \ \|\mathbf{w}\|_p \le W\right\}$. Suppose that $\frac{1}{p} + \frac{1}{q} = 1$. Then, there exists a universal constant $c \in (0, 1)$ such that

$$\max\left\{\mathfrak{R}_{\mathcal{S}}(\mathcal{F}), c\epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}\right\} \le \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \le \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}.$$

The adversarial Rademacher complexity, i.e., $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$ is always at least as large as the Rademacher complexity in the natural setting. This implies that uniform convergence in the adversarial setting is at least as hard as that in the natural setting. In addition, since $\max\{a, b\} \ge \frac{1}{2}(a + b)$, we have

$$\frac{c}{2}\left(\mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}\right) \le \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \le \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}.$$

Multi-class Linear Classifiers

Theorem 2 .cont

Corollary 2. Consider the multi-class classification setting. For any fixed $\gamma > 0$, we have with probability at least $1-\delta$, for all $f \in \mathcal{F}$,

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y \neq \arg\max_{y'\in[K]}[f(\mathbf{x})]_{y'}\right\} \leq$$

$$\frac{1}{n}\sum_{i=1}^{n}\not\Vdash\left(\left[f\left(\mathbf{x}_i\right)\right]_{y_i} \leq \gamma + \max_{y'\neq y}\left[f\left(\mathbf{x}_i\right)\right]_{y'}\right) + 2\Re_{\mathcal{S}}\left(\ell_{\mathcal{F}}\right) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}}$$

Where γ is the least margin in margin operator for true label $M(\mathbf{z},y) : \mathbb{R}^K \times [K] \rightarrow \mathbb{R}$ as $M(\mathbf{z},y) = z_y - \max_{y'\neq y} z_{y'}$

Linear Classifier Empirical Study

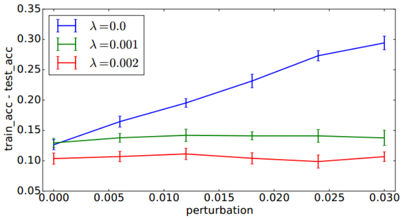


Figure: Adversarial generalization error vs ℓ_∞ perturbation ϵ and regularization coefficient λ.

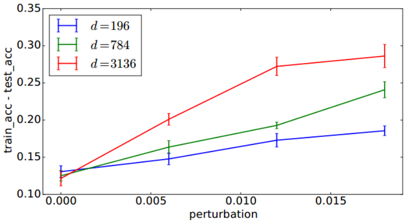


Figure: Adversarial generalization error vs ℓ_∞ perturbation ϵ and dimension of feature space d.

Figure: $\min_{\mathbf{W}} \frac{1}{n} \sum_{i=1}^{n} \max_{\mathbf{x}'_i \in \mathbb{B}^{\mathbf{x}}_{\mathbf{x}_i}(\epsilon)} \ell\left(f_{\mathbf{W}}\left(\mathbf{x}'_i\right), y_i\right) + \lambda \|\mathbf{W}\|_1$

Neural Networks

Theorem 5

Consider the neural network hypothesis class

$$\mathcal{F} = \left\{ f_{\mathbf{W}}(\mathbf{x}) : \mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L), \|\mathbf{W}_h\|_\sigma \leq s_h, \left\|\mathbf{W}_h^\top\right\|_{2,1} \leq b_h, h \in [L] \right\} \subseteq \mathbb{R}^{\mathcal{X}}$$

Then, we have

$$\mathfrak{R}_{\mathcal{S}}(\mathcal{F}) \leq \frac{4}{n^{3/2}} + \frac{26 \log(n) \log(2d_{\max})}{n} \|\mathbf{X}\|_F \left(\prod_{h=1}^{L} s_h \right) \left(\sum_{j=1}^{L} \left(\frac{b_j}{s_j} \right)^{2/3} \right)^{3/2}$$

Neural Networks Empirical Study

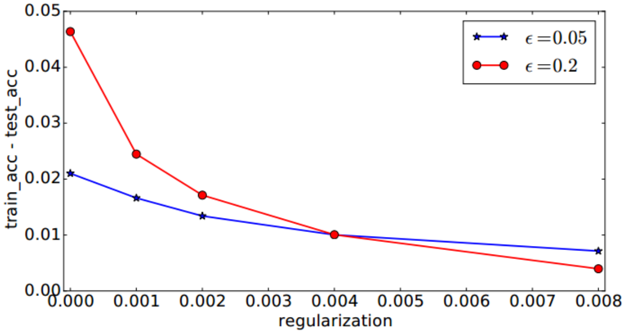


Figure: Adversarial generalization error vs regularization coefficient λ for a four-layer ReLU neural network with two convolutional layers and two fully connected layers. The results show that ℓ_1 regularization can effectively reduce the adversarial generalization error under the PGD attack.

$$\min_{\mathbf{W}} \frac{1}{n} \sum_{i=1}^{n} \max_{\mathbf{x}_i' \in \mathbb{B}_{\mathbf{x}_i}^{\mathbf{x}}(\epsilon)} \ell\left(f_{\mathbf{W}}\left(\mathbf{x}_i'\right), y_i\right) + \lambda \|\mathbf{W}\|_1$$

Precise Trade offs in Adversarial Training for Linear Regression [5]

Setup

$$y_i = \langle \boldsymbol{x}_i, \boldsymbol{\theta}_0 \rangle + w_i \text{ where } w_i \sim \mathrm{N}\left(0, \sigma_0^2\right).$$

$$\widehat{\boldsymbol{\theta}}^{\varepsilon} \in \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \max_{\|\boldsymbol{\delta}_i\|_{\ell_2} \le \varepsilon} \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \langle \boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{\theta} \rangle\right)^2$$

