AdvFoolGen: Creating Persistent Troubles for Deep Classifiers

Yuzhen Ding
Arizona State University
699 S. Mill Ave. Tempe, AZ 85281
Yuzhen.Ding@asu.edu

Nupur Thakur
Arizona State University
699 S. Mill Ave. Tempe, AZ 85281
nsthaku1@asu.edu

Baoxin Li
Arizona State University
699 S. Mill Ave. Tempe, AZ 85281
Baoxin.Li@asu.edu

Abstract

Researches have shown that deep neural networks are vulnerable to malicious attacks, where adversarial images are created to trick a network into misclassification even if the images may give rise to totally different labels by human eyes. To make deep networks more robust to such attacks, many defense mechanisms have been proposed in the literature, some of which are quite effective for guarding against typical attacks. In this paper, we present a new generative attack model termed AdvFoolGen, which can generate attacking images from the same feature space as that of the natural images, so as to keep baffling the network even though state-of-the-art defense mechanisms have been applied. We systematically evaluate our model by comparing with well-established attack algorithms. Through experiments, we demonstrate the effectiveness and robustness of our attack in the face of state-of-the-art defense techniques and unveil the potential reasons for its effectiveness through principled analysis. As such, AdvFoolGen contributes to understanding the vulnerability of deep networks from a new perspective and may, in turn, help in developing and evaluating new defense mechanisms.

1. Introduction

Deep neural networks have found wide applications in many computer vision tasks like face and object recognition, image segmentation, scene understanding etc., often delivering state-of-the-art performance for a given task. However, in recent years, it has been discovered that deep networks can be easily fooled/attacked: images can be created to trick a network into misclassification, although such created images may be classified correctly by humans. This has become a major concern for deep networks since they are becoming the backbone of real-world applications like access control and surveillance, where there may be adversarial agents constantly trying to outsmart the system.

Images generated using different attacks can be categorized as adversarial images or fooling images. Fooling images may look like random noise to human eyes but are given a class label with high confidence by a deep network. On the other hand, adversarial images may look just like some of the authentic images although often perturbations (either visible or imperceptible to human eyes) have been introduced to trick a network into misclassification.

While many earlier attack algorithms can deliver high fooling ratio in fooling typical deep classifiers, recent years have seen effective defense techniques [5, 22, 6, 28, 21, 30, 32], which can defeat many existing attacking schemes. A defense mechanism can be as simple as retraining the network with adversarial images as additional inputs [27]. More complex mechanisms often employ a new network for explicitly identifying the adversarial images [15, 25, 23]. Yet, new attack schemes keep emerging [20, 31, 26, 33, 35], and this attack-defense game will continue.

In this paper, we present a robust generator-based attack approach that employs a VAE-GAN [12] architecture as a building block and utilizes multiple target labels for constructing the objective function. These, in conjunction with employing a combination of both real and random images as input, help to produce new images lying in various portions of the original image feature space, hence creating a hurdle for existing defense mechanisms. As a result, our approach, termed as AdvFoolGen, can maintain a good fooling ratio in the face of many defense mechanisms. Moreover, the created images -'AdvFool images'- also cause trouble for human/automatic labeling since they are not random noise but also not exactly similar to the original images either. This in turn introduces additional uncertainty for both the

classifiers and the defense mechanisms.

We first evaluate our framework using initial fooling ratio as a criterion and compare with well-established attack approaches. We then showcase how these images are robust to existing defenses like retraining the network, adversarial training [16] and use of input transformations for retraining the network [9]. Moreover, we also illustrate why AdvFoolGen can fool the targeted network consistently through a principled analysis. As such, the work contributes to understanding the vulnerability of deep networks from a new perspective and may, in turn, help developing and evaluating new defense mechanisms.

2. Related Work

The existence of the adversarial negatives of neural networks, despite their high performance, was first discussed in [27]. [18] showed that it is extremely easy to fool a network with images that look like random noise to the human eye, crafted using evolutionary algorithms. There are two reasons mentioned for why these images get classified with high confidence. First, the fooling images contain the features matching one of the target class, because the evolutionary algorithms produce features unique to a class rather than features from all the classes. Second, the neural networks do not learn the global structure of the objects but only low-level and middle-level features.

[8] presented a simple approach called Fast Gradient Sign Method (FGSM) for producing adversarial images that look similar to the original ones. It is a white-box attack that works by adding or subtracting the sign of the gradient such that the loss increases. The linearity of the networks in high-dimensional spaces was mentioned as the reason for the existence of such adversarial images. [17] proposed an attack called DeepFool based on iterative linearization of the target network for generating adversarial images. These images have minimum perturbations when compared to those produced by the FGSM method.

