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

Hang Zhou¹, Dongdong Chen²*, Jing Liao³, Kejiang Chen¹, Xiaoyi Dong¹,
Kunlin Liu¹, Weiming Zhang¹,†, Gang Hua⁴, Nenghai Yu¹
¹University of Science and Technology of China, ²Microsoft Research
³City University of Hong Kong, ⁴Wormpex 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.

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 im-
age 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 perturbed 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.
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 $G$ that learns how to transform the input point clouds into targeted adversarial samples and a discriminator network $D$ that encourages the outputs of $G$ indistinguishable from clean point clouds. These two sub-networks are trained in an adversarial manner, and the discriminator $D$ is just an auxiliary network and not needed anymore after training.

3.1. Label guided Adversarial network $G$

Given a clean point cloud $P$ of total $N$ points and the target label $t$ to which we want the recognition network to misclassify, $G$ aims to learn how to transform $P$ into an adversarial sample $P_{adv}$ with minimal perturbations to $P$ 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 $E_l$, a hierarchical point feature encoder $E_p$, and a point decoder $D_p$. Specifically, $E_l$ 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 $E_p$ encodes $P$ into a multi-level feature embedding $\tilde{F}_P = (F_P^1, ..., F_P^m)$, then $z_t$ will be concatenated with $\tilde{F}_P$ and fed into $D_p$ to obtain the final adversarial sample $P_{adv}$. Mathematically:
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 \( P \) 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 \( F^i_\text{p} \), whose shape is \( \frac{N_i}{2^{i-1}} \times (64 \cdot 2^{i-1}) \).

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 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 \( F^\ast_\text{p} \), then:

\[
F^\ast_\text{p}(x) = \frac{\sum_{j=1}^{3} w_j(x)F^j_\text{p}(x_j)}{\sum_{j=1}^{3} w_j(x)},
\]

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 \( \ell_2 \) distance by default. Then each scale upsampled features will be processed by one \( 1 \times 1 \) convolutional layer similar to that in [39] to have the same dimension.

The \( m \) level of features \( F^\ast_\text{p}(x) \) together with the original point cloud coordinates are concatenated together along the feature dimension to obtain the final aggregated feature \( F^\ast_\text{p}(x) \). 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:

\[
F^{i+1'}_\text{p}(x) = \text{FC}([z^i_\text{r}, F^{i'}_\text{p}(x)]), i = 1, ..., l - 1.
\]

Here \( [x, y] \) means concatenating \( x, y \) along the feature dimension and \( \ell_2 \) is the number of fully connected layer (4 by default).

3.2. Graph Discriminator \( D \)

To help generate a more realistic adversarial point cloud, we further leverage a graph discriminator network \( D \) 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, \( D \) will encourage it to lie on the distribution of the clean point clouds.

For the detailed network structure, \( D \) 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:

\[
f_{\text{out}}(x) = w_0f_{\text{in}}(x) + w_1 \sum_{q \in N(x)} f_{\text{in}}(q), \quad \forall x \in v,
\]

where \( N(x) \) is defined by the \( k \) nearest neighbors of \( x \) in the Euclidean space. \( f_{\text{in}}, f_{\text{out}} \) 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 vertices. $k$ 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:

$$L_{\mathcal{G}} = L_{cls} + \alpha L_{rec} + \beta L_{dis}, \quad \text{(5)}$$

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

$$L_{cls} = -[t \log H(\hat{P}) + (1 - t) \log (1 - H(\hat{P}))], \quad \text{(6)}$$

where $\hat{P} = \mathcal{G}_\theta(P, t)$ is the generated adversarial point cloud, $H(\hat{P})$ is the predicted probability of the target model on the adversarial sample. $L_{rec}$ 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. $L_{dis}$ is a graph adversarial loss that has the same goal with $L_{rec}$. Inspired by LS-GAN [22],

$$L_{dis}(\hat{P}) = \|1 - D_\theta(\hat{P})\|^2_2. \quad \text{(7)}$$

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

$$L_D(P, \hat{P}) = \frac{1}{2}\|D_\theta(P)\|^2_2 + \frac{1}{2}\|1 - D_\theta(\hat{P})\|^2_2. \quad \text{(8)}$$

### 3.4. Implementation Details

Following the practice in [3], we adopt PointNet [24] and PointNet++ [25] as the targeted attack models $H$, 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 $\alpha$ and $\beta$ 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 $L_\infty$ 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.
Table 1: Attack success rate (%), 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%~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, \( \ell_2 \) loss and GAN loss.

<table>
<thead>
<tr>
<th>Attack type</th>
<th>C&amp;W + ( \ell_2 ) [36]</th>
<th>C&amp;W + Chamfer [36]</th>
<th>IFGM [20, 38]</th>
<th>LG + Chamfer (ours)</th>
<th>single-layered LG-GAN (ours)</th>
<th>LG (ours)</th>
<th>LG-GAN (ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PointNet [24]</td>
<td>White-box</td>
<td>100 / 0</td>
<td>100 / 0</td>
<td>73.0 / 0.4</td>
<td>96.1 / 0.45</td>
<td>97.6 / 0.16</td>
<td>97.6 / 0.45</td>
</tr>
<tr>
<td>PointNet++ [25]</td>
<td>Black-box</td>
<td>97.1 / 0.7</td>
<td>85.0 / 0.7</td>
<td>5.2 / 0.0</td>
<td>5.2 / 0.0</td>
<td>9.4 / 0.3</td>
<td>6.1 / 0.3</td>
</tr>
<tr>
<td>DGCNN [33]</td>
<td>Black-box</td>
<td>97.1 / 0.7</td>
<td>85.0 / 0.7</td>
<td>5.2 / 0.0</td>
<td>5.2 / 0.0</td>
<td>9.4 / 0.3</td>
<td>6.1 / 0.3</td>
</tr>
</tbody>
</table>

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 \( \alpha \) 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 \( \alpha \) (in blue lines) the average \( \ell_2 \) distance, and (in black lines) the average Chamfer distance.

Figure 4: Qualitative results of object “night stand” with different \( \alpha \). It shows (from left to right): the original point cloud from ModelNet40, the generated point cloud with \( \alpha \) 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...
better than single-layered one.

$\ell_2$ 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 $\ell_2$ loss. Because the output of LG-GAN shares the same point number with the input sample, both $\ell_2$ 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 $\ell_2$ 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 $\ell_2$ 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 $\alpha$ 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 $\alpha$ is zero), the target of the network is to learn the shape representation of the input object. When $\alpha$ 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. $\ell_2$ and Chamfer distances) with varying $\alpha$. When $\alpha$ 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 $\ell_2$ 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 $\alpha$. To reach 40% attack success rates ($\alpha$ 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 $\alpha$ 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 $\alpha$ 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
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% → 82.6%), still better than IFGM (73.0% → 72.8%) and C&W with $\ell_2$ loss (100% → 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 ($\alpha$ is 40) schemes from ModelNet40 dataset. As can be seen, C&W with $\ell_2$, 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


