A. PROOF FOR PROPOSITION 2

Proposition 2. *Consider a ball $\mathcal{B}_\epsilon(\boldsymbol{x})$ at an arbitrary point $\boldsymbol{x} \in \mathcal{X}$. For any $\boldsymbol{x}^* \in \mathcal{B}_\epsilon(\boldsymbol{x})$, we define $\breve{W}^{\boldsymbol{x}^*}$ and $\breve{\boldsymbol{x}}^*$ accordingly, as in equation (4). Let u be a constant such that, for all possible $\breve{W}^{\boldsymbol{x}^*}$, the following is satisfied*

$$\|\breve{W}^{\boldsymbol{x}^*}\|_F \leq u. \tag{9}$$

Assume $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ are arbitrary chosen points in $\mathcal{B}_\epsilon(\boldsymbol{x})$. Then for any

$$\boldsymbol{x}_\eta = \eta \cdot \boldsymbol{x}^1 + (1-\eta) \cdot \boldsymbol{x}^2, \tag{10}$$

we have

$$\boldsymbol{d}\breve{W}^{\boldsymbol{x}_\eta}\breve{\boldsymbol{x}}_\eta \geq \frac{1}{2}(\boldsymbol{d}\breve{W}^{\boldsymbol{x}^1}\breve{\boldsymbol{x}}^1 + \boldsymbol{d}\breve{W}^{\boldsymbol{x}^2}\breve{\boldsymbol{x}}^2) - u \cdot L, \tag{11}$$

where $L = \sqrt{2}(2\|\breve{\boldsymbol{x}}_\eta\| + \frac{1}{2}\|\breve{\boldsymbol{x}}^1 - \breve{\boldsymbol{x}}^2\|)$ is a constant.

Proof. The proof is straightforward. To simplify the notation, we define $s^1 \coloneqq \boldsymbol{d}\breve{W}^{\boldsymbol{x}^1}\breve{\boldsymbol{x}}^1$, $s^2 \coloneqq \boldsymbol{d}\breve{W}^{\boldsymbol{x}^2}\breve{\boldsymbol{x}}^2$ and $s_\eta \coloneqq \boldsymbol{d}\breve{W}^{\boldsymbol{x}_\eta}\breve{\boldsymbol{x}}_\eta$. Also, let l be the lower bound $\boldsymbol{d}\breve{W}^{\boldsymbol{x}^*}\breve{\boldsymbol{x}}^* \geq l$ for any $\boldsymbol{x}^* \in \mathcal{B}_\epsilon(\boldsymbol{x})$. Such l exits because $\mathcal{B}_\epsilon(\boldsymbol{x})$ is a closed and bounded ball and f is continuous. We first note that

$$|s^1 - l| - |s^1 - s_\eta| \leq |s_\eta - l|.$$

Since $s^1 - l > 0$ and $s_\eta - l \geq 0$, we can remove the absolute sign and have

$$s^1 - |s^1 - s_\eta| \leq s_\eta. \tag{16}$$

The term $|s^1 - s_\eta|$ can be upper bounded as

$$\begin{aligned} |s^1 - s_\eta| &= |\boldsymbol{d}\breve{W}^{\boldsymbol{x}^1}\breve{\boldsymbol{x}}^1 - \boldsymbol{d}\breve{W}^{\boldsymbol{x}_\eta}\breve{\boldsymbol{x}}_\eta| \\ &\leq |(\boldsymbol{d}\breve{W}^{\boldsymbol{x}^1} - \boldsymbol{d}\breve{W}^{\boldsymbol{x}_\eta})\breve{\boldsymbol{x}}_\eta| + |\boldsymbol{d}\breve{W}^{\boldsymbol{x}^1}(\breve{\boldsymbol{x}}^1 - \breve{\boldsymbol{x}}_\eta)| \\ &\leq 2u\|\boldsymbol{d}\|_2\|\breve{\boldsymbol{x}}_\eta\|_2 + u(1-\eta)\|\boldsymbol{d}\|_2\|\breve{\boldsymbol{x}}^1 - \breve{\boldsymbol{x}}^2\|_2. \end{aligned} \tag{17}$$

We further denote $m = 2u\|\boldsymbol{d}\|_2\|\breve{\boldsymbol{x}}_\eta\|_2$ and $v = u\|\boldsymbol{d}\|_2\|\breve{\boldsymbol{x}}^1 - \breve{\boldsymbol{x}}^2\|_2$. By replace $|s^1 - s_\eta|$ with its upper bounds equation (17), it follows that

$$s^1 - m - (1-\eta)v \leq s_\eta. \tag{18}$$

Similarly for s^2, we have

$$s^2 - m - \eta v \leq s_\eta. \tag{19}$$

Finally, combining equation (18) and equation (19), we get

$$\frac{1}{2}(s^1 + s^2) - m - \frac{1}{2}v \leq s_\eta, \tag{20}$$

the desired inequality. □

B. EXPERIMENTAL DETAILS

We give more details about training and attacks in the following.