The Carlini-Wagner (C&W) attack, a white-box attack that produces very strong adversarial images, was first introduced in [4]. It uses box-constraints and perturbation norm to form a cost function that is optimized to generate adversarial images. Though the fooling ratio is high, the method is computationally costly. In [20], an encoder-decoder architecture is used for generating the perturbations to be added to the original image to form the adversarial image. To generate adversarial images robust to the input transformations, [2] proposed an algorithm called Expectation Over Transformation.

More recently, there are works utilizing variants of Generative Adversarial Networks (GANs) for generating adversarial images to design fast and efficient attacks [31, 26, 34, 3, 29]. The adversarial examples generated by these approaches typically occupy a relatively small subspace of the space where the original images lie.

To tackle the hazard caused by the adversarial/fooling images, several defense methods have been proposed. Initially, [27] mentioned that training the network on a combined dataset of original and adversarial images improves the robustness of the network to adversarial attacks. [16] further discussed the adversarial robustness of the deep neural networks using MNIST and CIFAR-10 datasets. It also presents an iterative strategy called adversarial training. Here, the network is trained on adversarial images generated during the training stage.

Another effective defense, known as defensive distillation, was proposed in [19]. Based on the use of the knowledge from a DNN for training another neural network, this defense produced classifiers which are less sensitive to perturbations. With a Gaussian mixture prior on the latent variable, [7] makes the image and its corresponding label conditionally independent given the latent vector. It can defend the adversarial samples due to the conflict it entails in the latent space under such a framework.

[9] proposed to use input transformations (e.g., JPEG compression, bit-depth reduction, total variance minimization etc.) on the images before retraining to achieve higher robustness. A faster way of defending against adversarial images was proposed in [32]. The input images are passed through a random resizing and padding layer before feeding it to the neural network. This eliminates the need for retraining the network, which saves time and resources.

3. Proposed Approach: AdvFoolGen

We first present an intuitive understanding of why images lying between adversarial and fooling samples are more troublesome to deal with. Then we present the proposed attack mechanism, AdvFoolGen, in full detail.

3.1. The Unexplored Regions of the Feature Space

Though being different in appearance, adversarial samples and fooling samples behave similarly in the feature space, especially with the depth of the network. With the intention to be close to the authentic ones, the fooling examples, which are randomly generated initially, gradually evolve to a subspace near to that of the authentic images. On the other hand, the adversarial examples progressively shift to a subspace distancing from the original subspace so that the classifier eventually will misclassify. Despite this directional difference in the training stage, both the fooling and the adversarial samples typically cluster in a small region in the vast feature space. Thus, to defend or to attack is essentially the same task of identifying those regions that need to be taken care of. However, to our best knowledge, the samples that lie in between the fooling and adversarial samples are seldom studied, but they may be more bothering since they look neither like real images nor random noise and thus

can cause trouble for human/automatic labelling. Moreover, these in-between regions are vast to explore, providing the best opportunity for an attacker to exploit.

3.2. The Technique: AdvFoolGen

Consider a classification network P trained on clean images of C different classes, and the space of the normalized natural images X_o is represented by $\mathbb{R}^{[0,1]}$. Without loss of generality, we assume P achieves good performance on the clean images. Our aim is to generate adversarial or/and fooling images X_{af} that belong to the same space as the clean images but can achieve high fooling ratio even with various defense mechanisms applied. More specifically, if the correct class of an original image x_o is c, our goal is to make the pre-trained network P predict the corresponding adversarial or/and fooling image x_{af} anything other than c, with the constraint of the distance between x_o and x_{af} to be as small as possible. Note that we only test on network P and require no access to the parameters or gradients from P. This framework can also be extended to other applications like segmentation with appropriate changes in the similarity distance and loss. Without losing generality, we focus on the task of fooling the image classification models.

Although the feature space $\mathbb{R}^{[0,1]}$ of X is almost infinite in terms of the various combinations of different values, the original image set itself only occupies a relatively small region in $\mathbb{R}^{[0,1]}$ [14]. This leaves a huge space for the attackers to explore for success. On the other hand, the adversarial or/and fooling images generated by the current attack algorithms often gather in another small region in $\mathbb{R}^{[0,1]}$, which in turn leaves room for a defense scheme to work by enabling the target network P to recognize the small, 'poison' region where majority of the attacking images are present. We propose a new framework for generating diverse and robust adversarial/fooling images by forcing the generated images to utilize the feature space $\mathbb{R}^{[0,1]}$ greedily. Moreover, the min-max game training strategy is used for training a Generative Adversarial Network (GAN) to strengthen the generator in our attack model.

