Learn2Perturb: an End-to-end Feature Perturbation Learning to Improve Adversarial Robustness
Supplementary Material

Ahmadreza Jeddi[1], Mohammad Javad Shafiee[1]
Michelle Karg[2], Christian Scharfenberger[2], Alexander Wong[1]
[1]Waterloo AI Institute, University of Waterloo, Waterloo, Ontario, Canada
[2]ADC Automotive Distance Control Systems GmbH, Continental, Germany
[1]{a2jeddi, mjshafiee, a28wong}@uwaterloo.ca
[2]{michelle.karg, christian.scharfenberger}@continental-corporation.com

1. Detailed Analysis

Here we provide a more detailed analysis of the experiments evaluating the proposed method and other tested algorithms with regards to their theoretical background, training, and evaluation.

1.1. Embedding perturbation-injection modules in a network

Generally perturbation-injection modules can be embedded after the activation of each layer. However, for the ResNet baselines we choose to add them just to the output of every block and before the ReLU activation. We do this to reduce the amount of trainable parameters and reduce the training and inference times. Nevertheless, any other setup can be used as well.

1.2. Behaviour of noise distributions in PNI vs Learn2Perturb

As stated in Section 3.2, the trained noise parameters by the PNI approach fluctuate during the training because of the loss function. The min-max optimization applied in that methodology causes the training to enforce noise parameters to be zero as the number of training epoch increases. As such, it is crucial to select the right number of epochs in the training step.

This issue has been addressed in the proposed Learn2Perturb algorithm by introducing a new regularization term in the loss function. As a result, there is a trade-off between training proper perturbation-injection distribution and modeling accuracy during the training step. This trade-off would let the perturbation modules to learn properly and eventually converge to a steady state. To this end, a harmonic series term is introduced in the proposed regularization term which decreases the effect of regularization as the number of training epochs increases, and help the perturbation-injection modules to converge.

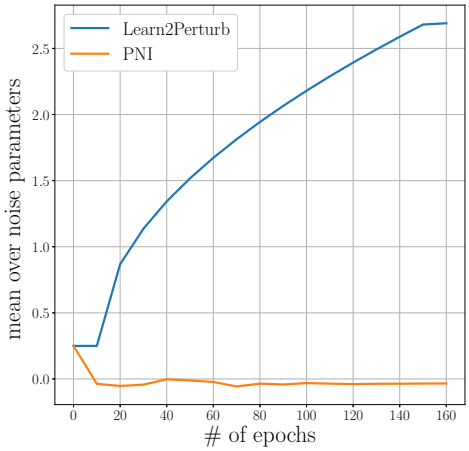

Figure 1. Evolution of mean over noise perturbation parameters through training epochs for ResNet V2. As seen, while the noise distributions are growing the in the Learn2Perturb algorithm, they converge close to zero in the PNI method.

Figure 1 shows the behaviour of noise distributions in both PNI and Learn2Perturb algorithm during the training. As seen, the proposed Learn2Perturb algorithm can handle the noise distributions properly and as a result, the noise distribution parameters are being trained as the number of training epoch increases until they converge to a steady state. However, the noise distributions are forced to zero for the model trained via the PNI algorithm due to the way the loss function is formulated.

1.3. Theoretical Background

It has been illustrated by Pinot *et al.* [6] that randomizing a deep neural network can improve the robustness of the model against adversarial attacks. A deep neural network M is a probabilistic mapping when it maps $\mathcal{X}$ to $\mathcal{Y}$ via $M : \mathcal{X} \rightarrow P(\mathcal{Y})$; to obtain a numerical output of this probabilistic mapping, one needs to sample y according to $M(x)$.

