Appendix for Improved Adversarial Training via Learned Optimizer

A Algorithm for TRADES

As we have emphasized, our proposed method can be incorporated into any adversarial training which can be formulated as a minimax optimization problem. Here we provide the detailed algorithm of RNN-TRADES in Algorithm 1.

Algorithm 1 RNN-based TRADES

Input: clean data $\{(\boldsymbol{x}, \boldsymbol{y})\}$, batch size B, step sizes α_1 and α_2, number of inner iterations T, classifier parameterized by θ, RNN optimizer parameterized by ϕ
Output: Robust classifer f_θ, learned optimizer m_ϕ
Randomly initialize f_θ and m_ϕ, or initialize them with pre-trained configurations
repeat
  Sample a mini-batch M from clean data.
  for $(\boldsymbol{x}, \boldsymbol{y})$ **in** B **do**
    Initialization: $\boldsymbol{h}_0 \leftarrow 0$, $\mathcal{L}_\theta \leftarrow 0$, $\mathcal{L}_\phi \leftarrow 0$
    Gaussian augmentation: $\boldsymbol{x}'_0 \leftarrow \boldsymbol{x} + 0.001 \cdot \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
    for $t = 0, \ldots, T-1$ **do**
      $\boldsymbol{g}_t \leftarrow \nabla_{\boldsymbol{x}'} \mathcal{L}(f_\theta(\boldsymbol{x}), f_\theta(\boldsymbol{x}'_t))$
      $\boldsymbol{\delta}_t, \boldsymbol{h}_{t+1} \leftarrow m_\phi(\boldsymbol{g}_t, \boldsymbol{h}_t)$, where coordinate-wise update is applied
      $\boldsymbol{x}'_{t+1} \leftarrow \Pi_{\mathbb{B}(\boldsymbol{x}, \epsilon)}(\boldsymbol{x}'_t + \boldsymbol{\delta}_t)$
      $\mathcal{L}_\phi \leftarrow \mathcal{L}_\phi + w_{t+1} \mathcal{L}(f_\theta(\boldsymbol{x}), f_\theta(\boldsymbol{x}'_{t+1}))$, where $w_{t+1} = t + 1$
    end for
    $\mathcal{L}_\theta \leftarrow \mathcal{L}_\theta + \mathcal{L}(f_\theta(\boldsymbol{x}), \boldsymbol{y}) + \mathcal{L}(f_\theta(\boldsymbol{x}), f_\theta(\boldsymbol{x}'_T))/\lambda$
  end for
  Update ϕ by $\phi \leftarrow \phi + \alpha_1 \nabla_\phi \mathcal{L}_\phi / B$
  Update θ by $\theta \leftarrow \theta - \alpha_2 \nabla_\theta \mathcal{L}_\theta / B$
until training converged

B Additional Analysis

B.1 Time comparison

In this section, we compared training time of our proposed methods with the original adversarial training. We still conduct analysis of VGG-16 on CIFAR-10 dataset. From results in Table 1, it can be observed that our methods approximately double the overall training time per epoch, which is not a heavy burden with improved performance taken into account.

Table 1. Time comparison with original adversarial training. Here we report the ratio of our proposed method to its original counterpart, for example, $T_{\text{RNN-Adv}}/T_{\text{AdvTrain}}$. In the parentheses, we report the training time per epoch of our proposed method including RNN-Adv and RNN-TRADES

	Training Time	Ratio of RNN counterpart
AdvTrain	122.24	2.20 (268.50)
TRADES	189.43	2.34 (443.52)

Besides, for the time of different methods generating adversarial examples, we report it in the following Table 2. Here we compute the average time (in second) for generating adversarial examples of one batch containing 100 images for the VGG-16 model. As we can see, our proposed RNN-Adv produces adversarial examples within similar time to PGD-10 while it takes L2LDA much more time.

Table 2. Time for generating adversarial examples of different attack methods

PGD-10	RNN-Adv	L2LDA
0.2163	0.2223	0.3511

B.2 Trajectory

A similar trajectory can be observed in terms of classification accuracy as well when the model is attacked by different attackers. In Figure 1, the robust accuracy under RNN-Adv drops most rapidly and also achieves the lowest point after the entire 10-step attack. It further verifies that our learned optimizer can guide the optimization along a better trajectory for the inner problem, meaning that crafted adversarial examples are much more powerful. This in turn contributes to a more robust model.

C Experiments on Restricted ImageNet

Without loss of generality, we conduct extra experimental analysis on a larger scale dataset, Restricted ImageNet [2]. It is a subset of 9 different super-classes extracted from the entire ImageNet to reduce the computational burden for adversarial training. We adopt the structure of ResNet-18 as the classifier for this dataset. Following the literature [1, 2], the inner attack strength of all defense methods is set to be 0.005 in l_∞ ball while models trained from these mechanisms are evaluated under the attack with the radius of 0.025. Note that in this

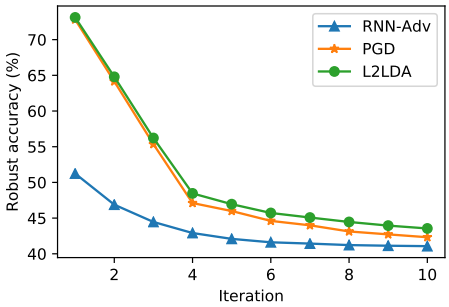

Fig. 1. Accuracy trajectory

experiment we only fine-tune the model starting from the naturally trained one for 2 epochs to observe different performance of defense strategies.

From Table 3, we can clearly see that our proposed methods such as RNN-Adv and RNN-TRADES consistently improve the robust accuracy compared with their original adversarial training algorithms. Specifically, models trained by our methods are always $2\% - 3\%$ percents more robust that others under various attacks.

Table 3. Robust accuracy under white-box attacks (Rest. ImageNet, ResNet18)

Defense \ Attack	Natural	PGD-10	PGD-100	CW100	Min
Plain	**97.66**	0.00	0.00	0.00	0.00
AdvTrain	91.43	8.50	4.88	9.04	4.88
TRADES	72.51	6.96	4.80	10.47	4.80
RNN-Adv	90.28	**10.13**	**6.04**	12.98	**6.04**
RNN-TRADES	73.84	9.76	5.79	**13.22**	5.79

References

1. Sinha, A., Singh, M., Kumari, N., Krishnamurthy, B., Machiraju, H., Balasubramanian, V.N.: Harnessing the vulnerability of latent layers in adversarially trained models. arXiv preprint arXiv:1905.05186 (2019)
2. Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., Madry, A.: Robustness may be at odds with accuracy. arXiv preprint arXiv:1805.12152 (2018)