

國立臺灣大學電機資訊學院電機工程學系
碩士論文



Department of Electrical Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

利用具可適性之生成對抗網路客製化對抗例生成器

Adaptive GAN: Customizing Generators for Adversarial
Examples

蔡仕竑

Shih-Hong Tsai

指導教授：陳銘憲 博士

Advisor: Ming-Syan Chen, Ph.D.

中華民國 107 年 8 月

August, 2018

國立臺灣大學 (碩) 博士學位論文
口試委員會審定書

利用具可適性之生成對抗網路客製化對抗例生成器
Adaptive GAN: Customizing Generators for Adversarial
Examples

本論文係蔡仕竑君 (R05921030) 在國立臺灣大學電機所完成之
碩士學位論文，於民國 107 年 7 月 24 日承下列考試委員審查通過及
口試及格，特此證明

口試委員：

陳韶曼

(簽名)

(指導教授)

邱克驊

李弘序

陳怡辰

楊得昇

系主任、所長

劉志文

(簽名)

(是否須簽章依各院系所規定)





ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to all those who have accompanied and supported me along the journey of my Master degree. I am grateful to have all of you by my side and to be the staunch backing of mine, helping me face all sweetness and bitterness.

First of all, I would like to express my sincere appreciation to my advisor, Prof. Ming-Syan Chen, for his support and inspiration. I have learned not only research capabilities but also the enthusiastic attitude to erudition under his guidance.

In addition, I would like to thank my colleagues and friends in Network and Database Laboratory of Department Of Electrical Engineering who give me a lot of assistance and encouragement.

Last but not least, I would like to thank my family for their support and endless love and to my father, who passed away during my graduate studies, for his dying wish that gave me a tons of motivation of hard work.





中文摘要

對抗例 (adversarial examples) 指的是那些爲了使神經網絡錯誤分類而特製的資料。當我們討論創造這些對抗例的方法時，我們通常會聯想到基於擾動的方法 — 在正常的資料上添加不可見的擾動來製造對抗例。對人類來說，由擾動法產生的對抗例將完全保留其原本資料的視覺外觀，從而使人類分不出和正常資料的差異，但 DNN 模型會將兩者視爲完全不同的外觀，從而產生誤導性的預測。然而，在本文中，我們認爲只依賴這個將現有資料轉化成對抗例的架構會限制對抗例的多樣性。我們提出了一個基於非擾動的框架，該框架以基於條件約束生成對抗網絡的生成模型直接生成對抗例。因此，生成的對抗例不會與任何現有的資料有外觀上的相似性，從而擴大了對抗例的多樣性，增加了防禦對抗例的難度。並且，我們將這個框架擴展到預先訓練的條件約束生成對抗網絡模型，其中，我們能將現有的普通生成模型經過些微的訓練後，轉變成一個專門生成對抗例的「對抗例生成模型」。我們針對 MNIST 和 CIFAR10 資料集進行了實驗，結果令人滿意，表明這種方法可做爲先前對抗例製造策略的替代方案。

關鍵字：對抗例、生成對抗網絡、條件約束生成對抗網絡





ABSTRACT

Adversarial examples are malicious data designed with the intention of causing misbehavior of neural networks. Typically, these examples are featured in terms of similar physical appearance to normal images, yet discrepancy in the prediction result when similar normal image and adversarial example are evaluated by the same DNN model. To create such examples, current methods rely mainly on techniques that overlay invisible perturbations onto normal images. The resulting adversarial examples therefore resemble the original images, but with different output in DNN's result. In this work, however, we consider crafting adversarial examples from existing data as a limitation to example diversity. We propose a non-perturbation-based framework that generates native adversarial examples from class-conditional generative adversarial networks. As such, the generated data will not resemble any existing data and thus expand example diversity, raising the difficulty in adversarial defense. We then extend this framework to pre-trained conditional GANs, in which we turn an existing generator into an "adversarial-example generator". We conduct experiments on our approach for MNIST and CIFAR10 datasets and have satisfactory results, showing that this approach can be a potential alternative to previous attack strategies.

Key words: adversarial examples, GAN(generative adversarial network), class-conditional GAN





Contents

| | |
|--|-------------|
| 口試委員會審定書 | i |
| Acknowledgements | iii |
| Chinese Abstract | v |
| English Abstract | vii |
| Contents | ix |
| List of Figures | xi |
| List of Tables | xiii |
| 1 Introduction | 1 |
| 2 Background | 3 |
| 3 Adaptive GAN | 7 |
| 3.1 Problem Setting | 7 |
| 3.2 Architecture | 7 |
| 3.3 Objective | 9 |
| 3.4 Turning Pre-trained Generator into Adversarial-example Generator | 11 |
| 4 Experimental results | 12 |
| 4.1 The Effect of Masked Loss Function | 12 |
| 4.2 Adversarial Attack on MNIST | 13 |
| 4.3 Adversarial Attack on CIFAR10 | 14 |
| 4.4 Cross-domain attacks | 16 |
| 5 Conclusion | 18 |
| Bibliography | 21 |





