Shape Prior Guided Attack: Sparser Perturbations on 3D Point Clouds

Zhenbo Shi[1], Zhi Chen[1], Zhenbo Xu[2], Wei Yang[1]*, Zhidong Yu[1]
Liusheng Huang[1]

[1]University of Science and Technology of China
[2]Hangzhou Innovation Institute, Beihang University, Hangzhou, China
qubit@ustc.edu.cn

Abstract

Deep neural networks are extremely vulnerable to malicious input data. As 3D data is increasingly used in vision tasks such as robots, autonomous driving and drones, the internal robustness of the classification models for 3D point cloud has received widespread attention. In this paper, we propose a novel method named SPGA (Shape Prior Guided Attack) to generate adversarial point cloud examples. We use shape prior information to make perturbations sparser and thus achieve imperceptible attacks. In particular, we propose a Spatially Logical Block (SLB) to apply adversarial points through sliding in the oriented bounding box. Moreover, we design an algorithm called FOFA for this type of task, which further refines the adversarial attack in the process of breaking down complicated problems into sub-problems. Compared with the methods of global perturbation, our attack method consumes significantly fewer computations, making it more efficient. Most importantly of all, SPGA can generate examples with a higher attack success rate (even in a defensive situation), less perturbation budget and stronger transferability.

Introduction

In recent years, deep learning algorithms have made significant and rapid progress in solving a number of tasks involving complex raw data. It has achieved promising performance in speech recognition (Wang et al. 2020), object detection (Kong et al. 2020), image segmentation (Kirillov et al. 2020), etc. However, neural networks have been verified to be vulnerable to adversarial attacks (Nguyen, Yosinski, and Clune 2015; Cisse et al. 2017). Even slight changes in the input data can result in a huge deviation in the output results, although these minor changes are indistinguishable to the human perception. The research of neural network vulnerability stems from the work of Szegedy et al. (2013) on 2D RGB images. Goodfellow, Shlens, and Szegedy (2014) proposed the concept of *adversarial example* for the first time, and pointed out that the vulnerability of deep model to adversarial example is mainly the existence of internal linear part.

The current popular methods of attacking point cloud classifiers are based on 2D image conversion. However, their performance is not satisfactory on sensitive indicators, such as perturbations budget and time consumption. Xiang, Qi, and Li (2019) proposed the point cloud attack algorithm named 3D-Adv firstly. They conducted attacks on point cloud by generating adversarial points or adversarial point perturbation. In fact, the earliest work 3D-Adv used an adversarial point cloud example to attack the objective model (the victim model) when the success rate is close to 99%. Later, Hamdi et al. (2020) presented a data-driven adversarial point cloud example generation method named AdvPC, and for the first time studied the transferability of adversarial point cloud examples.

As mentioned above, most of the previous works are model-specific, yet the problem that we should also take into account is the budget of perturbations and the consumption of time. These important factors have received little attention in earlier works. The existence space of these perturbation points is just too big. In fact, this space can be reduced even further. Moreover, there is no acceleration algorithm specially designed for the extremely sparse structure of point

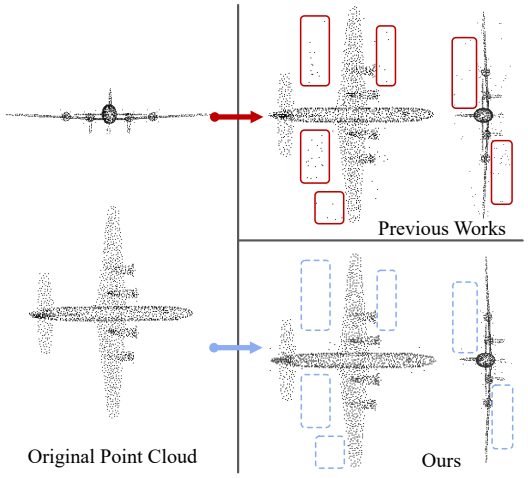

Figure 1: Illustration of attaching adversarial perturbation to clean point clouds to fool the classifiers. The left part is a clean point cloud. The upper part on the right comes from one of the previous popular works. Our method generates sparser perturbations in the bottom right.

*Corresponding Author.

cloud.

In view of the irregular shape of the point cloud and the observation of varying cardinality, it is challenging to effectively limit its perturbation space while ensuring the effectiveness of the attack. The L_p norm of 2D images cannot be simply applied to 3D point clouds, and the smoothness of point cloud adversarial examples is more difficult to ensure than in 2D images. When previous methods are used to attack critical points, perturbations are easily gathered in the local space, making it very hard to obtain perturbation sparsity. Furthermore, the generation process of point cloud adversarial examples is time-consuming, and there is no solution to speed up the task. In addition, enhancing the transferability while retaining a low perturbation budget is a problem that has yet to be solved.

To handle the above-mentioned problems, this paper proposes a novel attack method with shape prior guided and sparser perturbations, named SPGA. Different from previous methods, we limit the budget of perturbations, and it will not exceed 5% of the total number of initial points (this setting guarantees both success rate and imperceptibility). As shown in Fig. 1, the adversarial example generated by our SPGA needs fewer perturbations. SPGA utilizes Spatially Logical Block (SLB) to slide in the oriented bounding box of the point cloud, and selectively applies adversarial points. In particular, we propose Target Loss and Logical Structure Loss to further limit the distortion. In addition, we present an accelerated algorithm named Fast Optimization for Attacking (FOFA) to optimize this kind of task. FOFA can decompose a complex problem into several sub-problems and speed up the solution. We test the performance of SPGA on a much broader range of point cloud classifiers, which are more comprehensive than previous attacks and are particularly persuasive against transferability. Experiments show that our method has a higher success rate (even under defense), higher transferability and less time consumption.