Fig. 1 illustrates the architecture of our framework. The input is a four-channel image x_i consisting of an original colored image x_o and a gray-scale noise image of one channel x_n, with a relative magnitude. x_i is first fed to an encoder to learn the parameters of the latent distribution μ_x and $log(\sigma_x)$, followed by a re-sampling process from the learned μ_x and $log(\sigma_x)$. Next, the re-sampled latent representation goes through a decoder to reconstruct an image x_{af}. Note that in our case, the structure of the encoder and the decoder is not symmetric as the input consists of four channels and the reconstructed image consists of three channels only. The next step is to feed both x_o and x_{af} to a discriminator D to detect whether the image is real or fake. Simultaneously, x_o and x_{af} are passed through

Algorithm 1: AdvFoolGen

Input : Original images X_o, Noise mask X_n, model G, D and P
Output: AdvFool image X_{af}
for *each epoch* $e = 1, 2, ...$ **do**
  while *Training* **do**
    for *each batch* $b = 1, 2, ...$ **do**
      Generate a noise mask X_n;
      Construct input image as $cat(X_o, X_n)$;
      for $i = 1$ **to** 5 **do**
        Update model D;
      end
      Update model G;
    end
  end
  while *Testing* **do**
    Generate a noise mask X_n;
    Construct input image as $cat(X_o, X_n)$;
    Output X_{af};
    Compute fooling ratio using the predicted label given by P;
  end
end

the pre-trained model P to check if the predicted labels are different. We followed the WGAN [1] training strategy to ensure stability of training. The loss is defined by four components. The first one is the re-sampling loss L_{re} (Eq. (1)). The parameters of the latent representation are drawn from a multivariate Gaussian distribution $\mathcal{N}(0, 1)$.

$$L_{re} = D_{KL}(\mathcal{N}(\mu_x, \sigma_x)||\mathcal{N}(0, 1)) \tag{1}$$

where D_{KL} represents the KL divergence.

The second loss component is the similarity loss L_S (Eq. (2)), which aims at decreasing the pixel-wise distance between x_o and x_{af}.

$$L_S = ||x_o - x_{af}||_2 \tag{2}$$

The third loss component is the GAN loss L_{GAN} (Eq. (3)) that uses a discriminator to distinguish x_{af} from x_o and in turn makes the generator stronger.

$$L_{GAN} = log(D(x_o)) + log(1 - D(G(z))) \tag{3}$$

The last loss component is the fooling loss L_{af} (Eq. (4)). Based on the type of an attack (non-targeted attack or targeted attack), the fooling loss takes different forms. In our work, we force the predicted label of image x_{af} to be close to the two least likely classes simultaneously, unlike the previous fooling losses which force the predicted label to be one target label only [8, 4, 17].

$$L_{af} = -\sum_{i=1}^{2} log(H(P(x_{af})), 1_{t_i}) \tag{4}$$

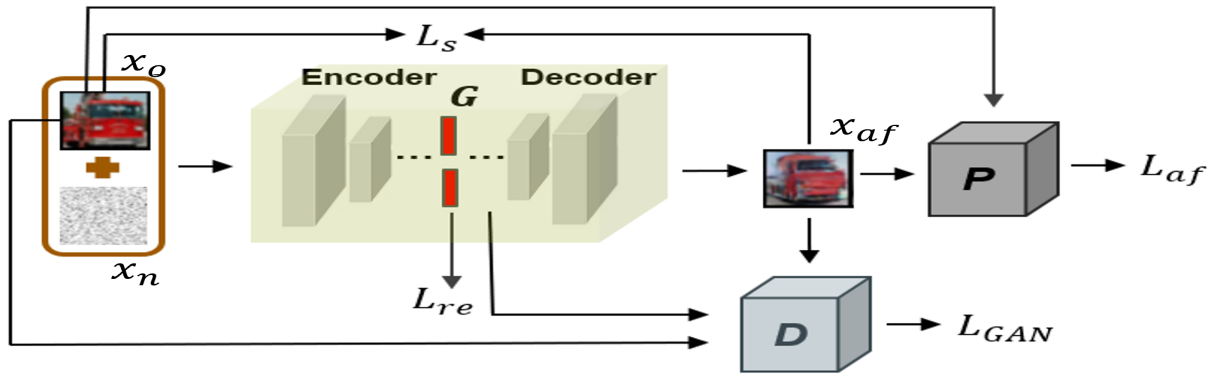

Figure 1. The proposed AdvFoolGen framework for generating AdvFool images. The input is a 4-channel image consisting of three original image channels and a noise channel. The discriminator network D and pre-trained network P compute their respective losses with the original image and the corresponding AdvFool image as inputs.

where $H(\cdot)$ is the cross-entropy function, and t_i represents the i-th target label. It is worth mentioning that this fooling loss forces the adversarial images to lie close to the decision boundary of the two least-likely classes, which causes more confusion for the classifier even with defenses applied. We discuss more details about it in Section 5.

