AdvCGAN: An Elastic and Covert Adversarial Examples Generating Framework

Baoli Wang[1,2], Xinxin Fan[1], Quanliang Jing[1,2], Haining Tan[1,2], Jingping Bi[1]
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China[1]
University of Chinese Academy of Sciences, China[2]
{wangbaoli, fanxinxin, jingquanliang, tanhaining, bjp}@ict.ac.cn

***Abstract*—Recently, a new methodology using generative adversarial network (GAN) has been proposed to produce adversarial examples, which breaks the limitations of the previous methods dependent on different norm-levels. It can efficiently generate perturbations for any instance once the generator is trained, arising from the learning to approximate the distribution of real instances. However, there are still two shortcomings for this category of GAN-based method: i) the predicted label in attacking stage totally depend on a fixed or randomly-chosen label in training stage, which cannot tackle the elasticity problem on how to elastically produce adversarial example with any arbitrarily-assigned label in targeted attack scene when the generator has finished training; and ii) it only considering the produced adversarial example is as close as the real instances, which cannot guarantee the generated adversarial example is visually indistinguishable from its corresponding original instance perceptually. The aboved two disadvantages make this kind of method lack of flexibility and covertness. To circumvent these two predicaments, we in this paper propose a simple and easy-to-use adversarial example generating framework AdvCGAN through training a conditional generative adversarial network under the co-consideration on the similarities in data distributions and the image labels between the adversarial examples and the original instances to be imperceptible to humans. Concretely, our proposed AdvCGAN trains the conditional GAN with both image data and label (normal and attack) information, by which the generator can utilizing the guidance of label information to appropriately produce the adversarial example with any specific label in attacking stage. Extensive experiments using the commonly used MNIST and CIFAR-10 datasets show that our proposed AdvCGAN significantly outperforms other methods in terms of multi-facet evaluation. The results exhibit that our AdvCGAN can elastically produce more realistic adversarial examples with any arbitrarily-assigned attack label and achieve higher attack accuracy, especially in targeted attack.**

I. Introduction

The recent breakthrough in deep learning has boosted the revival of machine learning and artificial intelligence in many applications, such as autonomous driving, medical diagnostics, smart city, object detection, etc. However, some existing researchers have found that the radical deep neural networks (DNNs) suffered from multi-facet vulnerabilities. They are surprisingly susceptible to adversarial attacks in the form of intended small perturbations to the original examples (images) [1]. These carefully fabricated examples by intentional adversaries are so-called adversarial examples, which are usually indistinguishable from the original image in human vision, and may cause the inference models to make catastrophic mistakes in different applications, e.g. the realistic traffic accident taken place in autonomous driving [2], the attacker evading from face recognition system or impersonating other individuals [3].

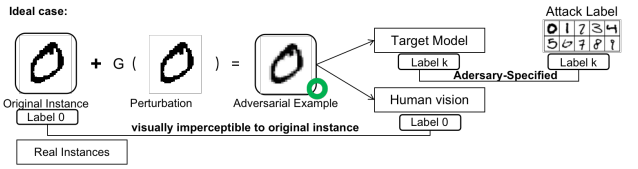
Jingping Bi (bjp@ict.ac.cn) is the corresponding author

The adversarial attacks can generate diversities of adversarial examples, and be broadly classified into two categories with regard to whether to assign the attack labels [1]: targeted attack and untargeted attack. The former aims to fool a target (inference) model into falsely predicting a specific label with high confidence for the victim example, while the latter predicts the label of the adversarial example as an arbitrary one except the correct label.

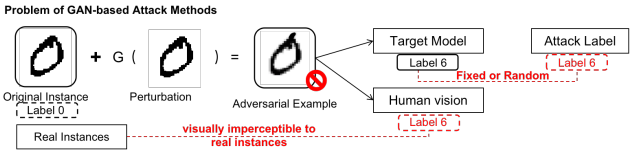
In many adversarial scenarios, such as surveillance, face recognition [3] or autonomous driving [2], targeted attack is in general more severe and meaningful than untargeted attack, along with more cost as well. For instance, the work advPattern [4], stated that the targeted attack to identify a victim person as another assigned one need make particular perturbation in a surveillance system. On the other hand, the existing works [3], [5]–[7] also show that if the attacker persistently fool an object or segment detector with targeted attack, compared to untargeted scene, the system is hard to detect the attack and causing traffic accidents. Accordingly, the complexity of the targeted attack is much hard than that of the untargeted attack.

In this paper, our investigation mainly focuses on targeted attack, despite we also show the strength of our proposed method in untargeted attack scene. Upon the extensive studying on adversarial examples, we think a decent and high-quality adversarial example ought to preserve two basic prerequisites to achieve attack success and covert: i) the predicted label by the target model should be same as pre-assigned label by the adversary, meanwhile, the label can be assigned arbitrarily and elastically; ii) the perturbed example should be indistinguishable in human vision to the original example as input, rather than simply similar to the real examples. The first endows adversaries sufficient attack capability to launch flexible and powerful perturbation to deceive target model, and the second guarantees the non-perception of perturbation of adversarial example in human vision even comparing with the original example.