Related Work

Point Cloud Classification Point cloud classification models usually first learn the embedding of each point, and then use an aggregation method to extract global shape features from the entire point cloud. Usually the global is embedded in several fully connected layers to achieve the classification goal. PointNet (Qi et al. 2017a) directly took the point cloud as the input, and realized the permutation invariance through the symmetric function. It learned the features of the points independently by inputting the MLP (Multilayer Perceptron) layer, and then extracted the global features through max-pooling. Further, PointNet++ (Qi et al. 2017b) captured the fine geometric structure from the field of each point, and learned features from the local geometric structure. Based on PointNet++, PointWeb (Zhao et al. 2019) used the context of the local domain to improve the feature of the point through Adaptive Feature Adjustment (AFA). ConvPoint (Boulch 2020) and A-CNN (Komarichev, Zhong, and Hua 2019) are methods based on convolution. The former divided the convolution kernel into a spatial part and a feature part to work, while the latter defined a circular convolution to learn the relationship between neighbor points in a local subset. In DensePoint (Liu et al. 2020), the feature learning is to make full use of the context information by connecting the features of all the previous layers. DGCNN (Wang et al. 2019) constructed a graph in the feature space, which is dynamically updated once through each layer, and performs channel-wise symmetric aggregation on the edge features associated with the neighbors of each point. View-GCN (Wei, Yu, and Sun 2020) used a directed graph and treated multiple views as Graph nodes to achieve point cloud classification.

In this paper, we conduct attack tests on the point cloud classifiers that are divided into three different strategies: Pointwise MLP methods, Convolution-based methods and Graph-based Methods.

Standard Attack Algorithms The adversarial attack can be classified into two categories, namely targeted attack and untargeted attack. On the other hand, based on the understanding of the model, it can also be divided into two categories, i.e., white-box attack and black-box attack. Szegedy et al. (2013) proposed a bounded constrained L-BFGS algorithm to generate adversarial examples. Goodfellow, Shlens, and Szegedy (2014) designed the Fast Gradient Notation Method (FGSM) to generate adversarial examples. Xu et al. (2019) proposed the concept of structured adversarial attack on 2D images, which inspires us to improve on 3D point clouds. Kurakin, Goodfellow, and Bengio (2016) redesigned the calculation method against interference and achieved relatively high accuracy and faster speed. Miyato et al. (2018) used the two-norm normalization result of the gradient as the added anti-interference, and also achieved a higher attack rate.

Compared with the previous attack methods, the optimization algorithm FOFA used in the background of our SPGA has higher speed and fewer perturbations.

Attack on Point Cloud Classification The adversarial examples for point clouds, which were first proposed by Xiang, Qi, and Li (2019), can be divided into two routes to generate adversarial examples, i.e., the perturbation of the adversarial point and the generation of adversarial points. Hamdi et al. (2020) put forward a data-driven adversarial attack against 3D point cloud networks, which can make adversarial examples migrate between different 3D object detection models. Tu et al. (2020) proposed a method to generate general 3D adversarial objects to fool LiDAR detectors. By preventing an opposing object on top of any target vehicle, the car can be prevented from being detected by LiDAR. Liu, Yu, and Su (2020) explored three feasible shape perturbations to attack point cloud classification. Tsai et al. (2020) proposed an attack named KNN to add K-Nearest Neighbor loss to the point cloud to make the point cloud attack easier to implement physically. Zhou et al. (2020) proposed a label-guided adversarial network LG-GAN for real-time and flexible attack point cloud classification model. By inputting the original point cloud and the target label into LG-GAN, the point cloud can be deformed with only one forward pass, thereby identifying the point cloud as a wrong specific label. Kim et al. (2021) attacked the point cloud classifiers while maintaining the perceptibility and the application of a minimum number of perturbation points. Imposing fewer per-

