Poisoning Attack on Defense GAN

J-How Huang
Department of Information Management
National Taiwan University
b07705026@ntu.edu.tw

Chi-Pin Huang
Department of Computer Science
National Taiwan University
b07501122@ntu.edu.tw

Yu-Chen Lin
Department of Computer Science
National Taiwan University
b06504025@ntu.edu.tw

Abstract

Generative adversarial network (GAN) aims to learn the distribution of training data. Utilizing this idea, Defense GAN (1) which is trained with benign examples becomes a helpful preprocessing skill for eliminating the possible adversarial characteristic of the input. Data poisoning attack (2) is a novel attack technique on deep neural network (DNN) which poison the training dataset and disable the discriminative ability of DNN model. Combining these two ideas, we use projected gradient descent (PGD) (3) and universal adversarial perturbation (UAP) (4) adversarial examples to poison training images of Defense GAN. Our results show that the defensibility of Defense GAN can be decrease after poisoning attack.

1 Introduction

1.1 Related Work

Defense GAN Defense GAN is trained to model the distribution of unperturbed images. In the training stage, clean benign images are used to train the generator and discriminator. In the inference stage, we would like to map the input images to the distribution of the benign images. With R different random initialization of $\mathbf{z}$ sampled from the normal distribution, we do gradient descent on $\mathbf{z}$ according to the mean square error between generated image $G(\mathbf{z})$ and input image $\mathbf{x}$. After the gradient descent, we choose $\mathbf{z}^*$ such that $G(\mathbf{z}^*)$ be closest to x from R results and feed $G(\mathbf{z}^*)$ into our classifier. The expectation is that if input $\mathbf{x}$ is a benign image, we can reconstruct well if our GAN trains well. And if input $\mathbf{x}$ is an adversarial image, we can eliminate the adversarial pattern by mapping $\mathbf{x}$ into benign distribution. Therefore, defense GAN is often used to pre-processing step prior to classification.

Poisoning Attack Most of adversarial attack is done in the inference stage, but poisoning attack, with a totally different thought, attack the model by inserting carefully constructed poison instance into the training data. This kind of attack is usually operated on two characteristics of the dataset, respectively modifying the features or the labels. In our work, we apply the concept of modifying the features of the dataset during the training stage of our model, and we would like to observe how strong is the poisoning effect.

Adversarial Attack Projected gradient descent (PGD) and universal adversarial perturbation (UAP) are two techniques to decrease the accuracy of the machine learning models by small amount of perturbation on the input. More details will be discussed on section 2.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

1.2 Problem Formulation

In our work, we are going to bring the idea of poisoning attack on to Defense GAN. At first, we imagine a company with powerful MLaaS products which is based on machine learning model. The company knows well about the potential problems of adversarial attack and would like to take advantage of defense GAN to be the add-on step before their powerful classifiers. However, the classifier is trained a couple of years ago, and the current dataset has been secretly poisoned by a wicked employee. The company then uses the poisoned dataset to train their defense GAN. Based on this kind of scenario, we construct a poisoned dataset containing some adversarial examples (PGD/UAP), and use the poisoned dataset to train our defense GAN. After the training, we would like to see whether the poisoned defense GAN can lead to serious failure on the classifier or not.

1.3 Contribution

In our work, we propose a poisoning algorithm on defense GAN, and prove the practicability of our method. First, we check that our poisoning attack works to decrease the defensibility of defense GAN and find that this kind of attack can be transferred to "black box" models. However, our method appear to have significant impact only when poisoning proportion is high enough. Next, we discover that under small proportion of poisoning, our purposed method actually hides one "backdoor" inside it. Then we can use specific attack (UAP-based) on the classifier even with add-on defense GAN.

2 Preliminaries

2.1 Dataset

Our dataset is based on CIFAR10 and CIFAR100. Experimental results are mainly shown on CIFAR10, since results of CIFAR100 are quite similar.

2.2 Target Models

Target models are classifiers after defense GAN. We use three powerful models in image classifying, respectively Inception-v4, ResNet-152, and VGG-19. We train those models by 50000 training images of CIFAR10 and CIFAR100 and each for 200 epochs using SGD optimizer. The accuracy is evaluated on 10000 tesing images of CIFAR10 and CIFAR100. Results are in the Table 1.

2.3 Projected Gradient Descent (PGD)

PGD is an iterative attack method which extend the FGSM into iterative version with view to getting better gradient update direction. The update formula is

$$x'_0 = x$$

$$x'_{n+1} = \text{Clip}^{\epsilon}_{x}(x'_n + \alpha \cdot \text{sign}(\nabla_x L(x, y; \theta))$$

where $\text{Clip}^{\epsilon}_{x}$ means that the adversarial instances should be within the ϵ-ball to the original instance, n denotes the iteration number and α is the step size. This PGD version does not have restart utilities.

