Transferable Sparse Adversarial Attack

Ziwen He[1,2], Wei Wang[2*], Jing Dong[2] & Tieniu Tan[2]
[1] School of Artificial Intelligence, University of Chinese Academy of Sciences
[2] Center for Research on Intelligent Perception and Computing, NLPR, CASIA
ziwen.he@cripac.ia.ac.cn,{wwang,jdong,tnt}@nlpr.ia.ac.cn

Abstract

Deep neural networks have shown their vulnerability to adversarial attacks. In this paper, we focus on sparse adversarial attack based on the ℓ_0 norm constraint, which can succeed by only modifying a few pixels of an image. Despite a high attack success rate, prior sparse attack methods achieve a low transferability under the black-box protocol due to overfitting the target model. Therefore, we introduce a generator architecture to alleviate the overfitting issue and thus efficiently craft transferable sparse adversarial examples. Specifically, the generator decouples the sparse perturbation into amplitude and position components. We carefully design a random quantization operator to optimize these two components jointly in an end-to-end way. The experiment shows that our method has improved the transferability by a large margin under a similar sparsity setting compared with state-of-the-art methods. Moreover, our method achieves superior inference speed, 700× faster than other optimization-based methods. The code is available at https://github.com/shaguopohuaizhe/TSAA.

1. Introduction

In the past few years, deep neural networks (DNNs) have been widely used in many computer vision tasks such as image classification [24], object detection [15] and action recognition [36], etc. However, some studies [16, 41] have shown that DNNs can easily be fooled by adversarial examples which are crafted by maliciously adding designed perturbations to the inputs. These examples are imperceptible to human eyes but result in wrong outputs, posing a potential threat to face recognition [12], autonomous driving [4] and other real-world applications [13, 17].

In most cases, an adversarial disturbance is constrained by the ℓ_p norm distance due to its concise mathematical expression. Much prior work [5, 10, 16, 26, 41] focuses on adversarial attack under ℓ_∞ or ℓ_2 constraints. Different from these attack methods disturbing all pixels of the image, sparse adversarial attack [7, 9, 27] only perturbs a few pixels but possibly with large modifications. The perturbations are indeed visible but do not alter the semantic content, and can even be applied in the physical world (adversarial patch) [3]. For comprehensively assessing the model robustness, it is equally important to explore this category of attack. Among them, ℓ_0 based attack attracts more attention. It is a typical NP-hard problem. To address this problem, many attempts have been made under both white-box and black-box settings.

However, existing ℓ_0 based methods [7, 9, 27, 31] still suffer from such a major issue: they have low transferability [25, 30, 32] when performing attack on black-box models. The transferability means that adversarial examples crafted on one model can successfully attack another model with different architecture and parameters. It plays an important role in black-box adversarial attacks. To fool a black-box model, attackers use a substitute model to craft adversarial examples and feed them into the target model. While ℓ_2 and ℓ_∞ norm constrained adversarial examples [10, 11, 22, 25] can achieve high transferability across different architectures, it is still unknown how transferable the ℓ_0 sparse adversarial examples can be.

In this paper, we explore this question and propose a method to generate transferable sparse adversarial examples. Previous ℓ_0 based attack methods rely on the target model's accurate gradient information or its approximation, causing an overfit of this model. To remedy this problem, we introduce a trainable generator architecture. It learns to translate a benign image into an adversarial example. Benefited from a large number of data in the training stage, the adversarial image does not overfit the fixed white-box model but can fool many black-box models.

To utilize a generator to craft sparse adversarial examples is not easy, since previous generator-based methods [1, 28, 29, 33, 38, 43] can only craft dense perturbations. To address this issue, we elaborately design a new framework. Our framework decouples the adversarial perturba-

[*] Corresponding author.

tion into two components which control distortion magnitude and perturbed pixel location respectively. In this way, we can apply sparse regularization on the location map to achieve a satisfactory sparsity. A still remaining question is that, each point of the map is a binary value, i.e., 0 or 1, causing a discrete optimization problem which hinders the end-to-end training. To optimize sparse perturbations in the training stage in an end-to-end manner, we further design a special 0-1 quantization operator. When the proposed framework finishes training, an original image is fed into the generator and the output adversarial image is fast obtained through only one feedforward inference.

Experiments on the ImageNet [8] dataset show that the transferability of our method is better than state-of-the-art methods. For images in the ImageNet, when there is no ℓ_∞ norm constraint, the recently superior GreedyFool [9] needs to perturb 0.67% pixels to get 15.09% fooling rate for transferring the InceptionV3 (IncV3) [40] to Resnet50 (Res50) [18]. In contrast, our method only needs to perturb 0.46% pixels to achieve the 63.76% fooling rate. When there is a constraint $\ell_\infty = 10$, our method only needs to modify 14.47% pixels to achieve a 45.32% transferability from IncV3 to Res50, while GreedyFool needs 18.19% pixels to obtain only 10.67% transferability. Meanwhile, our method demonstrates a much faster inference speed than state-of-the-art methods. To get a similar transferability from Res50 to IncV3 under the same sparsity setting, our method only needs 6 milliseconds to craft an adversarial image on average, while GreedyFool needs 20.49 seconds.