List of Figures

| | | |
|-----|--|----|
| 3.1 | Illustration of a typical perturbation-based method. (The iconic panda images are adopted from the representative work of Goodfellow et al. [8]) | 8 |
| 3.2 | The high level architecture of (a) a typical conditional-GAN variant and (b) the Adaptive GAN. | 9 |
| 4.1 | Comparison of the images sampled from models with attacking loss calculated using Equation 2 and Equation 1 respectively. | 13 |
| 4.2 | Adversarial examples generated from Adaptive GAN on MNIST with each row conditioned on different class label and each column conditioned on different attack target. | 14 |
| 4.3 | Adversarial examples sampled from the process of adaptive retraining on CIFAR10 Challenge, where the accuracy has been reduced by 12% after two epochs of retraining. | 16 |
| 4.4 | Cross-domain Adversarial examples. An image of horse and an image of car to be misinterpreted as a "watch for pedestrians sign" by some classifier on an autonomous vehicle are generated with Adaptive GAN. | 17 |





List of Tables

| | | |
|-----|--|----|
| 4.1 | Attack success rate of the target classifier (MNIST Challenge) | 14 |
| 4.2 | Accuracy of the target classifier (CIFAR10 Challenge) | 15 |





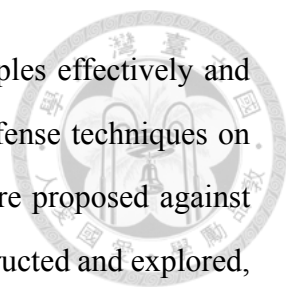
Chapter 1

Introduction

In the development of machine learning technology, the study of the model security against malicious data has always been a core topic of research. Researchers have found that deliberately manipulated training inputs are toxic in obfuscating machine learning models, increasing test errors, and leading to misclassified samples, while others have proposed security evaluation methods in various data-centric attack and defense scenarios [3, 1, 2]. In recent years, the research focus has gradually shifted to deep learning, one of the sub-field of machine learning, with the proliferation of its applications, and to the core of these research works are to defense against the so-called adversarial examples.

The term "adversarial example" first came into literature in 2013 in Szegedy's work [28], which stated that small perturbations to a DNN classifier's inputs would change the classifier's predictions. These examples also look indiscriminable to naked eyes; namely, they could easily fool DNN classifiers without being distinct to humans, making them hard to detect. In such a case, a purposely crafted adversarial example of dog may be misleading to some target classifier and be misclassified as a cat by the classifier, while humans still think of it as an image of dog. Such loophole imply an Achilles'heel inside the mechanism of deep neural networks, accompanying with security issues as adversarial examples allow adversaries to manipulate networks' outputs.

Ever since these blind spots of DNNs have been identified, related research has emerged



and exploded. Attack strategies on how to create adversarial examples effectively and efficiently have been developed [28, 8, 13, 20, 5, 31, 29], while defense techniques on enhancing model robustness and detecting adversarial examples were proposed against [18, 4, 30, 9]. Also, a variety of explanatory theories have been constructed and explored, attempting to identify the causes of adversarial examples [28, 8]. Other works investigate this phenomenon from different perspectives, including real-world application case studies [13, 27] and the visualization of the adversarial subspace [15].

As to the adversarial example space, however, we found an absence among attack techniques that could enrich adversarial example diversity. Current attack strategies focus mainly on the creation of adversarial perturbations and add them back to normal data. Through this process, the visual appearance of adversarial examples still depend on existing data, which might limit the example subspace we can achieve. We consider it as a research gap, and in this work we study generative models that can complement those so-called “perturbation-based methods” by directly creating adversarial examples instead of generating from existing data.

The structure of this work is organized as follows. Chapter 2 introduces the research and techniques related to our work. Chapter 3 formally describes the problem setting as well as the proposed method. Chapter 4 shows experimental results together with some discussions. Chapter 5 summarizes our work and future plans.



Chapter 2

Background

The background for this work is tripartite. First, the theoretical definitions of adversarial examples are described, and several precursors of adversarial-example generation methods such as FGSM and CW are introduced. Second, the basic principle of generative adversarial network (GAN) and its derived techniques that have been combined together to build our method will be covered later in this chapter. Third, some of the adversarial-example generating models that also take advantage of GAN will be mentioned and compared with our method.

Adversarial examples Szegedy et al. [28] first identified this perception inconsistency between humans and machine learning models. Given an image being slightly perturbed, a DNN model may treat it as a totally different image with a completely different prediction, whereas humans may not even perceive the difference. These blind spots expose neural network models to malicious attacks, and for such slightly-changed data, they have been termed as adversarial examples. Typically, attacks of adversarial examples are mainly categorized as two types: the targeted and the untargeted.