We use **Inception-v4** to generate our adversarial examples with $\epsilon = 8$ and 20 steps. Results are shown in Table 1. We can observe that PGD leads to nearly zero accuracy on white box settings, and there are still transferability shown on other black box models.

2.4 Universal Adversarial Perturbation (UAP)

UAP is a kind of adversarial attack, which aims on adding a single perturbation pattern to every image in a given dataset, to achieve the attack. The update formula of the UAP is

$$\Delta v_i \leftarrow \arg\min_{r} ||r||_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i)$$

$$v \leftarrow \text{Clip}^{\epsilon}_{0}(v + \Delta v_i)$$

Data \ Model	Inception-v4	ResNet-152	VGG-19
CIFAR10	95.2% / 0.00% / 35.1%	95.5% / 26.1% / 49.6%	93.6% / 43.8% / 55.4%
CIFAR100	78.2% / 0.03% / 36.4%	79.6% / 13.2% / 43.4%	71.7% / 19.0% / 37.8%

Table 1: Benign / PGD / UAP Accuracy of target models

Data \ Model	Inception-v4	ResNet-152	VGG-19
CIFAR10	46.8% / 49.7% / 45.6%	49.0% / 49.6% / 48.1%	52.2% / 53.0% / 50.7%
CIFAR100	22.5% / 22.5% / 21.8%	24.6% / 23.5% / 23.4%	22.9% / 21.7% / 21.1%

Table 2: Benign / PGD / UAP Accuracy of target models **with defense GAN**

where v is the UAP, $\hat{k}$ is the classifier, iterate through the images x_i in the dataset. We don't iterate through whole dataset, instead, we use a subset of it, which is proved effective in (4). The optimization problem in first line is done by Deepfool (5).

We set $\epsilon = 16$ and subset size 300 and to generate UAP in CIFAR10, $\epsilon = 10$ and subset size 1000 in CIFAR100. Note that UAP is also generated by use **Inception-v4**. The reason of the larger ϵ in CIFAR10 is models perform much better in CIFAR10, higher ϵ is required to observe significant attack. Other settings remain the same as original paper. The result of accuracy after UAP is in Table 1.

2.5 Reproduce Defense GAN

We reproduce the defense GAN using 50000 training images on CIFAR10 and CIFAR100. We use random initialization $\mathbf{z}$ of size 128 (i.e. R = 128), and do gradient descent for 500 epochs. The results are shown in Table 2. Three things are interesting to mention. First, the accuracy may seem lower than the original paper, but we use CIFAR10 and CIFAR100 as our dataset, while original paper evaluates on MNIST. We have surveyed on papers which have reproduced on the CIFAR10 and CIFAR100, and the results are quite corresponding with our results. Second, we can see the defense on PGD have largely increased in both white box and black box settings. Third, the results are a baseline for our experiments, and we want to decrease the accuracy as much as possible. Lower accuracy represents that defense GAN is not working well on preprocess-based defense.

3 Poisoning Attack on Defense GAN

3.1 Damage on Defensibility of Defense GAN

3.1.1 UAP-Poisoned Defense GAN

In this section, we will use the perturbation generated by UAP to poison the dataset. First, we would like to see whether the poisoning can decrease the defense ability of defense GAN. Then, we compare the performance on different poisoning proportion (10%, 50%, 100%), and we want to know how much proportion can take effect on destroying defense GAN. Results are shown in Table 3.

From Table 3, we can observe that our poisoning techniques actually work to decrease the performance of defense GAN. However, by only 10% and 50% poison proportion, the poisoning seems to have no effect on defense GAN performance. We can only largely decrease the performance of the defense GAN by 100% proportion, which is not practical.

Poison Proportion	100%	50%	10%
PGD examples	16.1%	47.7%	50.6%
DfGAN performance	-33.6%	-2.00%	+0.90%

Table 3: Acc of feeding PGD examples into UAP-perturbed DfGAN (Inception-v4 as classifier)

Poison Proportion	100%	50%	10%
Benign examples	16.2%	48.9%	52.0%
DfGAN performance	-33.5%	-0.80%	+2.30%

Table 4: Acc of feeding benign examples into UAP-perturbed DfGAN (Inception-v4 as classifier)

Table 4 has another thing to show. This time, we feed the benign images into our settings. Similar to previous discussion, the performance of defense GAN drops a lot for 100% poisoning. This means that not only PGD examples can lead to failure of defense GAN, but benign examples can also decrease the effect of defense GAN.