To date, a corpus of works [7]–[15] has been proposed to study how to generate adversarial examples from two popular techniques, namely gradient-based methods (e.g. FGSM



(a) The Ideal Adversarial Example Case



(b) The Problem of Adversarial Example Case

Fig. 1: The sketch of adversarial example

[16], BIM [17], ILCM [17]) and optimization-based methods (e.g. CW [18], L-BFGS [19]). All these works rely on the optimization schemes with simple pixel space metrics, such as L_∞ distance from a benign image. Differently, a GAN-based method AdvGAN [20] was recently proposed to train a generator to make the adversarial examples to be more similar to real examples in data distribution. It provides a new way to solve the limitation of previous methods which just rely on the different norm-levels, and more perceptually realistic adversarial examples, comparing with real examples, could be generated efficiently.

However, refer to the aforementioned covert and elasticity, AdvGAN exhibits two defects in the targeted attack scenario: First, the label of targeted attack is constrained to a fixed one or randomly-chosen one in training stage, which induces the adversarial example to be predicted as that fixed or randomly-chosen label with a higher probability; Second, it only considers the adversarial example's data distribution is close to real examples, but ignores whether the adversarial example is visually indistinguishable when comparing with its corresponding original instance. In this scene, assume a dataset with n-classes, AdvGAN must train out n sets of model parameters in the training course, and correspondingly select one set for a particular target attack label to perform the targeted attack. To sum up, AdvGAN lacks covertness in the generated adversarial examples and elasticity to handle the labels in attacking stage.

To clearly explain the drawbacks of AdvGAN, imagine the scenario: the $\{x_0 - x_9\}$ are real image sets used for training and $[0-9]$ are the label classes, our goal is to produce an adversarial example $x_0^{'}$ for an clean image x_0 with original label digit 0, which enforces the target model to classify it as any assigned label digit k other than the correct label 0 by the adversary.

As exemplified in Fig. 1b, AdvGAN has two facets of disadvantages: i) firstly if the generator was trained with a fixed label digit 6 in training stage, it is more likely generate adversarial examples with label digit 6, which cannot be any other assigned label digit k in attacking stage. Alternatively the generator was trained with a randomly-chosen label in training stage, the predicted label of the perturbed example will be same to the adversary-assigned one with a lower probability; ii) secondly, if we assume an extreme situation that we want make the perturbed image $x_0^{'}$ be misclassified as label digit 6, then the generator may directly produce an adversarial example $x_0^{'}$, which is almost same as a clean image x_6. There is no doubt that it will be classified as label 6, but the perturbed image may be no more visually indistinguishable from its corresponding original instance perceptually. That it to say, although AdvGAN can enforce the generated adversarial example to be similar to the real examples, it may not look indistinguishable in human vision when comparing to its corresponding original example.

To guarantee the attack success and covertness of the adversarial examples simultaneously, we propose a simple and easy-to-use approach AdvCGAN through resorting to both data and label information jointly to train the discriminator and the generator with conditional generative adversarial network (CGAN). Our proposed AdvCGAN can effectively match the two aforementioned prerequisites to generate high-quality adversarial examples successfully and covertly.

Concretely, our work involves three contributions.

- We propose a simple and easy-to-use framework to elastically generate adversarial examples for arbitrarily assigned labels by adversaries towards targeted attacks, namely, only need one-time training for multiple targeted attacks in our AdvCGAN, compared to multiple trainings to achieve the same goal in recently proposed work AdvGAN.
- Different from the existing works that only depending on the usage of date (image) information for adversarial attacks, we in this paper additionally utilize the label information as the input both in training and attacking stage, in this way our AdvCGAN can suffice the attack success and covertness at the same time.
- To validate our proposed AdvCGAN, we perform extensive experiments using the datasets MNIST and CIFAR-10 with/without defense mechanisms. The experimental results show that our AdvCGAN not only elastically generates adversarial example with the flexibly-assigned label, but also makes the adversarial example indistinguishable from its counterpart original example to conceal the perturbation from human-vision perspective. Our work significantly outperforms other baselines, such as AdvGAN, FGSM, L-BFGS.

II. Related Work

A. Adversarial Attacks

Currently, the adversarial attacks can be regularly classified as three categories as follows.

a) Optim-based adversarial attack method: Szegedy et al. [19] first demonstrated the existence of small perturbations to the images, i.e., the perturbed images could fool deep learning models into misclassification and they proposed an optim-based method L-BFGS to produce adversarial examples. Carlini and Wagner [18] applied more complex objective functions to optimize the adversarial perturbations with respect

to several constraints like L_0, L_2, L_∞ distance metrics. Many defense methods fail in the face of using CW loss function. This kind of optim-based methods achieves high attack accuracy in many senses for both targeted attack and untargeted attack, however, the optimization process is time costly and only can process one certain instance each time, which makes it hard to use for massive instances in practice.

b) Gradient-based adversarial attack method: Goodfellow et al. [16] proposed the fast gradient sign method (FGSM), it applies a first-order approximation of the loss function to construct adversarial examples, leading to the consequence that gradient direction moves towards attack label's gradient direction. Madry, Kurakin et al. [17] extended FGSM method by iteratively computing the adversarial perturbation, namely so-called Basic Iterative Method (BIM) and Iterative Least-likely Class Method (ILCM). Despite this kind of gradient-based attack methods computes very fast, however, the perturbation usually looks distinguishable obviously and unacceptable for human vision, furthermore, the attack efficiency sometimes is not stable.

c) GAN-based adversarial attack method: Different from the above two techniques, GAN is another fashion to generate adversarial examples. Xiao et al. [20] proposed AdvGAN method to generate adversarial examples using a GAN-based Network. Compared to prior works, it aims to produce such output results that are not only able to mislead the target learning models but also visually realistic. Concretely the adversarial loss in AdvGAN is expressed as bellow:

$$\mathcal{L}_{GAN} = \mathbb{E}_x \log \mathcal{D}(x) + \mathbb{E}_x \log(1 - \mathcal{D}(x + \mathcal{G}(x))). \quad (1)$$

where $\mathbb{E}_{(\cdot)}$ means $\cdot$ is sampled from a given distribution $p_{data}(\cdot)$.

