Towards Efficient Training and Evaluation of Robust Models against l_0 Bounded Adversarial Perturbations

Xuyang Zhong [1] Yixiao Huang [1 2] Chen Liu [1]

[1]City University of Hong Kong [2]University of Michigan

{xuyang.zhong, yixiao.huang}@my.cityu.edu.hk chen.liu@cityu.edu.hk

Formulation

Consider a l_0 bounded adversarial perturbation $\boldsymbol{\delta}$, we can decompose it into a magnitude tensor $\boldsymbol{p}$ and a binary sparsity mask $\boldsymbol{m}$. Therefore, the attacker aims to maximize the following objective function:

$$\max_{\|\boldsymbol{\delta}\|_0 \le k, 0 \le \boldsymbol{x}+\boldsymbol{\delta} \le 1} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}+\boldsymbol{\delta}) = \max_{\boldsymbol{p} \in \mathcal{S}_{\boldsymbol{p}}, \boldsymbol{m} \in \mathcal{S}_{\boldsymbol{m}}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}+\boldsymbol{p} \odot \boldsymbol{m})$$

The feasible sets for $\boldsymbol{p}$ and $\boldsymbol{m}$ are $\mathcal{S}_{\boldsymbol{p}} = \{\boldsymbol{p} \in \mathbb{R}^{h \times w \times c} | 0 \le \boldsymbol{x}+\boldsymbol{p} \le 1\}$ and $\mathcal{S}_{\boldsymbol{m}} = \{\boldsymbol{m} \in \{0,1\}^{h \times w \times 1} | \|\boldsymbol{m}\|_0 \le k\}$, respectively. k is the sparsity level.

Sparse-PGD (sPGD)

Similar to PGD, sPGD iteratively updates $\boldsymbol{p}$ and $\boldsymbol{m}$ until finding a successful adversarial example or reaching the maximum iteration number.

Update Magnitude Tensor $\boldsymbol{p}$: We utilize PGD in the l_∞ case, i.e., use the sign of the gradients, to optimize $\boldsymbol{p}$, with α being the step size:

$$\boldsymbol{p} \longleftarrow \Pi_{\mathcal{S}_{\boldsymbol{p}}}\left(\boldsymbol{p} + \alpha \cdot \mathtt{sign}(\nabla_{\boldsymbol{p}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}+\boldsymbol{p} \odot \boldsymbol{m}))\right)$$

Update Sparsity Mask $\boldsymbol{m}$: Instead of optimizing the discrete $\boldsymbol{m}$, we update its continuous alternative $\widetilde{\boldsymbol{m}} \in \mathbb{R}^{h \times w \times 1}$, and then project $\widetilde{\boldsymbol{m}}$ to the feasible set $\mathcal{S}_{\boldsymbol{m}}$ to obtain $\boldsymbol{m}$:

$$\widetilde{\boldsymbol{m}} \longleftarrow \widetilde{\boldsymbol{m}} + \beta \cdot \nabla_{\widetilde{\boldsymbol{m}}} \mathcal{L} / ||\nabla_{\widetilde{\boldsymbol{m}}} \mathcal{L}||_2$$
$$\boldsymbol{m} \longleftarrow \Pi_{\mathcal{S}_{\boldsymbol{m}}}(\sigma(\widetilde{\boldsymbol{m}}))$$

where β is the step size, $\sigma(\cdot)$ denotes sigmoid function. $\Pi_{\mathcal{S}_{\boldsymbol{m}}}$ is to set the k-largest elements in $\widetilde{\boldsymbol{m}}$ to 1 and the rest to 0.

Backward Function: We propose the ***sparse/projected*** gradient $g_{\boldsymbol{p}}$ and ***unprojected*** gradient $\widetilde{g}_{\boldsymbol{p}}$ of $\boldsymbol{p}$, which exhibit complementary performance:

$$g_{\boldsymbol{p}} = \nabla_{\boldsymbol{\delta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}+\boldsymbol{\delta}) \odot \boldsymbol{m}$$
$$\widetilde{g}_{\boldsymbol{p}} = \nabla_{\boldsymbol{\delta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}+\boldsymbol{\delta}) \odot \sigma(\widetilde{\boldsymbol{m}})$$

Random Reinitialization: When the attack fails and the current sparsity mask $\boldsymbol{m}$ remains unchanged for 3 consecutive iterations, $\widetilde{\boldsymbol{m}}$ will be randomly reinitialized.