(SR, AR) Region and its Pareto Optimal Curve. $\widehat{\boldsymbol{\theta}}$ we use $\mathrm{SR}(\widehat{\boldsymbol{\theta}})$ and $\mathrm{AR}(\widehat{\boldsymbol{\theta}})$ to denote the standard and adversarial risks achieved by $\widehat{\boldsymbol{\theta}}$. We can derive all the Pareto optimal points of the (SR, AR) region, by minimizing a weighted combination of these two accuracies for different weights λ.

$$\boldsymbol{\theta}^{\lambda} = \arg\min_{\boldsymbol{\theta}} \lambda \overbrace{\mathbb{E}\left\{\left(y - \langle \boldsymbol{x}, \boldsymbol{\theta} \rangle\right)^2\right\}}^{\text{standard risk}} + \overbrace{\mathbb{E}\left\{\max_{\|\boldsymbol{\delta}\|_{\ell_2} \le \varepsilon_{\text{test}}} \left(y - \langle \boldsymbol{x} + \boldsymbol{\delta}, \boldsymbol{\theta} \rangle\right)^2\right\}}^{\text{adversarial risk}}.$$

Analytical Expression of the Optimal Trade offs

Lemma 3.1

$$\mathrm{SR}(\widehat{\boldsymbol{\theta}}) := \frac{1}{p}\mathbb{E}\left[(y - \langle \boldsymbol{x}, \widehat{\boldsymbol{\theta}}\rangle)^2\right] = \frac{\sigma_0^2}{p} + \frac{1}{p}\left\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0\right\|_{\ell_2}^2,$$

$$\mathrm{AR}(\widehat{\boldsymbol{\theta}}) := \frac{1}{p}\mathbb{E}\left[\max_{\|\boldsymbol{\delta}\|_{\ell_2} \le \varepsilon_{\text{test}}} (y - \langle \boldsymbol{x} + \boldsymbol{\delta}, \widehat{\boldsymbol{\theta}}\rangle)^2\right]$$

$$\frac{1}{p}\left(\sigma_0^2 + \left\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0\right\|_{\ell_2}^2 + \varepsilon_{\text{test}}^2 \|\widehat{\boldsymbol{\theta}}\|_{\ell_2}^2\right) + 2\sqrt{\frac{2}{\pi}}\frac{\varepsilon_{\text{test}}}{\sqrt{p}}\|\widehat{\boldsymbol{\theta}}\|_{\ell_2}\left(\frac{\sigma_0^2}{p} + \frac{1}{p}\left\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0\right\|_{\ell_2}^2\right)^{1/2}.$$

The solution $\boldsymbol{\theta}^\lambda$ of the optimization problem is given by $\boldsymbol{\theta}^\lambda = \left(1 + \gamma_0^\lambda\right)^{-1}\boldsymbol{\theta}_0$, with γ_0^λ the fixed point of the following two equations:

$$\gamma_0^\lambda = \frac{\varepsilon_{\text{test}}^2 + \sqrt{\frac{2}{\pi}}\varepsilon_{\text{test}}\, A^\lambda}{1 + \lambda + \sqrt{\frac{2}{\pi}}\frac{\varepsilon_{\text{test}}}{A^\lambda}} \qquad A^\lambda = \frac{1}{\|\boldsymbol{\theta}_0\|_{\ell_2}}\left(\left(1 + \gamma_0^\lambda\right)^2\sigma_0^2 + \left(\gamma_0^\lambda\right)^2\|\boldsymbol{\theta}_0\|_{\ell_2}^2\right)^{1/2}.$$

Theoretical Analysis of Adversarial Learning: A Minimax Approach [6]

Setup

1. Push forward the original distribution P into a new distribution P' which id dependent on H using a transport map $T_h : \mathcal{Z} \to \mathcal{Z}$ satisfying

$$R_P(h, \mathcal{B}) = R_{P'}(h),$$

2. $T_h : \mathcal{Z} \to \mathcal{Z}$

$$z = (x, y) \to (x^*, y)\,,$$

where $x^* = \arg\max_{x' \in N(x)} l\left(h\left(x'\right), y\right)$

3. Define the radius of the adversary $\mathcal{B}$ as $\epsilon_{\mathcal{B}} := \sup_{x \in \mathcal{B}} d_{\mathcal{X}}(x, 0)$. For any hypothesis h and the corresponding $P' = T_h \# P$, we have

$$W_p\left(P, P'\right) \leq \epsilon_{\mathcal{B}}.$$

4.

$$R_P(h, \mathcal{B}) \leq R_{\epsilon_{\mathcal{B}},1}(P, h), \quad \forall h \in \mathcal{H}$$

Theoretical Analysis of Adversarial Learning: A Minimax Approach

Assumptions

1. The instance space $\mathcal{Z}$ is bounded: $\operatorname{diam}(\mathcal{Z}) := \sup_{z,z' \in \mathcal{Z}} d\mathcal{Z}\left(z, z'\right) < \infty$
2. functions in $\mathcal{F}$ are upper semicontinuous and uniformly bounded: $0 \leq f(z) \leq M < \infty$ for all $f \in \mathcal{F}$ and $z \in \mathcal{Z}$
3. For any function $f \in \mathcal{F}$ and any $z \in \mathcal{Z}$, there exists a constant λ such that $f\left(z'\right) - f(z) \leq \lambda d_{\mathcal{Z}}\left(z, z'\right)$ for any $z' \in \mathcal{Z}$

For any upper semicontinuous function $f : \mathcal{Z} \rightarrow \mathbb{R}$ and for any $P \in \mathcal{P}_p(\mathcal{Z})$
$R_{\epsilon_{\mathcal{B}},1}(P, f) = \min_{\lambda \geq 0} \left\{\lambda \epsilon_{\mathcal{B}} + \mathbb{E}_P\left[\varphi_{\lambda,f}(z)\right]\right\}$, where
$\varphi_{\lambda,f}(z) := \sup_{z' \in \mathcal{Z}} \left\{f\left(z'\right) - \lambda \cdot d_{\mathcal{Z}}\left(z, z'\right)\right\}$

