LG-GAN: Label Guided Adversarial Network for Flexible Targeted Attack of Point Cloud-based Deep Networks

Hang Zhou[1,*], Dongdong Chen[2,*], Jing Liao[3], Kejiang Chen[1], Xiaoyi Dong[1],
Kunlin Liu[1], Weiming Zhang[1,†], Gang Hua[4], Nenghai Yu[1]
[1]University of Science and Technology of China, [2]Microsoft Research
[3]City University of Hong Kong, [4]Wormpex AI Research
{zh2991,chenkj,dlight,lkl6949}@mail.ustc.edu.cn; cddlyf@gmail.com;
jingliao@cityu.edu.hk; {zhangwm, ynh}@ustc.edu.cn; ganghua@gmail.com

Abstract

*Deep neural networks have made tremendous progress in 3D point-cloud recognition. Recent works have shown that these 3D recognition networks are also vulnerable to adversarial samples produced from various attack methods, including optimization-based 3D Carlini-Wagner attack, gradient-based iterative fast gradient method, and skeleton-detach based point-dropping. However, after a careful analysis, these methods are either extremely slow because of the optimization/iterative scheme, or not flexible to support targeted attack of a specific category. To overcome these shortcomings, this paper proposes a novel label guided adversarial network (LG-GAN) for real-time flexible targeted point cloud attack. To the best of our knowledge, this is the first generation based 3D point cloud attack method. By feeding the original point clouds and target attack label into LG-GAN, it can learn how to deform the point clouds to mislead the recognition network into the specific label only with a single forward pass. In detail, LG-GAN first leverages one multi-branch adversarial network to extract hierarchical features of the input point clouds, then incorporates the specified label information into multiple intermediate features using the label encoder. Finally, the encoded features will be fed into the coordinate reconstruction decoder to generate the target adversarial sample. By evaluating different point-cloud recognition models (*e.g.*, PointNet, PointNet++ and DGCNN), we demonstrate that the proposed LG-GAN can support flexible targeted attack on the fly while guaranteeing good attack performance and higher efficiency simultaneously.*

[*]Equal contribution, † Corresponding author

1. Introduction

Deep neural networks (DNNs) have been successfully applied to many computer vision tasks [31, 12, 26, 6, 41, 19]. Recently, pioneering works such as DeepSets [40], PointNet [5] and its variants [25, 33] explored the possibility of reasoning with point clouds through DNNs for understanding geometry and recognizing 3D structures. These methods directly extract features from raw 3D point coordinates without utilizing hand-crafted features, such as normals and curvatures, and present impressive results for 3D object classification and semantic scene segmentation.

However, recent works have demonstrated that DNNs are vulnerable to adversarial examples, which are maliciously created by adding imperceptible perturbations to the original input by attackers. This would potentially bring security threats to real application systems such as autonomous driving [2], speech recognition [7] and face verification [21], to name a few. Similar to images, many recent works [36, 20, 42] have shown that deep point cloud recognition networks are also sensitive to adversarial examples and readily fooled by them.

Existing 3D adversarial attack methods can be roughly categorized into three: optimization-based methods such as L-BFGS [32] and C&W attack [3], gradient-based methods such as FGSM [8] and IFGM [9], and skeleton-detach based methods like [38]. Notwithstanding their demonstrated effectiveness in attack, they all have some significant limitations. For example, the former two rely on a very time-consuming optimization/iterative scheme. This renders them not flexible enough for real-time attack. In contrast, the last one is faster but has lower attack success rates. Moreover, it requires to delete some critical point cloud structures and only supports the untargeted attack.

To overcome these shortcomings and motivated by image adversarial attack methods [1, 37, 10], this paper proposes the first generation based point cloud attack method, which is faster, has better attack performance and supports flexible targeted attack of a specific category on the fly.

Specifically, we design a novel label guided adversarial network "LG-GAN". By feeding the original point clouds and target attack label into LG-GAN, it learns how to deform the point cloud with minimal perturbations to mislead the recognition network into the specific label only with a single forward pass. Specifically, LG-GAN first leverages one multi-branch generative network to extract hierarchical features of the input point clouds, then incorporates the specified label information into multiple intermediate features by a label encoder. Finally, the transformed features will be fed into the coordinate reconstruction decoder to generate the target adversarial sample. To further encourage the "LG-GAN" to produce a visually pleasant point cloud, a graph patch-based discriminator network is leveraged for adversarial training.

To demonstrate the effectiveness, we evaluate the proposed "LG-GAN" by attacking different state-of-the-art point-cloud recognition models (*e.g.*, PointNet, PointNet++ and DGCNN). Experiments demonstrate that it can achieve good attack performance and better efficiency simultaneously for flexible targeted attack.

In summary, our contributions are three fold:

- Motivated by the limitations of existing 3D adversarial attack methods, we propose the first generation based adversarial attack method for 3D point-cloud recognition networks.
- To support arbitrary-target attack, we design a novel label guided adversarial network "LG-GAN" by multiple intermediate feature incorporation.
- Experiments on different recognition models demonstrate that our method is both more flexible and effective in targeted attack while being more efficient.

2. Related Work

2.1. 3D point recognition

To obtain great 3D point recognition performance, the most important component is the feature extractor design. In the era before deep learning, various kinds of handcraft 3D descriptors are proposed. For example, the shape distribution is exploited in [23] to calculate the similarity based on distance, angle, area, and volume between random surface points. Based on the native 3D representations of objects, classical shape-based descriptors include voxel grid [23], polygon mesh [17], and 3D SIFT and SURF descriptors [27, 16]. Recently, thanks to the strong learning capability of deep networks, more deep learning-based descriptors are proposed.