To summarize, the main contributions of this paper are twofold: 1) We delve into the transferability of ℓ_0 based sparse adversarial examples, which has not been thoroughly explored yet. To enhance the transferability, we propose a generator-based framework, which could be end-to-end trained by our specially designed decoupling and 0-1 quantization. 2) We conduct extensive experiments to evaluate the transferability of sparse adversarial attacks. Results on Imagenet demonstrate the superior performance of our method, including transferability and speed. For both with or without ℓ_∞ norm constraint, our method outperforms state-of-the-art methods by a large margin.

2. Related work

White-box sparse attack. For white-box attack, JSMA [31] proposes to select the most effective pixels on the adversarial saliency map, which is used to identify the impact of features on output classification. PGD$_0$ [7] proposes to project the adversarial noise generated by the well-known PGD [26] to the ℓ_0-ball. SparseFool [27] converts the ℓ_0 constraint problem into an ℓ_1 constraint problem and exploits the boundaries' low mean curvature to compute adversarial perturbations. ADMM$_0$ [44] utilizes the alternating direction method of multipliers [42] to separate the ℓ_0 norm and the adversarial loss and facilitate the optimization of the sparse attack. SAPF [14] formulates the sparse attack as a mixed integer programming problem to jointly optimize the binary selection factors and continuous perturbation magnitudes of all pixels, with a cardinality constraint on selection factors to explicitly control the degree of sparsity. GreedyFool [9] builds on the greedy algorithm and introduces a GAN-based distortion map for better invisibility. These methods contribute to achieve a high white-box attack success rate under a low sparsity setting. We take a further deep step to explore the transferability obtained by performing attack on substitute models.

Black-box sparse attack. When it comes to black-box attack, One Pixel Attack [39] and Pointwise Attack [35] propose to apply evolutionary algorithms to achieve extremely sparse perturbations. CornerSearch [7] proposes to select the most effective subset of pixels by testing the score of target models after changing one pixel's value to one of the 8 corners of the RGB color cube. Recently, Sparse-RS [6] proposes a framework based on random search for score-based sparse attacks in the black-box setting. GeoDA [34] presents a geometric framework based on the observation that the decision boundary of deep networks usually has a small mean curvature near the data samples and achieves the best fooling rate with a limited query budget. These methods need to access the outputs of the target model via queries, while our method using local substitute model to craft adversarial examples without any query.

Generator-based attack. Some works [1, 28, 29, 33, 38, 43] adopt an image-to-image generator architecture in order to learn a mapping from the input image to a perturbed output image such that the perturbed image cannot be distinguished from the benign image for a classification model. ATN [1] and GAP [33] use trainable deep neural networks for transforming images to adversarial examples and perturbations respectively. AdvGAN [43] applies generative adversarial networks to craft visually realistic perturbations. Song et al. [38] synthesize unrestricted adversarial examples entirely from scratch by training a conditional generator. Mopuri et al. [28] present a generative model that utilizes the acquired class impressions to learn crafting universal adversarial perturbations. Naseer et al. [29] propose a generative framework that learns to generate strong adversaries using a relativistic discriminator. Above methods focus on all-pixels adversarial images. Instead, we propose a generator framework to craft sparse adversarial examples.

3. Transferable sparse adversarial attack

3.1. Problem analysis

Denote $\mathbf{x}$ as one benign image and y as its corresponding ground-truth label. Let f be the target model and thus we have $\arg\max_c f(\mathbf{x})_c = y$, where $f(\mathbf{x})_c$ is the output logit value for class c. To generate an adversarial sample, an adversary adds elaborately designed noise $\boldsymbol{\delta}$ to the original image $\mathbf{x}$. The resultant adversarial image $\mathbf{x}_{adv} = \mathbf{x} + \boldsymbol{\delta}$ is expected to satisfy $\arg\max_c f(\mathbf{x}_{adv})_c \neq y$. Meanwhile, the adversarial noise $\boldsymbol{\delta}$ should be small enough to guarantee the imperceptibility. In this paper, the constraint on $\boldsymbol{\delta}$ is measured by ℓ_0 and ℓ_∞ norm:

$$\begin{aligned} & \min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\|_0 \\ \textit{s.t.} \quad & \arg\max_c f(\mathbf{x} + \boldsymbol{\delta})_c \neq y \\ & \|\boldsymbol{\delta}\|_\infty < \epsilon, \end{aligned} \tag{1}$$

where ϵ is a hyper-parameter to control the ℓ_∞ norm constraint of adversarial perturbations. The above setting is non-targeted attack, and for targeted attack the condition is $\arg\max_c f(\mathbf{x}_{adv})_c = y_t$, where y_t is the target label.

In transfer-based attacks, attackers use the white-box model f to craft an adversarial example $\mathbf{x}_{adv}$ and feed $\mathbf{x}_{adv}$ into an unknown target model. For previous sparse attack methods, they craft the adversarial example $\mathbf{x}_{adv}$ heavily relying on the gradient information of f with respect to the single image $\mathbf{x}$, resulting in an overfitting issue. To alleviate the overfitting, we introduce a generator-based method. The generator learns a mapping between natural images and sparse adversarial images. Thus the generator's parameter is optimized by a data distribution rather than a single image. The increase of training data amount can reduce overfitting and therefore boost transferability. Under this assumption, the question is then transformed into how to design a generator-based framework to solve problem 1.

