Certifiable Robustness only formulas

Fabio Brau

March 1, 2023

Scuola Superiore Sant'Anna, Pisa.

Lp norms

$$\|\delta\|_p = \left(\sum_{i=1}^{n} |\delta_i|^p \right)^{\frac{1}{p}}$$

$$\|\delta\|_\infty = \max_i |\delta_i|$$

$$\|\delta\|_0 = \#\{i \,:\, \delta_i \neq 0\}$$

$$\mathcal{L} : \mathbb{R}^C \times \{1, \cdots, C\} \rightarrow \mathbb{R}$$

$$\mathcal{L}(y, c) = -\log\left(\frac{e^{y_c}}{\sum_{i=1}^{C} e^{y_i}}\right)$$

$$\begin{aligned} \max_{\delta \in \mathbb{R}^n} \quad & \mathcal{L}(f(x+\delta), l_{true}) \\ \text{s.t.} \quad & \|\delta\|_p \leq \varepsilon \\ & 0 \leq x + \delta \leq 1 \\ & Q * (x+\delta) \in \{0, \cdots, Q\} \end{aligned}$$

$$\begin{aligned} \min_{\delta \in \mathbb{R}^n} \quad & \|\delta\|_p \\ \text{s.t.} \quad & \mathcal{K}(x+\delta) \neq l_{true} \\ & 0 \leq x + \delta \leq 1 \\ & Q * (x+\delta) \in \{0, \cdots, Q\} \end{aligned}$$

$$\min_{\delta \in \mathbb{R}^n} \quad \mathcal{L}(f(x+\delta), l_{target})$$
$$\text{s.t.} \quad \|\delta\|_p \leq \varepsilon$$
$$0 \leq x+\delta \leq 1$$
$$Q*(x+\delta) \in \{0, \cdots, Q\}$$

$$\min_{\delta \in \mathbb{R}^n} \quad \|\delta\|_p$$
$$\text{s.t.} \quad \mathcal{K}(x+\delta) = l_{target}$$
$$0 \leq x+\delta \leq 1$$
$$Q*(x+\delta) \in \{0, \cdots, Q\}$$

$$\delta^* = \varepsilon \cdot \text{sign}\left(\nabla_x \mathcal{L}(f(x), l_{true})\right)$$

$$x^* = x + \delta^*, \quad \|\delta^*\|_\infty = \varepsilon$$

$$\tilde{\mathcal{L}}(f(x), l) = \alpha \cdot \mathcal{L}(f(x), l) \quad + \quad (1-\alpha) \cdot \mathcal{L}(f(x+\delta^*), l)$$

$$\begin{aligned} \min_{\delta \in \mathbb{R}^n} \quad & c\,\|\delta\|_p + \mathcal{L}(f(x+\delta), l_{target}) \\ \text{s.t.} \quad & 0 \le x + \delta \le 1 \\ & Q * (x+\delta) \in \{0, \cdots, Q\} \end{aligned}$$

$$\delta(c) \quad \mathcal{K}(x + \delta(c)) = l_{target}$$

$$c_{left} = 0,\ c_{right} = 100, \quad c_{test} = \frac{c_{left} + c_{out}}{2}$$

$$c_{left} = c_{left}, c_{right} = c_{test} \quad \text{if} \quad \mathcal{K}(x + \delta(c_{test})) = l$$

$$\delta = \frac{1}{2}\left(\tanh(w) + 1\right) - x$$

$$\mathcal{L}(y, l) = \left[y_l - \max_{j \neq l} y_j\right]^+$$

$$\min_{w \in \mathbb{R}^n} \quad \|\frac{1}{2}\left(\tanh(w) + 1\right) - x\|_p + c \cdot \mathcal{L}(f(\frac{1}{2}(\tanh(w) + 1)), l_{target})$$

$$\min_{w \in \mathbb{R}^n} \quad \| \underbrace{\frac{1}{2}\left(\tanh(w) + 1\right) - x}_{\delta} \|_p + c \cdot \mathcal{L}(f(\underbrace{\frac{1}{2}(\tanh(w) + 1)}_{x+\delta}), l_{target})$$

What is DeepFool Attack?