In [29], a multi-view convolutional neural network (MVCNN) is proposed to fuse multiple-view features by a pooling procedure. The pioneering work PointNet [25] further designs a novel neural network that can directly consume point clouds and achieves superior recognition performance. By considering more local structure information, it is further improved in the following work PointNet++ [25]. Another representative work is DGCNN [33], which exploits local geometric structures by constructing a local neighborhood graph and applying convolution-like operations on the edges connecting neighboring pairs of points in the spirit of graph neural networks. To evaluate the effectiveness and generalization ability of our method, we will try to attack these three representative methods respectively.

2.2. Existing 3D adversarial attack methods

As discussed before, 3D recognition networks are also vulnerable to adversarial attacks. We can roughly divide existing 3D adversarial attack methods into three different categories: optimization-based methods [36], gradient-based methods [20, 38, 42], and skeleton-based methods [38]. For optimization-based methods, Xiang *et al.* [36] propose a C&W based framework [3] to generate adversarial point clouds by point perturbation, point adding and cluster adding. This method uses an optimization objective to seek the minimal perturbated sample that can make the recognition network classify incorrectly.

Typical perturbation measurements include L_2 norm, Hausdorff distance, and Chamfer distance. For gradient-based methods, Liu *et al.* [20] extended the fast/iterative gradient method by constraining the perturbation magnitude onto an epsilon ball surface in different dimensions. Yang *et al.* [38] developed a variant of one-pixel attack [30] by using pointwise gradient method to only update the attached points without changing the original point cloud. Zheng *et al.* [42] proposed point dropping based attack by first constructing a gradient-based salience map and dropping points with lowest salience scores.

Though optimization-based and gradient-based methods can achieve pretty good attack performance, they are both extremely slow. Motivated by the fact that the recognition result of one 3D object in PointNet [24] is often determined by a critical subset, Yang *et al.* [38] developed a skeleton-detach based attack method by iteratively detaching the most important points from this critical subset. This method is faster but its attack success rate is not that high. More importantly, this method cannot support targeted attack. To overcome all the aforementioned limitations, we propose the first generation based attack method, which enables faster and better flexible targeted attacking.

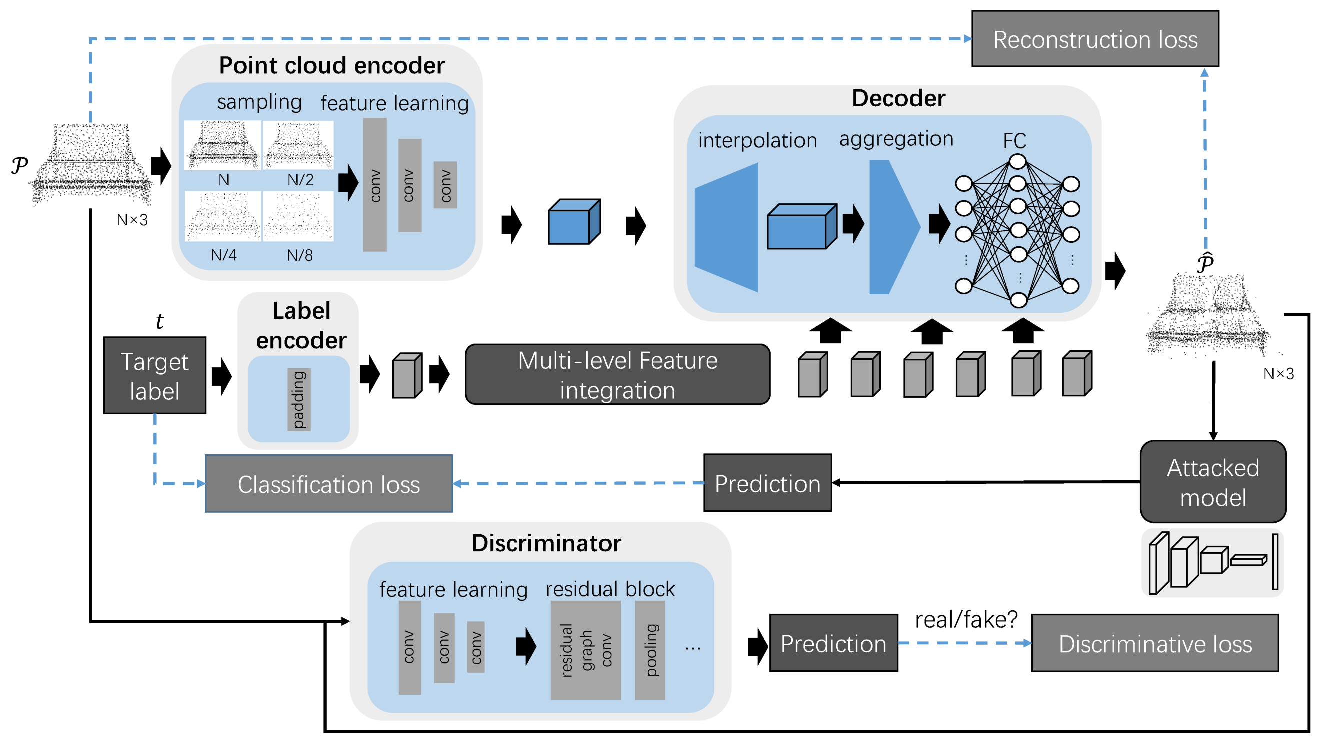


Figure 1: Illustration of the LG-GAN architecture for training the generation of targeted 3D adversarial point clouds. Given an input point cloud $\mathcal{P}$ of N points with XYZ coordinates, a hierarchical feature embedding network (implemented with PointNet++ [25] layers) subsamples and learns deep features on the points. Besides, a label encoder transforms the label information into a label feature to be better concatenated with the point cloud feature at each feature level. The decoder network includes interpolation, aggregation, and FC layers. The discriminator network $\mathcal{D}$ includes a feature learning net and multiple residual blocks. Image best viewed in color.