The total loss for the AdvFoolGen is the weighted summation of the above four losses, given by Eq. 5. The entire algorithm is summarized in Algorithm 1.

$$L = \alpha L_{re} + \beta L_S + \gamma L_{GAN} + \lambda L_{af}, s.t.\ \alpha+\beta+\gamma+\lambda = 1.$$

In the existing adversarial image generators, one critical constraint on the adversarial image is resemblance to the original image. In our case, we impose a relaxed version of such constraint because of the following considerations: 1) This constraint is a subjective one because the similarity threshold for the images may vary from person to person, making it an inconsistent criterion; 2) It confines the adversarial images to a small subspace which can be easily defended. To utilize the feature space greedily, our generator generates images that lie between the adversarial images and fooling images. Thus, we name the images as AdvFool images. The AdvFool images do not look like noise since it contains some patterns from the corresponding original image. On the other hand, the difference is obvious to the human eye. Fig. 2 shows some AdvFool images along with their corresponding original images from the CIFAR-10 dataset. In our experiments, we found that the AdvFool images yielded a better fooling ratio even with various defenses applied. The details are presented in the next section.

4. Experiments

This section includes the experimental results showing the success of our approach. First, the experiments are compared with well-established baseline attacks - Fast Gradient Sign Method (FGSM) [8], Iterative Fast Gradient Sign Method (I-FGSM) [11], DeepFool [17], Carlini & Wagner (C&W) [4] and Generative Adversarial Perturbations (GAP) [20] in terms of the initial fooling ratio explicitly. Next, we show the robustness of the AdvFool images by re-attacking the target network strengthened with state-of-the-art defense strategies like retraining the network, adversarial training [16], bit-depth reduction and JPEG compression [9]. We also include the details about the experimental settings and evaluation criteria in this section.

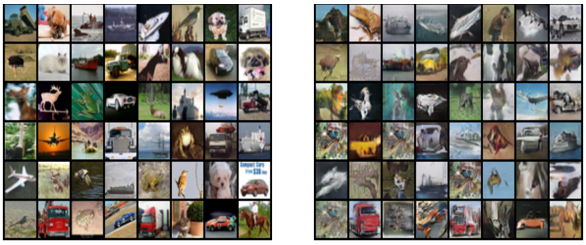

Figure 2. (a) The original images from CIFAR-10 dataset. (b) The corresponding AdvFool images. Though the difference between the AdvFool and original images is visible to human eye, these images still capture the colors and object patterns from the original images.

4.1. Experimental Setting

We use CIFAR-10 dataset [10] and TinyImageNet dataset [13] for our experiments. CIFAR-10 contains 10 classes, and each class has 5,000 training images and 1,000 test images. Each image has a height and width of 32 and consists of 3 channels. The target network is VGG-19 [24] classifier trained on clean CIFAR-10 images achieving a test accuracy of 92.42%. TinyImageNet is a smaller version of ImageNet dataset. It has 200 classes, and each class has 500 training and 50 validation images. Each image size is $64 \times 64 \times 3$. The target network for TinyImageNet is ResNet18 with Top1 and Top5 validation accuracy of 72.3%

Attack Algorithm	Initial Fooling Ratio		
	CIFAR10	TinyImageNet	
		Top1	Top5
FGSM	92.82% *	88.55% *	75.18% *
I-FGSM	99% *	100% *	98.86% *
DeepFool	99%	99%	83.77%
C&W	100%	99.12%	90.63%
GAP	82%	94.98%	87.01%
AdvFoolGen	68.5% - 78.36%**	95.41%-97.65%**	90.14%-93.07%**

Table 1. Initial fooling ratio for the AdvFoolGen compared with state-of-the-art attacks on CIFAR-10 and TinyImageNet Dataset. *$\epsilon = 0.07$ for FGSM and I-FGSM. **We report a range for AdvFoolGen attack as the fooling ratio varies from epoch to epoch.

and 91.2% respectively. No overfitting is observed while training the classifiers for both datasets. We use the test and validation data to report our results for CIFAR-10 and TinyImageNet respectively.

For the AdvFool images generator, the noise mask x_n is drawn from a uniform distribution of $\mathcal{U}(0, mgn)$, where mgn is the noise magnitude. Though the choice of noise magnitude is not unique, we found that different magnitudes yield similar performance through an empirical study. Thus, we used a fixed value of 0.1. The model is trained from scratch and it reaches a relatively stable stage after a few epochs. Thereafter, we use the AdvFool images generated from different epochs to attack the pre-trained network.