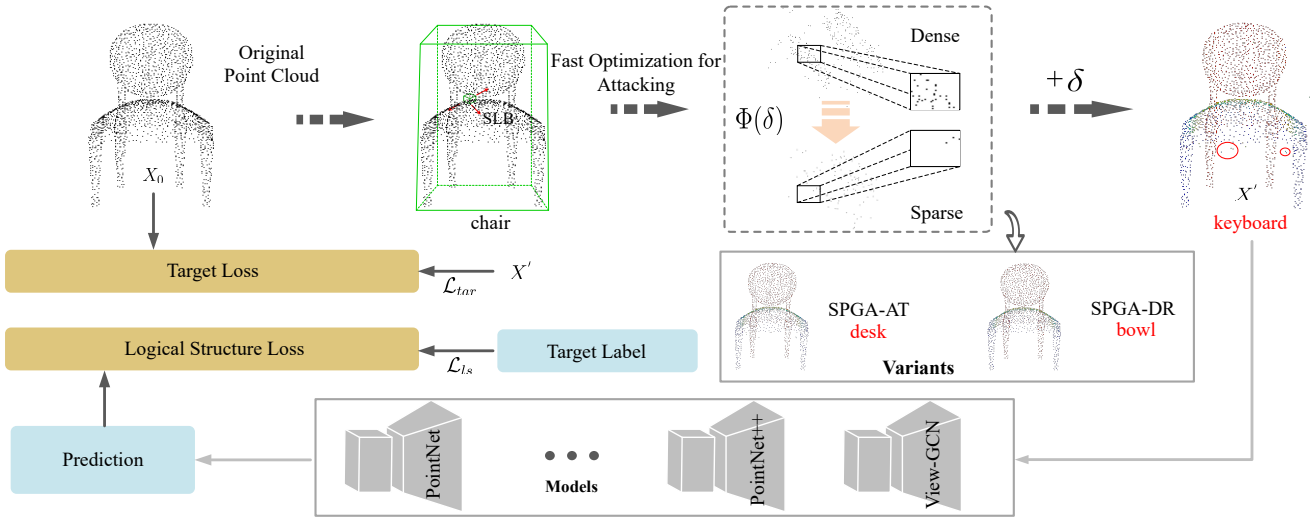

Figure 2: Overview of SPGA pipeline. Given a clean point cloud example, first attach perturbation points around the original point through SLB sliding within the oriented bounding box, and then retain influential perturbation points through FOFA algorithm. The relevant loss function can guarantee the fidelity against the point cloud and the effectiveness of targeted attacks.

turbations on the point cloud is a hot topic because it keeps the perturbation imperceptible. Therefore, we explored the sparse perturbation of the point cloud.

Methodology

In this section, we introduce the pipeline of SPGA (as shown in Fig. 2) and the implementation details of the FOFA optimization algorithm.

Spatial Feature Aggregation Can we fool the advanced point cloud classification methods by changing the logical structure inside the point cloud? To get the answer, we divide a point cloud into several sub-groups of points, and then punish the corresponding structure sparsity. As a result, non-dense adversarial point sets can be generated, which encode the sparser adversarial effect in the local structure of the original point cloud.

Given a point cloud X, we set its point set coordinates as (u^X, v^X, w^X). In order to reduce the cost of adversarial search, we first drop the point cloud into a 3D grid $G \in \mathbb{R}^{H\times W\times P}$. G is used to obtain a count feature. Note that this step is only for the improvement of calculation efficiency, and will not affect the size of the perturbation and the success rate of the attack. We will prove this through experiments later. For each point X_i whose coordinate is $(u^{X_i}, v^{X_i}, w^{X_i})$, we add 6 points around it respectively. These 6 points are located at the center of the 6 faces of the unit cube, and X_i is in the geometric center of the unit cube. We use a trilinear weight to add these 6 points:

$$G(u_i, v_i, w_i) = \sum_r (1 - \|u_r - u^{X_i}\|)(1 - \|v_r - v^{X_i}\|) \\ (1 - \|w_r - w^{X_i}\|) \tag{1}$$

where $r \in \mathcal{N}(u^{X_i}, v^{X_i}, w^{X_i})$ is the index of the 6 points around $(u^{X_i}, v^{X_i}, w^{X_i})$.

Structure Sparsity We define the attached perturbation as δ, and the coordinate of the perturbation point as $(u^\delta, v^\delta, w^\delta)$. We regard the oriented bounding box of the original point cloud as the effective space for attack. In order to describe the local characteristics of perturbation δ, we introduce the Spatial Logical Block (SLB) with size $l\times l\times C$, and step size S. The number of SLB sliding unit blocks can be controlled by S. By adjusting the step size S and the size l of the SLB, different spatially sparse group splitting schemes can be obtained. It should be noted that in the non-original point area, SLB (without the original point as the center and using the classic point cloud attack method as the positioning guide) has a low probability of imposing perturbations there.

SLB can decompose δ into a set of groups $\{\delta_{\Upsilon_{a,b,e}}\}$ for $a \in [A]$, $b \in [B]$ and $e \in [E]$, where $A = (H-1)/S + 1$, $B = (W-1)/S + 1$ and $E = P - C + 1$. $[n]$ denotes an integer set $\{1, 2, \cdots, n\}$. Given group $\{\delta_{\Upsilon_{a,b,e}}\}$, the structure sparsity can be described by the following formula:

$$\Phi(\delta) = \sum_{e=1}^{E}\sum_{a=1}^{A}\sum_{b=1}^{B}\|\delta_{\Upsilon_{a,b,e}}\|_2 \tag{2}$$

where $\{\delta_{\Upsilon_{a,b,e}}\}$ represents the set of adversarial points δ indexed by $\Upsilon_{a,b,e}$ and $\|\cdot\|_2$ is the L_2 norm.

Distortion Function

We design the distortion function $\mathcal{L}$, which consists of two loss functions, namely Target Loss ($\mathcal{L}_{tar}$) and Logical Structure Loss ($\mathcal{L}_{ls}$). We use the standard cross-entropy loss function as the Target Loss ($\mathcal{L}_{tar}$). Formally:

$$\mathcal{L}_{tar} = -[t' \log \mathcal{V}(X') + (1 - t') \log(1 - \mathcal{V}(X'))] \tag{3}$$

where $\mathcal{V}$ denotes the victim models, $t^{'}$ is the targeted label, and $\mathcal{V}(X^{'})$ represents the output of the victim models when we input the adversarial point cloud example.