Training We rely on the robust accuracy on a validation dataset to guide the training process. To be more specific, we first randomly select 5000 images from the training dataset to form a validation set before training starts. During training, after each epoch, we run a multi-step PGD attack to evaluate the model's robust accuracy on the validation set. If the validation robust accuracy does not improve for a fixed number of consecutive epochs (we refer to the number as plateau epoch number), we decrease the learning rate by five. If the robust accuracy does not improve for ten consecutive epochs, we terminate the training process.

MNIST We use SGD optimizer with a starting learning rate of 0.01 on MNIST. Batch size is set to be 128. After each epoch, we apply a 20-step PGD attack with step size 0.01 to determine validation robust accuracy. Similarly, when generating strong adversaries for multi-step methods, 20-step PGD

attack with step size 0.01 is used. For numbers reported in Table 1, zero-step JAC is trained with $\alpha_{\text{JAC}} = 0.5$; one-step and multi-step TRADES are trained with $\beta_{\text{TRADES}} = 1$; one-step LEAP is trained with $\alpha = 1e-05$ and $\beta = 0.3$ while multi-step LEAP is trained with $\alpha = 5e-06$ and $\beta = 0.2$.

CIFAR-10 and CIFAR-100 On both CIFAR-10 and CIFAR-100, we employ the SGD optimizer with a starting learning rate of 0.1, a momentum of 0.9 and a weight decay of $2e-4$. A batch size of 64 is used. In terms of computing validation robust accuracy and generating strong adversaries in multi-step cases, we apply 10-step PGD attack with step size 0.07. On the CIFAR-10 dataset, the following parameters are used for the numbers reported in Table 1: zero-step JAC is trained with $\alpha_{\text{JAC}} = 0.5$; one-step TRADES is trained with $\beta_{\text{TRADES}} = 2$; multi-step TRADES is trained with $\beta_{\text{TRADES}} = 4$; one-step LEAP is trained with $\alpha = 0.0002$ and $\beta = 5$ and multi-step LEAP is trained with $\alpha = 0.0001$ and $\beta = 5.0$. On the CIFAR-10 dataset, we used $\alpha_{\text{JAC}} = 0.5$ for zero-step JAC; $\beta_{\text{TRADES}} = 2$ for one-step TRADES; $\beta_{\text{TRADES}} = 4$ for multi-step TRADES; $\alpha = 0.0002$ and $\beta = 5.0$ for one-step LEAP and $\alpha = 0.0001$ and $\beta = 10.0$ for multi-step LEAP.

Attacks Random initialization is applied in all attacks. Untargeted and Multi-targeted attacks are implemented by following the descriptions given in (Qin et al., 2019). For these two attacks, we consider an attack is successful if an adversary is found after a gradient update at any point during the optimization procedure.

B.1. Plateau epoch number and FGSM step-size

Due to the simplicity of the MNIST dataset and similar behaviour on both CIFAR-10 and CIFAR-100, we determine the plateau epoch number and the FGSM step-size by experimenting with one-step ADV on CIFAR-10.

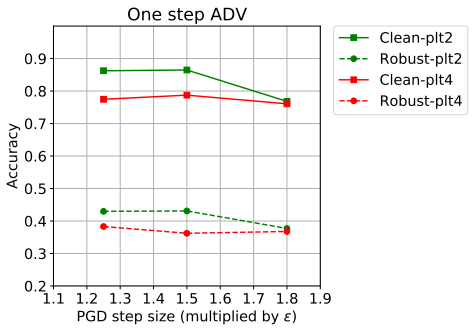


Figure 1: Clean and robust accuracy for one-step ADV at different step sizes

We use strongest Multi-T attack for evaluating model's robust accuracy. In Figure 1, the green lines show the results for plateau epochs to be 2 (plt2) while red lines are for plateau epochs to be 4 (plt4). In terms of step sizes, we tested 3 possible values: 1.25ϵ (the suggested value by Wong et al. (2020)), 1.5ϵ and 1.8ϵ. It is clear to see that with plt2, models perform better in both nominal accuracy and robust accuracy at all three step sizes. We thus use plateau epoch number 2 to adjust the learning rate in all our experiments for comparable results. When step size is considered, both 1.25ϵ and 1.5ϵ give satisfactory results. Since other methods show more stable performances with the step size 1.25ϵ, we use it for generating weak adversaries in our experiments.