The probabilistic mapping $M(x)$ is $d_{P(Y)}(\alpha, \epsilon, \gamma)$ robust if PC-Risk$_\epsilon(M, \epsilon) \leq \gamma$, where PC-Risk$_\epsilon(M, \epsilon)$ is defined as the minimum value of τ when $d_{P(Y)}\Big(M(x+t), M(x)\Big) > \epsilon$ and $d_{P(Y)}(\cdot)$ is a metric/divergence on $p(\mathcal{Y})$. If $M(x)$ follows an Exponential family distribution, it is possible to define the upper bound for the robustness of the model based on ϵ-perturbation.

1.4. Detailed Experimental Setup

In order to encourage the reproducible experimental results, in this section we provide a detailed explanation of the experimental setup and environment of the reported experiments. Pytorch version 1.2 was used for developing all experiments, and our codes will be open sourced upon the acceptance of this paper.

Following the observation made by Madry *et al.* [4], capacity of networks alone can help increasing the robustness of the models against adversarial attacks. As such, we compare Learn2Perturb and competing state-of-the-art methods for various networks with different capacities.

The ResNet [1] architectures has been selected as the baseline network followed by the state-of-the-art methods and the fast convergence property of this network. The effect of network depth were evaluated by examining the competing methods via ResNet-V1(32), (44), (56) as well as ResNet-V1(20) where (x) shows the depth of the network. Moreover, the effect of network width is examined similar to the work done by Zagoruyko and Komodakis [8]. To increase the width of the network (i.e, experiment performed on ResNet-V1(20)), the number of input and output channels of each layer is increased by a constant multiplier, $\times 1.5$, $\times 2$, and $\times 4$ which widen the ResNet architecture. However we do not follow the exact approach of [8] in which they applied dropout layers in the network; instead we just increase the width of the basic convolution at each layer by increasing the number of input/output channels.

We also consider a ResNet-V2(18), which has a very large capacity compared to ResNet-V1 architecture. Not only the number of channels have increased in this architecture but also it uses 1×1 convolutions to perform the down-sampling at each residual blocks.

The proposed Learn2Perturb, No defence, and Vanilla methods, used the same setup for gradient descent optimizer. SGD optimizer with momentum of 0.9 with Nesterov momentum and weight decay of $1e^{-4}$ is used for training of those methods. The noise injection parameters have weight decay equal to 0. We use the batch size of 128, and 350 epochs to train the model. The initial learning rate is 0.1, then changes to 0.01 and 0.001 at epochs 150 and 250, respectively.

For the parameter γ in equation 7 in the main manuscript, we choose value 10^{-4} for all of our experiments. In equation 8, we have τ which as we state is the output of a harmonic series given the epoch number. we formulate τ as below:

$$\tau(t) = \sum_{i=s}^{t} \frac{1}{i-s-1} \tag{1}$$

where t shows the current epoch, while s shows the first epoch number from which noise is being added to the network.

For training models with PNI, the same parameters reported by authors [2] are used.

The PGD adversarial training utilized alongside with the alternative back-propagation technique in the proposed method which can be formulated as:

$$\arg\min_{W,\theta} \Big[\arg\max_{\delta \in l_\infty - \epsilon} \mathcal{L}\Big(P(X+\delta; W, \theta), T\Big)\Big] \tag{2}$$

where W encodes the network parameters and θ shows the perturbation-injection parameters. In this formulation only adversarially generated samples are used in the training step for the outer minimization, following the original work introduced in [4].

Finally, in order to balance between the adversarial robustness and clean data accuracy [2, 9], we formulate the adversarial training as follow:

$$\begin{aligned} \arg\min_{W,\theta} \Big[\alpha \cdot \mathcal{L}\Big(P(X; W, \theta), T\Big) + \\ \beta \cdot \arg\max_{\delta \in l_\infty - \epsilon} \mathcal{L}\Big(P(X+\delta; W, \theta), T\Big)\Big] \end{aligned} \tag{3}$$

where the first term shows the loss associated to the clean data and α is the weight for the clean data loss term, while the second shows the loss associated with the adversarially generated data with weight β. The models trained with the proposed Learn2Perturb algorithm use $\alpha = \beta = 0.5$. (3) helps gain adversarial robustness, while maintaining a reasonably high clean data accuracy.