In order to make the attacked point cloud possess the smaller distortion in the spatial logic, we design the Logical Structure Loss ($\mathcal{L}_{ls}$). Formally:

$$\mathcal{L}_{ls} = \sum_{X_i \in X_0} \sum_{X_r \in \mathcal{N}(X_i)} \|\triangle X_i - \triangle X_r\|_2^2 \tag{4}$$

where $\triangle X_i = X_i^{'} - X_i$ denotes the displacement from the original point position to the adversarial point position, and $\mathcal{N}(X_i)$ represents the 6 points located at the center of the 6 faces of the cube centered on X_i.

In general, the combined loss is defined as:

$$\mathcal{L}_{dis} = \lambda_1 \mathcal{L}_{tar} + \lambda_2 \mathcal{L}_{ls} \tag{5}$$

where λ_1 and λ_2 are hyperparameters to balance the weights of the two losses.

Shape Prior Guided Attack

Generating the above-mentioned perturbations gives rise to optimization problems. Existing optimizers such as FISTA (for EDA attack (Chen et al. 2018)) and Adam (for C&W attack (Carlini and Wagner 2017)) are arduous to solve the problem of finding sparse adversarial points. We are inspired by stochastic/online ADMM (Ouyang et al. 2013) and linearized ADMM (Boyd, Parikh, and Chu 2011) to solve shared problems in linearization technology, that is, the positioning and searching of spatially sparse group and sparse logical points in this research. In order to take into account the decomposability and the convergence properties of the optimization algorithm, we propose an optimization algorithm called FOFA, which can effectively find spatially sparse adversarial points.

Assuming that the initial point cloud is $X_0 \in \mathcal{R}^{N\times 3}$, the adversarial cloud points after being disturbed is denoted as $X^{'} = X_0 + \delta$. The true label is defined as t, and the target label is $t^{'}$. By solving the optimization problem of this form, a better point cloud perturbation can be obtained:

$$\begin{aligned} \min_{\delta} \ & \mathcal{L}(X^{'}, t^{'}) + \alpha \mathcal{L}_{dis} + \beta \Phi(\delta) \\ & s.t. \quad \|\delta\|_\infty \leq \epsilon \end{aligned} \tag{6}$$

where $\mathcal{L}_{dis}$ is a distortion function. The definition of $\Phi(\delta)$ has been given in Eq. (2). In Eq. (6), strict constraints can ensure that the created adversarial points effective. α and β are non-negative regularization parameters. They can guide us to focus the attack on the distortion of adversarial examples and the logic of sparse space. We will discuss the ablation experiments of α and β in the experimental section.

Eq. (6) provides a general guidance for the generation of adversarial examples. If we remove the constraints of $\mathcal{L}_{dis}$ and L_∞, Eq. (6) represents a C&W attack. In this paper, we design the loss function of Eq. (6) as follows:

$$\mathcal{L}(X^{'}, t^{'}) = \max(\max_{i \neq t^{'}}(F(X^{'})_i) - F(X^{'})_{t^{'}} + \sigma, 0) \tag{7}$$

where $F(X^{'})_i$ is the $i-th$ element of the logarithm $F(X^{'})$, which represents the output from the last *Softmax* layer, and σ is a confidence parameter used to adjust the attack transferability of the adversarial example.

Fast Optimization for Attacking

To adapt Eq. (6) to FOFA, we rewrite it as follows:

$$\begin{aligned} \min_{\delta} \ & \mathcal{L}(X_0, \boldsymbol{x}) + \alpha \mathcal{L}_{dis} + \beta \sum_{i=1}^{ABE} \|\boldsymbol{y}_{\Theta_i}\|_2 + g(\boldsymbol{z}) \\ & s.t. \quad \delta = \boldsymbol{x}, \boldsymbol{y} = \boldsymbol{x}, \boldsymbol{z} = \boldsymbol{x} \end{aligned} \tag{8}$$

where $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ are newly introduced variables for solving optimization problems. To facilitate counting, we let $\Theta_{(b-1)A+a+e} = \Upsilon_{a,b,e}$. $g(\boldsymbol{z})$ be an indicator function, which is defined as follows:

$$g(\boldsymbol{z}) = \begin{cases} 0 & if \ \|\boldsymbol{z}\|_\infty \leq \epsilon \\ \infty & otherwise \end{cases} \tag{9}$$

The optimization problem of FOFA can be transformed into finding the minimum of an augmented Lagrangian problem. For the sake of simplicity, it can be equated to being decomposed to find the following sub-problems:

$$\begin{aligned} & \min_{\delta} \ \alpha \mathcal{L}_{dis} + \frac{d}{2}\|\delta - c_1\|_2^2 \\ & \min_{\boldsymbol{z}} \ g(\boldsymbol{z}) + \frac{d}{2}\|\boldsymbol{z} - c_2\|_2^2 \end{aligned} \tag{10}$$

where $c_1 = \boldsymbol{x}^k - (o^k/d)$ and $c_2 = \boldsymbol{x}^k - (p^k/d)$. We also need to solve the minimum value of the following problem:

$$\min_{\boldsymbol{y}} \ \beta \sum_{i=1}^{ABE} \|\boldsymbol{y}_{\Theta_i}\|_2 + \frac{d}{2}\|\boldsymbol{y} - c_3\|_2^2 \tag{11}$$

where $c_3 = \boldsymbol{x}^k - (q^k/d)$. o, p and q used here are all Lagrangian Multipliers, and k is an iteration index. $d(d > 0)$ is a penalty parameter. The key to FOFA is that the solution to Eq. (8) can be determined in parallel and accurately.