The loss function to fool the target model f under a targeted attack is defined as:

$$\mathcal{L}^f_{adv} = \mathbb{E}_x l_f(x + \mathcal{G}(x), t). \quad (2)$$

where t is the target class, which is a fixed or randomly-chosen one from label sets, and l_f denotes the loss function (e.g., cross-entropy loss) used to train the original model f.

Subsequently, on the basis of AdvGAN, some works are proposed, for examples, Surgan Jandial et al. [21] proposed AdvGAN++ to speed up training time through utilizing the features of image instead of the image itself as the generator's input. In addition, taking into account the noise and source label as the generator's inputs. Yu et al. [22] proposed two generative models to produce adaptive attack instances. The attack label information was used only for the target model but not the GAN's discriminator and generator. These methods have the capability to enforce the generated adversarial examples to be as similar as the real instances through resorting to the similarity constraint between real instances and adversarial examples in data distribution. Differently, our method additionally considers the guidance of label information in both training and attacking stages (condition 1 and 2 in Figure 1a), as well as regards whether the adversarial examples as close as its corresponding original instance (condition3 in Figure 1a). Through jointly referring to the image data and the label information (normal and attack), our AdvCGAN makes the generator more elastic to produce adversarial examples with any arbitrarily-assigned labels. Therefore, our method can significantly improve the attack efficiency in the adversarial attack especially in targeted adversarial attack scenes.

B. Conditional Generative Adversarial Networks

Goodfellow et al. [23] originally introduced the concept of Generative adversarial networks (GANs), which consists of two neural networks: a generator G and a discriminator D. In the training phase, the generator G and the discriminator D are typically learned in an adversarial fashion using actual input data samples x and random vectors z. While the generator G learns to generate outputs $G(z)$ that have a distribution similar to that of x, the discriminator D learns to discriminate between "real" samples x and "fake" samples $G(z)$. Next, Mirza [24] proposed an conditional version of generative adversarial networks to further improve the flexibility of the synthesis results. It can produce highly stochastic output with specific label through capturing the full entropy of the conditional distributions.

Nevertheless, the above works aim to produce examples similar to real ones, not to attack the target models. We in this paper adopt the idea of conditional GANs to produce adversarial examples with any arbitrarily-assigned labels that are able to fool the targeted models both in targeted attack and untargeted attack scenes. In addition, our work also fills in the deficiency of AdvGAN in targeted attack scene.

III. Methodology

A. Problem Definition

Given a target (inference) model f that has the capability to accurately map the image x sampled from a distribution p_{data} to its corresponding label y. The goal of the targeted attack launched by an adversary is to generate the adversarial example x_{adv} for a regular examples x by adding the perturbation $\mathcal{G}(x, y^{'})$ via a trained generator $\mathcal{G}$, wherein the generator takes the original image x and the attack label y' as its input. The predicted label of adversarial example x_{adv} by the target model will be the attack label $y^{'}$ with higher probability, also with the condition that the constant should be limited regularly by a certain value ϵ in p-norm, namely,

$$x_{adv} = x + \mathcal{G}(x, y^{'}). \quad (3)$$

s.t.

$$f(x_{adv}) = y^{'} \neq y. \quad (4)$$

$$||x - x_{adv}||_p < \epsilon. \quad (5)$$

B. Generating Adversarial Examples

Fig. 2 shows the overall framework of our proposed AdvCGAN method, it mainly contains three main components: a generator $\mathcal{G}$, a discriminator $\mathcal{D}$, and a target neural network f. The generator $\mathcal{G}$ takes both the original instance x and its corresponding attack label $y^{'}$ as the input to generate

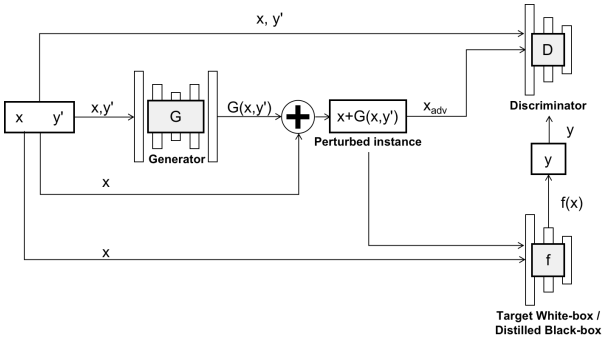


Fig. 2: AdvCGAN Architecture

perturbation $\mathcal{G}(x, y^{'})$ for x. Then, the generated perturbation will be added to the original instance x to make the perturbed example $x + \mathcal{G}(x, y^{'})$, i.e. adversarial example, subsequently, it will be sent to both of the discriminator $\mathcal{D}$ and the target model f respectively. For the discriminator $\mathcal{D}$, it tries to select the ideal adversarial examples to meet the covertness property from the combination of data and label $(X|X^{'}, Y|Y^{'})$, i.e. $(X, Y), (X^{'}, Y), (X, Y^{'}), (X^{'}, Y^{'})$, with respect to both the data distribution similarity and the label of perturbed and original example belongs to the same class visually. The target model f archives two manipulations: i) the target model f takes the x as input and outputs its predicted label; and ii) it also calculates the probability the generated adversarial example $x + \mathcal{G}(x, y^{'})$ belongs to the targeted attack label y'.