Theoretical Analysis of Adversarial Learning: A Minimax Approach

Theorem

Define the function class $\Phi := \{\varphi_{\lambda,f} : \lambda \in [a,b], f \in \mathcal{F}\}$ where $b \geq a \geq 0$. Then, the expected Rademacher complexity of the function class Φ satisfies

$$\Re_n(\Phi) \leq \frac{12\mathfrak{C}(\mathcal{F})}{\sqrt{n}} + \frac{6\sqrt{\pi}}{\sqrt{n}}(b-a)\cdot \mathrm{diam}(Z)$$

where $\mathfrak{C}(\mathcal{F}) := \int_0^\infty \sqrt{\log \mathcal{N}\left(\mathcal{F}, \|\cdot\|_\infty, u/2\right)}du$ and $\mathcal{N}\left(\mathcal{F}, \|\cdot\|_\infty, u/2\right)$ denotes the covering number of $\mathcal{F}$. If the assumptions $[1-3]$ hold, then for any $f \in \mathcal{F}$, we have below with probability at least $1-\delta$.

$$R_{e_B,1}(P,f) - R_{e_B,1}\left(P_n, f\right) \leq \frac{24\mathfrak{C}(\mathcal{F})}{\sqrt{n}} + M\sqrt{\frac{\log\left(\frac{1}{6}\right)}{2n}} + \frac{12\sqrt{\pi}}{\sqrt{n}}\Lambda_{e_B}\cdot \mathrm{diam}(Z)$$

Theoretical Analysis of Adversarial Learning: A Minimax Approach

Example

PCA:

$$R_P(f,\mathcal{B}) \le \frac{1}{n}\sum_{i=1}^{n} f\left(z_i\right) + \lambda^{+}_{f,P_n}\epsilon_{\mathcal{B}} + \frac{576B^2k\sqrt{m}}{\sqrt{n}} + \frac{24B\sqrt{\pi}}{\sqrt{n}}\Lambda_{\epsilon_{\mathcal{B}}} + B^2\sqrt{\frac{\log(1/\delta)}{2n}}$$

SVM:

$$R_P(f,\mathcal{B}) \le$$

$$\frac{1}{n}\sum_{i=1}^{n} f\left(z_i\right) + \lambda^{+}_{f,P_n}\epsilon_{\mathcal{B}} + \frac{144}{\sqrt{n}}\Lambda r\sqrt{d} + \frac{12\sqrt{\pi}}{\sqrt{n}}\Lambda_{\epsilon_{\mathcal{B}}}\cdot(2r+1) + (1+\Lambda r)\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2n}}$$

Contents

Stability of Adversarial Training

In this part we want to show that some upper and lower bound for adversarial Training. Let's begin with our problem:

define problem

The population adversarial loss is defined as :

$$R(\theta, \epsilon) := \mathbb{E}\left[l\left(f_\theta\left[x + A_\epsilon\left(f_\theta, x, y\right)\right], y\right)\right]$$

and the adversarial training minimizes the sample version is:

$$R_S(\theta, \epsilon) = \frac{1}{n}\sum_{i=1}^{n} l\left(f_\theta\left[x_i + A_\epsilon\left(f_\theta, x_i, y_i\right)\right], y_i\right)$$

Stability of Adversarial Training

now we can simply say :

Risk decomposition

$$R(\widehat{\theta}) - R\left(\theta_0\right) = \underbrace{R(\widehat{\theta}) - R_S(\widehat{\theta})}_{\mathcal{E}_{gen}} + \underbrace{R_S(\widehat{\theta}) - R_S(\bar{\theta})}_{\mathcal{E}_{\text{opt}}} +$$

$$+ \underbrace{R_S(\bar{\theta}) - R_S\left(\theta_0\right)}_{\leq 0} + \underbrace{R_S\left(\theta_0\right) - R\left(\theta_0\right)}_{\mathbb{E}=0}$$

it had shown that $\mathcal{E}_{gen}$ is upper bounded by algorithmic stability.

Stability of Adversarial Training

Uniform argument stability (UAS)

$$\sup_{S_1 \sim S_2} \left\| \widehat{\theta}\left(S_1\right) - \widehat{\theta}\left(S_2\right) \right\| := \sup_{S_1 \sim S_2} \lambda\left(S_1, S_2\right) \leq \lambda$$

so if we have

$$P\left(\left\|\lambda\left(S_1, S_2\right)\right\| \geq \gamma\right) \leq \kappa_0$$

we can say :

$$P\left[\left|\mathcal{E}_{gen}\right| \geq c\left(\gamma(\log n)(\log(n/\kappa)) + \sqrt{\frac{\log(1/\kappa)}{n}} \right) \right] \leq \kappa + \kappa_0$$

Stability of Adversarial Training

Upper Bound

For SGD and GD we will have:

$$\sup_{S_1 \sim S_2} \mathbb{E}\left[\left\|\theta_1^{(T)} - \theta_2^{(T)}\right\|\right] = O\left(\min\left\{r, L\sqrt{\sum_{t=1}^{T} \eta_t^2} + L\frac{\sum_{t=1}^{T} \eta_t}{n}\right\}\right)$$

Stability of Adversarial Training

Lower Bound

For SGD we will have:

$$\sup_{S_1 \sim S_2} \mathbb{E} \left\| \theta_1^{(T)} - \theta_2^{(T)} \right\| = \Omega \left(\min \left\{ 1, \frac{T}{n} \right\} \eta \sqrt{T} + \frac{\eta T}{n} \right)$$

And for GD we know:

$$\sup_{S_1 \sim S_2} \left\| \theta_1^{(T)} - \theta_2^{(T)} \right\| = \Omega \left(\min \left\{ 1, \eta \sqrt{T} + \frac{\eta T}{n} \right\} \right)$$