2.3. Existing 3D adversarial defense methods

To defend against the 3D adversarial attacks, we can derive from the well-known image adversarial defense strategies. For example, by augmenting the training set with adversarial examples, adversarial training [8, 18] can significantly increase the model's robustness. Other simple defense methods include random point sampling, Gaussian noising and quantification. Recently, Zhou *et al.* [43] deploy a statistical outlier denoiser and a data-driven upsampling network as the pre-processing operation before the input is fed into the recognition network. The denoiser contributes to removing outlier based noise patterns as a non-differentiable layer while the upsampling network help purify the perturbed point clouds. In this paper, these defense methods are considered to evaluate the attack performance.

3. Generation-based 3D Point Cloud Attack

Motivated by the shortcomings of existing methods, we propose a novel generation based 3D point cloud attack method "LG-GAN", as shown in Fig. 1. It consists of two sub-networks: a generative network $\mathcal{G}$ that learns how to transform the input point clouds into targeted adversarial samples and a discriminator network $\mathcal{D}$ that encourages the outputs of $\mathcal{G}$ indistinguishable from clean point clouds. These two sub-networks are trained in an adversarial manner, and the discriminator $\mathcal{D}$ is just an auxiliary network and not needed anymore after training.

3.1. Label guided Adversarial network $\mathcal{G}$

Given a clean point cloud $\mathcal{P}$ of total N points and the target label t to which we want the recognition network to misclassify, $\mathcal{G}$ aims to learn how to transform $\mathcal{P}$ into an adversarial sample $\hat{\mathcal{P}}$ with minimal perturbations to $\mathcal{P}$ based on t, which is constrained by three designed losses (classification loss, reconstruction loss and discriminative loss). It mainly consists of three different parts: a label encoder $\mathbf{E}_l$, a hierarchical point feature encoder $\mathbf{E}_p$, and a point decoder $\mathbf{D}_p$. Specifically, $\mathbf{E}_l$ is responsible for encoding label t into a latent label code z_t by repeating t for multiple times to match the point number of each feature layer i, and $\mathbf{E}_p$ encodes $\mathcal{P}$ into a multi-level feature embedding $\overrightarrow{F_{\mathcal{P}}} = (F^1_{\mathcal{P}}, ..., F^m_{\mathcal{P}})$, then z_t will be concatenated with $\overrightarrow{F_{\mathcal{P}}}$ and fed into $\mathbf{D}_p$ to obtain the final adversarial sample $\mathcal{P}_{adv}$. Mathematically:

$$\begin{aligned}
\overrightarrow{z_t} &= (z_t^1, ..., z_t^l) = \mathbf{E}_l(t), \\
\overrightarrow{F_{\mathcal{P}}} &= (F_{\mathcal{P}}^1, ..., F_{\mathcal{P}}^m) = \mathbf{E}_p(\mathcal{P}), \\
\mathcal{P}_{adv} &= \mathbf{D}_p(\overrightarrow{z_t}, \overrightarrow{F_{\mathcal{P}}}).
\end{aligned} \tag{1}$$

Here m is the feature level number (4 by default). Note that t is formatted as a one-hot vector whose tth value is 1.

Hierarchical Point Feature Learning. Progressively extracting features of different scales in a hierarchical manner has been proven to be an effective strategy for capturing both the local and global point structures. By default, in this paper, we use 4 different levels of point cloud representation from coarse to fine. Specifically, given the point cloud $\mathcal{P}$ consisting of N points, we use the iterative farthest point sampling algorithm as the sampling layer to evenly sample the points into four scales $(N, N/2, N/4, N/8)$. Then we adopt PointNet++ [25] to extract the feature embedding for each scale. In detail, for each point in level i, PointNet++ will utilize the local structure information and aggregate the features of neighboring points as the final feature F_P^i, whose shape is $\frac{N}{2^{i-1}} \times (64 \cdot 2^{i-1})$.

Feature Decoding and Label Concatenation to Reconstruct the Final Adversarial Sample. To aggregate the multi-scale features from each level, we follow the strategy used in [13, 11, 39] that directly combine features from different levels and let the network learn the importance of each level. Since the different scales have different point number, the downsampled point features will be upsampled to the original point number by using the interpolation method adopted in PointNet++ [25]. Formally, denote the upsampled feature as ${F_{\mathcal{P}}^i}'$, then:

$${F_{\mathcal{P}}^i}'(x) = \frac{\sum_{j=1}^3 w_j(x) F_{\mathcal{P}}^i(x_j)}{\sum_{j=1}^3 w_j(x)}, \tag{2}$$

where $w_j(x)$ is the contribution weight of the neighborhood point x_j defined as $1/d(x, x_j)$, and d is the ℓ_2 distance by default. Then each scale upsampled features will be processed by one 1×1 convolutional layer similar to that in [39] to have the same dimension.

