# Adversarial training with informed data selection

Marcele O. K. Mendonça*, Javier Maroto⋆, Pascal Frossard⋆ and Paulo S. R. Diniz*
* *SMT - Signals, Multimedia, and Telecommunications Lab.*
Universidade Federal do Rio de Janeiro, DEL/Poli & PEE/COPPE/UFRJ
P.O. Box 68504, Rio de Janeiro, RJ, 21941-972, Brazil,
⋆ École Polytechnique Fédérale de Lausanne (EPFL), Switzerland.
emails: {marcele.kuhfuss,diniz}@smt.ufrj.br, {javier.marotomorales,pascal.frossard}@epfl.ch

***Abstract*—With the increasing amount of available data and advances in computing capabilities, deep neural networks (DNNs) have been successfully employed to solve challenging tasks in various areas, including healthcare, climate, and finance. Nevertheless, state-of-the-art DNNs are susceptible to quasi-imperceptible perturbed versions of the original images – adversarial examples. These perturbations of the network input can lead to disastrous implications in critical areas where wrong decisions can directly affect human lives. Adversarial training is the most efficient solution to defend the network against these malicious attacks. However, adversarial trained networks generally come with lower clean accuracy and higher computational complexity. This work proposes a data selection (DS) strategy to be applied in the mini-batch training. Based on the cross-entropy loss, the most relevant samples in the batch are selected to update the model parameters in the backpropagation. The simulation results show that a good compromise can be obtained regarding robustness and standard accuracy, whereas the computational complexity of the backpropagation pass is reduced.**

***Index Terms*—data-selection, sampling strategy, adversarial training, robustness-accuracy tradeoff**

## I. Introduction

Over the past decade, the amount of available digital data has exponentially increased. Thanks to the advances in computing capabilities, deep neural networks (DNNs) have been successfully employed to solve challenging image and natural language processing tasks. However, state-of-the-art DNNs are known to be highly vulnerable to adversarial examples [1], [2]. These small but malicious perturbations of the network input can manipulate the trained model to produce incorrect predictions with high confidence, and some perturbations can even fool different network models [3]. Since adversarial attacks might lead to disastrous implications in critical areas like healthcare [4], climate [5] and finance [6], defending against them is critical.

So far, adversarial training is the most effective approach to mitigate the effect of strong attacks like the Projected Gradient Descent (PGD) attack [7], DeepFool [8], and AutoAttack [9]. Training the DNN with perturbed versions of the original samples makes it possible to improve the accuracy on unseen adversarial examples, also known as *robustness accuracy* [10]. However, generating adversarial examples during training can be highly computationally intense since each sample is usually built with several steps in the direction of the gradient as the model is trained. Moreover, adversarial training generally decreases the *standard accuracy*, that is, the accuracy on clean samples [11]. This *robustness-accuracy* tradeoff is reported to be highly data-dependent, especially regarding the data distribution [12] and its quality [13]. Furthermore, we only have access to a training dataset which is not necessarily representative for the problem we aim to learn. In this case, we could avoid using the entire training data. Since the dataset is reduced, we can save several computations during backpropagation and speed-up training. This hypothesis was already investigated for standard training in [14], [15]. In this work, we extend the work in [14], [15] and apply it to the adversarial training case. From each mini-batch composed of both clean and adversarial samples, the proposed data selection algorithm selects the most relevant samples based on the cross-entropy loss. Since only the selected samples are used to update the model parameters in the backpropagation, the training time is reduced. The selection also balances the necessary amount of clean and adversarial samples required to yield satisfactory robustness and standard accuracy.

The paper is organized as follows. Section II presents a brief overview of the adversarial training method and some notations. In section III, we propose a data selection technique for adversarial training. The proposed approach is tested via simulation results in section IV. Finally, section V includes some conclusion remarks.