Stability of Adversarial Training

Upper Bound Corollary

For SGD and GD we will have:

$$\sup_{S_1 \sim S_2} \mathbb{E} \left\| \theta_1^{(T)} - \theta_2^{(T)} \right\| = O\left(\min\left\{ r, L\sqrt{\sum_{t=1}^{T} \eta_t^2} + L\frac{\sum_{t=1}^{T} \eta_t}{n} + \kappa \Delta\varepsilon \sum_{t=1}^{T} \eta_t \right\} \right)$$

Stability of Adversarial Training

Corollary

$$\mathbb{E}\left[\left(R\left(\widehat{\theta}\left(S_1\right)\right)-R_{S_1}\left(\widehat{\theta}\left(S_1\right)\right)\right)^2\right] \leq \frac{M^2}{2n}+$$
$$4\mathbb{E}\left[\sup_{\theta\in B_2(0,r)} l^2\left(f_\theta\left(x+A_\epsilon\right),y\right)1\{\|x\|\geq\sqrt{d\log n}\}\right]+$$
$$3M\mathbb{E}\left[\left|l\left(f_{\widehat{\theta}(S_1)}\left(x_i+A_\epsilon\right),y_i\right)-l\left(f_{\widehat{\theta}(S_2^i)}\left(x_i+A_\epsilon\right),y_i\right)\right|\right]$$

Wasserstein distance

Let X and Y be two metric spaces with distance functions d_X and d_Y, respectively. Let μ and ν be two probability measures on X and Y, respectively. The Wasserstein distance between μ and ν, denoted by $W(\mu, \nu)$, is defined as follows:

$$W(\mu, \nu) = \inf_{\gamma \in \Pi(\mu, \nu)} \int_{X \times Y} d_X(x, y) d\gamma(x, y)$$

where $\Pi(\mu, \nu)$ is the set of all joint probability measures on $X \times Y$ whose marginal distributions are μ and ν, respectively.
In other words, the Wasserstein distance is the minimum possible cost of transporting the mass of μ to that of ν, where the cost of transporting a unit mass from x to y is given by $d_X(x, y)$. The infimum is taken over all possible transport plans (joint probability measures) that achieve this task.
It is important to note that the Wasserstein distance is a true metric, meaning that it satisfies the following properties:

Wasserstein distance

- Non-negativity: $W(\mu, \nu) \geq 0$ for all probability measures μ and ν.
- Identity of indiscernibles: $W(\mu, \nu) = 0$ if and only if $\mu = \nu$.
- Symmetry: $W(\mu, \nu) = W(\nu, \mu)$ for all probability measures μ and ν.
- Triangle inequality: $W(\mu, \rho) \leq W(\mu, \nu) + W(\nu, \rho)$ for all probability measures μ, ν, and ρ.

Wasserstein distance

p-Wasserstein distance

$$W_p\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \min_{\Pi \in \mathbb{R}_+^{(n\cdot m)\times(n\cdot m)}} < \Pi, C >,$$

$$\Pi \mathbf{1} = \boldsymbol{x}, \Pi^T \mathbf{1} = \boldsymbol{x}',$$

$$C_{(i,j),(i',j')} := \left[d\left((i,j),(i',j')\right)\right]^p$$

Wasserstein distance

1-Wasserstein distance

$$W_1\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \min_{\mathbf{g}} \sum_{(i,j)} \sum_{(i',j') \in \mathcal{N}(i,j)} \mathbf{g}_{(i,j),(i',j')}$$

where $\mathbf{g}$ is:

$$\sum_{(i',j') \in \mathcal{N}(i,j)} \mathbf{g}_{(i,j),(i',j')} - \mathbf{g}_{(i',j'),(i,j)} = \mathbf{x}'_{i,j} - \mathbf{x}_{i,j}$$

Wasserstein distance

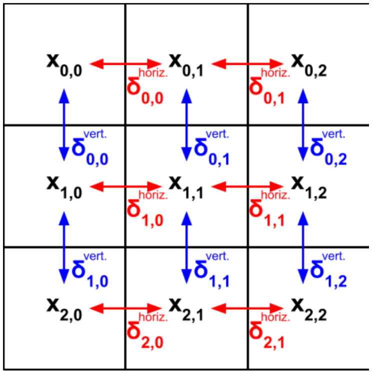

$x_{0,0}$
$x_{0,1}$
$x_{0,2}$
$x_{1,0}$
$x_{1,1}$
$x_{1,2}$
$x_{2,0}$
$x_{2,1}$
$x_{2,2}$
$\delta^{\text{horiz.}}_{0,0}$
$\delta^{\text{horiz.}}_{0,1}$
$\delta^{\text{horiz.}}_{1,0}$
$\delta^{\text{horiz.}}_{1,1}$
$\delta^{\text{horiz.}}_{2,0}$
$\delta^{\text{horiz.}}_{2,1}$
$\delta^{\text{vert.}}_{0,0}$
$\delta^{\text{vert.}}_{0,1}$
$\delta^{\text{vert.}}_{0,2}$
$\delta^{\text{vert.}}_{1,0}$
$\delta^{\text{vert.}}_{1,1}$
$\delta^{\text{vert.}}_{1,2}$

Figure: Indexing of the elements of the local flow map $delta$, in relation to the pixels of the image x, with $n = m = 3$.