To covertly generate adversarial examples to deceive human envision to the greatest extent, we define the aggregate loss function as follows:

$$\begin{aligned}\mathcal{L}_{CGAN}(\mathcal{G}, \mathcal{D}) =& \mathbb{E}_{x,y} \log \mathcal{D}(x, y) \\ &+ \mathbb{E}_{x,y^{'}} \log(1 - \mathcal{D}(x, y^{'})) \\ &+ \mathbb{E}_{x,y^{'}} \log(1 - \mathcal{D}(x + \mathcal{G}(x, y^{'}), y^{'})) \\ &+ \mathbb{E}_{x,y,y^{'}} \log(1 - \mathcal{D}(x + \mathcal{G}(x, y^{'}), y)).\end{aligned} \tag{6}$$

where the discriminator $\mathcal{D}$ aims to filter the high-quality (indistinguishable) adversarial examples with the co-consideration in both data distribution and label information.

Given the fact that the discriminator $\mathcal{D}$ receives image data X and label Y as the input information, furtherly, the X again includes real image x and generated adversarial example image x'; the Y similarly involves correct label y and attack label y'. therefore, our defined loss function naturally encompasses four types of inputs: (x, y), (x_{adv}, y), $(x, y^{'})$, $(x_{adv}, y^{'})$. Obviously, the ideal adversarial example is anticipated to be(x, y), which keeps not only close to the real instance in data distribution but also consistent to the correct label that the discriminator $\mathcal{D}$ encourages to produce. Note that in our method the four types inputs offer positive and negative samples for the discriminator $\mathcal{D}$, which is not realized in previous methods AdvGAN and AdvGAN++, they only contain two types of inputs: (x) and (x_{adv}), which can only ensure the adversarial example's data distribution is close to the real examples, but ignore whether the generated adversarial example is visually indistinguishable with its corresponding original instance.

Furthermore, to routinely fool the target model f, the generator ought to minimize $f_{y'}(x_{adv})$, which represents the $softmax$ probability of the adversarial example x_{adv} belonging to class $y^{'}$. We use a $\mathcal{L}_{adv}^{f}$ loss function to compute it, and present mathematically as:

$$\mathcal{L}_{adv}^{f} = \mathbb{E}_x l_f(x + \mathcal{G}(x, y^{'}), y^{'}). \tag{7}$$

where $y^{'}$ denotes the attack label and l_f indicates the loss function like cross-entropy or other loss functions. We use cross-entropy loss functions in the paper. The $y^{'}$ is equal to y^{target}, which is an adversary-specified attack label in the targeted attack scene ($y^{'} = y^{target} \in Y$). In the untargeted attack scene, different from AdvGAN, which performs adversarial attacks by maximizing the distance between the prediction and the ground truth, we define a label translation function to get a random label from the remaining-label set ($y^{'} \in (Y - y)$). This formulation can be unified in both untargeted attack and targeted attack scenes. In addition, in order to improve the capability of generalization of the generative model, we set the attack label $y^{'}$ as untargeted attack label in the training stage (i.e. allow to randomly-chosen any label other than the correct label for training) and targeted (adversary-specified) label in testing (attacking) stage in this our experiment. The detailed settings of untargeted attack label will be described in section IV.

Moreover, in order to adequately bound the magnitude of perturbation, we also use a $\mathcal{L}_{pert}$ loss to minimize the difference between the adversarial example x_{adv} and its corresponding original instance x on p-norm as:

$$\mathcal{L}_{pert} = \mathbb{E}_x \max(0, ||\mathcal{G}(x, y^{'})||_2 - c). \tag{8}$$

where c is a bound specified by the adversary, as a rule of thumb, we set c as 0.3.

Then, the total loss function can be expressed as:

$$\mathcal{L}(\mathcal{G}, \mathcal{D}) = \mathcal{L}_{CGAN} + \alpha \mathcal{L}_{adv}^{f} + \beta \mathcal{L}_{pert}. \tag{9}$$

where α and β control the relative importance of each objective. By solving the min-max game $\arg\min_{\mathcal{G}} \max_{\mathcal{D}} \mathcal{L}(\mathcal{G}, \mathcal{D})$, we can obtain the optimal parameters for $\mathcal{G}$ and $\mathcal{D}$. The generator $\mathcal{G}$ will be able to produce the ideal adversarial examples with covertness and elasticity under assigned labels.

The training procedure ensures that the adversarial image x_{adv} produced by our proposed AdvCGAN is close to its corresponding original instance x in data distribution and its label judged by the target model f is more likely to be the adversary-specified attack label $y^{'}$. The detailed algorithm of AdvCGAN is depicted in Algorithm 1.