The m level of features ${F_{\mathcal{P}}^i}'(x)$ together with the original point cloud coordinates are concatenated together along the feature dimension to obtain the final aggregated feature ${F_{\mathcal{P}}}''(x)$. Then, l fully connected layers with multi-layered label concatenation are utilized to reconstruct the detailed coordinates of the adversarial sample. At each fully connected layer i (except for the last layer), we obtain the integrated feature:

$${F_{\mathcal{P}}^{i+1}}''(x) = \text{FC}([z_t^i, {F_{\mathcal{P}}^i}''(x)]), i = 1, ..., l-1. \tag{3}$$

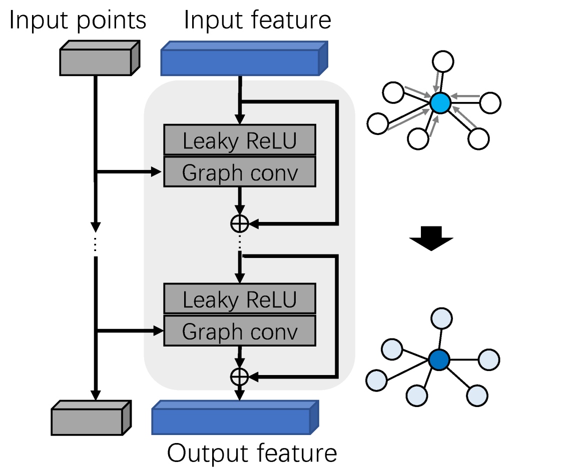


Figure 2: Residual graph convolutional block. The input points are utilized for querying the k nearest neighbors. The output feature of the block has the same dimension as the input feature.

Here $[x, y]$ means concatenating x, y along the feature dimension and l is the number of fully connected layer (4 by default).

3.2. Graph Discriminator $\mathcal{D}$

To help generate a more realistic adversarial point cloud, we further leverage a graph discriminator network $\mathcal{D}$ for adversarial learning. For the detailed network structure, we directly use the existing graph patch GAN [28, 34] to distinguish clean point clouds from generated adversarial point clouds. Then for every locally generated patch, $\mathcal{D}$ will encourage it to lie on the distribution of the clean point clouds.

For the detailed network structure, $\mathcal{D}$ consists of a PointNet++ like feature extraction head and a series of pooling blocks and residual graph convolution blocks. For the feature extraction head, it consists of several pointwise convolutional layers and aggregate the features of k nearest neighbors for each point by max pooling. For the pool block, it will first use the farthest point sampling algorithm to find some seed points then max-pool the corresponding features of their neighboring points to get the downsampled feature. For the residual graph convolution block, it is composed of several graph convolutions defined on the k nearest neighborhood-based graph $G = (v, N(x))$ and residual connections as shown in Fig. 2. Mathematically, the formulation of graph convolution is defined as:

$$f_{out}(x) = w_0 f_{in}(x) + w_1 \sum_{q \in N(x)} f_{in}(q), \quad \forall x \in v, \tag{4}$$

where $N(x)$ is defined by the k nearest neighbors of x in the Euclidean space. f_{in}, f_{out} are the input and output features of the vertex x respectively. w_0, w_1 are the learnable weights that determine how much information needs to be borrowed from the target vertex and its neighborhood vertexes. k is set to 8 by default.

3.3. Objective Loss Function

The objective loss function of the generator networks $\mathcal{G}$ consists of three parts: the classification loss, the reconstruction loss, and the discriminative loss. Formally:

$$\mathcal{L}_{\mathcal{G}} = \mathcal{L}_{cls} + \alpha \mathcal{L}_{rec} + \beta \mathcal{L}_{dis}, \tag{5}$$

where α, β are weight factors. $\mathcal{L}_{cls}$ is the classification loss that urges the attacked model $\mathcal{H}$ to make prediction to the target label t, and we use the standard cross-entropy loss:

$$\mathcal{L}_{cls} = -[t \log \mathcal{H}(\hat{\mathcal{P}}) + (1-t) \log (1 - \mathcal{H}(\hat{\mathcal{P}}))], \tag{6}$$

where $\hat{\mathcal{P}} = \mathcal{G}_\theta(\mathcal{P}, t)$ is the generated adversarial point cloud, $\mathcal{H}(\hat{\mathcal{P}})$ is the predicted probability of the target model on the adversarial sample. $\mathcal{L}_{rec}$ is the reconstruction loss that encourages the generated adversarial point cloud to resemble the original sample. We adopt an ℓ_2 distance, which is better than the Chamfer distance, as our measurement. $\mathcal{L}_{dis}$ is a graph adversarial loss that has the same goal with $\mathcal{L}_{rec}$. Inspired by LS-GAN [22],

$$\mathcal{L}_{dis}(\hat{\mathcal{P}}) = \|1 - \mathcal{D}_\theta(\hat{\mathcal{P}})\|_2^2. \tag{7}$$

The objective loss function of the discriminator networks $\mathcal{D}$ aims to distinguish real and fake point clouds by minimizing $\mathcal{L}_{\mathcal{D}}$ loss:

$$\mathcal{L}_{\mathcal{D}}(\mathcal{P}, \hat{\mathcal{P}}) = \frac{1}{2}\|\mathcal{D}_\theta(\hat{\mathcal{P}})\|_2^2 + \frac{1}{2}\|1 - \mathcal{D}_\theta(\mathcal{P})\|_2^2. \tag{8}$$

3.4. Implementation Details

Following the practice in [3], we adopt PointNet [24] and PointNet++ [25] as the targeted attack models $\mathcal{H}$, and use the default settings to train. Given the pretrained models, we train the proposed LG-GAN to attack the two models. The size of input point cloud is $2,048 \times 3$, and weights α and β are set with 0.001 and 1 respectively. The implementation is based on TensorFlow. For the optimization, we train the network for 200 epochs using the Adam [15] optimizer with a minibatch size of 4, and the learning rate of $\mathcal{G}$ and $\mathcal{D}$ are 0.001 and 0.00001 respectively. The whole training process takes about 8h on the NVIDIA RTX 2080 Ti GPU.