The SPGA attack needs to obtain the gradient of the loss function $\mathcal{L}$. We add the Bregman divergence term $\frac{\mu k}{2}\|\boldsymbol{x} - \boldsymbol{x}^k\|_2^2$. Then, we get the following question:

$$\begin{aligned} \min_{\boldsymbol{x}} \ & (\nabla \mathcal{L}(\boldsymbol{x}^k + X_0))^T(\boldsymbol{x} - \boldsymbol{x}^k) + \frac{\mu k}{2}\|\boldsymbol{x} - \boldsymbol{x}^k\|_2^2 \\ & + \frac{d}{2}\|\boldsymbol{x} - c_1\|_2^2 + \frac{d}{2}\|\boldsymbol{x} - c_2\|_2^2 + \frac{d}{2}\|\boldsymbol{x} - c_3\|_2^2 \end{aligned} \tag{12}$$

where $\frac{1}{\mu k} > 0$ is a given decaying parameter, and the Bregman divergence stabilizes the convergence of the $\boldsymbol{z}$ to minimization step. Eq. (12) belongs to a quadratic programming problem with a closed solution.

In summary, our proposed FOFA optimization algorithm obtains a closed solution in the following form:

$$\boldsymbol{x}^{k+1} = \frac{\mu k \boldsymbol{x}^k + dc_1 + dc_2 + dc_3 - \nabla \mathcal{L}(\boldsymbol{x}^k + X_0)}{\mu k + 3d} \quad (13)$$

Compared with the current optimization solution methods (such as C&W (Carlini and Wagner 2017), Zoo (Chen et al. 2017)), the two main advantages of FOFA are the efficiency of the solution and the generalization of the attack. We will go through this in detail on attacking different point cloud classification models in the experimental section. In other words, the calculation of each step has been executed more effectively, and FOFA can be employed in other attack algorithms.

Experiment

Datasets

We use two public datasets, 3D MNIST [1] and the aligned ModelNet40 [2] (Wu et al. 2015). The former contains 6000 original point clouds generated by 2D MNIST. The numbers of training set and test set are 5000 and 1000, respectively. Each point cloud has approximately 20,000 points. The latter possesses 12311 CAD models from 40 categories, and the numbers of training set and test set are 9,743 and 2,468, respectively. In the experiment, we randomly sample 2048 points on the surface of each object.

Implementation Details

In our implementation, we divide the attacks on all point cloud classification models into untargeted attack and targeted attack. The modifications of SPGA to the original point cloud are divided into moving points, attaching perturbation points and dropping original points. We also implement two variants of SPGA: SPGA-AT and SPGA-DR. The names suffixed with AT and DR indicate that SPGA is currently allowed to **at**tach and **dr**op points, respectively. The two hyperparameters of λ_1 and λ_2 are 0.3 and 0.7, respectively. The values of the two parameters α and β are both 0.5.

Qualitative Comparison

We carry out targeted attacks on different point cloud classifiers. As shown in Fig. 3, we select a slice of adversarial point cloud examples. The black text is the real label, and the brown text represents the target label. The original point cloud is attached, dropped or moved by points, all of which can achieve the goal of fooling the point cloud classifiers. Previous methods will generate clustered point cloud patches. By contrast, our method mainly generates sparse logical adversarial points. In these cases, the adversarial points do not seem to be overwhelmingly clustered. Most of the adversarial points are merged into the original normal points, so that the point cloud defense method cannot distinguish between the adversarial points and the normal points.

[1] https://www.kaggle.com/daavoo/3d-mnist/version/13

[2] http://modelnet.cs.princeton.edu/

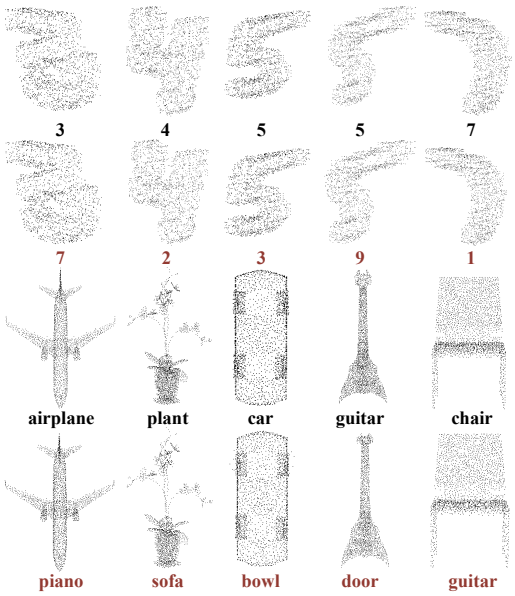

Figure 3: Illustration of the targeted adversarial examples generated by SPGA. The first and third rows represent the original point clouds, and the second and fourth rows denote the corresponding adversarial examples.

The goal of separating the two variants (SPGA-AT and SPGA-DR) is to analyze whether the addition and deletion of points will affect the efficiency of the attack, and in which way the perturbation is smaller in the senses. As shown in Fig. 4, the first column represents clean point cloud examples, and the next two columns are point cloud counter examples. Generally speaking, the perturbation points generated by the two variant methods are relatively few, but it is obvious that the operation of reducing the points is sensorily closer to the original example.