## II. Adversarial Training

Adversarial training continually creates and incorporates adversarial examples into the training process of a deep neural network classifier

$$f_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^N \rightarrow \{1 \cdots C\}, \tag{1}$$

with $\boldsymbol{\theta}$ weights, which maps an input image $\mathbf{x}$ to a label $y$ from a dataset

$$\mathcal{D} = \{(\mathbf{x}(1), y(1)), (\mathbf{x}(2), y(2)), \cdots, (\mathbf{x}(M), y(M))\}, \tag{2}$$

with $C$ possible classes. Adversarial training attempts to solve the min-max optimization problem

$$\begin{aligned} \min_{\boldsymbol{\theta}} \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x},y \in \mathcal{D}} & \max_{\boldsymbol{\eta}} \ \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}+\boldsymbol{\eta}), y) \\ & \text{s.t } ||\boldsymbol{\eta}||_p \leq \epsilon, \end{aligned} \tag{3}$$

where $\mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}+\boldsymbol{\eta}), y)$ is the loss function on the adversarial sample and $\boldsymbol{\eta}$ is a small perturbation constrained by $\epsilon$.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. This work was also supported by the Swiss Government Excellence Scholarships for Foreign Students.

Creating adversarial samples involves solving the inner maximization problem in equation (3), in which the loss function $\mathcal{L}$ is maximized in an effort to change the prediction, that is, $f_{\boldsymbol{\theta}}(\mathbf{x}+\boldsymbol{\eta}) \neq f_{\boldsymbol{\theta}}(\mathbf{x})$. The optimization constraints ensure that the distance between the adversarial and original example should be less than $\epsilon$ under a particular norm, $||\boldsymbol{\eta}||_p \leq \epsilon$. The norms aim to quantify how imperceptible to humans an adversarial example is. Some examples of norms are the $l_0$ norm, $l_2$ norm, and $l_\infty$. We then briefly review the most popular methods to create adversarial examples.

Introduced by [2], the Fast Gradient Sign Method (FGSM) attack generates adversarial examples by modifying the input towards the direction where the loss $\mathcal{L}$ increases

$$\mathbf{x}' = \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\boldsymbol{\theta}, \mathbf{x}, y)), \tag{4}$$

with sign$(\cdot)$ the sign function, and $\nabla_{\mathbf{x}} \mathcal{L}(\boldsymbol{\theta}, \mathbf{x}, y)$ the loss gradient with respect to $\mathbf{x}$. One of the strongest $l_\infty$-bounded attacks, the PGD attack [7] tries to solve the inner maximization problem in equation (3) following an iterative procedure. At each step $i$, the adversarial example is updated as

$$\mathbf{x}'_i = \text{clip}_{\mathbf{x}+\epsilon}(\mathbf{x}_{i-1} + \alpha \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\boldsymbol{\theta}, \mathbf{x}, y))), \tag{5}$$

in which function $\text{clip}_{\mathbf{x}+\epsilon}(\cdot)$ clips the input at the positions around the predefined perturbation range. In the context of $l_2$-bounded attacks, Deepfool [8] is an iterative attack optimized for the $l_2$-norm based on a linear approximation of the classifier. Using geometry concepts, DeepFool searches within the region of the space that describes the output of the classifier (polyhedron) for the minimal perturbation that can change the classifiers decision. Among black-box attacks, one pixel attack [16] is a $l_0$-bounded attack that employs differential evolution to create adversarial examples without knowing the network gradients and its parameters. Finally, the AutoAttack [9] method consists of an ensemble of four attacks: two versions of the PGD attack, the targeted version of the Fast Adaptive Boundary (FAB) attack [17] and the black-box Square Attack [18]. Currently, AutoAttack and PGD attack are the most popular methods to test adversarial robustness. Since the PGD attack is less computationally intense than AutoAttack, we consider the PGD attack in this work. However, other attacks can be used with the proposed data selection.

With the inner maximization problem addressed, the outer minimization problem in equation (3) is then solved to find the model parameters that minimize the loss on the generated adversarial examples. The original dataset $\mathcal{D}$ is split into small batches $\mathcal{B}$ and stochastic gradient descent (SGD) is employed to update the model parameters

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \mu \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x},y \in \mathcal{B}} \nabla_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x} + \boldsymbol{\eta}^*), y), \tag{6}$$

where the gradient is evaluated at the maximum point $\boldsymbol{\eta}^*$ found in the inner maximization problem, thanks to the Danskin's theorem [19].