IV. Experiment Evaluation

A. Datasets and Target Models

We perform experiments to verify the efficiency of our AdvCGAN using two commonly used datasets: MNIST [1] and CIFAR-10 [2]. The former has 10 types of label classes, 60k

[1] http://yann.lecun.com/exdb/mnist/

[2] http://www.cs.toronto.edu/~kriz/cifar.html

Algorithm 1: AdvCGAN training

Input: X: $\{x^{(1)}, ..., x^{(n)}\}$, dataset for training, which includes n instances

Output: optimal parameters for $\mathcal{G}$ and $\mathcal{D}$

for *number of training iterations* **do**

- Sample a mini-batch of m examples $\{x^{(1)}, ..., x^{(m)}\}$ from training data set X;
- Sample a mini-batch of m original labels $\{y^{(1)}, ..., y^{(m)}\}$ from training label set Y;
- Generate a mini-batch of m attack labels $y^{'} = \{y^{(1)}_{adv}, ..., y^{(m)}_{adv}\}$ using label translation function and takes original labels as input. $Y^{'} = T(Y)$;
- Update the discriminator $\mathcal{D}$ by ascending its stochastic gradient: $\nabla\theta_{\mathcal{D}} = \frac{1}{m}\sum_{i=1}^{m} \log \mathcal{D}(x^{(i)}, y^{(i)}) + \log(1 - \mathcal{D}(x^{(i)}, y^{'(i)})) + \log(1 - \mathcal{D}(x^{(i)} + \mathcal{G}(x^{(i)}, y^{'(i)}), y^{'(i)})) + \log(1 - \mathcal{D}(x^{(i)} + \mathcal{G}(x^{(i)}, y^{'(i)}), y^{(i)}))$;
- Update the generator $\mathcal{G}$ by descending its stochastic gradient: $\nabla\theta_{\mathcal{G}} = \frac{1}{m}\sum_{i=1}^{m} \log \mathcal{D}(x^{(i)} + \mathcal{G}(x^{(i)}, y^{'(i)}), y^{'(i)}) + ||\mathcal{G}(x, y^{'})||_2 + l_f(x^{(i)} + \mathcal{G}(x^{(i)}, y^{'(i)}), y^{'(i)})$;

end

images for training and 10k images for testing, and we employ LeNet as the target model. The latter involves 10 types of label classes, 50k images for training and 10k images for testing, wherein ResNet and VGGNet are utilized as the target model. We train our AdvCGAN using training set and perform efficiency evaluations using test set.

B. Implementation details

Besides our proposed AdvCGAN, there are several baselines methods (L-BFGS, FGSM and AdvGAN), we clarify their configurations respectively. In L-BFGS attack experiment, we set the number of max-iteration as 200, and learning rate from 0.01 to 0.0001 (start from 100 and 150 iteration with 0.001 and 0.0001). In the FGSM attack experiment, we set ϵ (pixel perturbation) from 0.05 to 0.3. In the AdvGAN and AdvCGAN attack experiment, we specify an encoder and decoder based architecture to realize the functions of discriminator $\mathcal{D}$ and generator $\mathcal{G}$ respectively, and set the training max-iteration as 60. The optimizer Adam equipped with learning rate from 0.01 to 0.0001 is used for optimizing generator and discriminator under the settings α = 1 and β = 10. Furthermore, for the fair comparison, all adversarial examples are generated by different attack methods under an L_∞ bound of 0.3. In addition, we shuffle the datasets in the training procedure, while keeping the data's order in the testing procedure.

As we know, the adversarial training can indeed enhance the robustness and attack resilience of target model, we here employ two types of adversarial training-oriented countermeasures as our defense strategies: FGSM-based adversarial training and BIM-based adversarial training. In the adversarial training procedure, we retrain the target model with original training dataset and the produced adversarial examples together to guarantee the target model can still predict clean instances with correct labels in high accuracy. In the training course of our proposed AdvCGAN, we use the costumed translation function alike the random permutation function in DAG [7], to elastically and arbitrarily generate attack label $y^{'}$ for both targeted and untargeted attack, which means $y^{'}$ can be any other label but not the original one corresponding to the original example. In this way, our AdvCGAN can validly improve the generator's generalization ability. In the testing procedure, we set the attack label as an adversary-specidied one for each original label.

TABLE I: Label sketch of original and targeted instances

Label	Training	Testing
Original	y	y
Targeted	$y^{'} = random(Y - y)$	$y^{'} = 9 - y$
Example	$3 \rightarrow random([1, 2, 4, 5, 6, 7, 8, 9, 0])$	$3 \rightarrow 6$

We regard targeted attack to be successful when the adversarial example is predicted same as the adversary-specified label, and untargeted attack to be successful when the adversarial example is predicted as any other label but not the correct one. Although our work focuses on the targeted attack in our experiment, we calculate attack success rate to extend to both of the two scenes. Concretely, we define the attack success rate (ASR) as:

$$ASR = N_s/N. \tag{10}$$

where N is the total number of adversarial examples for testing dataset and N_s refers to the number of adversarial examples which are classified into adversarial-specified label by the target model in targeted attack or any other but not the original label in untargeted attack. The sketch of the original and attack label is shown in Table I.