Assume that there exists a normal image x , the first type of attack refers to those methods that create examples x' such that $C(x') = t$, given C , the classifier of the attack target, and t , the class of the attack target. Contrary to the targeted attack, the untargeted attack urges to generate adversarial examples x' such that $C(x') \neq C(x)$. In both attacks, the distance (the perturbation size) between x and x' should be kept below a threshold to

keep them look the same. Without loss of generality, we only consider the targeted attack in this work.

To calculate the value of perturbations, however, adversaries need to have complete access to the architecture as well as the weights of the target network, which we called a white-box attack. Conversely, for those scenarios that we cannot gain access to the target network are called a block-box setting. To deal with such black-box situations, methods have been proposed to train a substitute network that imitates the target classifier itself [23]; white-box attacks are then applied to the substitute network.

In context of the definition of adversarial examples above, attack strategies are designed to generate minimal perturbations and to apply these perturbations back to normal data to create adversarial examples. Such methods are referred to as perturbation-based methods in this work.

As the first proposed perturbation-based method, Szegedy et al. [28] formalizes the perturbation generation process as a optimization problem, using box-constrained L-BFGS to solve the function $f(x + r) = l$ for minimal $\|\mathbf{r}\|_2$, given the original image x , the target label l and the classifier mapping function f . From another perspective, Goodfellow et al. proposed Fast Gradient Sign Method (FGSM) [8], which is basically a gradient-descent-based algorithm aiming to increase the loss between prediction and ground-truth class for the purpose of misclassification. Based on FGSM, Kurakin et al. proposed BIM [13] to perform the gradient descent iteratively, with small perturbation at each step. Carlini and Wagner et al. [5] also formulated this task as an optimization problem, yet they provides several techniques to make the generating process more effective, including using logit loss instead of the softmax-cross-entropy loss and converting the target to $argtanh$ space to make optimization easier. Their Carlini and Wagner's methods (abbreviated as CW's method) are now the state-of-the-art.

Generative Adversarial Networks (GANs) GANs are characterized by their powerful capability in data generation. Ian Goodfellow first introduced this adversarial framework [7], and within a few years following researchers have showed state-of-the-art performance among a wide range of applications [22, 11, 32, 25, 6]. We adopt GANs here to

generate realistic adversarial examples that look alike real ones.

Moreover, to gain control over the visual class of our generated adversarial examples, we used GANs in a conditioning setting [21, 19]. Conditional GAN (cGAN)[19], as the first proposed approach of conditioning GANs, takes an extra label vector y as input in addition to the original random input z . The class label input controls the visual class of image, whereas z provides image variability. The discriminator learns to tell the fake images $G(y, z)$ from real images x , while the generator learns to deceive the discriminator. Auxiliary Classifier GAN (ACGAN) [21], on the other hand, not only takes an extra input, but also gives an extra class probability output. More explicitly, the discriminator of ACGAN predicts both the image source (real or fake) and the image class, which functions as an auxiliary classifier. This modification helps stabilize training and improve image generation quality. We implemented both methods in this work.

Other GAN-based Adversarial Example Generation Methods It is worth mentioning that concurrent work Natural GAN[31] and AdvGAN[29] also utilizes GAN's generative nature to produce adversarial examples.

In the training stage, Natural GAN trains a generator that maps random noise from the latent space to the data space and an inverter that does the opposite. During the example-generating stage, a real image is inverted into its latent representation by the inverter, and then small perturbations are added to the latent vector to create a perturbed latent representation. Finally, the perturbed representation is fed back to the generator to generate the image of adversarial example. This generating strategy is of untargeted attack since we could not control the direction toward which the prediction of classifier changes when perturbations are added in the latent space. Also, the generated examples would resemble their original image since they share close latent representations in latent space.

AdvGAN, on the other hand, generates adversarial examples by producing the adversarial perturbations with GAN. The generated perturbations are then mixed to normal images and then fed to the discriminator and the target classifier. The goal of its generator is to produce perturbations that are to be added to normal images to deceive the discriminator and also the classifier, while its discriminator learns to distinguish between

normal and adversarial examples. Though GAN is involved in its architecture, the core idea is still perturbation-based. It still produces adversarial examples that share similar appearance with existing images.

Unlike these two methods, our method takes the goal to generate brand-new adversarial instances to further expand the example space. This could be achieved by directly generating adversarial examples with GAN, and we also adopt a targeted strategy so that we can decide the class of images we want to attack. The details of our method will be presented in the next chapter.



Chapter 3

Adaptive GAN

3.1 Problem Setting

Most adversarial example generation strategies are based on developing an algorithm that extrapolates appropriate perturbations from benign data and adds them back onto those data to turn into adversarial examples. Fig. 3.1 illustrates the high level flow of such generation process. As a consequence of perturbing from existing data, the generated images resemble their unperturbed version, which is not rigorously proved yet obviously limit the adversarial diversity. To complement the vacancy of the data diversity, we propose to generate adversarial examples directly from conditional generative adversarial models.