## III. Proposed Data Selection for Adversarial Training

When performing adversarial training, we are interested in learning a process or function $f(\cdot)$ that maps a data space $\mathcal{X}$ into an output space $\mathcal{Y}$. However, we do not have direct access to samples from $\mathcal{X}$ in order to train the model according to the adversarial objective. We only have access to a subset $\mathcal{D}$ which is split into batches used to update the model parameters in equation (6). However, there is no guarantee that this available subset or its batches consist of a good representation of the process $f(\cdot)$. In this regard, we propose a sampling strategy to select the most relevant samples to compose the batches in adversarial training.

We first consider the entire original dataset $\mathcal{D}$ of input-output pairs in equation (2). Then, at each mini-batch iteration, $b'$ clean samples are selected from the whole dataset to form the batch set $\mathcal{B}'$. By using PGD, $b'$ adversarial examples are generated from the samples in the set $\mathcal{B}'$ using equation (5). The resulting mini-batch $\mathcal{B}$ is then composed of $b = 2b'$ samples. The samples in the mini-batch flow through the network, the gradients are computed, and we obtain the network output as a one-hot-encoded vector $\mathbf{y}$, as shown in Figure 1. In order to quantify the relevance of the samples in the mini-batch, we define the error signal

$$E(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{c=1}^{C} e(\hat{y}_c, y_c), \tag{7}$$

which is based on the cross-entropy loss

$$e(\hat{y}_c, y_c) = \log \left( \sum_{c=1}^{C} \exp(\hat{y}_c) \right) - y_c, \tag{8}$$

where $C$ is the number of classes.

As a rule, the closer to zero the error signal is, the less informative or relevant will be the contribution of the correspondent data pair to the parameter update in equation (6). We then propose to select a portion $P_{\text{up}}$ of the samples in $\mathcal{B}$ based on the higher error values in equation (7), forming a selection set $\mathcal{S}$. After the forward propagation is completed, only the samples in $\mathcal{S}$ are used in the backpropagation to update the network parameters $\boldsymbol{\theta}$, as depicted in Figure 2. Since only a portion $P_{\text{up}}$ of the samples are used to update the parameters, we can save some computations and we alleviate the training burden.

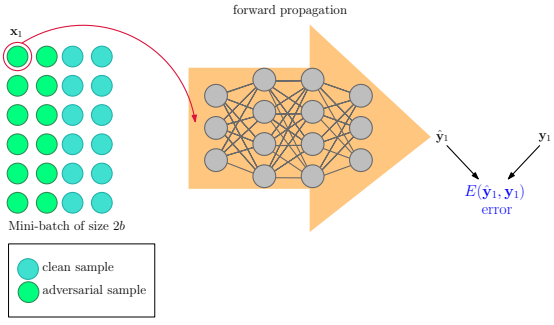


Fig. 1: Forward propagation and error signal computation.

One question remains about how to choose an adequate $P_{\text{up}}$ for our problem. As $P_{\text{up}} \to 0$, fewer samples are selected and we save more computations in the backpropagation. In this case, however, the selected samples might be insufficient lo

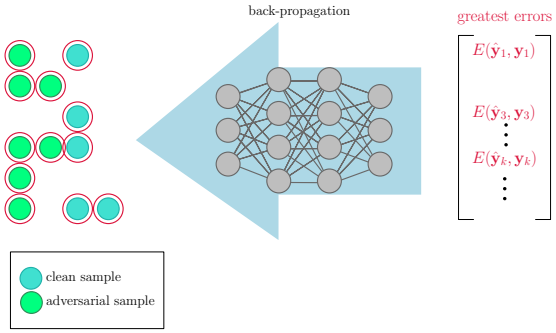


Fig. 2: Selected samples being used in the backpropagation.