1.5. Black-Box Attacks

In this section, the robustness of the proposed method and the competing algorithms against black-box attacks are evaluated. Two different attacks including few-pixel attack [7] and transferability attack [5] are used to evaluate the competing methods.

Table 1. Few-pixel attack; the competing methods are evaluated via few-pixel [7] attack base on two network architectures of ResNet-V1(20) and ResNet-V2(18). {1,2,3} pixels are changed in the test samples to perturbed the images.

Network Architecture	Attack Strength	No defence	Vanilla	PNI	Adv-BNN	Learn2Perturb
ResNet-V1(20)	**1-pixel**	21.45	65.20	67.40	58.40	**70.15**
	2-pixel	2.55	48.35	61.75	56.20	**63.90**
	3-pixel	1.10	36.40	58.10	55.70	**61.85**
ResNet-V2(18)	**1-pixel**	23.44	56.10	50.90	**68.60**	64.45
	2-pixel	3.20	33.20	39.00	**64.55**	60.05
	3-pixel	0.95	23.95	35.40	**59.70**	53.90

Few-pixel attack (here in the range of one to three pixels) utilizes differential evolution technique to fool deep neural networks under the extreme limitation of only altering at most few pixels. We use population size of 400 and maximum iteration steps of 75 for the differential evolution algorithm. The attack strength is controlled by the number of pixels that are allowed to be modified. In this comparison we consider the {1,2,3}-pixel attacks.

Table 1 shows the comparison results of the competing methods against few-pixel attack. Two different network architectures (ResNet-V1(20) and ResNet-v2(18)) are used to evaluate the competing algorithms. As seen, the proposed Learn2Perturb method outperforms other state-of-the-art methods when the baseline network architecture is ResNet-V1(20). However, Adv-BNN provides better performance when the baseline network architectures is ResNet-V2(18), while the proposed Learn2Perturb algorithm provides comparable performance for this baseline.

Table 2 demonstrates the comparison results for the proposed Learn2Perturb and state-of-the-art methods based on Transferability attack. Results again show that the proposed Learn2Perturb method provides robust prediction against this attack as well.

1.6. CIFAR-100

A more detailed analysis of the experimental setup and results for the CIFAR-100 dataset is provided as follows. The CIFAR-100 dataset [3] is very similar to CIFAR-10 dataset, however the image samples are categorized to 100 fine class labels. All the models involving PGD adversarial training are trained with $\epsilon = \frac{8}{255}$ during training. Figures 2 and 3 demonstrate the performance comparison of the proposed Learn2Perturb with other state-of-the-art methods on CIFAR-100 dataset based on FGSM and PGD attacks.

As seen, the proposed Learn2Perturb method outperforms other competing algorithms for ϵs up to $\frac{8}{255}$, however for bigger ϵs it provides comparable performance with Adv-BNN, which has the best result.

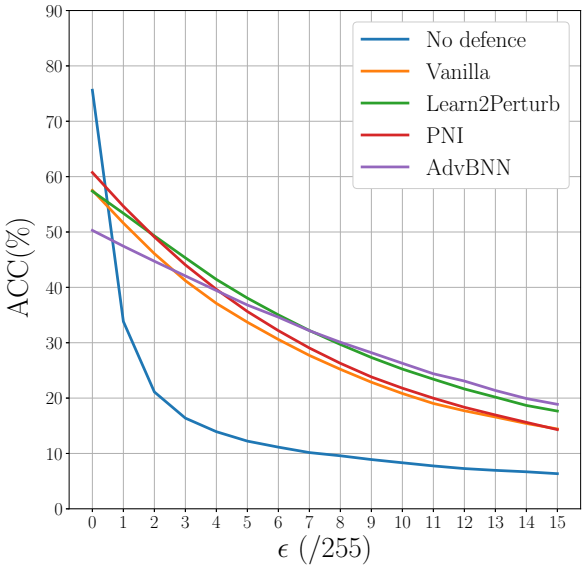

Figure 2. FGSM attack on CIFAR-100 with different epsilons for the l_∞ ball on ResNet-V2(18).