To demonstrate the robustness of AdvFool images, we attack the networks equipped with defenses. It is worth mentioning that initially, we set the ground truth label for each x_{af} to be t_c, the true label of the corresponding original image x_o. However, the AdvFool images are visually between original images and fooling images, and thus using the original labels is not fair for the defense. Therefore, we assign additional labels for x_{af} when applying defenses that require ground truth of AdvFool images. More information about the labeling of AdvFool images is discussed in the later part of this section.

4.2. Initial Fooling Ratio

To evaluate the success of the attack, we use the fooling ratio metric which is calculated using Eq. (5).

$$F_{ratio} = \frac{\sum_{n=0}^{N} t_n^{x_{af}} \neq t_n}{N} \tag{5}$$

where $t_n^{x_{af}}$ represents the predicted label for x_{af}, t_n is the predicted label for x_o and N is the number of images in the test set.

Table 1 shows the initial fooling ratio of different attack methods on CIFAR-10 dataset and TinyImageNet. AdvFoolGen achieves a reasonable (more than half the images can fool) but not a competitive (15%-30% lower than others) fooling ratio as compared to the existing approaches. As TinyImageNet dataset is a smaller version of ImageNet dataset, to remain consistent with the notations the latter dataset uses, we report two fooling ratios - Top1 and Top5. It can be seen that Top1 fooling ratio is always higher than the Top5 fooling ratio, which is intuitive since the Top1 accuracy is always lower than the Top5 accuracy. The moderate Top1 accuracy of the target classifier explains the high Top1 and Top5 fooling ratio for all the attacks. Similar to the results obtained on CIFAR-10 dataset, we observe that the fooling ratio of AdvFoolGen attack is lower than most of the state-of-the-art attacks. However, the majority of the AdvFool images can successfully fool the network.

Although the initial fooling ratio is a common measure to evaluate how good an attacker is, it has several drawbacks. Using just the fooling ratio, we cannot evaluate the diversity of the generated images nor robustness of the attack, which are two critical properties contributing to consistent fooling in real-world applications. In other words, if an attacker can only fool the pre-trained network once and fails under one simple defense technique, it is not a strong attacker. Thus, the adversarial/fooling images that can fool the pre-trained network strengthened with defenses consistently have drawn more attention recently. In the next section, we demonstrate the robustness of AdvFool images with experiments using various defenses.

4.3. Effect of Defenses on Fooling Ratio

Table 2 shows the fooling ratio of different attacks as the defenses are employed for CIFAR-10 dataset. The results on TinyImageNet are included in the supplementary material. The 2^{nd} column shows the fooling ratio against retrained networks. For the existing attackers, we retrain the pre-trained network with the same number of the adversarial images as that of the original images. For AdvFoolGen, we retrain the target network by slightly revising the structure: more output labels instead of 10, the details of which are discussed next.

As AdvFool images are between adversarial and fooling images, we tested them using three different retraining strategies. First, we used their original class labels i.e. treating them as adversarial images for retraining. Second, we created 10 new corresponding classes for the AdvFool im-

Attack	Retraining*	Adv Training	BDR-3	BDR-8	JPEG
FGSM	9.76%	35.9%	18.21%	16.2%	18.6%
I-FGSM	8.22%	39.3%	12.32%	11.2%	13.1%
DeepFool	9.87%	26.5%	14.55%	14.1%	14.8%
C&W	9.2%	41.25%	12.97%	12.19%	15.67%
GAP	8.91%	9.04%	14.99%	15.09%	19.89%
AdvFoolGen	**27.3%-58.1%**	**59.56%-65.26%**	**37.08%-52.82%**	**24.76%-35.4%**	**24.44%-50.64%**

Table 2. Fooling ratio after the defenses are applied. The fooling ratio for AdvFoolGen is higher than existing attacks when it comes to networks with added defense mechanisms. For Bit-Depth Reduction (BDR), the results are reported for bit-depth of 3 and 8. For JPEG, all the images are compressed at the quality level of 75 (out of 100). *Equal number of original and adversarial images are used for retraining. For AdvFoolGen attack, 5000 AdvFool images are used for training and 1000 AdvFool images are used for testing and the total number of classes is 11.

ages i.e. if the original label of an AdvFool image is 2, then it will be assigned label 12 while retraining. In this case, we get a total of 20 classes. Lastly, we created a new class for all the AdvFool images making the total number of classes equal to 11. For the first two types of retraining, the network could not learn the AdvFool images very well (around 25% test accuracy on AdvFool images) and eventually led to more confusion, reducing the accuracy on original images significantly. However, for the 11-class retraining strategy, the network could learn the AdvFool images very well (around 95% test accuracy on AdvFool images). Therefore, we present all the results for AdvFoolGen attack with one additional class (11-class strategy). If the network does not classify an AdvFool image as belonging to 11^{th} class, it is considered to be fooling the network.