C LEAP: Ablation studies

We perform ablation studies for LEAP. We consider the one-step setting. Recall that LEAP consists of two components: a local component and a global component. We test the effect of the local component by setting $\beta = 0$ to get **LEAP-l**:

$$\ell_{\text{LEAP-l}} = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{x} \in \mathcal{X}} \ell^{\text{CE}}(\boldsymbol{x}') + \alpha \cdot \mathcal{A}(J(\boldsymbol{x}')), \quad \boldsymbol{x}' \in \mathcal{B}_\epsilon(\boldsymbol{x}). \tag{21}$$

and the global component by setting $\alpha = 0$ to get **LEAP-g**:

$$\ell_{\text{LEAP-g}} = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{x} \in \mathcal{X}} \ell^{\text{CE}}(\boldsymbol{x}') + \beta \cdot \text{KL}(p(\boldsymbol{x}'; \boldsymbol{\theta}) \| p(\boldsymbol{x}; \boldsymbol{\theta})), \quad \boldsymbol{x}' \in \mathcal{B}_\epsilon(\boldsymbol{x}). \tag{22}$$

We apply the strongest Multi-T attack on CIFAR-10 and Un-T attack on CIFAR-100. Results are summarised in Table 2. When CIFAR-10 is considered, both LEAP-l and LEAP-g alone are effective in improving the model's robust accuracy. Combining the local and global components (one-step LEAP) lead to a further robustness improvement. In terms of time per batch, apart from the cross-entropy loss term at the adversary in both LEAP-l and LEAP-g, LEAP-l requires computing the gradient of a Jacobian approximation term and is computationally more expensive than LEAP-g, which calculates the gradient of a KL penalty term instead. Furthermore, the fact that LEAP-g is computationally cheaper than TRADES is because it uses cross-entropy loss to compute gradients for FGSM while TRADES uses KL distance. Similar performance is observed on CIFAR-100.

Table 2: Robustness performance for CIFAR-10 and CIFAR-100 on Wide-Resnet 28-8. The higher the better.

	Model	Clean accuracy	Un-T attack	Multi-T attack	Time per batch
CIFAR-10	one-step ADV	86.24%	-	42.97%	0.25s
	one-step TRADES	**86.84%**	-	38.14%	0.55s
	one-step LEAP-l	85.24%	-	44.46%	0.60s
	one-step LEAP-g	82.07%	-	44.43%	0.38s
	one-step LEAP	84.52%	-	**46.55%**	0.73s
CIFAR-100	one-step ADV	48.88%	19.04%	-	0.25s
	one-step TRADES	**62.58%**	14.83%	-	0.53s
	one-step LEAP-l	58.46%	24.73%	-	0.60s
	one-step LEAP-g	51.88%	19.71%	-	0.37s
	one-step LEAP	60.98%	**26.28%**	-	0.73s

We show model's performance when trained with LEAP-l and LEAP-g at different parameter values. Results for CIFAR-10 are summarised in Figure 2 and 3 while results for CIFAR-100 are summarised in Figure 4 and 5. For CIFAR-10, we see that the clean accuracy decreases with the increase of α value for LEAP-l but robust accuracy retains at the similar level. In terms LEAP-g, the same trend is observed and when the value of β is large, both clean and robust accuracy decline. On CIFAR-100, there is a big drop in robust accuracy for LEAP-l when α rises while robust accuracy for LEAP-g is relatively insensitive to the change of β value.