The minimum optimization problem 1 is NP-hard. To solve this problem, many prior works solve the approximate ℓ_1 constraint problem. For instance, SparseFool [27] is an algorithm that exploits such a relaxation, by adopting an iterative procedure that includes a linearization of the classifier at each iteration, in order to estimate the minimal adversarial perturbation. However, this iterative solution includes non-differentiable steps, thus cannot be used in an end-to-end training on a generator. To solve the approximate ℓ_1 constraint problem within a generator-based framework, one idea is to directly add an ℓ_1 regularization on $\boldsymbol{\delta}$. However, the ℓ_1 regularization on $\boldsymbol{\delta}$ will make the value of perturbation converge around 0 and thus result in a dense solution. If binary quantization is applied on this solution to obtain a real sparse perturbation, the obtained example is probably not adversarial due to information loss. We solve this problem by factorizing the perturbation to the element-wise product of two variables, including a vector which controls perturbation magnitudes and a binary mask which controls where to perturb. Formally,

$$\boldsymbol{\delta} = \mathbf{r} \odot \mathbf{m}, \tag{2}$$

where $\mathbf{r} \in \mathbb{R}^N$ denotes the vector of perturbation magnitudes and $\mathbf{m} \in \{0,1\}^N$ denotes the vector of perturbed positions, with N being the data dimension, and $\odot$ represents element-wise product. Then we jointly optimize $\mathbf{m}$ and $\mathbf{r}$, and only apply ℓ_1 regularization on $\mathbf{m}$ instead of $\boldsymbol{\delta}$. The ℓ_1 regularization on $\mathbf{m}$ can lead to a sparse map and meanwhile does not affect the optimization of perturbation magnitude for a successful attack.

In our framework, we utilize two different branches to optimize the two variables respectively. One branch outputs $\mathbf{r}$, which is bounded by a predefined ℓ_∞ norm ϵ. The other branch outputs the location map $\mathbf{m}$, each element of which is a binary value (0 or 1). One pixel (i,j) is perturbed if $\mathbf{m}_{(i,j)} = 1$, otherwise it is not perturbed. The binary value of $\mathbf{m}$ causes a challenging discrete optimization problem, as it cannot be directly optimized using any gradient-based continuous solver. To solve this problem, a 0-1 random quantization operator is designed, which translates continuous vector into discrete vector and enables gradient backpropagation.

3.2. Framework

Figure 1 illustrates the overall architecture. A generator is designed to translate a benign image into an adversarial image. Denote the generator as $\mathcal{G}$, the adversarial image is crafted by $\mathbf{x}_{adv} = \mathbf{x} + \mathcal{G}(\mathbf{x})$. The generator mainly includes one encoder and two decoder branches. Firstly the encoder $\mathcal{E}$ takes the original instance $\mathbf{x}$ as its input and generates a latent code $\mathbf{z} = \mathcal{E}(\mathbf{x})$. Then $\mathbf{z}$ is fed into two decoders, denoted as $\mathcal{D}_1$ and $\mathcal{D}_2$.

$\mathcal{D}_1$ is similar to the decoder in ℓ_∞ norm generators [29, 33]. It outputs a vector which represents perturbation magnitudes. A projection operator is used to bound the vector in a valid range, such as [-255,255]. We use a scale operation to achieve the projection and the whole process can be formulated as $\mathbf{r} = \epsilon \cdot \mathcal{D}_1(\mathbf{z})$, where ϵ is the hyper-parameter to control the ℓ_∞ norm constraint of adversarial perturbations.

$\mathcal{D}_2$ outputs a vector $\boldsymbol{\varrho} \in [0,1]^N$. To get the discrete vector $\mathbf{m}$ which controls the perturbed pixels' positions, we need to pass $\boldsymbol{\varrho}$ through a binary quantization operator q. Normally, a hard label quantization operator is

$$q(\varrho_{i,j}) = \begin{cases} 0 & \varrho_{i,j} \leqslant \tau \\ 1 & \varrho_{i,j} > \tau, \end{cases} \tag{3}$$

where τ is a predefined threshold and $\varrho_{i,j}$ represents the value of $\boldsymbol{\varrho}$ at pixel (i,j). Obviously, such a quantization operator will cause gradient vanishing if used in training (the derivative of all differentiable points is 0). Therefore, we design an operator including randomness in the training stage. Denote whether to quantize $\varrho_{i,j}$ as a random variable $X \in \{0,1\}$. If $X = 1$, $\varrho_{i,j}$ is quantized to 0 or 1 using Equation 3, otherwise keep its original value. We make X

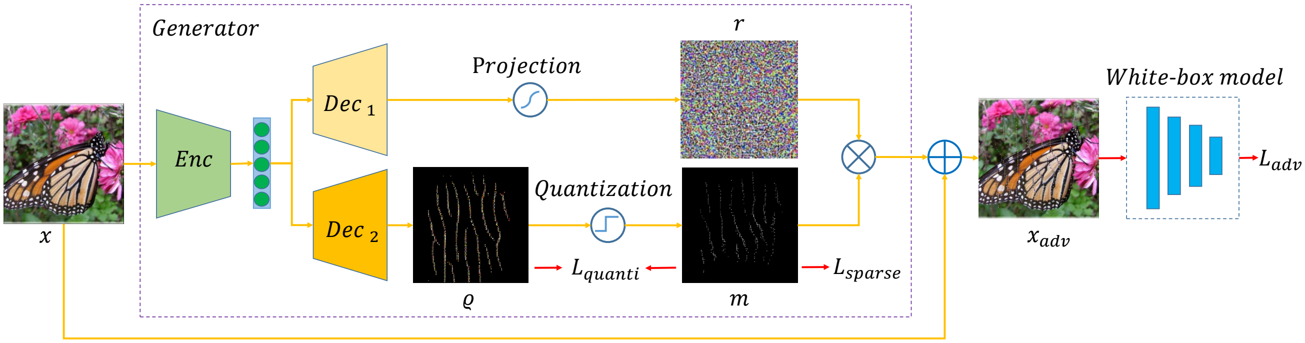

Figure 1. Our framework for generating transferable sparse adversarial examples.