Pseudo-code

The pseudo-code of sPGD is presented below:

Algorithm 1 Sparse-PGD

Input: Clean image: $\boldsymbol{x} \in [0,1]^{h \times w \times c}$; Model parameters: $\boldsymbol{\theta}$; Max iteration number: T; Tolerance: t; l_0 budget: k; Step size: α, β; Small constant: $\gamma = 2 \times 10^{-8}$
Random initialize $\boldsymbol{p}$ and $\widetilde{\boldsymbol{m}}$
for $i = 0, 1, ..., T-1$ **do**
  $\boldsymbol{m} = \Pi_{\mathcal{S}_{\boldsymbol{m}}}(\sigma(\widetilde{\boldsymbol{m}}))$
  Calculate the loss $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}+\boldsymbol{p} \odot \boldsymbol{m})$
  if unprojected **then**
    $g_{\boldsymbol{p}} = \nabla_{\boldsymbol{\delta}} \mathcal{L} \odot \sigma(\widetilde{\boldsymbol{m}})$ $\{\boldsymbol{\delta} = \boldsymbol{p} \odot \boldsymbol{m}\}$
  else
    $g_{\boldsymbol{p}} = \nabla_{\boldsymbol{\delta}} \mathcal{L} \odot \boldsymbol{m}$
  end if
  $g_{\widetilde{\boldsymbol{m}}} = \nabla_{\boldsymbol{\delta}} \mathcal{L} \odot \boldsymbol{p} \odot \sigma'(\widetilde{\boldsymbol{m}})$
  $\boldsymbol{p} = \Pi_{\mathcal{S}_{\boldsymbol{p}}}(\boldsymbol{p} + \alpha \cdot \mathtt{sign}(g_{\boldsymbol{p}}))$
  $\boldsymbol{d} = g_{\widetilde{\boldsymbol{m}}} / (||g_{\widetilde{\boldsymbol{m}}}||_2)$ **if** $||g_{\widetilde{\boldsymbol{m}}}||_2 \ge \gamma$ **else** 0
  $\boldsymbol{m}_{old}, \widetilde{\boldsymbol{m}} = \boldsymbol{m}, \widetilde{\boldsymbol{m}} + \beta \cdot \boldsymbol{d}$
  if attack succeeds **then**
    break
  end if
  if $||\Pi_{\mathcal{S}_{\boldsymbol{m}}}(\sigma(\widetilde{\boldsymbol{m}})) - \boldsymbol{m}_{old}||_0 \le 0$ for t consecutive iters **then**
    Random initialize $\widetilde{\boldsymbol{m}}$
  end if
end for
Output: Perturbation: $\boldsymbol{\delta} = \boldsymbol{p} \odot \boldsymbol{m}$

Sparse-AutoAttack (sAA)

For comprehensive robustness evaluation against l_0 bounded perturbations, we propose **sparse-AutoAttack (sAA)**, which is a parameter-free ensemble of both white-box and black-box attacks. We adopt two variants of sPGD with different backward functions: **sPGD**$_{\text{proj}}$ using $g_{\boldsymbol{p}}$, and **sPGD**$_{\text{unproj}}$ using $\widetilde{g}_{\boldsymbol{p}}$. Additionally, **Sparse-RS** is adopted as the black-box attack. These attacks run in a cascade way.