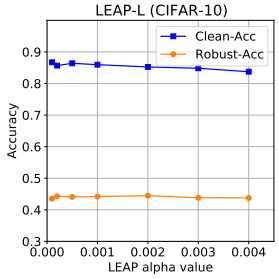


Figure 2: Clean and robust accuracy for LEAP-l at different α on CIFAR-10

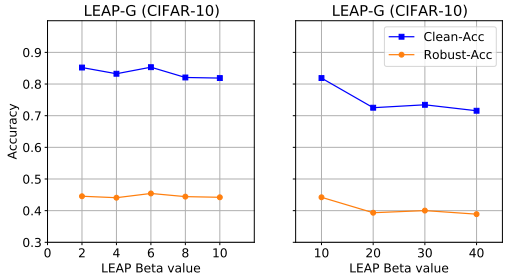


Figure 3: Clean and robust accuracy for LEAP-l at different β on CIFAR-10

C.1 LEAP-G VS. TRADES

Leap-g and TRADES take similar forms. The only differences between these two methods are: firstly, Leap-g computes cross-entropy loss at the adversary while TRADES at the natural image; secondly,

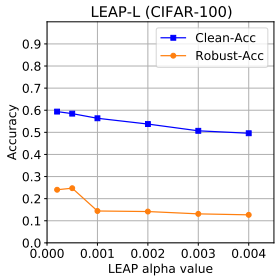


Figure 4: Performance of LEAP-l at different α on CIFAR-100

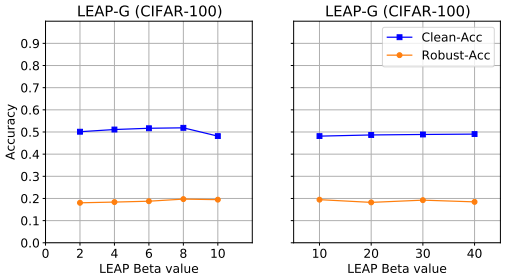


Figure 5: Clean and robust accuracy for LEAP-l at different β on CIFAR-100

LEAP-g uses cross-entropy loss to find a gradient in FGSM while TRADES employs KL distance. On CIFAR-10, it is clear that LEAP-g outperforms TRADES on both clean and robust accuracy for all tested β values. This observation supports the fact that LEAP-g, by focusing on local patches around adversary points, weakens the potential dominating effects of natural images. LEAP-g allows a more effective use of weak adversaries. In terms of CIFAR-100, LEAP-g outperforms TRADES on robust accuracy. However, TRADES achieves higher clean accuracy on small β values. The regularization effect of LEAP-g could be higher than TRADES.

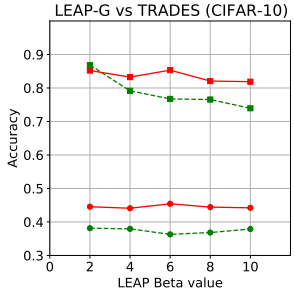


Figure 6: LEAP-g vs. TRADES on CIFAR-10

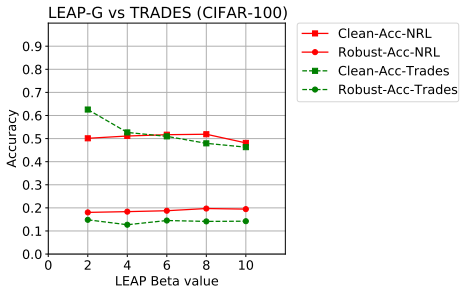


Figure 7: LEAP-g vs. TRADES on CIFAR-100

D Parameter Choices: One-Step Case

We show clean accuracy and robust accuracy for each method at various parameter choices.

D.1 Zero-Step Jac

Although zero-step Jac does not require adversaries, we used 2 epochs for adjusting learning rate via validation robust accuracy to be consistent. Model's performance over a range of α_{Jac} values is shown in Figure 8 for CIFAR-10 and 9 for CIFAR-100. There is a large trade-off between clean and robust accuracy for small α_{Jac} values and then both clean and robust accuracy decrease with the increase of α_{Jac}.

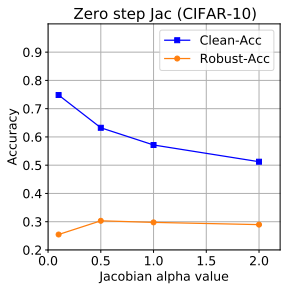


Figure 8: Clean and robust accuracy for zero-step JAC at different α_{Jac} on CIFAR-10

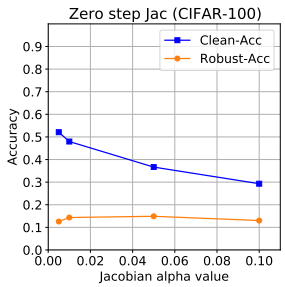


Figure 9: Clean and robust accuracy for zero-step JAC at different α_{Jac} on CIFAR-100

D.2 ONE-STEP TRADES

In Figure 10 (CIFAR-10) and 11 (CIFAR-100), we show one-step TRADES at various β_{TRADES} values. It is easy to see that increasing the value of β_{TRADES} mainly hurts the clean accuracy without improving robust accuracy.