Compared with baseline attacks, although all the fooling ratio decreased significantly on the retrained networks, at least 30% of the AdvFool images still fooled the retrained network. Also, the accuracy on the original images only slightly decreased to 88.05%.

Considering the 3^{rd} column, it is observed that the adversarially trained networks can no longer be fooled by the state-of-the-art attacks with a high fooling ratio. However, AdvFool images can fool such a network effortlessly. Moreover, we observed that with increased number of training AdvFool images from different epochs, the accuracy on both original and AdvFool images decreased significantly. The high fooling ratio indicates that AdvFoolGen can get past the adversarial training strategy with ease.

The 4^{th} and 5^{th} columns show the results for Bit-Depth Reduction on the input images followed by retraining. We use a bit depth of 3 and 8. The last column shows the JPEG compression defense results where all the images are compressed at the quality level of 75 (out of 100). Yet, a significant number of AdvFool images can fool the network retrained with the transformed images.

The fooling ratio for all the baseline attacks drop dramatically after the defenses are applied. A decrease in the fooling ratio is observed for the AdvFoolGen attack as well but it is relatively high compared to the baseline attacks. AdvFoolGen attack suffers only 20%-30% decrease in the fooling ratio. On an average, 30% of the AdvFool images still fool the network. Therefore, in real-world applications, AdvFool images can be considered as more 'poisonous' than the adversarial images generated by existing attacks.

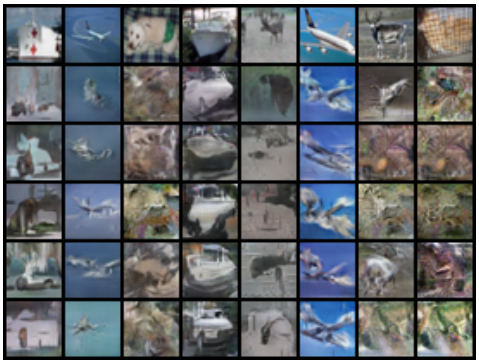

Figure 3. The first row shows the original images from CIFAR-10 dataset. Row 2-5 are AdvFool images generated using generators of different epoch. The variations between the images from different epochs are clearly visible. Though the images are morphed, they do not resemble objects from any other classes.

Taking a closer look at Table 2, we found that the I-FGSM and C&W attacks with almost 100% initial fooling ratio, completely failed when simple defenses were employed. On the contrary, FGSM which has the lowest fooling ratio among the baseline approaches achieves high fooling ratio against the defenses. This supports our claim that the initial fooling ratio may not be a good criterion for evaluating the adversarial attacks. Furthermore, the difference between the initial fooling ratio and re-attack fooling ratio implies that the adversarial images generated by C&W attack overfit the target network. Thus, these images fail immediately with only a few changes to the target network.

5. Why AdvFool Images can Fool the Network?

We discuss further why AdvFool images can fool the network in the face of various defense techniques from two different perspectives. We first carefully examine the architec-

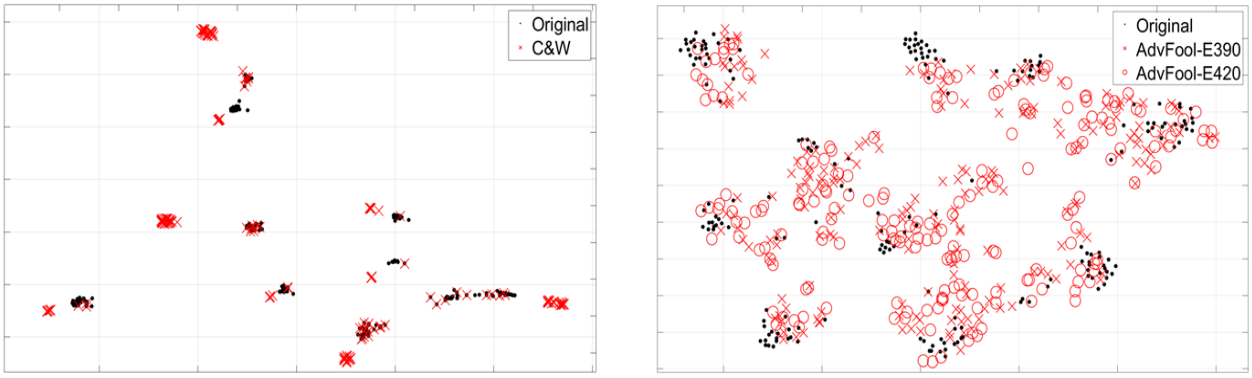

Figure 4. Visualizing the features from the layer before classification layer of VGG-19 network in 2-D for C&W (left) adversarial CIFAR-10 images and the AdvFool (right) CIFAR-10 images. The corresponding original images are represented in both of the figures by black dots. Two sets of AdvFool images are shown, each from a different epoch generator. While the C&W images remain clustered in a small subspace, the AdvFool images are spread out and between the classes explaining its higher fooling ratio for defense-equipped networks. Besides, the two sets of AdvFool images do not overlap completely explaining its fooling ability across various epochs.