learn the problem. For standard training, the most favorable $P_{\text{up}}$ choice mainly depends on the dataset complexity [14]. Simpler datasets like MINIST requires $P_{\text{up}} = 0.3$, whereas for more complex datasets as CIFAR10, $P_{\text{up}} = 0.5$ is a better choice. Thus, one option is to set a fixed $P_{\text{up}}$ for the whole training process. In this way, we can set the amount of saved computations from the beginning. Nevertheless, in cases where the dataset complexity is unknown and it is difficult to prescribe a $P_{\text{up}}$ for all the epochs, an automatic $P_{\text{up}}$ can be advantageous. In this way, we can obtain the $P_{\text{up}}$ for each epoch in an adaptive manner as the training is performed. This can be achieved by considering the accuracy at each epoch as a criterion. Hence, we can estimate the number of selected samples $P_{\text{up}}$ at each epoch $t$.

$$P_{\text{up}}^{(t)} = (1 - \lambda_{acc}^{(t-1)}) P_{\text{up}}^{(t-1)} \tag{9}$$

where $P_{\text{up}}^{(0)} = 1$ and $\lambda_{acc}^{(t-1)}$ is the last available accuracy. We need more samples in the mini-batch to improve learning when the accuracy is low, whereas fewer samples are required to continue the learning process when the accuracy increases.

As it will be shown in the simulations, updating the $P_{\text{up}}$ using equation (9) accelerates the convergence for $P_{\text{up}}^{(0)} = 1$ because, in this case, it selects more samples in the first epochs. Our motivation was to provide more samples to the model at the beginning to improve and accelerate its learning. Therefore, early stopping methods [20] can be employed to further reduce the training time. Since we do not consider the early stopping approach in the simulations, we propose using a fixed prescribed $P_{\text{up}}$ in this work. The main proposed algorithm is detailed in Algorithm 1.

**Algorithm 1** Proposed Data Selection for adversarial training

1: Given dataset $\mathcal{D}$, mini-batch size $b'$, and prescribed $P_{\text{up}}$
2: **for** epoch = $1 \cdots T$ **do**
3:  **for** mini-batch $\mathcal{B} \subset \mathcal{D}$ **do**
4:  Create adversarial examples $\{x'_1, \cdots x'_{b'}\}$ from clean samples $\{x_1, \cdots x_{b'}\}$ using current state of the network and obtain $\mathcal{B}' = \{x'_1, \cdots x'_{b'}, x_1, \cdots x_{b'}\}$;
5:  Forward propagation with samples in $\mathcal{B}'$;
6:  Compute the error signal for each sample in $\mathcal{B}'$ using equation (7);
7:  Select the $P_{\text{up}} \times 100\%$ of the samples in $\mathcal{B}'$ with greatest error values;
8:  Update model parameters by back propagation using only the data samples in $\mathcal{S}$;

## IV. Simulation Results

In this section, we assess the performance of the proposed data selection method in the CIFAR10 dataset using the Resnet18 model. The PGD attack with $\epsilon = 8/255$, $\alpha = 0.01$ and 20 iterations is employed to build the adversarial examples. We consider the following methods in the simulations. The standard method trains only with clean samples with a mini-batch $\mathcal{B}$ of size $b = 256$. Also using $\mathcal{B}$ with $b = 256$, the robust method is trained only with adversarial examples. The DS robust method is trained with the selection set $\mathcal{S}$ of size $b = 256$, which is composed of both clean and adversarial samples, and it is obtained using our selection strategy with $P_{\text{up}}$ fixed or varying. The random robust method is trained with a mini-batch of size $b = 256$, composed of clean and adversarial samples selected at random. We also consider the selection method proposed in [13] in which the samples are selected based on their learning stability. In this case, we used $50\%$ of the samples with high quality in order to perform a fair comparison in terms of number of samples used.

First, we vary the portion of selected samples $P_{\text{up}}$ in Figure 3 to investigate the impact on the standard and robustness accuracy at the last epoch. By using $P_{\text{up}} = 0.5$, we slightly outperform the approach that consider all the samples ($P_{\text{up}} = 1$) in terms of standard accuracy, with the benefit of requiring only $50\%$ of the samples in the batch. In terms of robustness, the methods with $0.5 \leq P_{\text{up}} < 1$ perform quite close to the method with $P_{\text{up}} = 1$. If we reduce $P_{\text{up}}$ even further, we do not observe a gain in performance. In such case, the model would require more epochs to achieve the same performance or it would need more samples to learn the problem.