subject to Bernoulli distribution. The probability distribution is

$$P(X=k)=p^k(1-p)^{1-k}, k\in\{0,1\}, \tag{4}$$

where p is the probability of $X = 1$. We block gradient backpropagation when $X = 1$ and only permit gradient flow when $X = 0$. In other words, if one pixel $m_{i,j}$ is not quantized in the training stage, its gradient cannot be backpropagated to $\varrho_{i,j}$. In this way, we make $\varrho_{i,j}$ gradually approaching 0 or 1 and ensure accurate gradient information in the training process. In the inference stage, we set p = 1 and thus all points of $\boldsymbol{\varrho}$ are quantized to a binary value.

Finally, the adversarial image can be obtained by $\mathbf{x}_{adv} = \mathbf{x} + \mathbf{r}\cdot\mathbf{m}$. Once $\mathcal{G}$ is trained on the training data and the white-box model, it can produce perturbations for any input instance to perform a transfer attack.

3.3. Loss functions

Adversarial loss. In order to achieve our goal of fooling a target black-box model, we need a local substitute model f to supervise the training of generator $\mathcal{G}$. For a non-targeted attack, the goal is to generate an adversarial image that is not classified as the ground-truth label. At each training iteration, the generator tries to maximize the adversarial image's probability of a wrong label via a loss function. To achieve this goal, we use the loss function from C&W [5] as the adversarial loss:

$$\begin{aligned}&\mathcal{L}_{adv}(\mathbf{x}_{adv},y,f)\\&=\max\left(f(\mathbf{x}_{adv})_y-\max_{i\neq y}\left\{f(\mathbf{x}_{adv})_i\right\},-\kappa\right),\end{aligned} \tag{5}$$

where κ is a confidence factor to control the attack strength. The targeted attack can be achieved simply by replacing the first term with $\max_{i\neq y_t}\left\{f(\mathbf{x}_{adv})_i\right\}-f(\mathbf{x}_{adv})_{y_t}$, where y_t is the target label.

Sparse loss. For general adversarial attack, perturbation should also be minimal such that the adversarial image appears as a legitimate image similar to the input image. Here for the ℓ_0 sparse adversarial attack, the constraint on perturbation is a small ℓ_0 norm. In our framework the perturbation is generated by $\boldsymbol{\delta}=\mathcal{G}(\mathbf{x})=\mathbf{r}\cdot\mathbf{m}$ and $\mathbf{r}$ is continuous, so the degree of sparsity is mainly controlled by $\mathbf{m}$. Since the ℓ_0 norm is non-differentiable, we cannot incorporate it directly as a regularization term in the objective function. As we discussed in subsection 3.1, the minimization of ℓ_0 can be relaxed to ℓ_1. Thus we use the ℓ_1 regularization on $\mathbf{m}$,

$$\mathcal{L}_{sparse}(\mathbf{m})=\|\mathbf{m}\|_1. \tag{6}$$

The above function urges the perturbation to be as sparse as possible and the resultant perturbation has a dynamic pixel number according to the convergence degree of generator $\mathcal{G}$, hyper-parameters setting, etc.

Quantization loss. We have designed a special quantization operator which is in different state in training and test. This may bring some test error due to the information loss caused by quantization. To reduce the gap between training and test, a straightforward solution is to reduce the quantization error of $\boldsymbol{\varrho}$. The quantized parameter should approximate the full-precision parameter as closely as possible, expecting the test performance will be close to that of training. The quantization loss is:

$$\mathcal{L}_{quanti}(\boldsymbol{\varrho})=\|\boldsymbol{\varrho}-\mathbf{m}\|_2. \tag{7}$$

Overall loss. The overall loss is:

$$\mathcal{L}=\mathcal{L}_{adv}+\lambda_s\mathcal{L}_{sparse}+\lambda_q\mathcal{L}_{quanti}, \tag{8}$$

where λ_s and λ_q are hyper-parameters controlling the relative importance of the sparse and quantization losses, respectively. $\mathcal{L}_{sparse}$ and $\mathcal{L}_{quanti}$ encourage the generated perturbation to be sparse in the inference stage, while $\mathcal{L}_{adv}$ optimizes for a high attack success rate. After training, the generator $\mathcal{G}$ can generate an adversarial image for any input image and can be tested on any black-box model.

4. Experiments

Experimental setup. We perform experiments on the widely used ImageNet [8] dataset. We generate adversarial samples by attacking an Inception-v3 (IncV3) model [40] and a Resnet50 (Res50) model [18] respectively. For our proposed method, we use the source model to supervise the training of generator and directly translate original images into adversarial images during inference time. When using IncV3 as the source model, the input image size is cropped to 299×299 while for Res50 is 224×224. For target models, we also use a VGG16 without batch normalization (VGG16) [37] and a Densenet161 (Dense161) [19]. To make an accurate comparison, we generate adversarial samples with 5000 images randomly selected from the ImageNet validation set.