3.1.2 More Insight on UAP-Poisoned Defense GAN and UAP pattern

First, if we have our Defense GAN well trained, it will learn the distribution of the training data, which is actual benign data plus UAP perturbation. Hence, whatever generated by Defense GAN should be an image fitting this distribution plus the UAP perturbation. The reconstruction procedure is as follows:

$$\mathbf{z}^* = \arg\min_z \|G(\mathbf{z}) - \mathbf{x}\|_2^2$$

$$G(\mathbf{z}^*) = \mathbf{x}' + \mathbf{UAP}$$

Starting with random initialized $\mathbf{z}$, to minimize the mean square error, $\mathbf{x}'$ here should be $\mathbf{x}$ plus some random noise with constant mean less than zero. We split $\mathbf{x}'$ into $\mathbf{x}$ plus $\mathbf{g_{GAN}}$, and move $\mathbf{x}$ to the left hand side.

$$G(\mathbf{z}^*) = (\mathbf{x} + \mathbf{g_{GAN}}) + \mathbf{UAP}$$

$$G(\mathbf{z}^*) - \mathbf{x} = \mathbf{g_{GAN}} + \mathbf{UAP}$$

Here, we take the expectation value from both sides

$$\mathbb{E}[G(\mathbf{z}^*) - \mathbf{x}] = \mathbb{E}[\mathbf{g_{GAN}} + \mathbf{UAP}] = c + \mathbf{UAP}$$

Now the UAP term is left alone with a constant in the right hand side. Based on this result, we

reconstruct 1000 benign images with 100% UAP-Poisoned Defense GAN. And then we subtract each reconstructed image by corresponding benign one and take the average over these 1000 deviations. We show the result below:

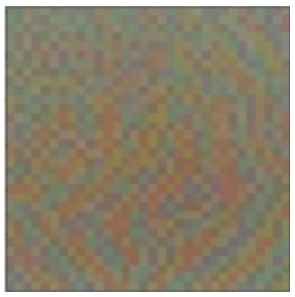

Figure 1: UAP perturbation

Figure 2: Average of 1000 deviations

It is obvious that Figure 1 and Figure 2 have some similar patterns, which can be verified by the formulation above.

3.2 Transferability of Poisoning

In the previous section, we poison defense GAN with UAP pattern generated from Inception-v4, and then test the UAP-poisoned DfGAN with Inception-v4 as classifier. In both attacking and inference stages, we use the same model, which is somewhat like "white box" setting. In this section, we are going to test the transferability of our purposed method. Two different models (ResNet-152 and VGG-19) would be adopted in attacking and inference stages. Results are shown in Table 5.

Poison Proportion	100%	50%	10%
Inception-v4	16.2%	48.9%	52.0%
ResNet-152	20.3%	48.0%	54.3%
VGG-19	24.8%	53.2%	57.4%

Table 5: Acc of feeding benign examples into UAP-perturbed DfGAN with different classifiers

Poison Proportion	100%	50%	10%
PGD-perturbed dfGAN performance	-0.02%	+1.40%	+2.80%
UAP-perturbed dfGAN performance	-33.6%	-2.00%	+0.90%

Table 6: Performance decrease of PGD/UAP-perturbed DfGAN (Inception-v4 as classifier)

From Table 5, we can observe that under the so-called "white box" setting (Inception-v4), the accuracy decrease more than other "black box" settings (ResNet-152 and VGG-19).

3.3 Method of Poisoning: PGD versus UAP

UAP pertubation is famous for its universality and may be simply learned by GAN, while PGD pertubation could be extremely diverse for each image. Thus in this section, we would poison Defense GAN by PGD adversarial examples and compared with UAP-poisoned DfGAN.

From Table 6, we can see that the performance of PGD-perturbed defense GAN is similar to the performance of benign-trained defense GAN. It means that GAN cannot learn the distribution of PGD in this case. As mentioned before, since PGD adversarial examples are generated according to each image, so the pattern may not be easy for simple GAN training. In opposite, GAN seems to learn much more better on UAP pattern, which leads to large decrease on the performance of UAP-perturbed defense GAN.

3.4 Subtle Enemy Under Poisoning Attack

The poisoned (little proportion poisoned) defense GAN may look robust to our proposed poisoning attack in fact. However, when defense GAN encounter the poison which corresponds to UAP attack that added in training data, defense GAN will have only limited defense ability.