C. Performance Evaluation

a) Performance under no defense: We compare the attack success rates (include targeted and untargeted attack scenes) of adversarial examples generated by FGSM, L-BFGS, AdvGAN and AdvCGAN on the target models without using any defense strategy on them.

We first compare our AdvCGAN with FGSM, L-BFGS and AdvGAN using targeted and untargeted attacks under no defense strategies. The ASR results in Table II show that L-BFGS and AdvCGAN achieve higher attack success rates than AdvGAN and FGSM, especially in targeted attack scene both on the two datasets MNIST and CIFAR-10 datasets. For untargeted attack scene, all four methods can achieve ASR more than 80% and AdvCGAN and L-BFGS almost totally fool the classifier (target model) with ASR more than 95%. The methods of FGSM and AdvGAN can also launch ASR with greater than 80%. For targeted attack scene, FGSM compromise the images with a certain perturbation ($\epsilon = 0.3$), and the ASR is only at the low-level, i.e. 16.56%, 28.07%

TABLE II: ASR (%) under no defense

DataSet	Model	Type	FGSM	L-BFGS	AdvGAN	AdvCGAN
MNIST	LeNet	Targeted	16.56	90.91	14.15	**91.15**
		Untargeted	80.45	95.65	87.13	**97.28**
CIFAR-10	ResNet	Targeted	28.07	99.31	12.56	**99.66**
		Untargeted	91.65	99.69	83.54	**99.84**
	VGGNet	Targeted	14.17	86.85	11.79	**89.97**
		Untargeted	80.83	98.67	90.73	**99.95**

and 14.17% over three target models/networks LeNet, ResNet and VGGNet. Compared to FGSM, AdvGAN under targeted attack performs more poorly, namely 14.15%, 12.56% and 11.79% over the three target networks, this is because of the generator it doesn't know which label is the targeted label in the training stage. Hence it cannot use label information efficiently, as a result, it only can produce adversarial examples that are close to real ones but cannot achieve high attack success rate. However, L-BFGS under targeted attack behaves quite well, since it can compromise each image with different perturbation, which on the other hand also results in time consuming and depends on many hyper parameters.

Our AdvCGAN under targeted attack has the highest ASR compared to the other three methods, this is because the generator uses not only the image information but also the attack and original label information in the training stage, thus, it can covertly generate proper adversarial examples in vision with adversary-specified labels quickly once it finished training. It's worth noting that the number of epochs to obtain a stable and high ASR by AdvCGAN is less than that by AdvGAN when we train the generators for them.

b) Performance under defense: At the start, we use the target model f under no defense to generate adversarial examples using FGSM and BIM attacks with $\epsilon = 0.3$. Next, these generated adversarial examples in return are utilized to retrain the target model f to improve the attack resilience, i.e. so-called adversarial training, in this way, two categories of target models $f^{'}_{FGSM}$ and $f^{'}_{BIM}$ with higher robustness are rebuilt, in the light of three original target models LeNet, ResNet and VGGNet, for details in Table III. Then, we verify the performance through testing whether the adversarial examples generated by FGSM, L-BFGS, AdvGAN and AdvCGAN can fool the two newborn target models.

The experimental results are depicted in Table III, and it shows that the attack accuracy of all of the four methods decreases under defense, despite our method performs still better than the other three methods (including Optim-based, Gradient-based and GAN-based methods) obviously. This unveils that: i) the adversarial training can indeed improve the attack resilience as one representative fashion; and ii) our proposed AdvCGAN can still generate high-quality adversarial examples to deceive target models even equipped with defense strategies.

For L-BFGS, which performs well in the previous undefended scenario, differently, its ASR in targeted attack seriously reduces to 2.27% with adversarial training and 1.78% with iterative adversarial training on MNIST, and 4.92%, 17.40% w.r.t the target model ResNet and 10.39%, 18.60% w.r.t target model VGGNet on CIFAR-10under defense strategy. The behind reason lies in that L-BFGS only optimizes the adversarial examples using p-norm in pixel-level, which can easily be resisted by adversarial trainings.

Also FGSM and AdvCGAN deteriorate, however, their downward trend is more gentle than L-BFGS. FGSM modifies the image with a certain pixel value, which is probably greater or smaller than the suitable value changed in L-BFGS, thus makes its attack accuracy a little higher than L-BFGS in most untargeted attack scenes. AdvGAN uses the popular GAN network to ensure the data distribution similar with real ones, which endows it stronger anti-detection ability in untargeted attack scene, but a poor performance in targeted attack because it cannot utilize the guidance of label information to make adversarial examples with any adversary-specified attack label.