Quantitative Comparison

The number of perturbed points We complete the experiment of the number of sparse logical adversarial points. The results are shown in Fig. 5, where 3D MNIST and ModelNet 40 are with suffixes 3M and M40, respectively. **RN** stands for Random Noise, which means randomly attaching noise points according to Gaussian sampling. Experimental results demonstrate that merely attaching random noise does not significantly reduce the recognition accuracy of the model. In contrast, on 3D MNIST dataset, when the number of adversarial points generated by SPGA reaches about 100, the recognition accuracy is close to 0%. On ModelNet40 dataset, when the number of perturbation points reaches about 90, the model can no longer accurately classify objects.

Transferability We also conduct experiments on the attack transferability of SPGA. It should be noted that the transferability between the same models is 100%. Among the 8 point cloud classifiers, based on the migration performance between the 3 models PointNet, PointNet++ and DGCNN, we list the best results of the previous works, as

Methods	PointNet		PointNet++		DGCNN		A-CNN		PointWeb		DensePoint		ConvPoint		View-GCN	
	RO	**RS**	**RO**	**RS**	**RO**	**RS**	**RO**	**RS**	**RO**	**RS**	**RO**	**RS**	**RO**	**RS**	**RO**	**RS**
3D-Adv	53.4	55.3	57.1	58.4	61.2	63.4	64.7	71.2	60.7	57.6	64.6	62.0	51.5	57.4	52.1	52.2
KNN	64.8	65.1	62.8	67.7	66.4	68.2	72.0	80.1	65.5	61.6	80.0	73.4	62.7	59.3	63.1	60.8
AdvPC	64.1	73.3	63.0	72.0	72.4	76.6	77.5	82.6	68.2	66.2	84.5	80.7	66.5	67.4	76.2	73.3
ASP	86.8	**89.5**	**87.3**	84.5	85.5	81.3	**86.2**	82.9	85.7	80.3	84.0	82.5	85.4	83.1	83.2	78.1
SPGA-AT	83.2	85.4	84.8	83.3	84.5	80.4	81.1	78.0	85.6	76.3	81.8	78.8	82.6	83.3	83.0	77.5
SPGA-DR	85.3	86.5	85.8	85.3	84.3	81.5	83.0	81.8	84.4	77.8	83.6	81.4	85.8	86.2	82.1	76.2
SPGA	**87.5**	88.6	86.4	**86.7**	**85.7**	**82.9**	84.3	**83.0**	**86.9**	**81.2**	**85.7**	**82.8**	**86.3**	**87.5**	**84.6**	**79.3**

Table 1: Different attack methods generate untargeted adversarial examples for 8 point cloud classifiers. Note that these point cloud classification models have taken defensive measures, which are represented by RO (Remove Outliers) and RS (Remove Salient), respectively. Results in bold and underlined indicate the best and the second-best. The experiments are conducted on ModelNet40 dataset.

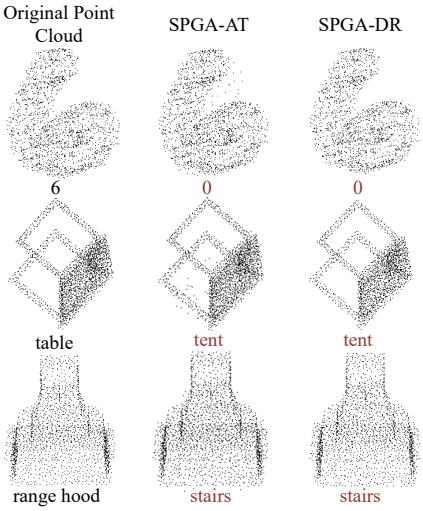

Figure 4: Illustration of the targeted attack performance of two variants of SPGA: SPGA-AT and SPGA-DR.

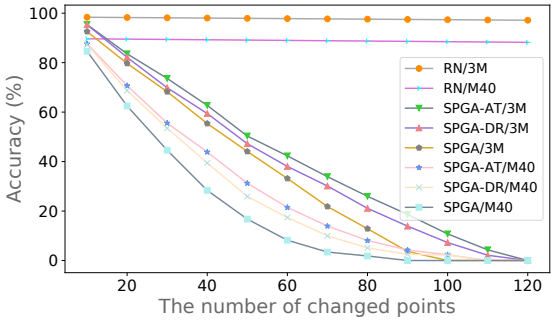

Figure 5: Illustration of the influence of the number of adversarial points on the accuracy of point cloud classification.

shown in Fig. 6. The transferability of the other 5 point cloud classification models is not comparable to previous work. It can be seen from Fig. 6 that the success rate of SPGA transferability between different models can exceed 50% in a multitude of cases, which is sufficient to prove the successful performance of transferability.

Perturbation budget We conduct untargeted attacks on the same point cloud classifier without defensive measures on ModelNet40 dataset. We compare the perturbation budgets between 3D-Adv and AdvPC attack algorithms. As shown in Table 2, SPGA w/o FOFA means that SPGA uses other optimization algorithms and the main structure remains unchanged. Table 2 reveals that SPGA can achieve the goal of the attack with a perturbation budget of less than 5%.

