Adversarial Sample Generation and Training using Neural Network

Ho Yub Jung

Abstract

The neural network classifier is known to be susceptible to adversarial attacks, where projected gradient descent-like noise is added to the data, causing misclassification. These attacks can be prevented by min-max training, where the neural network is trained to handle adversarial attack data. Although min-max training is very effective, it requires a large amount of training time because each adversarial attack data generation requires several iterations of gradient back-propagation to produce. In this paper, convolutional layers are used to replace the projected gradient descent-based production of adversarial attack data in an attempt to reduce the training time. By replacing the adversarial noise generation with the output of convolutional layers, the training time becomes comparable to that of a simple neural network classifier with a few additional layers. The proposed approach significantly reduced the effects of smaller-scale adversarial attacks, and under certain circumstances, was shown to be as effective as min-max training. However, for severe attacks, the proposed approach was not able to compete with modern min-max-based remedies.

Keywords : adversarial attack| min-max training| adversarial training| neural network

I. INTRODUCTION

The recent breakthroughs in neural networks are allowing practical solutions to computer vision problems. Applications such as autonomous cars, biometric surveillance, and robotics are expected to be mature into everyday technology. Nevertheless, there are important security problems with neural networks involving adversarial attacks especially in computer vision classification problems.

Adversarial attacks refer to various methods use to trick neural network into misclassification [1]. Typically, the adversarial attack noise, produced by projected gradient descent, is applied to an image. The noise is semantically meaningless; however, it can force the neural network to misclassify an image to a particular label or prevent the correct classification of certain labels [2-4]. For example, a person can wear a t-shirt with adversarial noise that prevents the neural network from correctly classifying a wearer as a person. More alarmingly, a car can have a cover with adversarial noise so that it cannot be recognized as car which can make autonomous driving more difficult.

Off course, there are proven methods to deal with adversarial attacks, called min-max training, or more commonly referred as adversarial training [5]. The

*Chosun University, Dept. of Computer Engineering

*This study was supported by research fund from Chosun University, 2023.

Manuscript : 2024. 07. 29
Revised : 2024. 09. 12
Confirmation of Publication: 2024.09.20
Corresponding Author: Ho Yub Jung
e-mail: hoyub@chosun.ac.kr

basis is simple, during min−max training, the neural network produces adversarial attack samples from the neural network classifier and training samples. The neural network is then trained to correctly classify the adversarial samples as well as the original training samples.

Min−max training is very effective for adversarial attacks; however, there are two glaring disadvantages to this training system. First, the accuracy on noiseless data is decreased significantly; the network trained with the original training set have significantly higher accuracy than the network trained with additional adversarial attack samples [4]. Second, min−max training is much more time−consuming than regular training methods [5]. This is so because the network has to frequently produce adversarial attack samples throughout the entirety of the training. Adversarial noise has to be produced by multiple iteration of projected gradient descent, and each descent requires a forward and backward pass [5].

Depending on the quality of adversarial attack samples, up to 40 iterations of projected gradient descent for construction of adversarial samples may be required for their construction [5]. Also, as the training of the network progresses, the adversarial attack samples have to be reconstructed multiple times because they depend on the training state of the classifier [5].

In this paper, we examine a way to possibly reduce the training time of min−max training by replacing adversarial sample generation with neural network layers. The production of adversarial samples requires multiple iterations of gradient descent, and it has to be updated as the classifier is trained. By replacing the generation of adversarial samples with output from simple neural layers, the training time can be greatly reduced.

The proposed method only requires additional layers compare to conventional classifier networks, thus greatly reducing the complexity of training. In the evaluation, it is shown to reduce adversarial attack effectiveness when low level adversarial noise is applied, similar to min−max training. However, under high levels of adversarial noise, it has been shown to be ineffective compared to recent advanced min−max training.

Next, we will examine the details of min−max training. The proposed method will be introduced, and the effectiveness and weaknesses of the proposed method will be discussed. The paper concludes with possible future work.

II. Background

The following equation defines the min−max training for adversarial attacks [5].

$$\min_{\theta} \rho(\theta), where \qquad (1)$$
$$\rho(\theta) = E_{(x,y)\sim D}[\max_{\delta\in S} L(\theta, x+\delta, y)].$$

In equation (1), x and y are the sample and label, respectively. θ is the set of trainable weights of $\rho(\theta)$ which is a classifier network. δ is adversarial noise, and $x+\delta$ would make the adversarial sample. L is the loss or the distance between the adversarial sample classification and the label. E is the classification error loss.