4. Experiments

In this section, we firstly compare our LG-GAN with previous state-of-the-art methods on a CAD object benchmark (Sec. 4.1). We then provide analysis experiments to understand the effectiveness and efficiency of LG-GAN (Sec. 4.2). We also validate translation-based attack (Sec. 4.3). Finally we show qualitative results of our LG-GAN (Sec. 4.4). More analysis and visualizations are provided in the supplementary materials.

4.1. Comparing with State-of-the-art Methods

Dataset. ModelNet40 [35] is a comprehensive clean collection of 3D CAD models for objects, which contains 12,311 objects from 40 categories, where 9,843 are used for training and the other 2,468 for testing. As done by Qi *et al.* [24], before generating adversarial point clouds, we first uniformly sample 2,048 points from the surface of each object and rescale them into a unit cube.

ShapeNetCore [4] is a subset of the full ShapeNet dataset with single clean 3D models and manually verified category and alignment annotations. It covers 55 common object categories with 52,472 unique 3D models, where 41,986 are used for training and the other 10,486 for testing. All the data are uniformly sampled into 4096 points.

Attack evaluations. The attackers generate adversarial examples using the targeted models and then evaluate the attack success rate and detection accuracy of these generated adversarial examples on the target and defense models.

Methods in comparison. We compare our method with a wide range of prior art methods. C&W [36] is an optimization based algorithm with various loss criteria including ℓ_2 norm, Hausdorff and Chamfer distances, cluster and object constraints. FGSM and IFGM [20, 38] are gradient-based algorithms constrained by the ℓ_∞ and ℓ_2 norms, where FGSM is a naïve baseline that subtracts perturbations along with the direction of the sign of the loss gradients with respect to the input point cloud, and IFGM iteratively subtracts the ℓ_2-normalized gradients. Since FGSM is not strong enough to handle targeted attack (less than 30% attack success rate) and point-detach methods [24, 38] cannot targetedly attack pretrained networks, we do not compare these methods in the following experiments.

Results are summarized in Table 1 and Table 2. FGSM has low attack success rates (12.2%). LG-GAN outperforms IFGM methods by at least 23.1% of attack success rates and $5\times$ of generation speed. LG-GAN has similar attack success rate with optimization-based C&W methods, but outperforms speed by $100\times$. In terms of visual quality, LG-GAN performs a little worse than C&W methods since C&W attempt to modify points as little as possible to implement attacks by querying the network multiple times, and thus it is difficult to attack in real time. LG-GAN has better attack ability on robust defense models (gray-box models), with 88.8% and 84.8% attack success rates on simple random sampling (SRS) and DUP-Net [43] defense models, respectively. Notably, we achieve great improvements with regard to attack black-box models. As shown in Table 2, LG-GAN has 11.6% and 14.5% attack success rates on PointNet++ [25] and DGCNN [33] respectively while IFGM [20, 38] only has 3.0%, 2.6% and C&W [36] based methods with 0 and 0.

	Target [5]	Defense (SRS) [43]	Defense (DUP-Net) [43]	ℓ_2 dist (meter)	Chamfer dist (meter)	Time (second)
C&W + ℓ_2 [36]	100	0	0	**0.01**	0.006	40.80
C&W + Hausdorff [36]	100	0	0	—	0.005	42.67
C&W + Chamfer [36]	100	0	0	—	**0.005**	43.73
C&W + 3 clusters [36]	94.7	2.7	0	—	0.120	52.00
C&W + 3 objects [36]	97.3	3.1	0	—	0.064	58.93
FGSM [20, 38]	12.2	5.2	2.8	0.15	0.129	0.082
IFGM [20, 38]	73.0	14.5	3.3	0.31	0.132	0.275
LG + Chamfer (ours)	96.1	75.4	13.9	0.63	0.137	0.037
single-layered LG-GAN (ours)	97.6	80.2	37.8	0.27	0.032	0.053
LG (ours)	97.1	85.0	72.0	0.25	0.028	**0.033**
LG-GAN (ours)	**98.3**	**88.8**	**84.8**	0.35	0.038	0.040

Table 1: Attack success rate (%, second to fourth column), distance (fifth-sixth column) between original sample and adversarial sample (meter per object) and generating time (second per object) on attacking PointNet from ModelNet40. "Target" stands for white-box attacks. The hyper-parameter setting of two gray-box attacks is: for the simple random sampling (SRS) defense model, percentage of random dropped points is 60%$\sim$90%; for DUP-Net defense model, $k = 50$ and $\alpha = 0.9$ from [43]. The default LG-GAN (ours) consists of multi-layered label embedding, ℓ_2 loss and GAN loss.