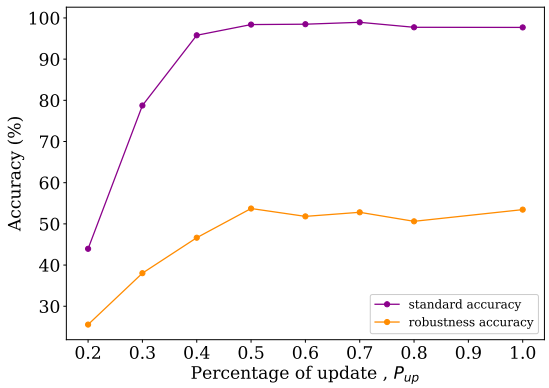


Fig. 3: Evolution of the standard and robustness accuracy as the portion of selected samples $P_{\text{up}}$ is varied.

We then evaluate the proposed DS robust method with varying $P_{\text{up}}$ and compare it with the fixed $P_{\text{up}} = 0.5$, the standard and robust methods in terms of standard and robust accuracy in Figures 4 and 5. We show in Figure 6 the obtained $P_{\text{up}}$ for each epoch following equation (9). By using both a varying $P_{\text{up}}$ and $P_{\text{up}} = 0.5$, we observe an

improvement in terms of standard accuracy when compared with the standard and robust methods. Moreover, reducing the number of samples in the mini-batch does not affect the robust accuracy, as shown in Figure 5.

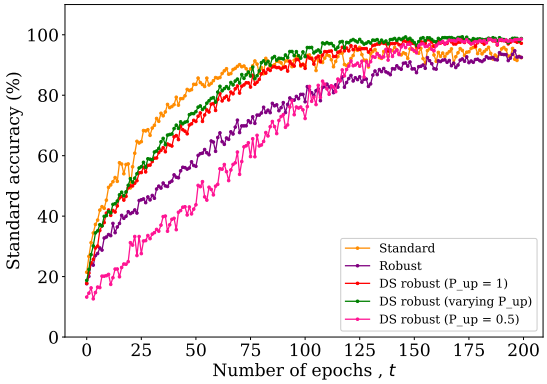


Fig. 4: Standard accuracy as a function the number of epochs.

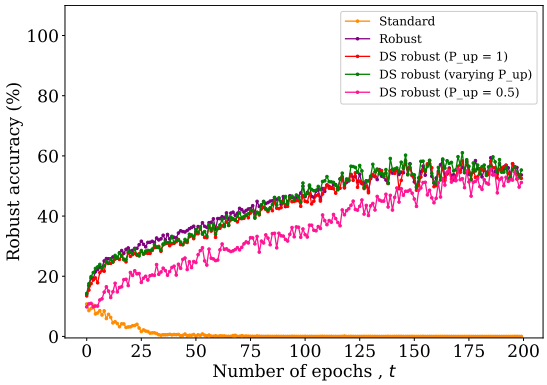


Fig. 5: Robust accuracy as a function the number of epochs.

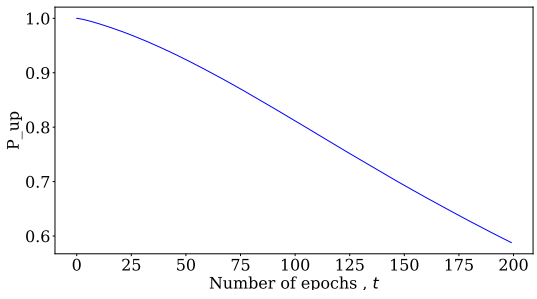


Fig. 6: Portion of selected samples $P_{\text{up}}$ obtained for each epoch following equation (9).

This improvement in robustness-accuracy tradeoff is reasonable since our method includes the most potential relevant clean and adversarial samples in the mini-batch. Some claim that such a tradeoff exists because the standard and robust objectives conflict [21], [22]. We can then observe in Figure 7 that the model trained with $P_{\text{up}} = 0.5$ starts by selecting more adversarial samples than clean samples. However, after a few epochs, this behavior changes, and the number of selected clean samples increases. This feature potentially suggests that the model tries to learn the adversarial problem first. When it is done, the DS method attempts to improve the clean accuracy. Moreover, the number of selected minimum adversarial examples increases as the model is trained, as depicted in Figure 8. The minimum adversarial examples are generated by slowly increasing the perturbation constraint $\epsilon$ until the prediction changes.