Essentially, equation (1) wants to find

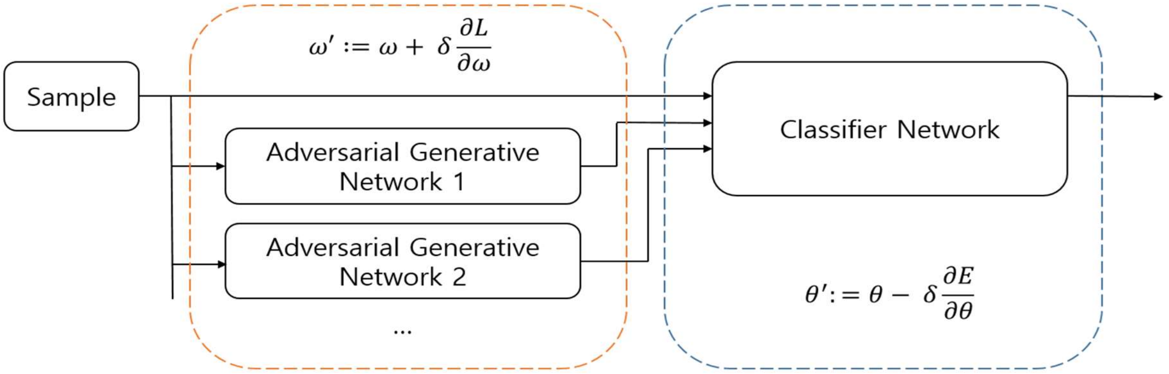
$\omega' := \omega + \delta \frac{\partial L}{\partial \omega}$
Sample
Adversarial Generative Network 1
Adversarial Generative Network 2
...
Classifier Network
$\theta' := \theta - \delta \frac{\partial E}{\partial \theta}$

Figure 1. The figure shows the proposed adversarial training approach, where the neural network is trained to produce adversarial samples instead of updating the samples directly. The adversarial generative network' s weights are updated for misclassification, and the classifier' s weights are updated for correct classification.

adversarial noise δ of confined size that would result in the wrong classification of $x+\delta$. At the same time, equation (1) is trying to find trainable weights θ that would correctly classify $x+\delta$. Thus, the min-max training seemly has contradicting goals, which makes the training very difficult.

The computational complexity comes from the calculation of adversarial noise δ. Since the error between the classification label and the true label has to be maximized, δ is found by multiple iterations of gradient ascent on the sample [5]. However, instead of a single update per sample, a multiple gradient ascent steps must be performed in order to maximize the loss. The gradient has to be recalculated with a forward and backward pass every time. Also, new adversarial samples have to be reconstructed whenever there is a significant change to the trainable weights, which are constantly updated to minimize classification error. Thus, depending on the size confinement of δ and the frequency of adversarial sample generation, min-max training can be up to 40 times computationally intensive than traditional classifier training [4].

III. Proposed Method

In this paper, we introduce a method to estimate the adversarial sample $x+\delta$ from a sequence of convolution layers. The min-max training criteria are adjusted into the following equation.

$$\min_{\theta} \rho(\theta)\,, where \qquad (2)$$
$$\rho(\theta) = E_{(x,y)\sim D}[\max_{\omega} L(\theta, g_{\omega}(x), y)].$$

In the equation (2), the adversarial sample $x+\delta$ is replaced by the convolution network $g_{\omega}(x)$ with trainable weights ω. Thus, for the maximization part, the loss between the classification result of $g_{\omega}(x)$ and the true label is maximized. Specifically, the multi-categorical cross entropy loss between $\rho(g_{\omega}(x))$ and y can be maximized by updating the weights of $g_{\omega}(x)$ but not the weights of classifier $\rho(\theta)$. We will refer to $g_{\omega}(x)$ as the adversarial sample generator.

The problem with using the convolution

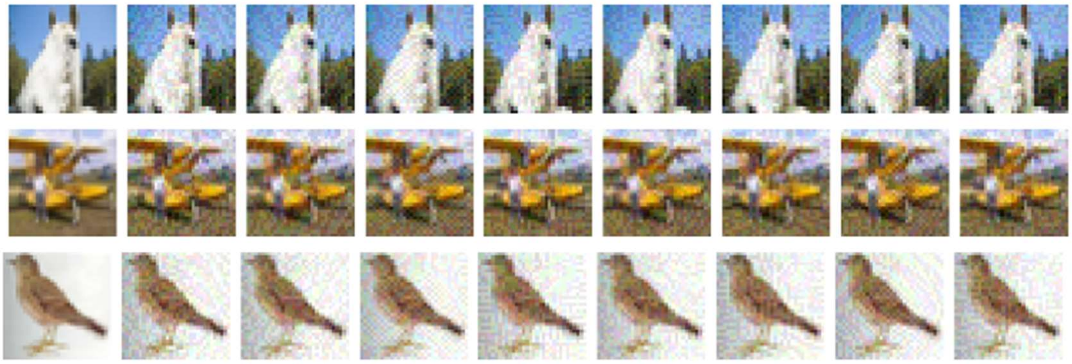

Figure 2.This figure shows examples of adversarial samples generated by the proposed approach. The left most images are the originals and the rest are adversarial samples. The adversarial noises are added using convolution layers instead of projected gradient descent. Each adversarial sample has different noises that were produced by different generator networks.