Implementation of generator. In our experiments we use a popular residual network generator architecture to translate natural images into adversarial examples. Similar networks are often used in an image-to-image translation task [21, 23, 45]. Since the residual network generator is fully convolutional, it can be applied to images of any resolution. The input and output are both color images of identical shape. For the encoder, it contains three stride-2 convolutions and six residual blocks [18]. For both two decoders, we use the same architecture which consists of three 1/2-strided convolutions, except that the channel of output layer is 3 for $\mathcal{D}_1$ and 1 for $\mathcal{D}_2$ respectively. The generator in the following experiments is trained with the whole ImageNet training set.

Baselines. Throughout our experiments we rely on three standard sparse attack strategies which can be adapted to the transfer-based setting: PGD$_0$ [7], SparseFool [27], and GreedyFool [9]. We use the official implementation of these methods. PGD$_0$ needs a pre-defined number of perturbation pixels and calculate the fooling rate under such a pre-defined number, while SparseFool and GreedyFool perturb an image with a dynamic pixel number and run until a successful attack or iteration upper bound. Thus for a fair comparison, we report results after finetuning hyper-parameters to get a comparable sparsity.

Hyper-parameters. For PGD$_0$, it does not find an adversarial example for each test point due to the fixed maximum number of pixels that can be modified. Therefore, we directly pre-define a sparsity number similar to our method for it. For SparseFool, λ is its only hyper-parameter and can be easily adjusted to meet the corresponding needs in terms of fooling rate, sparsity, and complexity. For GreedyFool, since it also uses the C&W loss, the κ in Equation 5 controls the attack strength. For example, when $\kappa = 0$, the attack stops once the generated adversarial sample is adversary. When $\kappa > 0$, pixels keep increasing until the logit difference $\max_{i \neq y} \{f(\mathbf{x}_{adv})_i\} - f(\mathbf{x}_{adv})_y > \kappa$. We finetune λ and κ until their sparsity is similar to our method and then use the fooling rate on black-box target model for evaluation. For the sake of completeness, we also report results which inherit all hyper-parameter settings from their respective papers. For our method, we set $\tau = 0.5$ in Equation 3, $p = 0.5$ in Equation 4, $\kappa = 0$ in C&W loss. λ_s and λ_q in Equation 8 are finetuned in different sparsity settings.

4.1. Transferability evaluation

In this section, we evaluate the transferability under different perturbation ℓ_∞ norm constraint. We report both fooling rates on white-box and black-box models under a similar sparsity setting. A higher fooling rate on black-box target models indicates a better adversary. The sparsity is the average proportion of disturbed pixels in a single image. We also report the average speed of computing an adversarial image.

Non-targeted result. First of all, we show some visual examples in Figure 2. When $\ell_\infty = 255$, the perturbation is marginally visible for both GreedyFool and ours. Quantitative results for $\ell_\infty = 255$ on ImageNet are shown in Table 1. For this setting, under a similar sparsity, the inference speed of our method is only 0.006 seconds, which is several orders of magnitude faster than other methods. As the sparsity increases, the transferability of baselines increases, while our method is still better than others with a large margin. When the source model is IncV3, comparing PGD$_0$, SparseFool (λ = 10) and GreedyFool (κ = 15) with ours, our method obtains better transferability with lower sparsity and faster inference speed. When it comes to Res50 as the source model, we find that our method still outperforms other methods in most settings. Note we mainly focus on the transfer rate (the white-box attack success rate is shown for comprehensiveness). The test data is unseen when updating our generator, and therefore our method gets a lower success rate on the white-box model compared with some baselines methods like GreedyFool.

Currently, our method encourages sparsity indirectly. It is easy to achieve a version that can satisfy a hard constraint. For example, to satisfy a maximum number of perturbed pixels k, iteratively choose k pixels on the generated sparse map $\mathbf{m}$ to modify, based on the largest magnitude of $\mathbf{r}$ in Equation 2. Now, we test with a fixed threshold, 500 for the ℓ_0 norm. Results in Table 2 show the superiority of our method in such a hard constraint setting. Designing a more adaptive method for crafting ℓ_0-based perturbation with a fixed upper bound is left for an interesting future work.

When ℓ_∞ = 10, the inference process of our method is still very fast, shown in Table 3. The transferability of our method is also still better than others with a large margin except when transferring from Res50 to IncV3.

Targeted attack result. We then explore the much harder targeted attack. As SparseFool operates as a non-targeted attack, here we compare with PGD$_0$ and Greedy-

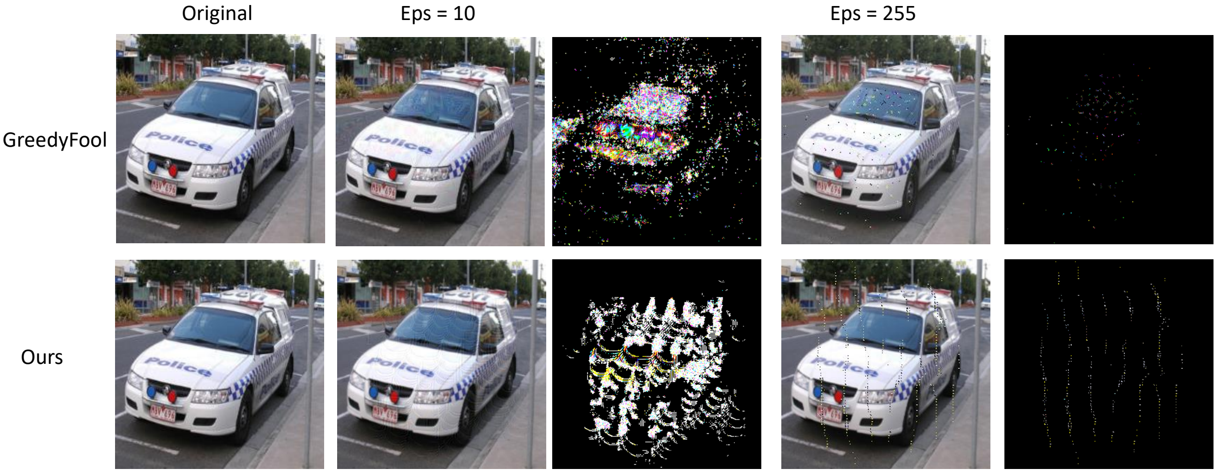

Figure 2. Visual comparison between a state-of-the-art method, GreedyFool (top row), and our method (bottom row).