Wasserstein distance

The local flow plan application function $\Delta \in \mathbb{R}^{n\times m} \times \mathbb{R}^{r} \to \mathbb{R}^{n\times m}$ is defined as:

$$\Delta(\boldsymbol{x},\boldsymbol{\delta})_{i,j} = \boldsymbol{x}_{i,j} + \boldsymbol{\delta}^{\text{vert.}}_{i-1,j} - \boldsymbol{\delta}^{\text{vert.}}_{i,j} + \boldsymbol{\delta}^{\text{horiz.}}_{i,j-1} - \boldsymbol{\delta}^{\text{horiz.}}_{i,j}$$

Note that the local flow plan are additive:

$$\Delta\left(\Delta(\boldsymbol{x},\boldsymbol{\delta}),\boldsymbol{\delta}'\right) = \Delta\left(\boldsymbol{x},\boldsymbol{\delta}+\boldsymbol{\delta}'\right)$$

lemma

$$W_1\left(\boldsymbol{x},\boldsymbol{x}'\right) = \min_{\boldsymbol{\delta}:\boldsymbol{x}'=\Delta(\boldsymbol{x},\boldsymbol{\delta})} \|\boldsymbol{\delta}\|_1$$

Wasserstein distance

We denote by $\mathcal{L}(\sigma) =$ Laplace $(0, \sigma)^r$ as the Laplace noise with parameter σ in the flow domain of dimension r.

Wasserstein-smoothed classification

$$\overline{\boldsymbol{f}} = \mathop{\mathbb{E}}_{\boldsymbol{\delta} \sim \mathcal{L}(\sigma)} [\boldsymbol{f}(\Delta(\boldsymbol{x}, \boldsymbol{\delta}))]$$

Wasserstein distance

Theorem

For any normalized probability distribution $\boldsymbol{x} \in [0,1]^{n \times m}$, if

$$\overline{\boldsymbol{f}}_i(\boldsymbol{x}) \geq e^{2\sqrt{2}\rho/\sigma} \max_{i' \neq i} \overline{\boldsymbol{f}}_{i'}(\boldsymbol{x})$$

then for any perturbed probability distribution $\tilde{\boldsymbol{x}}$ such that $W_1(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \leq \rho$, we have:

$$\overline{\boldsymbol{f}}_i(\tilde{\boldsymbol{x}}) \geq \max_{i' \neq i} \overline{\boldsymbol{f}}_{i'}(\tilde{\boldsymbol{x}}).$$

Wasserstein distance

We consider a hard smoothed classifier approach: we set $\boldsymbol{f}_j(\mathbf{x}) = 1$ if the base classifier selects class j at point $\mathbf{x}$, and $\boldsymbol{f}_j(\mathbf{x}) = 0$ otherwise. We also use a stricter form of the condition given as:

hard smoothed classifier

$$\overline{\boldsymbol{f}}_i(\boldsymbol{x}) \geq e^{2\sqrt{2}\rho/\sigma} \left(1 - \overline{\boldsymbol{f}}_i(\boldsymbol{x})\right)$$

Contents

References

[1] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[2] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

[3] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale, 2017.

[4] Dong Yin, Kannan Ramchandran, and Peter Bartlett. Rademacher complexity for adversarially robust generalization, 2020.

[5] Adel Javanmard, Mahdi Soltanolkotabi, and Hamed Hassani. Precise tradeoffs in adversarial training for linear regression, 2020.

[6] Zhuozhuo Tu, Jingwei Zhang, and Dacheng Tao. Theoretical analysis of adversarial learning: A minimax approach, 2019.

[7] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. *CoRR*, abs/2102.01356, 2021.

[8] Peilin Kang and Seyed-Mohsen Moosavi-Dezfooli. Understanding catastrophic overfitting in adversarial training, 2021.

[9] Hoki Kim, Woojin Lee, and Jaewook Lee. Understanding catastrophic overfitting in single-step adversarial training, 2020.

[10] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[11] Fabian Latorre, Igor Krawczuk, Leello Tadesse Dadi, Thomas Pethick, and Volkan Cevher. Finding actual descent directions for adversarial training. In *The Eleventh International Conference on Learning Representations*, 2023.

[12] Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming, 2019.

[13] Yue Xing, Qifan Song, and Guang Cheng. On the generalization properties of adversarial training, 2021.

Contents

Neural Networks Adversarial Training

Neural Networks [7]

1. Neural Networks, by the nature of their loss surfaces, are especially prone to adversarial examples
2. So how to solve inner maximization

$$\underset{\|\delta\|\leq\epsilon}{\text{maximize}}\, \ell\left(h_\theta(x), y\right)$$

3. Three main strategies
    1. Lower Bounds
    2. Exact Solutions
    3. Upper Bounds

Adversarial Regularization

More robust models usually have smaller values of local linearity, and the linearity of neural networks is attributed to the existence of adversarial examples.

1. FGSM-based Regularization

$$L(\theta, x + \varepsilon sign(\nabla_x L(\theta, x, y))) \tag{26}$$

2. Local Linearity Regularization (TRADE)

$$\min_f \mathbb{E}\left\{\mathcal{L}(f(x), y) + \max_{x' \in \mathbb{B}(x,\epsilon)} \mathcal{L}\left(f(x), f\left(x'\right)\right)/\lambda\right\} \tag{27}$$

3. Misclassification Aware adveRsarial Training (MART): Emphasizes on misclassified examples with weights of $1 - P_y(x, \theta)$.
4. Adversarial Logit Pairing (ALP): encouraging logits for pairs of examples to be similar and the alignment of representations of natural data and their adversarial counterparts.

Curriculum-based Adversarial Training

Adversarial examples generated by strong attacks significantly cross over the decision boundary, this leads to overfitting of adversarial examples, so the worst-case samples are not always suitable for adversarial training. We can improve the generalization of clean data while preserving adversarial robustness by using weaker attacks in early training.

1. Curriculum Adversarial Training (CAT): gradually increasing the iteration steps of PGD until the model achieves a high accuracy.
2. Friendly Adversarial Training (FAT): early stopping when performing PGD attacks and returning adversarial data near the decision boundary for training.
3. Other methods using First-Order Stationary Condition to estimate the convergence quality of the inner maximization problem.