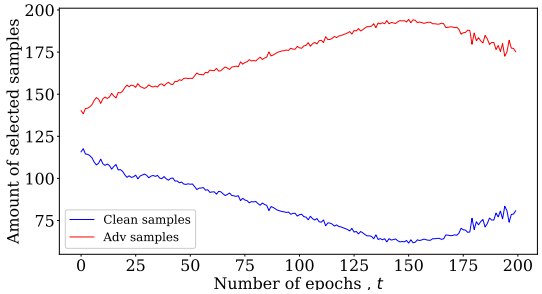


Fig. 7: Averaged amount of selected clean and adversarial samples at each epoch for $P_{\text{up}} = 0.5$.

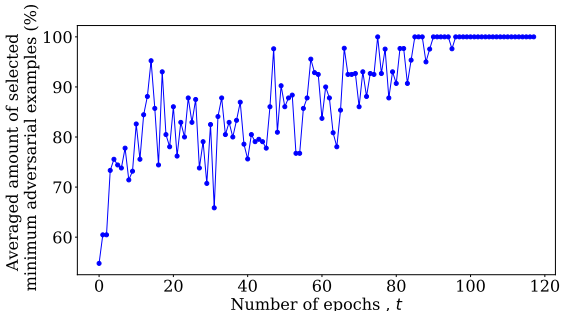


Fig. 8: Averaged amount of selected minimum adversarial examples at each epoch for $P_{\text{up}} = 0.5$.

Finally, our methods are compared with other selection methods in terms of standard and robust accuracy in Figures 9 and 10, respectively. The DS approach outperforms both the random method and the selection method with $50\%$ of high quality samples from [13], especially in terms of standard accuracy.

The benefits of the proposed methods in terms of performance are followed by a reduction in computational complexity. Since only $P_{\text{up}}$ samples in the mini-batch are backpropagated through the network to update its parameters; we can save some computations. For example, we present the total training time after 200 epochs in Table I. The simulations were performed in a computer with two GTX-1080 GPUs. With $P_{\text{up}} = 0.5$, the training time is reduced when compared with $P_{\text{up}} = 1$ and varying $P_{\text{up}}$. However, if we stop the training by the 150th epoch, the training time for the varying $P_{\text{up}}$ can be reduced to $15261.29$s. Therefore, the varying $P_{\text{up}}$ strategy can be applied if an early stopping method is also employed. We also outperform the method introduced in [13] in terms of total training time as their method needs a pre-training to rank the samples by the learning stability values.

TABLE I: Total training time after 200 epochs.

| Method | Time (s) |
|---|---|
| Selection approach from [13] with 50% of samples removed | 39970.71 |
| Robust with $P_{\text{up}} = 1$ | 20200.33 |
| DS Robust with $P_{\text{up}} = 0.5$ | 19770.51 |
| DS Robust with $P_{\text{up}}$ varying as in equation (9) | 20161.29 |

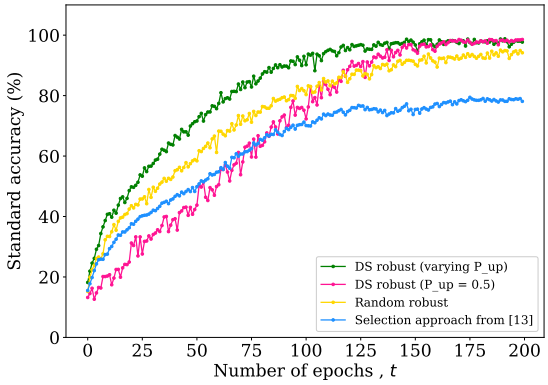


Fig. 9: Comparing the proposed method with other selection methods in terms of standard accuracy.