We first denote the target classifier TC that adversaries aim to attack $TC : \mathbb{R}^m \rightarrow \mathbb{R}^n$, where m is the dimension of images and n is the dimension of the image class. Given a class label c and the attack target label t , our goal is to train a generator G that generates adversarial images $x' = G(c, z)$ such that x' corresponds to the class c and meanwhile causes misclassification that satisfies $TC(x') = t$.

3.2 Architecture

The design thinking behind our model is based on an intuitive idea: imposing an additional objective on a GAN-based generator so that the generator learns to produce image samples carrying characteristics that correspond to the objective. We would also like the objective

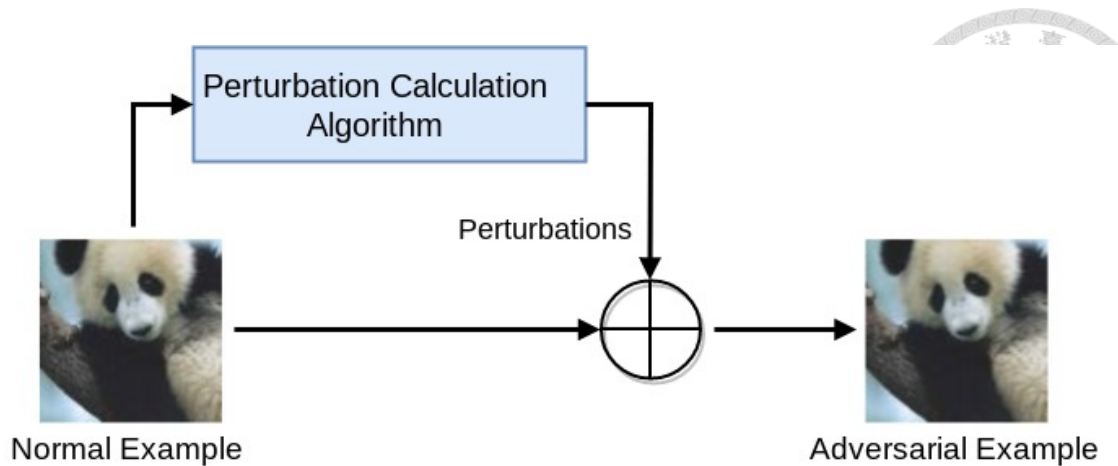


Figure 3.1: Illustration of a typical perturbation-based method. (The iconic panda images are adopted from the representative work of Goodfellow et al. [8])

to be less influential, keeping the generated images as realistic as those expected from typical GANs.

To achieve our goal of adversarial attacking, we thus set the objective to minimize a certain classifier’s loss w.r.t. certain attack target class. Then, realistic image samples produced by the generator might be able to cause misclassification to the classifier if trained properly.

Besides, controlling the class of the generated image samples is essentially needed for the targeted attack. Hence, we chose class-conditional GANs [19, 21] as our mainframe GAN network, and impose the generator with the additional objective in expect to produce class-specified images that can mislead the target classifier.

In order to realize the objective in practice, we designed our model as a class-conditional-GAN network concatenated by the classifier of the attack target, as shown in Fig. 3.2b. This architecture design allows us to obtain the attacking loss (which is calculated by feeding the generated images to the target classifier) and back-propagates it to the generator in a straightforward manner.

At the training stage, the generator takes class labels c , the random noise z and especially the label of attack target t as inputs (Fig. 3.2b), contrary to conventional class-conditional GANs which only take the first two (Fig. 3.2a). Then, it is trained to minimize both the adversarial loss and the attacking loss at the same time. Under this scenario, the generator acts like an adaptable animal that can adapt to external environmental con-