Ensemble Adversarial Training

In Ensemble Adversarial Training (EAT), adversarial examples generated from different target models are used instead of a single model to reduce sharp curvatures caused by the single-step attacks. There are variations of this method since similar predictions or representations and share the adversarial subspace:

1. Forcing different models to be diverse in non-maximal predictions
2. Forcing different models to be diverse in non-maximal predictions
3. Maximizing the cosine distances among each target models' input gradients
4. Maximizing the vulnerability diversity (sum of losses)

Adversarial Training with Adaptive ϵ

Individual data points might have different intrinsic robustness and different distances to the classifier's decision boundary. Adversarial training with fixed ϵ treats all data equally.

1. Adaptive Adversarial Training (IAAT): ϵ is selected to be as large as possible, ensuring images within ϵ-ball of x are from the same class.
2. Margin Maximization Adversarial Training (MMA): maximizes the margin-distances between data points and the model's decision boundary (estimated by the adversarial perturbations with the least magnitudes).

Adversarial Training with Semi/Unsupervised Learning

- The adversarial accuracy in testing is much lower than in training and there is a large generalization gap in adversarial training.
- It is theoretically proved that adversarially robust training requires substantially larger datasets than standard training but quality datasets with labels are expensive to collect.
- Training with additional unlabeled data to reduce the sample complexity gap between standard training and adversarial training.
- Use unlabeled data for stability while labeled data for classification by decomposing the adversarial robustness like TRADES.

Generalization Problem in Adversarial Training

Standard Generalization

Adversarial training improves robustness but harms standard accuracy. There is a trade-off between adversarial robustness and standard accuracy, with some studies showing a negative correlation.

Adversarially Robust Generalization

Adversarially trained models perform poorly on adversarially perturbed test data, indicating overfitting. More data and empirical techniques like early stopping are used to improve generalization.

Generalization on Unseen Attacks

Adversarially trained models struggle to generalize to unseen attacks. Fixed constraints in training methods limit their effectiveness. Approaches like diversifying targeted models or modeling adversarial example distributions aim to approximate optimal solutions.

Lower Bound

Lower Bound Methods

1. By definition any feasible δ will give us a lower bound
2. Equivalent to just "trying to empirically solve the optimization problem" or "find an adversarial example".
3. Most common strategy for solving the inner maximization (motivated largely by the fact that for neural networks in general, problems of local optima don't seem as bad as initially may be thought)
4. Idea is quite simple: using back propagation, we can compute the gradient of the loss function with respect to the perturbation δ itself, so let's just perform gradient descent on δ to maximize our objective. However, we also need to ensure that δ stays within the norm bound ϵ, so after each step, we can project back into this space

Fast Gradient Sign Method (FGSM)

FGSM

FGSM [1] suggests For some given example x, we know that we are going to adjust δ in the direction of it's gradient, we will first compute the gradient

$$g := \nabla_{\delta}\ell\left(h_{\theta}(x+\delta), y\right)$$

for some step size α and then project back into the norm ball defined by $\|\delta\| \leq \epsilon$. For particular case of the ℓ_∞ norm $\|\delta\|_\infty \leq \epsilon$, where, as we mentioned before, projecting onto this norm ball simply involves clipping values of δ to lie within the range $[-\epsilon, \epsilon]$.

$$\delta^{t} := \delta^{t-1} + \mathrm{clip}(\alpha g, [-\epsilon, \epsilon])$$

but Doing it until convergence is slow so we estimate sign of gradient with only one step.

$$\delta := \epsilon \cdot \mathrm{sign}(g)$$

+ Catastrophic Over fitting occurs here [8] [9]

Projected Gradient Descent (PGD)

PGD

PGD [10] inspire more powerful adversary is the multi-step variant, which is essentially projected gradient descent (PGD) on the negative loss function. the basic PGD algorithm simply iterates the updates.
Repeat:

$$\delta := \mathcal{P}\left(\delta + \alpha \nabla_{\delta} \ell\left(h_{\theta}(x+\delta), y\right)\right)$$

where $\mathcal{P}$ denotes the projection onto the ball of interest (for example, clipping in the case of the ℓ_{∞} norm).
At the initial zero point, we need to scale it by a relatively large α to make any progress at all. Once we "break out" of the initial region around $\delta = 0$, the gradients typically increase in magnitude substantially, and at this point our large α is too large, and the method takes too big a step toward the boundary.

Projected Gradient Descent (PGD)

Steepest Descent

Speaking generically, if we want to minimize some function $f : \mathbb{R}^n \to \mathbb{R}$ over the input z, the traditional gradient descent algorithm repeats the update

$$z := z - \alpha \nabla_z f(z)$$

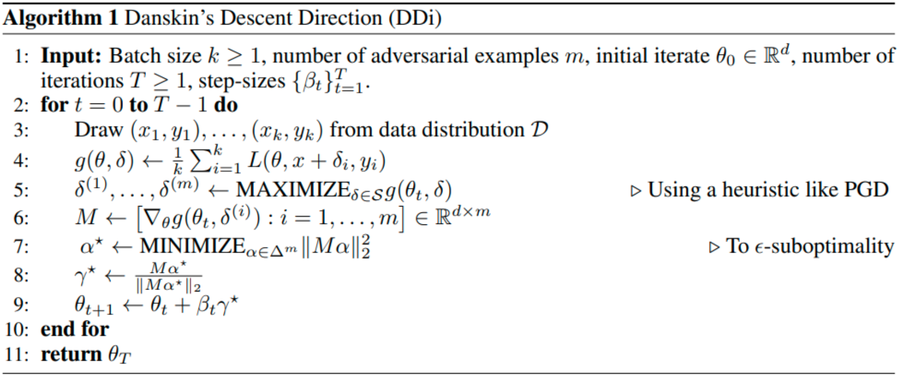
The normalized steepest descent method applies find some negative update direction v, where we choose v to maximize the inner product between v and the gradient subject to a norm constraint on v. That is normalized steepest descent performs the update

$$z := z - \underset{\|v\| \leq \alpha}{\mathrm{argmax}} v^T \nabla_z f(z)$$

In the case that we use an ℓ_2 norm constraint on v, the argmax has the analytical solution

$$\underset{\|v\|_2 \leq \alpha}{\mathrm{argmax}}\, v^T \nabla_z f(z) = \alpha \frac{\nabla_z f(z)}{\|\nabla_z f(z)\|_2}$$