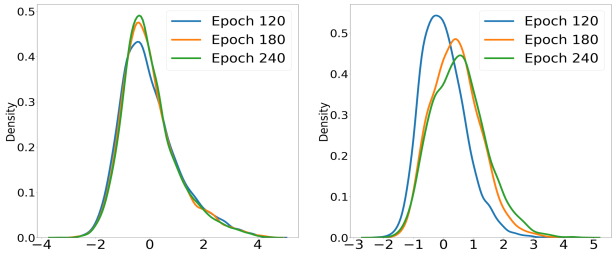

Figure 5. Distributions of the mean (left) and variance (right) used for latent representation of AdvFool images in three different epochs of the AdvFoolGen attack. The distributions are Gaussian with different parameters.

ture of our model to discover potential factors leading to the effectiveness and robustness of AdvFool images. Next, we use statistical tools to analyze AdvFool images from various epochs to verify several conjectures that explain the good fooling ratio of AdvFool images.

5.1. Revisiting the architecture of AdvFoolGen

From Fig. 1, it is clear that the reference image of an AdvFool image is not a pure original image, but a 4-channel image which is an integration of the original colored image and an added channel of gray-scale noise image of relatively small magnitude. We introduce the noise to bring in randomness in the input which helps the generator to explore the feature space deeper. Although the noise magnitude has to be chosen from a limited range, the variations of the value within this limited range are unlimited. The re-sampling step adds extra randomness which forces the generator to explore the untouched regions as well. Fig. 3 shows some AdvFool images from different epochs. It is clear that the images from different epochs have different perturbations which are visible to human eye. Due to the almost unlimited range of noise magnitude and the re-sampling process, every single time a different set of AdvFool images are generated. Therefore, the network can be fooled even if images from different epochs are used for retraining. Besides, as the number of AdvFool images used for retraining increase, the accuracy on the original images starts to decrease which is highly undesirable.

We defined fooling loss L_{af} in such a way that the AdvFool images should be on/near the boundary region of two least-likely classes, where the original images rarely occur. The relaxed constraint of the similarity between the original image and AdvFool image along with the discriminator network ensures that the AdvFool image does not visually go too far away from the corresponding original image. As a result of these constraints in the fooling loss, the AdvFool images are morphed but never completely change into images that belong to any of the remaining classes. Thus, these images lie between adversarial images and fooling images, making it hard for the classification model to determine their correct labels required for retraining the network. Moreover, the AdvFool images capture features from three different classes (the original class and two least-likely classes). It tends to confuse humans while labeling resulting in inconsistent labels. Therefore, human labeling for AdvFool images is not feasible.

5.2. Analytical Study of AdvFool Images

To further support our claims about the AdvFool images, we analyze the images systematically using statistical tools. Fig. 4 is the visualization of the deep features extracted for a typical batch of CIFAR-10 dataset by the VGG-19 network just before the classification layer in 2-D. The original images (black dots) are well-clustered in both the plots. In the first plot, C&W adversarial images are present in a small

Epoch	Mean		Variance	
	KLD (P‖Q)	**KLD (Q‖P)**	**KLD (P‖Q)**	**KLD (Q‖P)**
120 &180	0.0491	0.0123	1.35	0.299
120 & 240	0.0102	0.0129	2.513	0.313
120 & 330	0.0103	0.0129	2.539	0.314
120 & 360	0.0131	0.0130	2.546	0.314

Table 3. KL-Divergence between the distributions of the mean and variance of the latent representation. P and Q are the first and second epoch number mentioned in a particular row in the first column. All values have 10^{12} as a multiplication factor.

subspace which is in the opposite direction of the original image categories. Therefore, these images can be easily defended with defenses like retraining. However, it is clear from the second plot that the AdvFool images are spread out, lying near the decision boundary of two classes. Also, the set of AdvFool images from two different epochs do not overlap, which makes defending them difficult.

In the re-sampling process of AdvFoolGen, the values of mean and variance are sampled assuming that the distribution is Gaussian. We claim that the AdvFool images from different epochs can fool the network because they belong to different distributions. To verify this claim, we use density estimation and show that the mean and variance sampled for the latent distribution at different epochs belong to different distributions. We used a non-parametric density estimation technique called Parzen-Window with Gaussian window function to estimate the distribution of the mean and variance. The smoothing parameter for this technique is found using Grid Search.