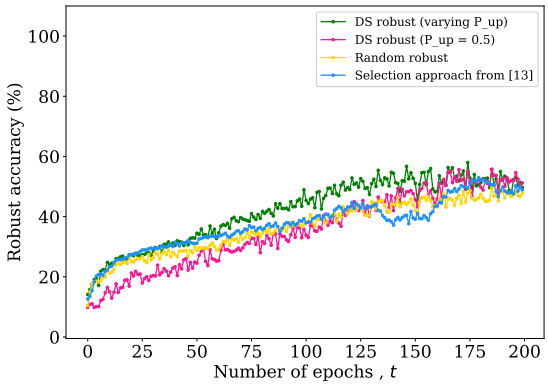


Fig. 10: Comparing the proposed method with other selection methods in terms of robust accuracy.

## V. Conclusion

Adversarial training is the most popular solution to mitigate the effect of malicious attacks on the deep neural networks. Although adversarial training is able to improve the robustness accuracy, it usually sacrifices standard accuracy in its way. Motivated by this drawback and also seeking to reduce the computational complexity during training, we proposed a data selection strategy to include the data samples that bring about a novelty to the learning process. The simulation results with CIFAR10 using the Resnet18 model indicate that the method is beneficial to improve the robustness-accuracy tradeoff and reduce the computational complexity of the training. In the future investigation, one can employ the data selection method to other CNNs models and other datasets.

## References


[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[3] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1765–1773.

[4] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, "A guide to deep learning in healthcare," *Nature medicine*, vol. 25, no. 1, pp. 24–29, 2019.

[5] Y. Liu, E. Racah, J. Correa, A. Khosrowshahi, D. Lavers, K. Kunkel, M. Wehner, W. Collins, et al., "Application of deep convolutional neural networks for detecting extreme weather in climate datasets," *arXiv preprint arXiv:1605.01156*, 2016.

[6] M. Dixon, D. Klabjan, and J. H. Bang, "Classification-based financial markets prediction using deep neural networks," *Algorithmic Finance*, vol. 6, no. 3-4, pp. 67–77, 2017.

[7] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[8] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.

[9] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International conference on machine learning*. PMLR, 2020, pp. 2206–2216.

[10] G. Ortiz-Jiménez, A. Modas, S.-M. Moosavi-Dezfooli, and P. Frossard, "Optimism in the face of adversity: Understanding and improving deep learning through adversarial robustness," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 635–659, 2021.

[11] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *International conference on machine learning*. PMLR, 2019, pp. 7472–7482.

[12] G. W. Ding, K. Y. C. Lui, X. Jin, L. Wang, and R. Huang, "On the sensitivity of adversarial robustness to input data distributions.," in *ICLR (Poster)*, 2019.

[13] C. Dong, L. Liu, and J. Shang, "Data quality matters for adversarial training: An empirical study," *arXiv preprint arXiv:2102.07437*, 2021.

[14] J. O. Ferreira, M. O. K. Mendonça, and P. S. R. Diniz, "Data selection in neural networks," *IEEE Open Journal of Signal Processing*, vol. 2, pp. 522–534, 2021.

[15] M. O. K. Mendonça, J. O. Ferreira, and P. S. R. Diniz, "Data selective deep neural networks for image classification," in *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 1376–1380.

[16] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[17] F. Croce and M. Hein, "Minimally distorted adversarial examples with a fast adaptive boundary attack," in *International Conference on Machine Learning*. PMLR, 2020, pp. 2196–2205.

[18] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: a query-efficient black-box adversarial attack via random search," in *European Conference on Computer Vision*. Springer, 2020, pp. 484–501.

[19] J. M. Danskin, "The theory of max-min, with applications," *SIAM Journal on Applied Mathematics*, vol. 14, no. 4, pp. 641–664, 1966.

[20] L. Prechelt, "Early stopping-but when?," in *Neural Networks: Tricks of the trade*, pp. 55–69. Springer, 1998.

[21] A. Raghunathan, S. M. Xie, F. Yang, J. C. Duchi, and P. Liang, "Adversarial training can hurt generalization," *arXiv preprint arXiv:1906.06032*, 2019.

[22] A. Javanmard, M. Soltanolkotabi, and H. Hassani, "Precise tradeoffs in adversarial training for linear regression," in *Conference on Learning Theory*. PMLR, 2020, pp. 2034–2078.