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–597, 2019.

[3] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[4] 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.

[5] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[6] Rafael Pinot, Laurent Meunier, Alexandre Araujo, Hisashi Kashima, Florian Yger, Cédric Gouy-Pailler, and Jamal Atif. Theoretical evidence for adversarial robustness through ran-

Table 2. Transferability attack comparison. The competing methods are attacked within the context of Transferability where the perturbed images utilized to evaluate the robustness of the model are generated by one another method. The 'Source Model' is the model which the perturbed samples are generated from to attack each competing algorithm.

		Vanilla		PNI		Adv-BNN		Learn2Perturb	
Network Architecture	**Source Model**	**FGSM**	**PGD**	**FGSM**	**PGD**	**FGSM**	**PGD**	**FGSM**	**PGD**
Resnet20 - V1	**Vanilla**	–	–	60.32±0.05	58.27±0.01	49.22±0.90	48.63±3.10	58.86±0.03	56.75 ± 0.01
	PNI	66.31±0.02	63.04±0.01	–	–	51.12±1.22	49.59±0.83	63.26±0.10	59.31 ± 0.06
	Adv-BNN	74.38±0.16	72.02±0.02	73.05±0.12	70.26±0.05	–	–	72.48±0.05	69.25 ± 0.06
	Learn2Perturb	70.66±0.01	67.32±0.01	68.46±0.03	64.77±0.01	54.16±2.36	52.23±1.52	–	–
Resnet18 - V2	**Vanilla**	–	–	69.52±0.02	68.01±0.02	67.20±0.04	65.88±0.03	67.32±0.04	65.58 ± 0.04
	PNI	69.56±0.01	67.09±0.02	–	–	67.63±0.03	64.87±0.04	67.63±0.05	64.61 ± 0.01
	Adv-BNN	73.66±0.01	71.23±0.01	74.02±0.03	71.51±0.02	–	–	71.67±0.09	68.75 ± 0.04
	Learn2Perturb	73.49±0.01	70.44±0.00	73.89±0.04	70.61±0.01	70.22±0.04	67.33±0.04	–	–

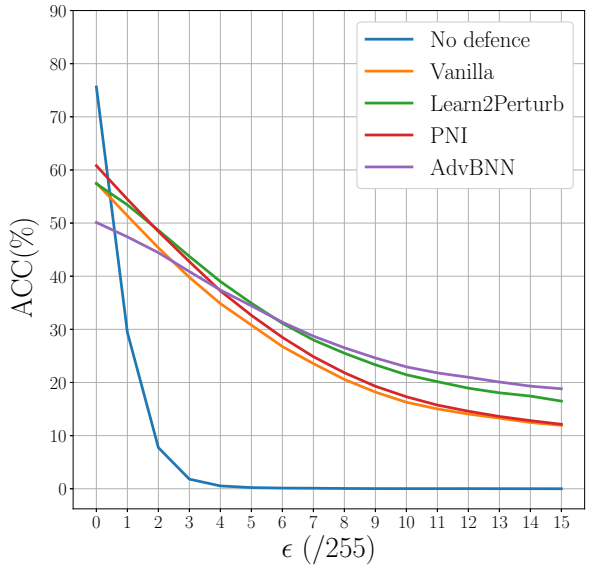

Figure 3. PGD attack on CIFAR-100 with different epsilons for the l_∞ ball on ResNet-V2(18).

domization: the case of the exponential family. *arXiv preprint arXiv:1902.01148*, 2019.

[7] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019.

[8] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[9] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.