Source	Method	Sparsity (%)	Speed (s)	IncV3 (%)	Res50 (%)	VGG16 (%)	Dense161 (%)
IncV3	PGD_0	0.56	62.19	56.50*	21.95	23.60	9.69
	SparseFool	0.26	13.80	99.90*	7.34	14.24	5.04
	SparseFool(λ=10)	0.52	8.80	100.00*	11.76	24.50	6.96
	GreedyFool	0.11	7.05	100.00*	2.16	5.38	1.38
	GreedyFool(κ=15)	0.67	63.12	100.00*	15.09	26.37	11.94
	Ours	0.46	**0.006**	61.24*	**63.76**	**85.94**	**46.22**
Res50	PGD_0	0.60	18.62	20.54	75.74*	43.50	16.72
	SparseFool	0.41	14.23	21.56	98.74*	25.34	9.90
	SparseFool(λ=10)	0.66	5.81	27.18	100.00*	35.40	13.56
	GreedyFool	0.22	6.74	2.52	100.00*	8.88	1.80
	GreedyFool(κ=15)	0.75	22.87	**29.12**	100.00*	43.88	30.09
	Ours	0.59	**0.006**	25.90	79.04*	**85.96**	**60.18**

Table 1. $\ell_\infty = 255$ constrained non-targeted attack transferability comparison on ImageNet dataset. The best speed and transfer rate are shown in bold. '*' means white-box setting.

Method	IncV3 (%)	Res50 (%)	VGG16 (%)	Dense161 (%)
PGD_0	56.50*	21.95	23.60	9.69
GreedyFool	100.00*	9.58	10.15	6.32
Ours	58.42*	**61.80**	**84.30**	**43.72**

Table 2. $\ell_\infty = 255$ constrained non-targeted attack transferability comparison on ImageNet dataset, with a hard constraint $\ell_0 = 500$. The best transfer rate are shown in bold. '*' means white-box setting. The source model is IncV3.

Fool on ImageNet dataset in Table 4. Running on all 1000 target classes of ImageNet is too time consuming, so we randomly choose a target class 'bubble'(ID:971). We find it is hard for all methods to find transferable targeted sparse adversarial samples. Though, our method still apparently improves transferability.

4.2. Comparison with generator-based methods

Our framework is based on a generator architecture, which directly outputs a sparse perturbation. Here we compare it with generator-based dense attack methods, including GAP [33] and cross-domain perturbations [29]. We follow their work to set ℓ_∞ = 10 and use IncV3 as the source model. Results are shown in Table 5. Both dense attacks achieve a high transfer rate with near all-pixel perturbations. We tune the λ_s in Equation 8 to obtain results with diverse sparsity. With λ_s increasing, both the sparsity and transfer rate drops, showing the number of modified pixels affects transferability when the pixel budget is limited. However, when $\lambda_s = 5\times10^{-6}$, the transfer rate of our method is com-

Source	Method	Sparsity (%)	Speed (s)	IncV3 (%)	Res50 (%)	VGG16 (%)	Dense161 (%)
IncV3	PGD_0	14.54	63.28	97.89*	9.70	12.73	8.16
	SparseFool	1.65	29.60	99.98*	4.94	9.10	4.08
	SparseFool(λ=10)	12.56	83.85	100.00*	7.99	12.63	11.40
	GreedyFool	0.55	2.79	100.00*	0.94	0.58	2.08
	GreedyFool(κ=40)	18.19	84.67	100.00*	10.67	11.24	6.67
	Ours	14.47	**0.006**	87.72*	**45.32**	**50.38**	**28.98**
Res50	PGD_0	9.96	18.36	11.38	99.54*	21.42	20.74
	SparseFool	1.27	7.80	2.92	99.96*	2.94	2.02
	SparseFool(λ=15)	9.72	36.00	11.87	100.00*	13.39	14.23
	GreedyFool	0.59	1.22	3.20	100.00*	2.76	1.42
	GreedyFool(κ=30)	12.64	20.49	**12.35**	100.00*	17.09	20.89
	Ours	10.52	**0.006**	9.20	72.90*	**39.48**	**51.18**

Table 3. $\ell_\infty = 10$ constrained non-targeted attack transferability comparison on ImageNet dataset. The best speed and transfer rate are shown in bold. '*' means white-box setting.