	Attack type	C&W + ℓ_2 [36]	C&W + Chamfer [36]	IFGM [20, 38]	LG + Chamfer (ours)	single-layered LG-GAN (ours)	LG (ours)	LG-GAN (ours)
PointNet [24]	White-box	100/0	100/0	73.0/0.4	96.1/0.45	97.6/0.16	97.6/0.45	**98.3**/0.54
PointNet++ [25]	Black-box	0/3.1	0/6.2	3.0/3.4	2.8/4.1	9.4/5.3	6.1/8.0	**11.6**/5.2
DGCNN [33]	Black-box	0/9.8	0/5.3	2.6/4.1	5.8/6.2	13.5/7.4	9.4/8.2	**14.5**/15.3

Table 2: Black-box attack success rate/accuracy (%). The adversarial point clouds of ModelNet40 are generated from PointNet [24], and then attack PointNet++ [25] and DGCNN [14].

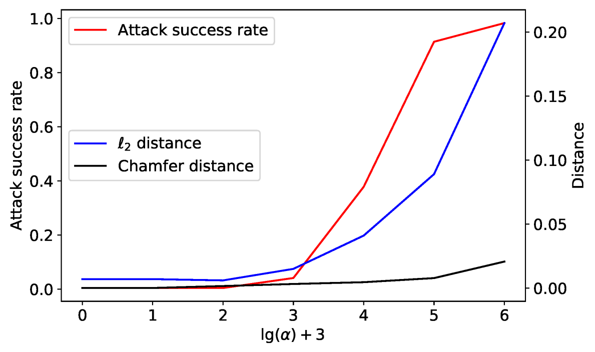


Figure 3: Larger α helps more generated point clouds successfully target-attack PointNet, but affects visual quality of generated objects from ModelNet40. We show the average attack success rate (in red lines) of LG-GAN w.r.t. balancing weight α, (in blue lines) the average ℓ_2 distance, and (in black lines) the average Chamfer distance.

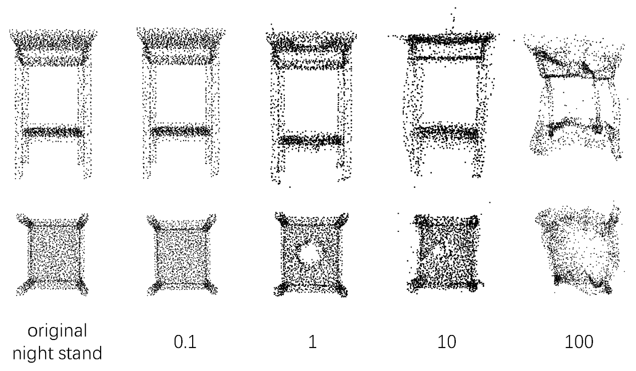
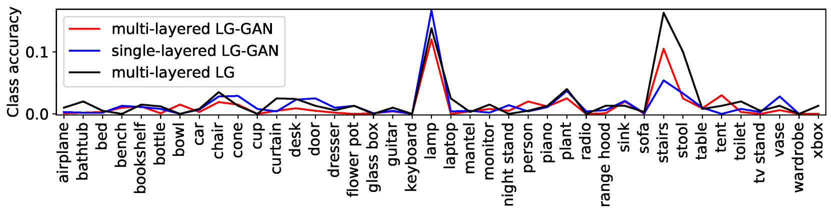


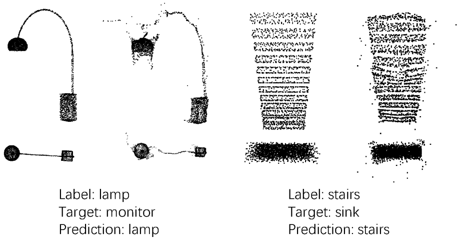
Figure 4: Qualitative results of object "night stand" with different α. It shows (from left to right): the original point cloud from ModelNet40, the generated point cloud with α ranging among 0.1, 1, 10, and 100. Images from top to bottom are the front view and bottom view.

4.2. Ablation Study

Multi-layered or single-layered concatenation? In this part, we explore the influence of multi-layered concatenation for reconstruction result. We retain the bottom label concatenation layer and train such single-layered LG-GAN. Since multi-layered LG-GAN concatenates features repeatedly along the convolution operation, it helps the point cloud feature and label feature bond together in a progressive way. In other words, it is easier for the multi-layered LG-GAN to attack successfully without sacrificing visual quality too much. As shown in Table 1 and Fig. 6, under the same attack success rate, multi-layered LG-GAN performs



(a)



(b)

Figure 5: (a) Class accuracy of generated adversarial point clouds of ModelNet40 averaged over 10 random testing. Some objects from "lamp" and "stairs" classes are harder to targetedly attack; **(b) Visualization of difficult crafts** with failed attacks: objects from "lamp" and "stairs" categories and their adversarial examples.

better than single-layered one.

ℓ_2 loss or Chamfer loss? Chamfer loss is a typical metric for constraining a desirable point cloud to the original sample, which is a looser loss compared with the point-to-point ℓ_2 loss. Because the output of LG-GAN shares the same point number with the input sample, both ℓ_2 loss and Chamfer loss can serve as the reconstruction loss. We argue that since the generation network tries to deform the generated sample to fit the targeted class, it is more difficult to preserve the shape of the point cloud than tackling generation tasks such as surface reconstruction and point cloud upsampling. In the 3D space, Chamfer loss cannot restrain point drift to some extent, and thus visually produces isolated points, which are known as "outliers". Table 1 shows that using ℓ_2 loss performs better than Chamfer loss.

Effect of the GAN loss. Compared to LG-GAN, we remove the discriminator network from the proposed structure, denoted as LG. We argue that the GAN loss helps further constrain the manifold surface and increases surface smoothness, but at the same time bends the shape more. Another benefit of GAN loss is helping suppress the outliers and improving the attack ability on gray-box defense including SRS (+3.8%) and DUP-Net [43] (+12.8%). Both ℓ_2 and Chamfer distances have slightly enlarged because despite point clouds being more compact, they are more distant from the original sample in the distance evaluation in the Cartesian coordinate system.