Steepest Descent [11]



Algorithm 1 Danskin's Descent Direction (DDi)

```
1: Input: Batch size $k \geq 1$, number of adversarial examples $m$, initial iterate $\theta_0 \in \mathbb{R}^d$, number of
   iterations $T \geq 1$, step-sizes $\{\beta_t\}_{t=1}^T$.
2: for $t = 0$ to $T - 1$ do
3:     Draw $(x_1, y_1), \ldots, (x_k, y_k)$ from data distribution $\mathcal{D}$
4:     $g(\theta, \delta) \leftarrow \frac{1}{k} \sum_{i=1}^k L(\theta, x + \delta_i, y_i)$
5:     $\delta^{(1)}, \ldots, \delta^{(m)} \leftarrow \text{MAXIMIZE}_{\delta \in \mathcal{S}} g(\theta_t, \delta)$        ▷ Using a heuristic like PGD
6:     $M \leftarrow \left[\nabla_\theta g(\theta_t, \delta^{(i)}) : i = 1, \ldots, m\right] \in \mathbb{R}^{d \times m}$
7:     $\alpha^\star \leftarrow \text{MINIMIZE}_{\alpha \in \Delta^m} \|M\alpha\|_2^2$        ▷ To $\epsilon$-suboptimality
8:     $\gamma^\star \leftarrow \frac{M\alpha^\star}{\|M\alpha^\star\|_2}$
9:     $\theta_{t+1} \leftarrow \theta_t + \beta_t \gamma^\star$
10: end for
11: return $\theta_T$
```

Non ℓ_∞ Norm

ℓ_2

We just use the projected normalized steepest decent method for the ℓ_2 ball, which as discussed above has the form

$$\delta := \mathcal{P}_\epsilon\left(\delta - \alpha\frac{\nabla_\delta \ell\left(h_\theta(x+\delta), y\right)}{\|\nabla_\delta \ell\left(h_\theta(x+\delta), y\right)\|_2}\right)$$

where $\mathcal{P}_\epsilon$ now denotes the projection onto the ℓ_2 ball of radius ϵ. This projection in turn is just given by normalizing δ to have ℓ_2 norm ϵ if it is greater than ϵ,

$$\mathcal{P}_\epsilon(z) = \epsilon\frac{z}{\max\left\{\epsilon, \|z\|_2\right\}}$$

Exactly solving the inner maximization

Exactly solving the inner maximization

$$\begin{aligned} z_1 &= x \\ z_{i+1} &= f_i\left(W_i z_i + b_i\right), \quad i, \ldots, d \\ h_\theta(x) &= z_{d+1} \end{aligned}$$

Consider targeted adversarial attack

$$\begin{aligned} \underset{z_{1,\ldots,d+1}}{\text{minimize}} \; & \left(e_y - e_{y_{\text{targ}}}\right)^T z_{d+1} \\ \text{subject to } \; & \|z_1 - x\|_\infty \le \epsilon \\ & z_{i+1} = \max\left\{0, W_i z_i + b_i\right\}, i = 1, \ldots, d-1 \\ & z_{d+1} = W_d z_d + b_d \end{aligned}$$

Exactly solving the inner maximization

Exactly solving the inner maximization

It's provable that above problem is equal to

$$
\begin{gathered}
z_{i+1} = \max\{0, W_i z_i + b_i\} \\
z_{i+1} \geq W_i z_i + b_i \\
z_{i+1} \geq 0 \\
u_i \cdot v_i \geq z_{i+1} \\
W_i z_i + b_i \geq z_{i+1} + (1 - v_i)\, l_i \\
v_i \in \{0, 1\}^{|v_i|}
\end{gathered}
$$

Exactly solving the inner maximization

Final Formulation

Final formulation [12] would be

$$\begin{aligned}
\underset{z_{1,\dots,d+1}, v_{1,\dots,d-1}}{\text{minimize}} \quad & \left(e_y - e_{y_{\text{targ}}}\right)^T z_{d+1} \\
\text{subject to} \quad & z_{i+1} \geq W_i z_i + b_i, \quad i = 1\dots,d-1 \\
& z_{i+1} \geq 0, \quad i = 1\dots,d-1 \\
& u_i \cdot v_i \geq z_{i+1}, \quad i = 1\dots,d-1 \\
& W_i z_i + b_i \geq z_{i+1} + (1 - v_i)\, l_i, \quad i = 1\dots,d-1 \\
& v_i \in \{0,1\}^{|v_i|}, \quad i = 1\dots,d-1 \\
& z_1 \leq x + \delta \\
& z_1 \geq x - \delta \\
& z_{d+1} = W_d z_d + b_d.
\end{aligned}$$

Upper bounding the inner maximization

Upper bounding the inner maximization

It is important to be able to obtain fast upper bounds on the inner maximization problem. If, for example, we can attain an upper bound which still shows that no targeted attack can change the class label, this also provides a verification that no attack is possible

1. Convex Relaxation
$$0 \leq v_i \leq 1$$
2. Interval-propagation-based

Upper bounding the inner maximization

Interval-propagation-based

Bit more concretely, for a d-layer network our goal as we saw above was to minimize some linear function of the last layer $c^T z_{d+1}$. If we have an interval bound on the second to-last layer $\hat{l} \leq z_d \leq \hat{u}$, then we can simply solve the optimization problem

$$\begin{array}{ll} \underset{z_d, z_{d+1}}{\text{minizize}} & c^T z_{d+1} \\ \text{subject to} & z_{d+1} = W_d z_d + b_d \\ & \hat{l} \leq z_d \leq \hat{u}. \end{array}$$

We just eliminate the z_{d+1} variable using the first constraint, this problem is equivalent to

$$\underset{z_d}{\text{minimize}} c^T \left(W_d z_d + b_d\right) \equiv \left(W_d^T c\right)^T z_d + c^T b_d$$

subject to $\hat{l} \leq z_d \leq \hat{u}$.