Target class	Method	Sparsity (%)	Speed (s)	IncV3 (%)	Res50 (%)	VGG16 (%)	Dense161 (%)
'bubble' (ID:971)	PGD_0	0.56	58.53	0.00*	2.25	6.50	0.38
	GreedyFool	0.42	25.42	99.90*	0.10	0.16	0.06
	Ours	0.55	**0.006**	35.38*	**10.38**	**9.08**	**3.66**

Table 4. Targeted attack transferability comparison. The source model is IncV3 and attacks are performed on ImageNet dataset, with $\ell_\infty = 255$ constraint. The best speed and transfer rates are shown in bold. '*' means white-box setting.

petitive with the two dense attacks while our perturbation is sparser. This demonstrates that some modified pixels of dense attacks are redundant for a successful attack. Therefore, sparse adversarial examples as a natural consequence of removing redundancy can be as transferable as dense examples.

4.3. Ablation study

We further analyze the contributions of key parts in our framework toward sparsity and transferability. Here we set ℓ_∞ = 255 and use IncV3 as the source model. Results are shown in Table 6.

Effects of decoupling. In the following, we evaluate the effect of modules in our framework. To prove the importance of decoupling, we delete the path of $\mathcal{D}_2$ in the proposed framework and then use single path of $\mathcal{D}_1$ to output $\boldsymbol{\delta}$, on which sparse loss $\mathcal{L}_{sparse}(\boldsymbol{\delta})$ is added. We mark this setting as 'w/o decoupling' and its sparsity is 82.88%, demonstrating that directly applying sparse regularization on the disturbance $\boldsymbol{\delta}$ instead of the decoupled mask $\mathbf{m}$ cannot lead to a real sparse solution.

Effects of quantization. Then we turn to the quantization operator. To show the effectiveness of our proposed operator, we compare it with training without quantization (p = 0). Results show that without quantization in training, the attack can still achieve a competitive sparsity but the transferability drops. We further compare with straight through estimator (STE) [2]. STE is a popular technique in binary neural networks [20] to address the gradient problem occurring when training deep networks binarized by function. It chooses the identity function to approximate the derivative of the sign function. When using STE, we just change the quantization operator to STE and keep all the other settings. We find that the proposed method is better than STE in both sparsity and transfer rate.

Effects of losses. We study the effect of sparse loss and quantization loss. Without sparse loss, the attack degrades to a dense attack with a 100% sparsity. Without quantization loss, the sparsity raises and meanwhile the transferability decreases, which indicates that the information loss in quantization will lead to test error.

More data boosts transferability. Our generator is trained with a large amount of data because we assume that more data can help alleviate overfitting. Here we empirically prove this assumption by comparing our method with 'ad-hoc'. The 'ad-hoc' setting means that, for each test image $\mathbf{x}$, we utilize the architecture in Figure 1 to optimize a generator, which learns a mapping between $\mathbf{x}$ and its corresponding adversarial image $\mathbf{x}_{adv}$. The output of this generator, i.e., the generated adversarial image $\mathbf{x}_{adv}$, is then used

Method	Sparsity	IncV3	Res50	VGG16	Dense161
GAP	98.98	91.62*	82.42	86.26	72.14
Cross-domain perturbations	99.91	98.10*	88.96	95.86	83.76
Ours (λ_s=5$\times$10^{-6})	39.26	99.06*	86.84	90.54	76.82
Ours (λ_s=1$\times$10^{-5})	27.54	97.82*	70.90	75.82	54.34
Ours (λ_s=1$\times$10^{-4})	14.47	87.72*	45.32	50.38	28.98
Ours (λ_s=2$\times$10^{-4})	7.98	71.36*	32.82	36.00	20.60

Table 5. Comparison with generator-based dense attacks. Results are sparsity (%) and fooling rate (%) on different models. '*' means white-box setting.

Method	Sparsity	IncV3	Res50	VGG16	Dense161
The proposed	0.46	61.24*	63.76	85.94	46.22
w/o decoupling	82.88	-	-	-	-
$p = 0$	0.47	19.92*	29.06	50.88	21.44
q = STE [2]	3.26	39.30*	50.06	71.36	38.16
w/o sparse loss	100.00	-	-	-	-
w/o quantization loss	0.93	53.04*	51.94	75.98	40.92
ad-hoc	0.47	87.07*	43.97	65.52	23.28

Table 6. Ablation study of the proposed framework. Results are sparsity (%) and fooling rate (%) on different models (fooling rate is not studied if sparsity is not satisfactory). '*' means white-box setting.

for evaluation. The 'ad-hoc' optimizes a generator only with a single image, while our method train a generator with a number of natural images. The result in Table 6 shows that 'ad-hoc' achieves a higher white-box success rate but a lower transfer rate. This demonstrates adversarial examples crafted by 'ad-hoc' are easier to overfit the white-box model. By training with more data, our method alleviates the overfitting and promotes the transferability.

5. Discussion

Limitations. The limitations of the proposed method are two folds. First, compared with optimization-based methods, our method increases the cost of training, including data amount and training time. Second, our method needs to carefully finetune the hyperparameter λ_s to get a trade-off between sparsity and transferability. For the assumption that more data can alleviate the overfit issue and therefore boost transferability, we have only empirically validate it and theoretical analysis may be considered in the future.

Social impact. Adversarial attacks may cause security problems. Note that our purpose is to verify the existence of transferable sparse adversarial attacks. If the sparse adversarial perturbations have low attack transferability, there is no need to specifically design a defense method for this type of attack. Our work highlights the potential danger of transfer-based black-box sparse adversarial attacks and the need for the community to defend against this type of attack. One straightforward defense method is adversarial training by using our generated adversarial examples.