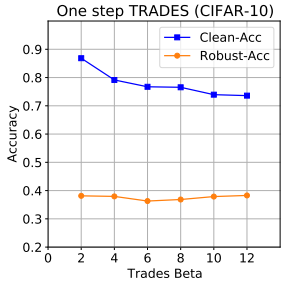


Figure 10: Clean and robust accuracy for one-step TRADES at different β_{TRADES} on CIFAR-10

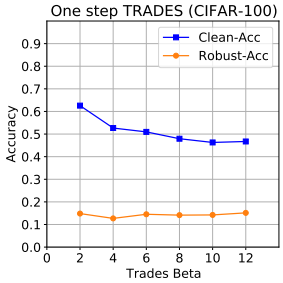


Figure 11: Clean and robust accuracy for one-step TRADES at different β_{TRADES} on CIFAR-100

D.3 ONE-STEP NRL

In Figure 12, we give clean and robust accuracy for models trained at different α and β values on CIFAR-10. In Figure 13, we give clean and robust accuracy for models trained at different α and β values on CIFAR-100. A slight trade-off between clean and robust accuracy can be observed.

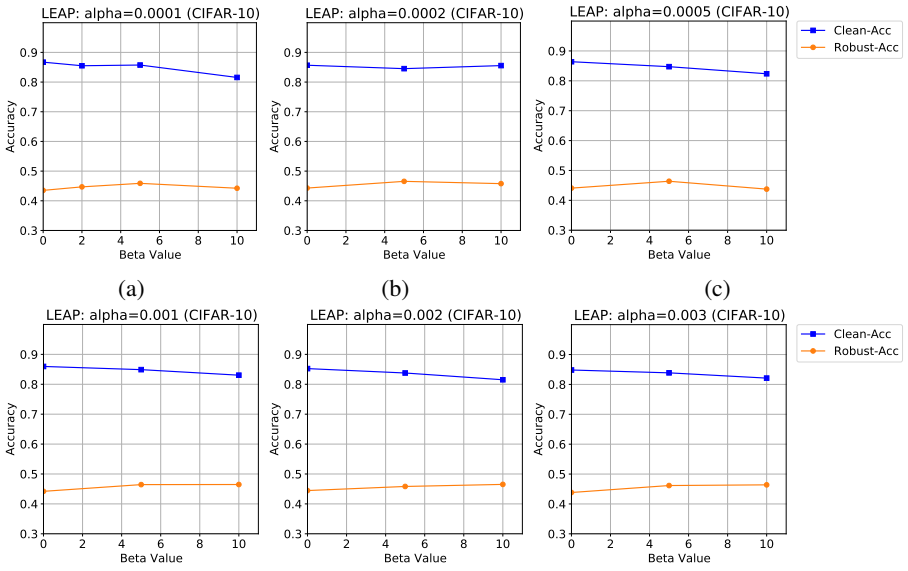
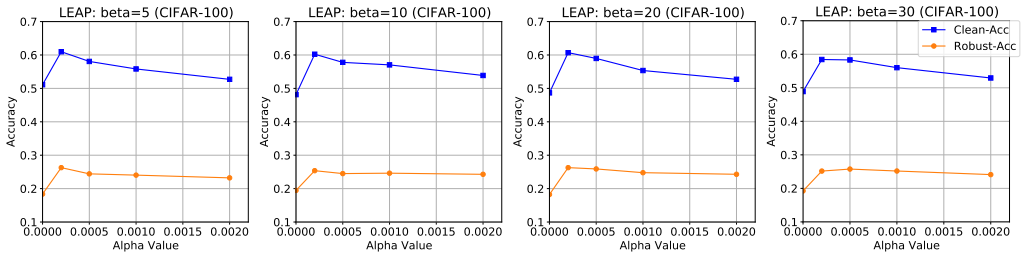


Figure 12: Clean and robust accuracy for one-step NRL at different α, β on CIFAR-10



(a) (b) (c) (d)

Figure 13: Clean and robust accuracy for one-step NRL at different α, β on CIFAR-100