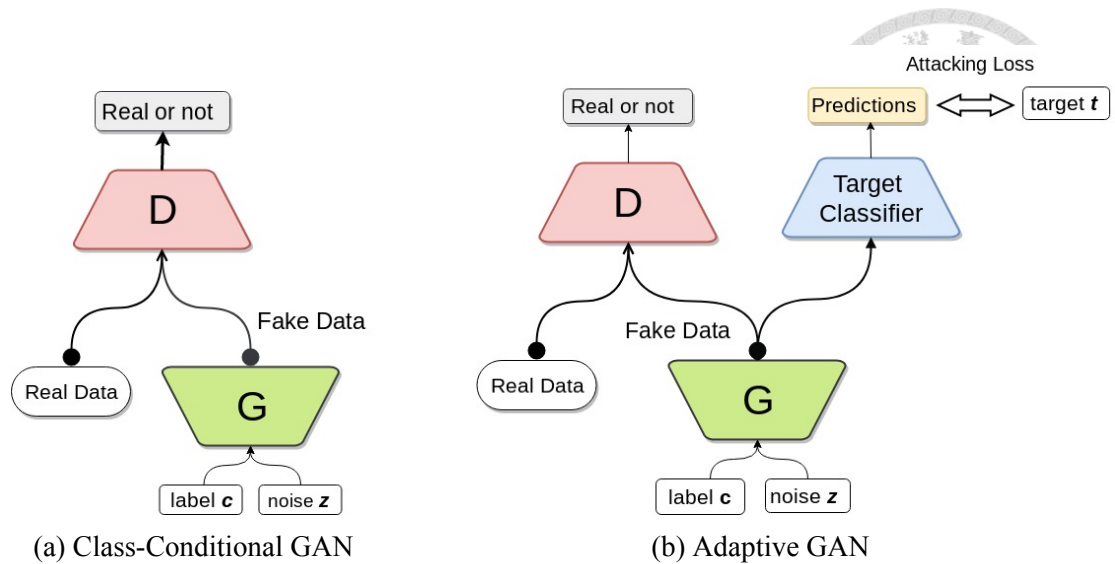


Figure 3.2: The high level architecture of (a) a typical conditional-GAN variant and (b) the Adaptive GAN.

straints. Actually, it is this behavior of adaptation that we call our framework Adaptive GAN.

The discriminator, on the other hand, is trained as conventional GANs to distinguish between fake and real images, while the target classifier stays weight-frozen. Note that our architecture is somehow similar to AdvGAN in [29], we think it’s a coincidence on vision, and each architecture actually functions differently from one another..

After the training stage, the generated images shall meet the following two goals: realistic for the discriminator (for humans), but misleading for the target classifier.

Note that since we need to calculate the attack loss during the training step, we must have access to the target classifier architecture as well as its weight (i.e. in the white-box settings), which is the basic limitation of Adaptive GAN. Yet, the recent work [23] to address the black-box attack problem by training substitute networks can be combined with our approach.

3.3 Objective

In correspondence to the attacking loss that the target classifier contributed, an additional objective L_{attack} is added accordingly:

$$L_{attack} = \frac{1}{n} \sum_{n=1}^N I(TC(x') \neq t) \cdot J(TC(x'), t) \quad (3.1)$$

where I is an indicator function that takes the value 1 iff the predicted label $TC(x')$ does not equal to the target label t and is zero otherwise, and J denotes the classification loss function used for the target classifier w.r.t. the attacking target t (e.g. cross-entropy).

Intuitively, L_{attack} may take the value of the expected classification loss for $TC(x)$ w.r.t. to the attack target t :

$$L_{attack} = E[J(TC(x'), t)] = \frac{1}{n} \sum_{n=1}^N J(TC(x'), t) \quad (3.2)$$

, which would directly reflect the attack effectiveness and is exactly Equation (3.1) without the indicator function.

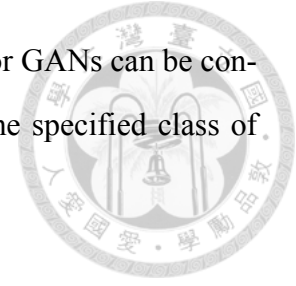
We observed that directly take Equation (3.2) as the attacking loss would make the generator fit too much on this objective. Namely, the generated images would end up in a strange shape between the given class and the target class. To alleviate this problem, we set an indicator function to mask out only the losses of images associated with the unsuccessful attack, and leave successful ones less influential. Thus, by optimizing on this adjusted objective, the generator is only trained to produce images to be misclassified as the target label, but not tuned to increase the confidence of misclassification. Therefore, the resulting adversarial examples would end up with little distortion that is just enough to fool the target classifier. An alternative choice for balancing the loss contribution between the successful attack and the unsuccessful attack may be using a weighted mask with weight $\beta < 1$. We will later discuss the impact of the indicator function in the experiment in Chapter 4.1.

Combining L_{attack} with L_{cGAN} , the loss function for the mainframe conditional GAN, yields the final objective function:

$$\min_G \max_D L_{cGAN}(G, D) + \alpha L_{attack}(G(z, c)) \quad (3.3)$$

where α serves as the controlling parameter to balance the influence between the adver-

serial loss and the attacking loss. Then the optimization procedure for GANs can be conducted to train the generator to meet both objectives: generating the specified class of images, while keeping them to be misclassified as the target type.



3.4 Turning Pre-trained Generator into Adversarial-example Generator

Instead of training from scratch, however, the "adapting procedure" can be directly applied to pre-trained class-conditional GANs. We found that this procedure takes only a few epochs to complete, since slight differences on the output pixels are enough to cause the desired misclassification.

We call it adaptive retraining that provides a way to instantly turn a pre-trained model into an adversarial-example generator within few epochs of retraining, which results in slight changes to the output images given the same class condition and the noise input. In addition, as a result of limited changes, the generative power of the original class-conditional GAN as well as the visual appearance of images produced by it are both reserved.