From the results, we can see our AdvCGAN achieves the highest ASR both in targeted attack and untargeted attack scenes. Specifically, we use the attack label to enforce the generated adversarial example's prediction result to be more likely to the adversary-specified one, and simultaneously, its corresponding original label to guarantee the adversarial example looks like the original label in human vision. In targeted attack scene, the ASRs of our AdvCGAN can be increased by at least 2 times compared to the other three methods.

c) Virtual results of the adversarial examples: Fig. 3 shows the original images and adversarial images with fixed labels produced by FGSM, L-BFGS, AdvGAN and AdvCGAN on MNIST and CIFAR-10 datasets. For clearly comparing from the lens of human vision, we set $\epsilon = 0.2$ in MNIST dataset and 16 pixels in CIFAR-10 dataset. Fig. 4 shows that all of the original images can be successfully misclassified as any adversary-assigned labels in targeted attack. This strongly illustrates our proposed AdvCGAN can receive any input in the test dataset to elastically produce adversarial example with any intended targeted label. On the other hand, from the lens of perturbation, L-BFGS has the smallest values than the other three methods, and our AdvCGAN has smaller value than the other two methods. An ideal adversarial example ought to match two conditions, i.e. attack-success and covertness. Although L-BFGS has a smaller perturbation than our AdvCGAN, it performs more poorly to a large extent especially under the defense, in addition to a high-cost overhead. Comparatively, given the superior performance on attack success rate of our AdvCGAN in conjunction with a small perturbation, our AdvCGAN provides a proper high-performance adversarial attack manner.

TABLE III: ASR (%) under defense

DataSet	Model	Defense	Type	FGSM	L-BFGS	AdvGAN	AdvCGAN
MNIST	LeNet	Adv.	Targeted	2.88	2.27	1.95	**8.64**
			Untargeted	22.58	13.46	21.59	**38.51**
		Iter.Adv.	Targeted	3.92	1.78	1.97	**9.87**
			Untargeted	29.95	10.03	21.03	**36.84**
CIFAR-10	ResNet	Adv.	Targeted	8.98	4.92	9.63	**67.43**
			Untargeted	20.31	10.61	85.49	**89.43**
		Iter.Adv.	Targeted	19.24	17.40	10.24	**72.10**
			Untargeted	81.52	40.05	81.52	**84.57**
	VGGNet	Adv.	Targeted	8.43	10.39	10.03	**49.03**
			Untargeted	46.00	43.97	88.62	**90.60**
		Iter.Adv.	Targeted	11.91	18.60	10.88	**49.37**
			Untargeted	75.68	65.51	89.17	**92.61**

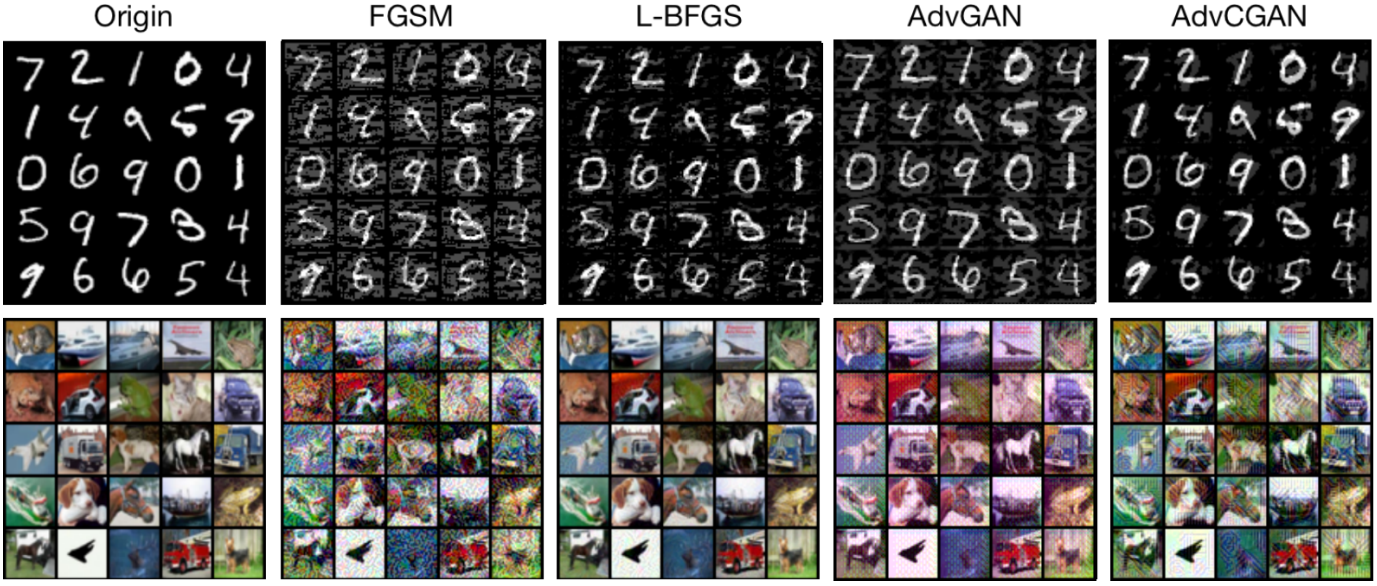


Fig. 3: Adversarial examples generated by different attack methods. Row 1: MNIST dataset, Row 2: CIFAR-10 dataset.