network as the adversarial sample generator is that it can output an adversarial sample that is too far away from the original data. Thus, we have to provide a min and max range layer for the adversarial samples.

$$g_{\omega}(x) = min(max(\varphi_{\omega}(x), x - s), x + s) \quad (3)$$

In the equation (3), $\varphi_{\omega}(x)$ is a convolutional neural network with trainable weights. s is the max differences allowed to the samples by the adversarial sample generator. This way, we can limit the changes to the adversarial sample, similar to the original min−max loss, where the adversarial noise is confined to a specified space $\delta \in S$. Finally, if L is the cross−entropy loss, it can be denoted as follows.

$$L(\theta, g_{\omega}(x), y) = CE(\rho(g_{\omega}(x)), y). \quad (4)$$

In the minimization part, the weights of classifier are updated by the gradient descent on the classification loss E. In contrast, the updates for adversarial sample generators are performed by gradient ascend. In the following equations (5), θ' and ω' are updated weights for the classifier and the adversarial sample generator.

$$\begin{aligned} \theta' &:= \theta - \delta \frac{\partial E}{\partial \theta} \\ \omega' &:= \omega + \delta \frac{\partial L}{\partial \omega} \end{aligned} \quad (5)$$

This is different from the min−max formulation, where the sample is updated by the gradient ascend. The advantage is that the trainable weights in the adversarial sample generator is updated concurrently with the weights in the classifier, whereas in min−max training, gradient ascend on the adversarial sample has to be iterated many times for each significant training progression of the classifier network. See Fig. 1 for the diagram of the proposed method. Some of the generated adversarial images are shown in Fig. 2.

IV. Evaluation

We evaluated the proposed method

using the CIFAR−10 dataset with projected gradient descent squared (PGD ℓ_2) adversarial samples. A pre−trained ResNet−20 has been used as the base model for the comparison test. The same pre−trained base model is used as the initial network for both min−max training and the proposed training approach.

For our method, we employ 32 adversarial generator networks to retrain the classifier. The maximum difference between the original data and the adversarial sample has been set to $s = 0.2$. Stochastic gradient descent is used with a learning rate of 0.005. Similarly, the min−max training started with the same ResNet−20 base model. The adversarial samples during the training are generated using the PGD method with an epsilon value of 0.05. The maximum iteration is set to be 40, and the learning rate of the projected gradient descent was 0.005.

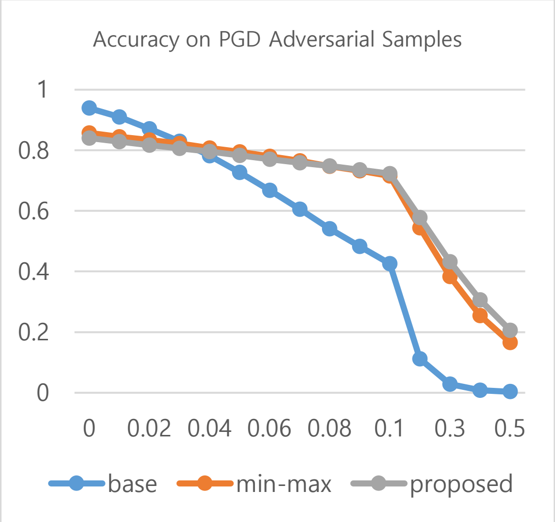

Figure 3. The y−axis is accuracy ratio. The x−axis is different ℓ_2 unitless values for PGD adversarial samples produced. The base model is a pre−trained ResNet−20 for the CIFAR−10 dataset. Both min−max [5] and the proposed approach were trained using the base model. Although the accuracies are very similar, the proposed approach has a much shorter training time.

Table 1. Computation time comparison between the proposed method and min−max training [5] on the same computer.

	Proposed	Min−max [5]
Average training time for 1 epoch	105 seconds	551 seconds

Both methods were trained for 8 epochs. As shown in Table 1, the proposed approach consumed 105 seconds for 1 epoch training. The min−max training consumed 551 seconds for each epoch, using the same computer. The training time may be vary depending on the number of adversarial generators used as well as the maximum iteration number setting. However, there is a clear computational advantage to the proposed method.