Note that there are situations in which the generated adversarial examples are not as natural and convincing as real-world images, we argue that it caused by insufficient ability of GAN to describe the dataset, since the retraining only has a limited impact on the output. A more advanced model design for GAN shall be able to address this problem.



Chapter 4

Experimental results

In this chapter, first we compare the effect of using different attacking loss functions in the training process. Then, we apply our method on two public adversarial attack challenges. For those experimental results hard to be quantitatively evaluated, we offer images for human assessment. Our code and models will be available upon publication.

4.1 The Effect of Masked Loss Function

We first discuss the effect of adding an indicator function when calculating the attacking loss. The indicator function serves as a mask to filter out only the loss of the unsuccessful attack (Equation 3.1), instead of the overall classification loss (Equation 3.2). We claimed that without masking, the generated adversarial examples would overfit the attack target, ending up in a shape between the original class and the target class.

To depict this scenario, we sampled "1" and "7" images targeted to be classified as "2" from models using Equation 3.2 and Equation 3.1 as the attacking loss respectively. As can be seen in Fig. 4.1a, the "1" and "7" images trained with the former would grow spots in the head and tail, which makes them more like the attack target "2". On the other hand, adopting the masked loss could prevent such overfitting, as shown in Fig. 4.1b.

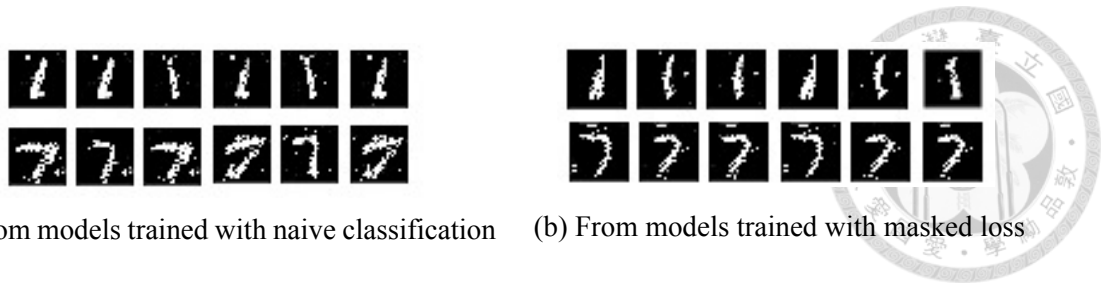


Figure 4.1: Comparison of the images sampled from models with attacking loss calculated using Equation 2 and Equation 1 respectively.

4.2 Adversarial Attack on MNIST

We then applied our attack strategy on MNIST Challenge [17], a public challenge proposed by [18] specifically for MNIST dataset [14]. The challenge offers a robust classifier trained with adversarial training based on CW’s method [18], and calls for adversarial attacks that could achieve the highest attack success rate. Attackers participating in challenge are allowed to perturb each pixel of target images by at most $\epsilon=0.3$ to prevent image distortion.

Apparently, non-perturbation-based methods (i.e. Adaptive GAN) is hard to follow this max-norm constraint, and, in fact, should not follow such constraint by design. This makes direct comparison between two kinds of approaches a bit of unfair. However, due to lack of direct evaluation metrics, we still chose to apply our attack strategy on the challenge to see the attack effectiveness on a robustly-trained network. Also, the performance of other perturbation-based methods on leaderboard are listed for benchmarking (the performance has been converted from accuracy to attack successful rate).

From Table 4.1, we can see the highest attack success rate that Adaptive-GAN contributed to the target classifier. We consider it as a reasonable result since there is no max-perturbation constraint on non-perturbation-based method. Yet the result still supports that our approach is effective in creating adversarial examples against robust classifiers.

To check the recognizability of adversarial examples, we randomly sampled 100 generated adversarial examples with 10 for each class of attack target in Fig. 4.2. Our mainframe class-conditional model is constructed based on conditional GAN [19] and DC-GAN [24].



Table 4.1: Attack success rate of the target classifier (MNIST Challenge)

| Attack | Attack success rate |
|---|---------------------|
| 100-step PGD on the cross-entropy loss with 50 random restart | 10.38% |
| 100-step PGD on the CW loss with 50 random restarts | 10.29% |
| 100-step PGD on the cross-entropy loss | 7.48% |
| 100-step PGD on the CW loss | 6.96% |
| FGSM on the cross-entropy loss | 3.64% |
| FGSM on the CW loss | 3.60% |
| *Adaptive GAN | 99.90% |

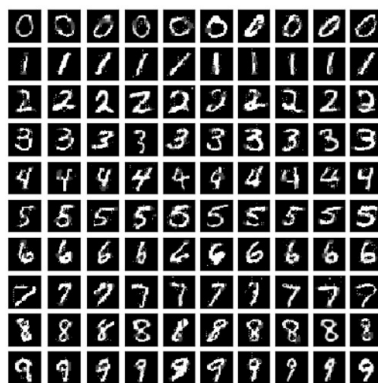


Figure 4.2: Adversarial examples generated from Adaptive GAN on MNIST with each row conditioned on different class label and each column conditioned on different attack target.