Visual quality *vs.* attack success rate. Weight factor α controls the balance between the visual quality of an adversarial point cloud and the attack success rate it has. Without the constraint of classification loss (when α is zero), the target of the network is to learn the shape representation of the input object. When α increases, more generated point clouds can successfully target-attack the recognition network, but visual quality gradually decreases. In Fig. 3, we show the average attack success rate and distortion evaluation (*i.e.* ℓ_2 and Chamfer distances) with varying α. When α is set with 100, a superior attack success rate is achieved with 90% or so, and the Chamfer distance is around 0.007, which is slightly larger than C&W with ℓ_2 loss based attacks (0.006). Thus, arguably, LG-GAN is close to optimization-based attacks in most cases. Besides, in Fig. 4 we visualize the generated point clouds with different α. To reach 40% attack success rates (α is 10), it is effortless for LG-GAN to reshape the input object a little bit to achieve the goal of attack with slight perturbation of points on the object surface. The object gradually bends out of shape when α increases, but barely produces scattered outliers compared with gradient-based and optimization-based methods.

The curvilinear trend also reveals that some categories are harder to be attacked targetedly.We average over 10 times of the class accuracy of adversarial samples with LG-GAN and its variances under α is 100, as shown in Fig. 5(a). As can be seen, "lamp" and "stairs" are two categories that are hard to be operated to attack. Similar results can be obtained when attack success rates are low ($< 50\%$). Furthermore, we visualize the objects from the two categories that fail to attack PointNet in Fig. 5(b), and realize the attack ability of generating adversarial point clouds depends on the shape diversity among object classes. It is more difficult to deform "lamp" or "stairs" to other classes. In short, shapes determine the difficulty level of attacks; it is unavoidable to sacrifice visual quality to increase attack ability.

Attack transferability. The results in Table. 2 illustrate that all the attack schemes have limited transferability. On the whole, adversarial examples generated from PointNet [24] have better attack success rate on DGCNN [33] than PointNet++ [25] which indicates that the structures of PointNet and DGCNN are more similar. Nevertheless, LG-GAN performs better than C&W and IFGM methods because adversarial examples generated by LG-GAN have better fluid-like shapes than others, which are still able to fool other unknown networks.

Speed. Our proposed model is very efficient since it only

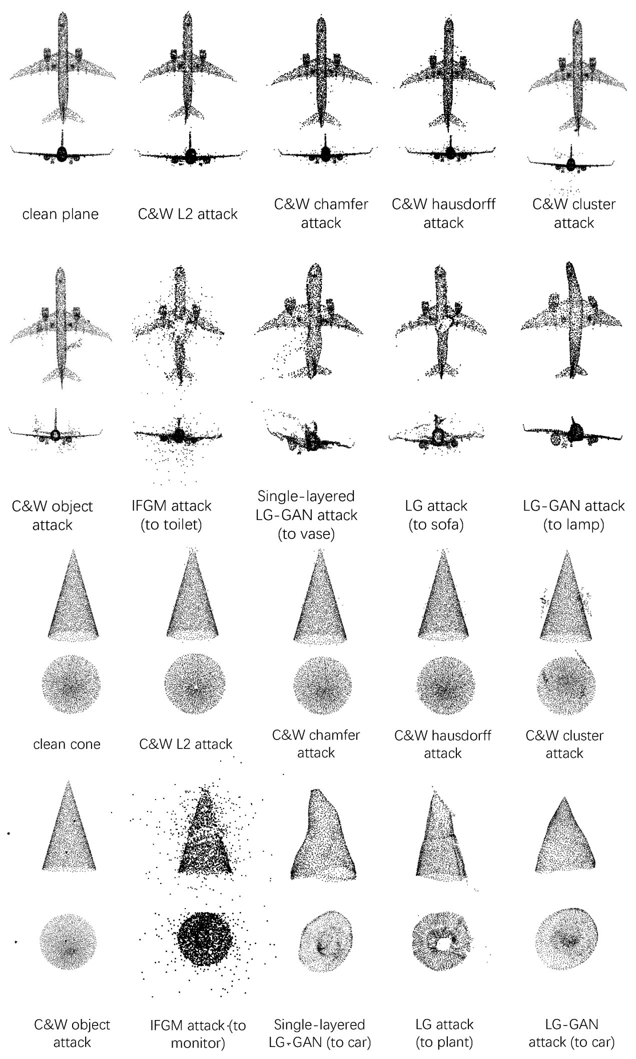


Figure 6: Qualitative results on two point clouds from ModelNet40, "plane" and "cone". See Sec. 4.4 for details. Enlarge to see details.

accesses the target model in the inference stage only once. Compared to previous best methods (Table 1), our model is more than $1000\times$ times faster than C&W and $7\times$ faster than IFGM in speed. Therefore, LG-GAN is more friendly to be utilized in real-time systems.

4.3. Translation Attack

We have designed an alternative attack based on geometric translation. It is observed that the centroids of generated adversarial point clouds are not on the origin of the Cartesian coordinate system compared with the original point clouds. By normalizing to center, attack success rate of LG-GAN has dropped slightly ($98.2\% \rightarrow 82.6\%$), still better than IFGM ($73.0\% \rightarrow 72.8\%$) and C&W with ℓ_2 loss ($100\% \rightarrow 0\%$). We proceed with a second analysis by translating a whole point cloud to random direction and find that the networks are fragile to monolithic translation. More details are given in the supplementary material.