DeepFool is a white-box attack that aims to deceive deep neural networks by crafting adversarial examples. The goal of the attack is to perturb an input image in such a way that it is misclassified by the network, while the perturbation is imperceptible to humans.

How does it work?

DeepFool works by iteratively finding the direction of the minimum amount of perturbation required to move an input image across the decision boundary of a neural network. The algorithm computes the gradient of the neural network's output with respect to the input image, and then finds the direction of the smallest perturbation that can cause a change in the predicted class.

DeepFool Attack (cont.)

Iterative algorithm

The DeepFool algorithm is iterative, and repeats the following steps until the input image is classified as the target class:

Compute the gradient of the neural network's output with respect to the input image.

Calculate the minimum perturbation required to move the input image across the decision boundary of the network.

Add the computed perturbation to the input image.

Check if the image is now classified as the target class. If yes, stop the algorithm. If no, repeat from step 1.

Effectiveness

DeepFool has been shown to be effective in producing imperceptible adversarial examples that fool state-of-the-art neural networks. The attack can be applied to a wide range of deep learning models, including image classifiers and object detectors.

Defenses against DeepFool Attack

Several defenses have been proposed to mitigate the effectiveness of the DeepFool attack, including:

Adversarial training: Training the neural network on a mix of original and adversarial examples.

Defensive distillation: Training the network to output smoothed probabilities instead of hard probabilities.

Input transformation: Applying a non-linear transformation to the input image before feeding it to the neural network.

Gradient masking: Hiding the gradient information from the attacker by adding random noise to the gradients.

Conclusion The DeepFool attack is a powerful tool for crafting adversarial examples that can deceive deep neural networks. While several defenses have been proposed, the arms race between attacks and defenses continues, highlighting the importance of ongoing research in this area.

$$f(x) = Wx + b, \quad W = \begin{bmatrix} w^{(1)T} \\ \vdots \\ w^{(C)T} \end{bmatrix} \in \mathbb{R}^{C \times n}$$

$$\delta^{(i)} = -\frac{f_i(x)}{\|w^{(i)}\|^2} w^{(i)}$$

$$\rho_{adv}(\mathcal{X}) = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \frac{\|\delta(x)\|_2}{\|x\|_2}$$

$$\tilde{x}^{(t)} = x^{(t-1)} + \alpha \operatorname{sgn}\left(\nabla_x \mathcal{L}(f(x^{(t-1)}), l_{true})\right)$$

$$x^{(t)} = \Pi_{B_p(x,\varepsilon)}\left(\tilde{x}^{(t)}\right)$$

$$\min_{\theta} \rho(\theta), \quad \rho(\theta) = \mathbb{E}_{(x,l)\in\mathcal{X}}\left[\max_{\|\delta\|_p<\varepsilon} \mathcal{L}\left(f_\theta(x+\delta), l; \theta\right)\right]$$

$$\mathcal{L}(y, l; T) = -\log\left(\frac{e^{\frac{y_l}{T}}}{\sum_{i=1}^{C} e^{\frac{y_i}{T}}}\right)$$

Thanks for the attention

Fabio Brau

Scuola Superiore Sant'Anna, Pisa

fabio.brau@santannapisa.it

retis.santannapisa.it/~f.brau

linkedin.com/in/fabio-brau

TELECOMMUNICATIONS, COMPUTER ENGINEERING, AND PHOTONICS INSTITUTE

Sant'Anna School of Advanced Studies – Pisa

ECCELLENZA MIUR 2018-2022

ROBOTICS & AI

Sant'Anna Scuola Universitaria Superiore Pisa

retis Real-Time Systems Laboratory