6. Conclusion

In this paper, we propose a generator-based sparse adversarial attack framework. Under the same sparsity setting, it can achieve stronger transferability than existing state-of-the-art methods with a faster inference speed. Empirically, we observe that our approach leads to state-of-the-art results when generating attacks on the large scale ImageNet. Our work sheds light on the existence of transferable ℓ_0 based sparse adversarial examples and illustrates state-of-the-art white-box sparse attack methods tend to find adversarial examples which have the least number of modified pixels but do not transfer. Both types of sparse adversarial attack are equally important to analyse the vulnerability of DNNs and evaluate security risks such as creating transferable adversarial patches in the physical world to deceive autonomous cars.

Acknowledgement

This work was supported by the National Key Research and Development Program of China under Grant No.2020AAA0140003 and National Natural Science Foundation of China under Grant 61972395.

References

[1] Shumeet Baluja and Ian Fischer. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387*, 2017. 1, 2

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 7, 8

[3] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017. 1

[4] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z. Morley Mao. Adversarial Sensor Attack on LiDAR-based Perception in Autonomous Driving. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2267–2281, 2019. 1

[5] Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 39–57, 2017. 1, 4

[6] Francesco Croce, Maksym Andriushchenko, Naman D Singh, Nicolas Flammarion, and Matthias Hein. Sparse-rs: a versatile framework for query-efficient sparse black-box adversarial attacks. *arXiv preprint arXiv:2006.12834*, 2020. 2

[7] Francesco Croce and Matthias Hein. Sparse and imperceivable adversarial attacks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4724–4732, 2019. 1, 2, 5

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 2, 5

[9] Xiaoyi Dong, Dongdong Chen, Jianmin Bao, Chuan Qin, Lu Yuan, Weiming Zhang, Nenghai Yu, and Dong Chen. Greedyfool: Distortion-aware sparse adversarial attack. *Advances in Neural Information Processing Systems*, 33, 2020. 1, 2, 5

[10] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting Adversarial Attacks with Momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9185–9193, 2018. 1

[11] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading Defenses to Transferable Adversarial Examples by Translation-Invariant Attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4312–4321, 2019. 1

[12] Yinpeng Dong, Hang Su, Baoyuan Wu, Zhifeng Li, Wei Liu, Tong Zhang, and Jun Zhu. Efficient Decision-Based Black-Box Adversarial Attacks on Face Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7706–7714, 2019. 1

[13] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust Physical-World Attacks on Deep Learning Visual Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018. 1

[14] Yanbo Fan, Baoyuan Wu, Tuanhui Li, Yong Zhang, Mingyang Li, Zhifeng Li, and Yujiu Yang. Sparse adversarial attack via perturbation factorization. In *Computer Vision European Conference*, volume 12367 of *Lecture Notes in Computer Science*, pages 35–50, 2020. 2

[15] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014. 1

[16] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *Proceedings of International Conference on Learning Representations*, 2015. 1

[17] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. Adversarial Examples for Malware Detection. In *Proceedings of the European Symposium on Research in Computer Security*, pages 62–79, 2017. 1

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 2, 5

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017. 5

[20] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 4114–4122, 2016. 7

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 5

[22] Lin Jiadong, Song Chuanbiao, He Kun, Wang Liwei, and John E Hopcroft. Nesterov Accelerated Gradient and Scale Invariance for Adversarial Attacks. In *Proceedings of International Conference on Learning Representations*, 2020. 1

[23] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711, 2016. 5

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 1

[25] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Xiaodong Song. Delving into transferable adversarial examples and black-box attacks. In *Proceedings of International Conference on Learning Representations*, 2017. 1

[26] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of International Conference on Learning Representations*, 2018. 1, 2

[27] Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Sparsefool: a few pixels make a big difference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9087–9096, 2019. 1, 2, 3, 5

[28] Konda Reddy Mopuri, Phani Krishna Uppala, and R. Venkatesh Babu. Ask, acquire, and attack: Data-free UAP generation using class impressions. In *Computer Vision European Conference*, volume 11213, pages 20–35, 2018. 1, 2

[29] Muzammal Naseer, Salman H. Khan, Muhammad Haris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-domain transferability of adversarial perturbations. In *Advances in Neural Information Processing Systems*, pages 12885–12895, 2019. 1, 2, 3, 6

[30] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Asia Conference on Computer and Communications Security*, pages 506–519, 2017. 1

[31] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *European Symposium on Security and Privacy*, pages 372–387, 2016. 1, 2

[32] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016. 1

[33] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge J. Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4422–4431, 2018. 1, 2, 3, 6

[34] Ali Rahmati, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Huaiyu Dai. Geoda: a geometric framework for black-box adversarial attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8446–8455, 2020. 2

[35] Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on MNIST. In *Proceedings of International Conference on Learning Representations*, 2019. 2

[36] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 568–576, 2014. 1

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of International Conference on Learning Representations*, 2015. 5

[38] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models. In *Advances in Neural Information Processing Systems*, pages 8322–8333, 2018. 1, 2

[39] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019. 2

[40] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016. 2, 5

[41] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna Estrach, Dumitru Erhan, Ian Goodfellow, and Robert Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. 1

[42] Baoyuan Wu and Bernard Ghanem. ℓ_p-box admm: A versatile framework for integer programming. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1695–1708, 2018. 2

[43] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 3905–3911, 2018. 1, 2

[44] Pu Zhao, Sijia Liu, Yanzhi Wang, and Xue Lin. An admm-based universal framework for adversarial attacks on deep neural networks. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1065–1073, 2018. 2

[45] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 5