When 100% of training data is poisoned by a single UAP pattern, defense GAN may learn to generate this certain pattern, further send this pattern into classifier. However, when smaller proportion of training data is poisoned, defense GAN may not be able to generate this pattern, so it would not change benign images into adversarial attack images. We guess, by intuition, defense GAN still "remember" this pattern, as long as we ask it to reconstruct a image with such pattern, it can still recall this pattern and reconstruct it. By this idea, the UAP survives the defense of Defense GAN and further sends into classifier.

We summary our results are shown in Table 7. The white box setting means that we use generated UAP to perturb defense GAN during training and this generated UAP also become an input during testing. The black box setting means that we use one kind of UAP to perturb defense GAN during training and another kind of UAP (generated by different images) becomes an input during testing. From white box setting, we can observe that 5% poisoning can do effect on defense GAN, and 10% poisoning is enough to do significant damage on defense GAN. From black box setting, we can also see kind of tranferability preserves during 10% and 50% poisoning.

Note that we call it a subtle enemy because this weakness is difficult to discover as it only occur when sending such UAP pattern. Service designer may found its defense GAN works fine under both

Poison Proportion	50%	10%	5%	1%
White box performance	**-32.4%**	**-20.1%**	**-5.50%**	+2.80%
Black box performance	**-11.5%**	**-10.6%**	+0.07%	+0.12%

Table 7: Performance decrease of PGD/UAP-perturbed DfGAN (Inception-v4 as classifier)

Image Proportion	Benign	PGD
100%	51.0% / 53.6% / 57.9%	49.5% / 49.5% / 55.0%
50%	53.0% / 51.8% / 56.1%	51.1% / 53.0% / 55.8%
10%	52.5% / 52.4% / 56.1%	50.5% / 51.7% / 54.2%

Table 8: CIFAR10 images into PGD-poisoned DfGAN on Inception-v4 / ResNet-152 / VGG-19

Image Proportion	Benign	PGD
100%	27.0% / 28.9% / 25.6%	23.6% / 23.6% / 23.3%
50%	27.1% / 27.6% / 26.5%	23.2% / 25.1% / 23.4%
10%	27.4% / 29.6% / 26.4%	22.6% / 24.8% / 22.5%

Table 9: CIFAR100 images into PGD-poisoned DfGAN on Inception-v4 / ResNet-152 / VGG-19

Image Proportion	Benign	PGD
100%	16.2% / 20.3% / 24.8%	16.1% / 21.6% / 24.6%
50%	48.9% / 48.0% / 53.2%	47.7% / 47.2% / 52.2%
10%	52.0% / 54.3% / 57.4%	50.6% / 52.8% / 53.0%

Table 10: CIFAR10 images into UAP-poisoned DfGAN on Inception-v4 / ResNet-152 / VGG-19

Image Proportion	Benign	PGD
100%	16.2% / 20.3% / 24.8%	16.1% / 21.6% / 24.6%
50%	48.9% / 48.0% / 53.2%	47.7% / 47.2% / 52.2%
10%	52.0% / 54.3% / 57.4%	50.6% / 52.8% / 53.0%

Table 11: CIFAR10 images into PGD-poisoned DfGAN on Inception-v4 / ResNet-152 / VGG-19

benign and adversarial (e.g. PGD) images. However, some adversaries may have already poisoned defense GAN and know the weakness of defense GAN.

4 Experiments

This section is a summarization of detailed experiments we have conducted. The experiment setting is that we input 1000 benign and PGD images respectively by different poisoned portion (100%, 50%, 10%) defense GAN. The results are evaluated on CIFAR10 / CIFAR100 and three target models Inception-v4 / ResNet-152 / VGG-19. Please refer to Table 8 to 11 for all results.

5 Conclusion

In this work, we propose a poisoning attack method on defense GAN. We conduct several experiments to illustrate the achievability of our proposed method. In the experiment results, the defensibility of our purposed UAP-poisoned defense GAN decrease, while naive PGD-poisoned defense GAN seems to have same performance as benign-trained defense GAN. We also examine and confirm the transferability of this poisoning attack. However, our method appear to useless when poisoning proportion is low. We further find that even under small proportion of poisoning (5%-10%), our purposed method actually hides one backdoor. It means that the minor-poisoned defense GAN is vulnerable to UAP attack, especially the identical UAP perturbation we used to poison the Defense GAN training dataset.

We list two future work here. First, our UAP method utilize ϵ larger than 8. However, the generated images may appear to be easily detected by human eyes. Decrease the ϵ of UAP or consider more about human vision is a great field to explore more. Second, we can have more practical attack applied in the real world, which will be make our proposed method more convinced.

References

[1] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," 2018.

[2] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," 2018.

[3] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2019.

[4] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," 2017.

[5] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," 2016.