The accuracy comparison is illustrated in Fig. 3, which shows the changes in accuracy as PGD ℓ_2 adversarial noise is applied to the test data. When there is no adversarial noise present, the base model achieves the highest accuracy. The proposed and min−max method show a degradation of accuracy at noiseless data, which is typical for robust training. However, as the adversarial noise increases, the accuracy degradations of the proposed and the min−max training methods are much slower than that of the base model.

As shown in Fig. 3, the proposed adversarial training method has very similar accuracy performance as the min−max training. However, the proposed training approach also has a clear advantage in training time, as iterative backpropagation calculations are not required to produce adversarial samples.

However, many different factors contribute to the training time and robustness. First, the number of adversarial sample generators can be adjusted, where more generator can result in greater robustness in exchange for longer training time. Parameters like s and δ also effect the robust accuracy and training time. The choice of dataset as well as the pre-trained network choice might impact the result. There are also many different adversarial attack methods such as PGD ℓ_∞, one pixel, jitter and such [6-8].

Compared to the state-of-the-art robust classifiers, however, the proposed approach is still far behind in terms of accuracy. For the $\ell_2 = 0.5$ attacks, the state-of-the-art methods can reach up to 84.97% accuracy using diffusion augmentation techniques [9, 10]. This is much higher than the proposed approach with no augmentation other than the random flip. Many different methods can be combined to achieve contemporary the robustness and accuracy [11, 12]. More testing in combination with more recent approaches is needed to fully evaluate the proposed approach.

V. Conclusion

We proposed a generative neural network to produce the adversarial samples. Conventional adversarial samples are generated by iterating projected gradient descent on the original samples. The proposed approach allows for the fast generation of adversarial samples because the generative neural network can be trained concurrently with the classifier. The conventional min-max training approach, however, requires multiple iteration of gradient descent for each update to the classifier. Evaluation on CIFAR-10 dataset showed that the proposed training approach can achieve comparable accuracy with relatively shorter training time per each epoch.

Nevertheless, the evaluation is strictly confined to the simplest of circumstances. Many factors such as model choice, parameter settings, adversarial attack methods, and dataset that can affect the comparison. Additionally, the recent advancements in min-max training and the inclusion of complex data-augmentation techniques have not been explored in this paper. More evaluation and incorporation of current augmentations and other advancements are needed to fully evaluate the potential of the proposed method.

REFERENCES

[1] Szegedy, C. "Intriguing properties of neural networks." *arXiv preprint arXiv:1312.6199* (2013).

[2] Bai, Tao, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. "Recent advances in adversarial training for adversarial robustness," *arXiv preprint arXiv:2102.01356* , 2021.

[3] Qian, Zhuang, Kaizhu Huang, Qiu-Feng Wang, and Xu-Yao Zhang. "A survey of robust adversarial training in pattern recognition: Fundamental, theory, and methodologies," *Pattern Recognition* 131, 2022.

[4] Tsipras, Dimitris, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. "Robustness may be at odds with accuracy," *arXiv preprint arXiv: 1805.12152* , 2018.

[5] Mądry, Aleksander, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. "Towards deep learning models resistant to adversarial attacks," *stat 1050*, no. 9, 2017.

[6] Su, Jiawei, Danilo Vasconcellos Vargas, and Kouichi Sakurai. "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation 23*, no. 5, 2019.

[7] Wang, Zekai, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. "Better diffusion models further improve adversarial training," *In International Conference on Machine Learning,* pp. 36246-36263. PMLR, 2023.

[8] Schwinn, Leo, et al. "Exploring misclassifications of robust neural networks to enhance adversarial attacks," *Applied Intelligence*, vol. 53, 2023.

[9] https://robustbench.github.io/(accessed Jul., 28, 2024).

[10] You, Zhonghui, Jinmian Ye, Kunming Li, Zenglin Xu, and Ping Wang. "Adversarial noise layer: Regularize neural network by adding noise." In 2019 IEEE International Conference on Image Processing (ICIP), pp. 909-913. IEEE, 2019.

[11] Sankaranarayanan, Swami, Arpit Jain, Rama Chellappa, and Ser Nam Lim. "Regularizing deep networks using efficient layerwise adversarial training." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, 2018.

[12] Tramèr, Florian, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel, "Ensemble adversarial training: Attacks and defenses," arXiv preprint arXiv:1705.07204, 2017.

Authors

Ho Yub Jung

He received the B.S. degree in electrical engineering from The University of Texas at Austin, in 2002, and the M.S. and Ph.D. degrees in electrical engineering and computer science from Seoul National University, in 2006 and 2012, respectively.