4.3 Adversarial Attack on CIFAR10

Next, we applied our attack strategy on CIFAR10 challenge [16], a public challenge proposed by [18] for CIFAR10 dataset [12]. The challenge provides a robust classifier trained with adversarial training based on CW’s method, and invite adversaries to submit adversarial examples against the classifier.

Instead of training an adversarial-example generator from scratch, we used adaptive retraining on top of a pre-trained model this time. By starting from a pre-trained model, we can then shorten the overall training time since CIFAR10 is a more time-consuming dataset for build GANs. Also, we can take the images generated from the pre-trained model as ground truth in some sense, to verify the claim the re-training would not cause

too much change to the original conditional GAN’s output. A pair of the images sampled from the original and retrained models is shown in Figure 4.3.

Table 4.2 shows the attack success rate to the target classifier under different attacks, including Adaptive GAN and other perturbation-based methods on leaderboard (the performance has been converted from accuracy to successful rate). We can see that Adaptive-GAN outperforms others. Again, we have to emphasize that unlike other perturbation-based strategies, Adaptive GAN cannot limit the size of perturbation by nature, so it is not surprising that it achieves higher attack success rate than other methods. Yet the result still suggests that our method is competitive with other approaches.

As far as image quality is concerned, we have to admit that some of the generated adversarial examples are not convincing enough as those generated from perturbation-based methods. We think it is a practical issue for all generative models. Since we only provides a retraining framework which causes little change to the output of the pre-trained generative model. And, to the best of our effort, we have constructed our model based on ACGAN[21] and DCGAN[24] together with state-of-the-art techniques including batch normalization [10], leaky relu as activation function, label smoothing [26] and minibatch discrimination [26] in this experiment. We think to improve the overall quality of images generated from GAN is beyond the scope of this work, and await for further research.

Table 4.2: Accuracy of the target classifier (CIFAR10 Challenge)

| Attack | Attack success rate |
|---------------------------------------|----------------------------|
| 20-step PGD on the cross-entropy loss | 52.94% |
| 20-step PGD on the CW loss | 52.24% |
| FGSM on the CW loss | 45.08% |
| FGSM on the cross-entropy loss | 44.45% |
| *Adaptive GAN | 61.90% |

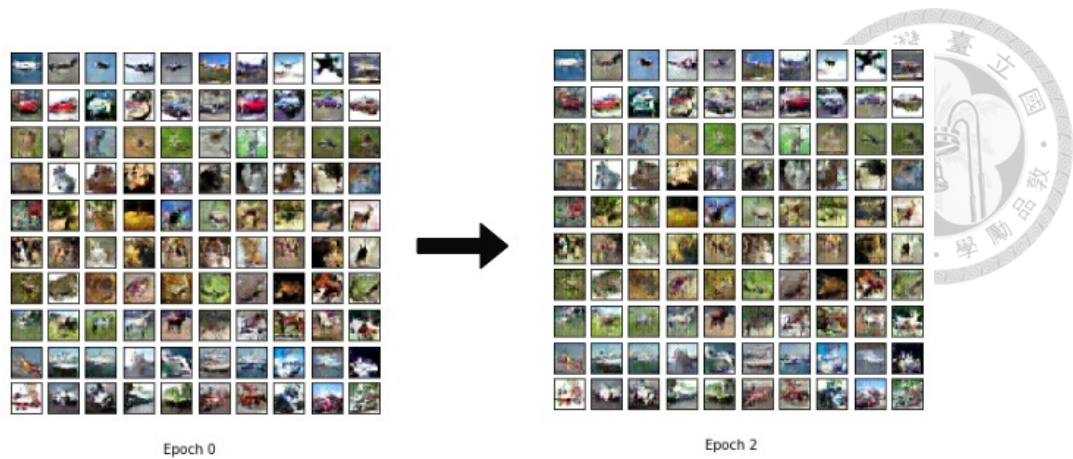


Figure 4.3: Adversarial examples sampled from the process of adaptive retraining on CIFAR10 Challenge, where the accuracy has been reduced by 12% after two epochs of re-training.

4.4 Cross-domain attacks

Adaptive GAN can also be applied to cross-domain attacks: for example, Logo Attacks or Traffic Sign Attacks [27]. We take a scenario of Traffic Sign Attack as example, and plot our attack images in Fig. 4.4.

In the attack scenario, we want to misguide an autonomous vehicle by faking non-traffic-sign images that would be interpreted by the vehicle as some traffic signs, then causing misbehavior of the vehicle. We applied adaptive retraining on a GAN pre-trained with CIFAR-10 dataset and a classifier trained to classify traffic signs. After the retraining process, our generator is capable of generating images of transportation or animals to be classified as traffic signs. In Fig. 4.4 images of a car and a horse are presented, and both of them are interpreted as a "watch for pedestrians sign" by the classifier serving as simulated computer vision system of autonomous vehicle in our experiment.