Attack Success Rate and Distortion

SPGA attacks can achieve a 100% success rate on any target label. Compared with other methods, the distortion of the point cloud is smaller. It is worth mentioning that on ModelNet40 dataset, the example quality of SPGA is better. We conduct tests on 3D MNIST and ModelNet40 datasets to study the relationship between the amount of disturbance and the attack success rate. In Fig. 5, we plot the relationship between the number of changed points and the classification accuracy.

Comparison with the State-of-the-arts

We compare our methods with 3D-Adv (Xiang, Qi, and Li 2019), KNN (Tsai et al. 2020), AdvPC (Hamdi et al. 2020) and ASP (Liu, Yu, and Su 2020), which use different strategies and perform well. Since Kim et al. (2021) does not have open source code, we cannot make a fair comparison with it under the same experimental background. We carry out defensive attacks against different point cloud models on ModelNet40 dataset. Considering that when the previous attack methods meet those victim models, the attack success rate is close to 99%, consequently we change the traditional route to test the success rate of an attack and adopt certain defensive measures. Table 1 shows the performance of these attack methods when attacking 8 point cloud classification models. There are two types of defense used, Remove Outliers (**RO**) and Remove Salient (**RS**) (Liu, Yu, and Su 2019). Salient points are identified and deleted based on the calculated significance. As shown in Table 1, SPGA outper-

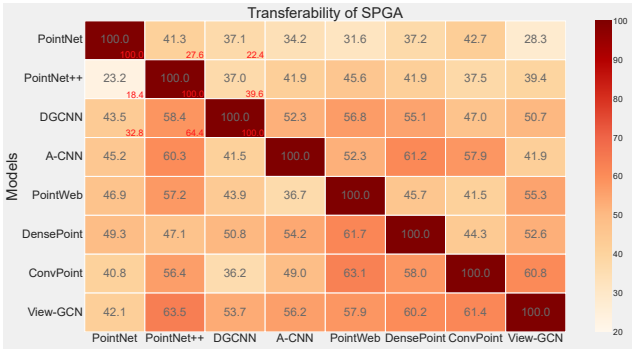

Figure 6: Adversarial examples transferability of SPGA between different point cloud classifiers. The red font results come from the best performance in previous works.

Models	3D-Adv	AdvPC	**w/o FOFA**	**SPGA**
PointNet	7.64	5.29	4.82	**4.36**
PointNet++	6.21	5.03	4.65	**4.12**
DGCNN	7.38	4.87	4.43	**3.85**
A-CNN	6.92	5.63	5.27	**4.50**
PointWeb	7.16	5.74	5.59	**4.73**
DensePoint	7.05	5.31	4.78	**4.09**
ConvPoint	6.74	4.80	4.67	**3.82**
View-GCN	5.82	4.45	3.84	**3.67**

Table 2: The average perturbation budgets (%) of different attack methods when conducting untargeted attacks.

forms state-of-the-arts in most cases. This is attributed to the sparse logic of the adversarial points generated by SPGA, and SPGA has a positive attack performance in front of most defense methods. As an example, in the Random Outliers defense method, the sparse adversarial points generated by SPGA are closer to the original point cloud on the surface of the point cloud, and will not produce a large outlier effect.

Interaction Object and Standard Attack

To some extent, the existence of interaction objects will worsen the performance of classifiers. Sometimes we can even use interaction objects to classify objects into specified labels. As shown in Fig. 7, we attach additional interaction objects (512 sampling points) to the clean point cloud examples to make them misclassified. We compare the adversarial point cloud with the attached interaction objects and the adversarial examples generated by SPGA. Through experiments, we find that the conditions for interaction objects to effectively interfere with the point cloud are harsh, and sometimes only when they are placed in a specific location can they enjoy an effective attack effect. Furthermore, when the number of sampling points of the interaction object is small, the attack effectiveness is relatively not so good.

Ablation Study

Table 3 shows an ablation study for the acceleration of FOFA (measured in seconds), where w/o FOFA and w/ FOFA denote that FOFA is substituted and is working, respectively.

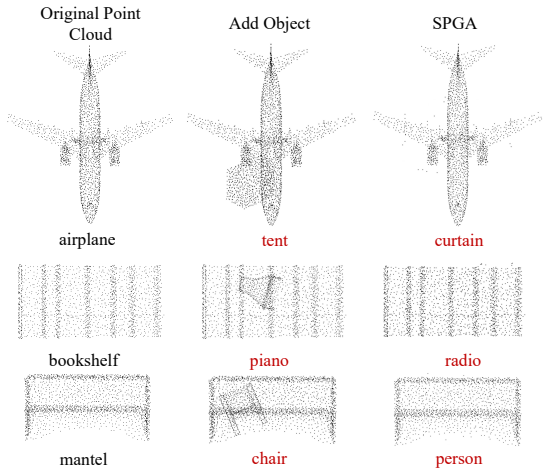

Figure 7: Illustration of the interaction objects that are attached to the original point clouds.

	SPGA-AT	SPGA-DR	SPGA
w/o FOFA	24.47	24.03	**23.87**
w/ FOFA	18.63	18.15	**18.09**

Table 3: Ablation study of FOFA acceleration.

When FOFA is used, the speed of generating adversarial samples is accelerated by about 31.8%.