Upper bounding the inner maximization

Interval-propagation-based

The analytic solution is just to choose $(z_d)_j = \hat{l}_j$ if $\left(W_d^T c\right)_j > 0$, and $(z_d)_j = \hat{u}_j$ otherwise. This results in the optimal objective value

$$\max\left\{c^T W_d c, 0\right\} \hat{l} + \min\left\{c^T W_d, 0\right\} \hat{u} + c^T b_d$$

Where c is class so verification can be Done.

On the Generalization Properties of Adversarial Training [13]

Setup

Consider the linear regression model: $f_\theta(x) = \theta^\top x$

$$R_L(\theta, \epsilon) = \|\theta - \theta_0\|_\Sigma^2 + \sigma^2 + \epsilon^2\|\theta\|^2 \quad +2\epsilon c_0\|\theta\|\sqrt{\|\theta - \theta_0\|_\Sigma^2 + \sigma^2},$$

where $\|a\|_\Sigma^2 := a^\top \Sigma a, \|a\| := \|a\|_2$ for any vector a, and $c_0 := \sqrt{2/\pi}$. Given any $\epsilon \geq 0$, define

$$\theta^*(\epsilon) := \underset{\theta}{\operatorname{argmin}} R_L(\theta, \epsilon) \text{ and } R^*(\epsilon) := \min_\theta R_L(\theta, \epsilon).$$

$$A_\epsilon\left(f_\theta, x, y\right) := \underset{z \in \mathcal{R}(0,\epsilon)}{\operatorname{argmax}} \left\{l\left(f_\theta(x+z), y\right)\right\}.$$

$$\widehat{R}_f(\theta, \epsilon) = \frac{1}{n}\sum_{i=1}^{n} l\left(f_\theta\left[x_i + A_\epsilon\left(f_\theta, x_i, y_i\right)\right], y_i\right),$$

Surrogate Attack

Surrogate Attack

$R_L(\theta, \epsilon)$ is not differentiable even if we smooth the standard loss to improve the quality of the gradients the non-differentiable issues remains for the adversarial loss. the adversarial training loss will fluctuate during training rather than strictly decreases over iterations. To solve this problem and improve the gradient quality, for both models under consideration, we introduce $A_{\epsilon,\xi}$ that is a surrogate for $\mathcal{L}_2$ attack A_ϵ :

$$A_{\epsilon,\xi}\left(f_\theta, x, y\right) = \frac{\left\|\partial l\left(f_\theta(x), y\right)/\partial x\right\|}{\sqrt{\left\|\partial l\left(f_\theta(x), y\right)/\partial x\right\|^2 + \xi^2}} A_\epsilon\left(f_\theta, x, y\right)$$

Surrogate Attack Curve

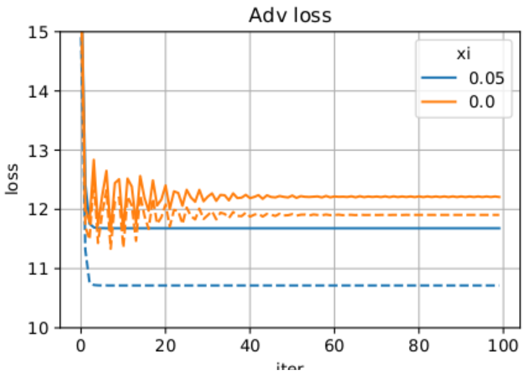


Figure: Adversarial training and testing loss in linear regression. Solid line: adversarial testing loss. Dashed line: adversarial training loss. The training process with $\xi = 0$ is more fluctuate

Convergence

Theorem 2

If there exists some constant B_0 such that $\left\|\theta_\xi^{(0)}\right\| \leq B_0 v$, and the dimension growth rate satisfies $\log n \sqrt{d^2/n} \to 0$, then with probability tending to 1 , the surrogate loss $R_{L,\xi}\left(\theta_\xi^{(t)}, \epsilon\right)$ decreases in each iteration, and

$$\frac{R_{L,\xi}\left(\theta_\xi^{(T)}, \epsilon\right) - R^*}{v^2} \to 0, \text{ and } \frac{\left\|\theta_\xi^{(T)} - \theta^*\right\|}{v} \to 0,$$

given $\eta = \xi / \left(v^2 L\right)$ for some large constant $L, T = \left(v^2 \log\log n\right) / \xi$ and $\xi = v^2 d / \sqrt{n} \log n$

Convergence error

Theorem 2 extension

Adversarial training hurts standard estimation. $\widehat{\theta}_{\text{OLS}}$ as the common least square estimator for un-corrupted data (i.e. without attack) and trivially $\widehat{\theta}_{\text{OLS}} \to \theta_0$. By Theorem 2 , $\theta_\xi^{(T)} \to \theta^* \neq \theta_0$, thus

$$\frac{R_L\left(\theta_\xi^{(T)}, 0\right) - R_L\left(\widehat{\theta}_{\text{OLS}}, 0\right)}{v^2} \to \frac{R_L\left(\theta^*, 0\right) - R_L\left(\theta_0, 0\right)}{v^2} \quad = c(\epsilon) > 0,$$