As mean and variance from the re-sampling process have high dimensions, we used Principal Component Analysis (PCA) to reduce their dimensions for visualization. The dimensionality reduction is done in such a way that all of them are mapped to the same dimension to make a fair comparison. Fig. 5 shows the distributions of the mean (left) and variance (right) used in the re-sampling process during 3 different epochs. It is evident that all of these are Gaussian distributions but with different parameters. Though the difference between the distributions look very minute in the figures, it is enough to result in distinct distributions that can produce different set of AdvFool images. Such sets of AdvFool images can fool the network successfully. It becomes difficult to defend these images which are generated from different epoch generators as they belong to different distributions. In other words, even if the network learns AdvFool images from one epoch, it can be easily fooled by another set of AdvFool images from a different epoch as they do not belong to the same distribution.

We calculated the KL-Divergence between two different distributions to show that even a tiny difference in the parameters leads to very distinct distributions. Table 3 lists the KL-Divergence between the distributions from different epochs. It is clear that the KL-Divergence increases as we move farther away from a particular epoch number. This justifies the increase in the fooling ratio as a farther epoch generator is used for generating the AdvFool images. For example, if a network is familiar with the AdvFool images from epoch 120, the AdvFool images from epoch 360 will achieve high fooling ratio than those from epoch 180. As KL-Divergence is not symmetric, we calculated it in the reverse manner as well and observed the same trend. Therefore, the order in which the KL-Divergence is calculated does not make an impact on the results.

6. Conclusion

In this paper, we proposed a new generator-based scheme, AdvFoolGen, for attacking deep classifiers. AdvFoolGen generates images (termed as AdvFool images) which can consistently fool a network that has the help of many existing defense techniques. From experiments, we observed that the initial fooling ratio is not a good metric for evaluating an attack scheme, as the scheme may fall apart under some defense mechanisms. Though AdvFoolGen has relatively lower initial fooling ratio, it can keep deceiving the defense strategies over and over again.

The principled analysis shows that the mean and variance used for the latent representation of the images in our framework belong to different Gaussian distributions for different epochs. Thus, the images from AdvFoolGen at a certain epoch can fool the network trained on images produced from the AdvFoolGen at another epoch. The analysis also showed that the AdvFool images do not occupy a small subspace and are highly spread. Also, sets of AdvFool images from different epoch generators do not overlap in space, explaining why they can fool the network after defenses are applied using a set of AdvFool images. The success of our attack on the defenses shows the susceptibility of the current defense mechanisms and raises a need for more robust approach. We envision this better understanding of the pitfalls of the neural networks will be useful for building more generic and advanced defense mechanisms.

Acknowledgement The work was supported in part by a grant from ONR. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of ONR.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[2] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

[3] Tao Bai, Jun Zhao, Jinlin Zhu, Shoudong Han, Jiefeng Chen, Bo Li, and Alex Kot. Ai-gan: Attack-inspired generation of adversarial examples. *arXiv preprint arXiv:2002.02196*, 2020.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[5] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E Kounavis, and Duen Horng Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–204. ACM, 2018.

[6] Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

[7] Partha Ghosh, Arpan Losalka, and Michael J Black. Resisting adversarial attacks using gaussian mixture variational autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 541–548, 2019.

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[9] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[10] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 55, 2014.

[11] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[12] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *International Conference on Machine Learning*, pages 1558–1566, 2016.

[13] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 2015.

[14] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[15] Jiajun Lu, Theerasit Issaranon, and David Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 446–454, 2017.

[16] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[18] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[19] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[20] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4422–4431, 2018.

[21] Aaditya Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James Storer. Deflecting adversarial attacks with pixel deflection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8571–8580, 2018.

[22] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

[23] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[25] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

[26] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models. In *Advances in Neural Information Processing Systems*, pages 8312–8323, 2018.

[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[28] Deepak Vijaykeerthy, Anshuman Suri, Sameep Mehta, and Ponnurangam Kumaraguru. Hardening deep neural networks via adversarial model cascades. *arXiv preprint arXiv:1802.01448*, 2018.

[29] Xiaosen Wang, Kun He, Chuanbiao Song, Liwei Wang, and John E Hopcroft. At-gan: An adversarial generator model for non-constrained adversarial examples. *arXiv preprint arXiv:1904.07793*, 2019.

[30] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme

value theory approach. *arXiv preprint arXiv:1801.10578*, 2018.

[31] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

[32] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.

[33] Zhewei Yao, Amir Gholami, Peng Xu, Kurt Keutzer, and Michael W Mahoney. Trust region based adversarial attack on neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11350–11359, 2019.

[34] Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. In *International Conference on Learning Representations*, 2018.

[35] Tianhang Zheng, Changyou Chen, and Kui Ren. Distributionally adversarial attack. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2253–2260, 2019.