We conduct ablation experiments on λ_1, λ_2, α and β involved. Firstly, we fix the values of λ_1 and λ_2, and then test the combination of α and β. In the experiment, we found that the best combination of λ_1 and λ_2 is 0.3 and 0.7, respectively.

Conclusion

In this paper, we presented a novel method of attacking point cloud classifiers called SPGA. We utilized the point cloud shape prior information to obtain the sparse logical adversarial points within the oriented bounding box. Particularly, we proposed the Spatial Logical Block (SLB), novel loss functions and FOFA optimization algorithms. Furthermore, the FOFA optimization algorithm can decompose complex problems into multiple sub-problems to be solved, which can be applied to other attack tasks. Experiments demonstrate that the perturbations of SPGA are fewer than previous works, and the attack success rate and the performance of transferability are improved. SPGA also has strong attack abilities in the face of point cloud classification models with defensive measures. In conclusion, our method achieves the state-of-the-art performance on public 3D MNIST and ModelNet40 datasets.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 62172385 and 61602400), and the Anhui Initiative in Quantum Information Technologies (No. AHY150300).

References

Boulch, A. 2020. ConvPoint: Continuous convolutions for point cloud processing. *Computers & Graphics*, 88: 24–34.

Boyd, S.; Parikh, N.; and Chu, E. 2011. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 39–57. IEEE.

Chen, P.-Y.; Sharma, Y.; Zhang, H.; Yi, J.; and Hsieh, C.-J. 2018. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Chen, P.-Y.; Zhang, H.; Sharma, Y.; Yi, J.; and Hsieh, C.-J. 2017. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 15–26.

Cisse, M.; Adi, Y.; Neverova, N.; and Keshet, J. 2017. Houdini: Fooling deep structured prediction models. *arXiv preprint arXiv:1707.05373*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Hamdi, A.; Rojas, S.; Thabet, A.; and Ghanem, B. 2020. Advpc: Transferable adversarial perturbations on 3d point clouds. In *European Conference on Computer Vision*, 241–257. Springer.

Kim, J.; Hua, B.-S.; Nguyen, T.; and Yeung, S.-K. 2021. Minimal adversarial examples for deep learning on 3d point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7797–7806.

Kirillov, A.; Wu, Y.; He, K.; and Girshick, R. 2020. Pointrend: Image segmentation as rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9799–9808.

Komarichev, A.; Zhong, Z.; and Hua, J. 2019. A-cnn: Annularly convolutional neural networks on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7421–7430.

Kong, T.; Sun, F.; Liu, H.; Jiang, Y.; Li, L.; and Shi, J. 2020. Foveabox: Beyound anchor-based object detection. *IEEE Transactions on Image Processing*, 29: 7389–7398.

Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2016. Adversarial examples in the physical world. *CoRR*, abs/1607.02533.

Liu, D.; Yu, R.; and Su, H. 2019. Extending Adversarial Attacks and Defenses to Deep 3D Point Cloud Classifiers. *CoRR*, abs/1901.03006.

Liu, D.; Yu, R.; and Su, H. 2020. Adversarial Shape Perturbations on 3D Point Clouds. In *European Conference on Computer Vision*, 88–104. Springer.

Liu, Y.; Fan, B.; Meng, G.; Lu, J.; Xiang, S.; and Pan, C. 2020. DensePoint: Learning Densely Contextual Representation for Efficient Point Cloud Processing. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*.

Miyato, T.; Maeda, S.-i.; Koyama, M.; and Ishii, S. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8): 1979–1993.

Nguyen, A.; Yosinski, J.; and Clune, J. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 427–436.

Ouyang, H.; He, N.; Tran, L.; and Gray, A. 2013. Stochastic alternating direction method of multipliers. In *International Conference on Machine Learning*, 80–88. PMLR.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Tsai, T.; Yang, K.; Ho, T.-Y.; and Jin, Y. 2020. Robust adversarial objects against deep learning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 954–962.

Tu, J.; Ren, M.; Manivasagam, S.; Liang, M.; Yang, B.; Du, R.; Cheng, F.; and Urtasun, R. 2020. Physically realizable adversarial examples for lidar object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13716–13725.

Wang, Y.; Mohamed, A.; Le, D.; Liu, C.; Xiao, A.; Mahadeokar, J.; Huang, H.; Tjandra, A.; Zhang, X.; Zhang, F.; et al. 2020. Transformer-based acoustic modeling for hybrid speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6874–6878. IEEE.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5): 1–12.

Wei, X.; Yu, R.; and Sun, J. 2020. View-GCN: View-based graph convolutional network for 3D shape analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1850–1859.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1912–1920.

Xiang, C.; Qi, C. R.; and Li, B. 2019. Generating 3d adversarial point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9136–9144.

Xu, K.; Liu, S.; Zhao, P.; Chen, P.; Zhang, H.; Fan, Q.; Erdogmus, D.; Wang, Y.; and Lin, X. 2019. Structured Adversarial Attack: Towards General Implementation and Better Interpretability. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zhao, H.; Jiang, L.; Fu, C.-W.; and Jia, J. 2019. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5565–5573.

Zhou, H.; Chen, D.; Liao, J.; Chen, K.; Dong, X.; Liu, K.; Zhang, W.; Hua, G.; and Yu, N. 2020. LG-GAN: Label Guided Adversarial Network for Flexible Targeted Attack of Point Cloud Based Deep Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10356–10365.