4.4. Qualitative Results

Fig. 6 shows two representative examples of adversarial point clouds ("plane" and "cone") using C&W [36], IFGM [20, 38] and LG-GAN (α is 40) schemes from ModelNet40 dataset. As can be seen, C&W with ℓ_2, Chamfer and Hausdorff losses have the least perturbations among the three schemes, but create outliers far away from the object surface relatively. For C&W with cluster and object losses, they only add points and the original point clouds are not modified. For IFGM, plenty of points are moved to outside or inside of the object, which makes the object more obscure to observe. For LG, without GAN loss, the shapes of point clouds are preferable and easy to recognize. For LG-GAN, due to the constraint of GAN loss, the density of points on the surface has been changed but the points still adjoin the manifold, which is more difficult to recognize. However, as stated above, the advantage of LG-GAN lies in that it can attack outlier-removal based defences with a high success ratio. In a word, there is still a long way to design a nice and distortionless adversarial point cloud that can totally deceive human eyes.

5. Conclusion

In this work we have introduced LG-GAN: a novel label guided adversarial network for arbitrary-target point cloud attack. By feeding the original point clouds and target attack label into LG-GAN, it can learn how to deform the point cloud with minimal perturbations to mislead the recognition network into the specific label only with a single forward pass. In details, LG-GAN first leverages one multi-branch adversarial network to extract hierarchical features of the input point clouds, then utilizes a label encoder to incorporate the specified label information into multiple intermediate features. Finally, the transformed features will be fed into the coordinate reconstruction decoder to generate the target adversarial sample. Experiments show that the proposed LG-GAN can support more flexible targeted attack on the fly while guaranteeing good attack performance and higher efficiency simultaneously.

Acknowledgement This work was supported in part by the Natural Science Foundation of China under Grant U1636201 and 61572452, by Anhui Initiative in Quantum Information Technologies under Grant AHY150400, by National Key Research and Development Program of China under Grant 2018YFB0804100, and by Exploration Fund Project of University of Science and Technology of China under Grant YD3480002001. Gang Hua was supported partly by National Key RD Program of China Grant 2018AAA0101400 and NSFC Grant 61629301.

References

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

[2] Yulong Cao, Chaowei Xiao, Dawei Yang, Jing Fang, Ruigang Yang, Mingyan Liu, and Bo Li. Adversarial objects against lidar-based autonomous driving systems. *arXiv preprint arXiv:1907.05418*, 2019.

[3] 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.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] R Qi Charles, Hao Su, Mo Kaichun, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 77–85. IEEE, 2017.

[6] Dongdong Chen, Jing Liao, Lu Yuan, Nenghai Yu, and Gang Hua. Coherent online video style transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1105–1114, 2017.

[7] Moustapha Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured prediction models. *arXiv preprint arXiv:1707.05373*, 2017.

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

[9] Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

[10] Jiangfan Han, Xiaoyi Dong, Ruimao Zhang, Dongdong Chen, Weiming Zhang, Nenghai Yu, Ping Luo, and Xiaogang Wang. Once a man: Towards multi-target attack via learning multi-target adversarial network once. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5158–5167, 2019.

[11] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 447–456, 2015.

[12] 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.

[13] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip HS Torr. Deeply supervised salient object detection with short connections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3203–3212, 2017.

[14] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 984–993, 2018.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Jan Knopp, Mukta Prasad, Geert Willems, Radu Timofte, and Luc Van Gool. Hough transform and 3d surf for robust three dimensional classification. In *European Conference on Computer Vision*, pages 589–602. Springer, 2010.

[17] Iasonas Kokkinos, Michael M Bronstein, Roee Litman, and Alex M Bronstein. Intrinsic shape context descriptors for deformable shapes. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 159–166. IEEE, 2012.

[18] Alex Kurakin, Dan Boneh, Florian Tramèr, Ian Goodfellow, Nicolas Papernot, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. 2018.

[19] Suichan Li, Dapeng Chen, Bin Liu, Nenghai Yu, and Rui Zhao. Memory-based neighbourhood embedding for visual recognition. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[20] Daniel Liu, Ronald Yu, and Hao Su. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. *arXiv preprint arXiv:1901.03006*, 2019.

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

[22] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017.

[23] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *ACM Transactions on Graphics (TOG)*, 21(4):807–832, 2002.

[24] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4, 2017.

[25] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[27] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360. ACM, 2007.

[28] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017.

[29] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks

for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[30] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019.

[31] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[32] 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.

[33] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 38(5):146, 2019.

[34] Huikai Wu, Junge Zhang, and Kaiqi Huang. Point cloud super resolution with adversarial residual graph networks. *arXiv preprint arXiv:1908.02111*, 2019.

[35] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[36] Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9136–9144, 2019.

[37] 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.

[38] Jiancheng Yang, Qiang Zhang, Rongyao Fang, Bingbing Ni, Jinxian Liu, and Qi Tian. Adversarial attack and defense on point sets. *arXiv preprint arXiv:1902.10899*, 2019.

[39] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018.

[40] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pages 3391–3401, 2017.

[41] Jie Zhang, Dongdong Chen, Jing Liao, Han Fang, Weiming Zhang, Wenbo Zhou, Hao Cui, and Nenghai Yu. Model watermarking for image processing networks. *AAAI 2020*, 2020.

[42] Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. Pointcloud saliency maps. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1598–1606, 2019.

[43] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1961–1970, 2019.