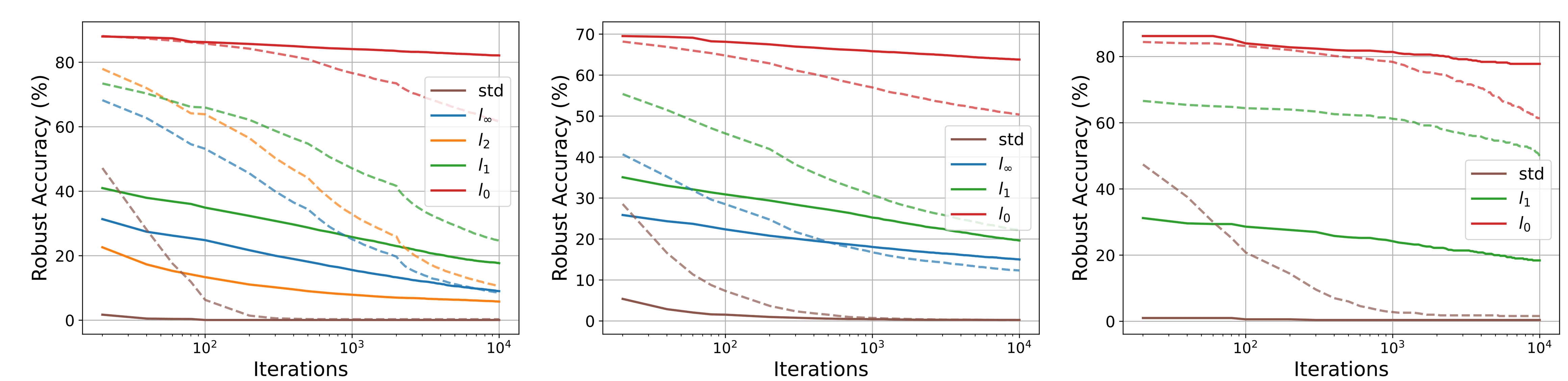
Adversarial Training

To accommodate the scenario of adversarial training, we make the following modifications to sPGD: **a)** random backward function, **b)** multi-k strategy, **c)** higher tolerance for reinitialization. In experiments, we incorporate sPGD in vanilla adversarial training and TRADES and name corresponding methods **sAT** and **sTRADES**, respectively.

Experiments

Table 1. Robust accuracy of various models on different attacks that generate l_0 bounded perturbations, where the sparsity level $k = 20$. The models are trained on **CIFAR-10**.

Model	Network	Clean	Black-Box CS	Black-Box RS	White-Box SF	White-Box PGD$_0$	White-Box SAIF	White-Box **sPGD**$_{\text{proj}}$	White-Box **sPGD**$_{\text{unproj}}$	sAA
Vanilla	RN-18	93.9	1.2	0.0	17.5	0.4	3.2	0.0	0.0	**0.0**
l_∞-adv. trained, $\epsilon = 8/255$										
GD	PRN-18	87.4	26.7	6.1	52.6	25.2	40.4	9.0	15.6	**5.3**
PORT	RN-18	84.6	27.8	8.5	54.5	21.4	42.7	9.1	14.6	**6.7**
l_1-adv. trained, $\epsilon = 12$										
l_1-APGD	PRN-18	80.7	32.3	25.0	65.4	39.8	55.6	17.9	18.8	**16.9**
Fast-EG-l_1	PRN-18	76.2	35.0	24.6	60.8	37.1	50.0	18.1	18.6	**16.8**
l_0-adv. trained, $k = 20$										
PGD$_0$-A	PRN-18	77.5	16.5	2.9	62.8	56.0	47.9	9.9	21.6	**2.4**
PGD$_0$-T	PRN-18	90.0	24.1	4.9	85.1	61.1	67.9	27.3	37.9	**4.5**
sAT	PRN-18	84.5	52.1	36.2	81.2	78.0	76.6	75.9	75.3	**36.2**
sTRADES	PRN-18	89.8	69.9	61.8	88.3	86.1	84.9	84.6	81.7	**61.7**



(a) CIFAR-10, $k = 20$ (b) CIFAR-100, $k = 10$ (c) ImageNet-100, $k = 200$

Figure 1. Comparison between sPGD and RS attack under different iterations. The results of sPGD and RS attack are shown in solid lines and dotted lines, respectively.

Takeaway Messages

1. We propose **sparse-PGD (sPGD)** to generate l_0 bounded adversarial perturbations.
2. We combine sPGD with a black-box attack to construct **sparse-AutoAttack (sAA)** for a more comprehensive robustness evaluation against sparse attacks.
3. sPGD achieves better performance under limited iterations. Models adversarially trained by sPGD have the strongest robustness against sparse attacks.

Codes on Github