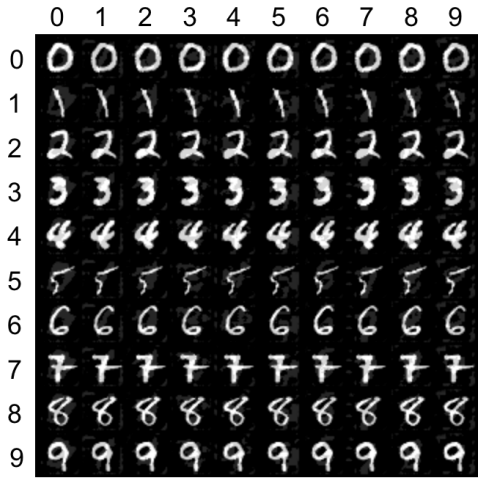


Fig. 4: Elastically generating adversarial examples with any adversary-specified label (under ϵ=0.2), the numbers of horizontal axis are source labels and the numbers of vertical axis are target attack labels.

d) Time cost: Table IV shows the time cost. Compared to FGSM and L-BFGS, although AdvGAN and AdvCGAN need extra time to train the generator, i.e. 10 min. and 40 min. for 60 epochs at training stage, both of them can run very fast once the trainings are finished, i.e. less than 1.5 seconds to produce 10k adversarial examples. The running time of FGSM is also very fast within 1 second. However, the running time of L-BFGS is far more than others, i.e. more than 3 hours, it is unpractical to produce large-scale examples, or apply into real-time tasks (e.g. auto-driving).

TABLE IV: Time cost comparison of different adversarial attack methods

Time	FGSM	L-BFGS	AdvGAN	AdvCGAN
Training (60 epoch)	-	-	10min	40min
Running (10k adv. examples)	<1s	>3h	<1.5s	<1.5s

e) Transferability to other models: Table V shows the transferability performance of our AdvCGAN across different target models on CIFAR-10 dataset. We first train the generator with one target model and evaluate the performance with another target model using ResNet and VGGNet. We calculate the ASR under targeted attack and untargeted attack respectively. From the results in Table V, we can see the ASRs are 18.94% and 65.70% in targeted attack, and 88.22% and 94.59% in untargeted attack. This may be caused by the linear

TABLE V: Transferability of adversarial examples generated by AdvCGAN

DataSet	Target model	Other Model	Type	ASR (%)
CIFAR-10	ResNet	VGGNet	Targeted	18.94
			Untargeted	88.22
	VGGNet	ResNet	Targeted	65.70
			Untarget	94.59

characteristics of the target model, i.e. the base networks of the two target models have similar architecture, or the features extracted by different classifiers are largely coincident. The result also shows that the ASR of adversarial examples on VGGNet is lower than that on ResNet. This stems from the different-level qualities of representation between VGGNet and ResNet. This kind of transferability provides the threat risks of black-box attacks even the adversary does not have the information of the target model. For instance, an adversary can easily simulate several proper local models instead of the target model, and make adversarial examples by ensemble training with these local models.

V. Conclusion

In our work, we proposed an easy-to-use adversarial attack framework to elastically generate adversarial examples and simultaneously guarantee the covertness from the human vision. Compared to the state-of-the-art works, our proposed AdvCGAN has several novelties. First, we employ the conditional generative adversarial network to enforce the generator to elastically produce adversarial examples with any adversary-assigned labels through resorting to jointly training image data and label knowledge. Second, Not only does it consider the similarity in data distribution between adversarial examples and original instances, but it also leverages label information to train the discriminator for the goal of indistinguishability from the corresponding original example. Last but not least, we design two cases to validate the efficiency of our AdvCGAN, i.e. under adversarial training-based defense and under no-defense. In addition, we also evaluate the transferability of our AdvCGAN and analyze the computation overhead for the four attack methods.

Acknowledgment

This work has been supported by the National Natural Science Foundation of China under Grant No.: 62077044, 61702470, 62002343.

References

[1] N. Akhtar and A. S. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, 2018.

[2] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning visual classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1625–1634.

[3] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the 2016 acm sigsac conference on computer and communications security*, 2016, pp. 1528–1540.

[4] Z. Wang, S. Zheng, M. Song, Q. Wang, A. Rahimpour, and H. Qi, "advpattern: Physical-world attacks on deep person re-identification via adversarially transformable patterns," in *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019*.

[5] Y. Jia, Y. Lu, J. Shen, Q. A. Chen, Z. Zhong, and T. Wei, "Fooling detection alone is not enough: First adversarial attack against multiple object tracking," *arXiv preprint arXiv:1905.11026*, 2019.

[6] A. Arnab, O. Miksik, and P. H. S. Torr, "On the robustness of semantic segmentation models to adversarial attacks," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018, pp. 888–897.

[7] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1369–1378.

[8] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.

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

[10] 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.

[11] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. D. McDaniel, "Adversarial perturbations against deep neural networks for malware classification," *CoRR*, 2016.

[12] N. Papernot, P. D. McDaniel, and I. J. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," *CoRR*, 2016.

[13] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," *arXiv preprint arXiv:1611.02770*, 2016.

[14] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. D. McDaniel, "On the (statistical) detection of adversarial examples," *CoRR*, 2017.

[15] S. H. Huang, N. Papernot, I. J. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial attacks on neural network policies," in *5th International Conference on Learning Representations, ICLR 2017*.

[16] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[17] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

[18] N. Carlini and D. A. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, 2017.

[19] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

[20] C. Xiao, B. Li, J. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, 2018.

[21] S. Jandial, P. Mangla, S. Varshney, and V. Balasubramanian, "Advgan++: Harnessing latent layers for adversary generation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[22] P. Yu, K. Song, and J. Lu, "Generating adversarial examples with conditional generative adversarial net," *CoRR*, 2019.

[23] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2672–2680.

[24] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.