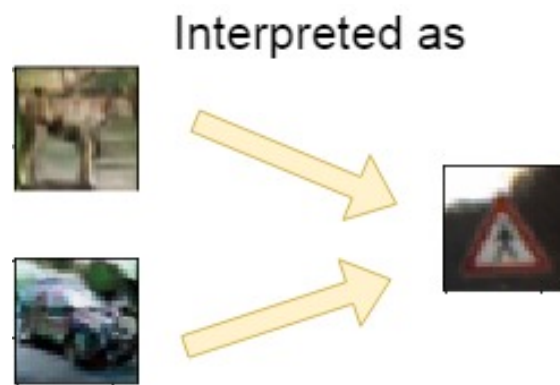


Figure 4.4: Cross-domain Adversarial examples. An image of horse and an image of car to be misinterpreted as a "watch for pedestrians sign" by some classifier on an autonomous vehicle are generated with Adaptive GAN.



Chapter 5

Conclusion

In this work, we study the perturbation-based nature for most adversarial-example generating methods that create examples mainly by perturbing existing data which limits the data space in some manner. We then engage to complement these perturbation-based strategies by proposing Adaptive GAN to directly produce examples with a modified conditional-GAN framework. Our method expands the adversarial-example subspace, since images will not have to take the appearance of existing data. We then experimented on MNIST and CIFAR10 to verify our original thinking.

A clear downside for Adaptive GAN is that it needs to be trained several times for different targets and be retrained every time when the defender modifies its model. However, we have showed in experiments that our method can apply to pre-trained GANs, turning them into adversarial-example generators. A re-training process to slightly modify the pre-trained model takes only a few epochs, which means the issue of training time would not limit our method.

We think the practical limitation to Adaptive GAN is that the perceptual quality of the generated adversarial examples depends heavily on the generative power of its mainframe GAN model. Joint effort across several GAN training techniques is required to generate acceptable quality of adversarial examples. And, in fact, the quality may not be as good when compared to those generated from traditional perturbation-based methods. In general, we still think Adaptive GAN is a promising alternative to current attack strategies since it provides a generative framework to create native adversarial examples which is

irreplaceable to other methods.




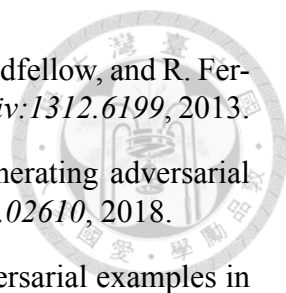




Bibliography

- [1] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.
- [2] B. Biggio, G. Fumera, and F. Roli. Security evaluation of pattern classifiers under attack. *IEEE transactions on knowledge and data engineering*, 26(4):984–996, 2014.
- [3] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.
- [4] J. Bradshaw, A. G. d. G. Matthews, and Z. Ghahramani. Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks. *arXiv preprint arXiv:1707.02476*, 2017.
- [5] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.
- [6] C. Donahue, J. McAuley, and M. Puckette. Synthesizing audio with generative adversarial networks. *arXiv preprint arXiv:1802.04208*, 2018.
- [7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [8] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- [9] C. Guo, M. Rana, M. Cissé, and L. van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.
- [10] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- [11] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks.
- [12] A. Krizhevsky, I. Sutskever, and G. Hinton. The cifar-10 dataset. : <http://www.cs.toronto.edu/kriz/cifar.html>, 2014.
- [13] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

- 
- [14] Y. LeCun. The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>, 1998.
- [15] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, M. E. Houle, G. Schoenebeck, D. Song, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.
- [16] A. Madry, A. Makelov, Schmidt, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. CIFAR10 Adversarial Examples Challenge. https://github.com/MadryLab/cifar10_challenge. Accessed: 2018-05-06.
- [17] A. Madry, A. Makelov, Schmidt, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. MNIST Adversarial Examples Challenge. https://github.com/MadryLab/mnist_challenge. Accessed: 2018-05-06.
- [18] 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.
- [19] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- [20] S. M. Moosavi Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, number EPFL-CONF-218057, 2016.
- [21] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.
- [22] A. Osokin, A. Chessel, R. E. C. Salas, and F. Vaggi. Gans for biological image synthesis. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2252–2261. IEEE, 2017.
- [23] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.
- [24] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- [25] S. Rajeswar, S. Subramanian, F. Dutil, C. Pal, and A. Courville. Adversarial generation of natural language. *arXiv preprint arXiv:1705.10929*, 2017.
- [26] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.
- [27] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal. Darts: Deceiving autonomous cars with toxic signs. *arXiv preprint arXiv:1802.06430*, 2018.

- 
- [28] 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.
- [29] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.
- [30] W. Xu, D. Evans, and Y. Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.
- [31] Z. Zhao, D. Dua, and S. Singh. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*, 2017.
- [32] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.