Passive Defense Against 3D Adversarial Point Clouds Through the Lens of 3D Steganalysis

Jiahao Zhu
Nanjing University of Aeronautics and Astronautics
jiahaozhu@nuaa.edu.cn

Abstract

Nowadays, 3D data plays an indelible role in the computer vision field. However, extensive studies have proved that deep neural networks (DNNs) fed with 3D data, such as point clouds, are susceptible to adversarial examples, which aim to misguide DNNs and might bring immeasurable losses. Currently, 3D adversarial point clouds are chiefly generated in three fashions, i.e., point shifting, point adding, and point dropping. These point manipulations would modify geometrical properties and local correlations of benign point clouds more or less. Motivated by this basic fact, we propose to defend such adversarial examples with the aid of 3D steganalysis techniques. Specifically, we first introduce an adversarial attack and defense model adapted from the celebrated Prisoners’ Problem in steganography to help us comprehend 3D adversarial attack and defense more generally. Then we rethink two significant but vague concepts in the field of adversarial example, namely, active defense and passive defense, from the perspective of steganalysis. Most importantly, we design a 3D adversarial point cloud detector through the lens of 3D steganalysis. Our detector is double-blind, that is to say, it does not rely on the exact knowledge of the adversarial attack means and victim models. To enable the detector to effectively detect malicious point clouds, we craft a 64-D discriminant feature set, including features related to first-order and second-order local descriptions of point clouds. To our knowledge, this work is the first to apply 3D steganalysis to 3D adversarial example defense. Extensive experimental results demonstrate that the proposed 3D adversarial point cloud detector can achieve good detection performance on multiple types of 3D adversarial point clouds.

1. Introduction

With the advancement of 3D acquisition techniques, 3D sensors, e.g., LiDARs and depth cameras, are increasingly accessible in our daily life [4]. As such, 3D data, including meshes, point clouds, voxels, and depth images, draws more and more research concerns. In particular, the 3D point cloud is a preferred 3D data representation in the computer vision field due to its easy availability, simple structure, and low memory footprint. Recently, the emergence of some deep learning technologies, like PointNet [19] and its variant PointNet++ [20], further consolidates the position of the 3D point cloud in some specific fields, such as object classification and detection, and scene semantic segmentation.

Notwithstanding these techniques have reached prominent achievements in the 3D computer vision, extensive studies have pointed out that DNNs are susceptible to an attack dubbed adversarial example, which can misguide DNNs to make wrong judgments and cause tremendous losses to the deep model holders. Motivated by this basic fact, some scholars extend 2D adversarial example generation methods to their 3D counterparts. Unlike traditional 2D images, point clouds are more irregular in structure, and thus there exist more potential manners in which we can craft 3D adversarial point clouds.

Existing adversarial attacks in 3D settings can be implemented in three basic manners: point shifting [8,14,15,27,28,32], point adding [8,27,28], and point dropping [24,28]. For the first manner, one can modify points in a given point cloud with a small perturbation, like [8,14,27,28,32], or...
shift a part of points to a large extent such that point clouds are deformed, such as three shape attacks proposed in [15], i.e., perturbation resampling, adversarial sticks, and adversarial sinks. Since point clouds are structurally irregular, it is possible to generate adversarial examples by only adding or deleting points. Moreover, one can certainly adopt multiple of these basic methods to create 3D adversarial examples. We provide several classical examples so as to render readers have a more intuitive understanding of 3D adversarial point clouds.

Since the above attack methods are indeed effective to deceive some target DNNs, e.g., PointNet, PointNet++, and DGCNN [23], some defense methods that are based on adversarial training [6] and example pretreatment [14, 15, 33] come into being. For the former, researchers will regard adversarial examples as input to retrain the model to resist adversarial attacks robustly. For the latter, some studies propose to employ point cloud processing methods, e.g., outlier removal [14,33], point cloud resampling [33], and point cloud restoration [26], to mitigate the impact of adversarial examples on the target DNNs. However, these methods will involve the retraining of target models or modifications to input examples. As we know, the cost of training DNNs, especially enterprise-level models, is expensive. What’s more, model holders may not know what exact type of adversarial examples the attacker will use for a model attack. As for the preprocessing for point clouds, especially for benign point clouds, it is extremely unfriendly to certain applications with strict precision requirements. As a matter of fact, shifting, adding, or deleting points will change certain fundamental geometrical properties of point clouds, such as the principal curvature and surface normal at each point. Motivated by this basic fact and to overcome the drawbacks exposed in previous defense methods, we put forward a new detection based method, dubbed as a type of passive defense, to defend a 3D adversarial attack by crafting a set of discriminative features to detect various types of 3D adversarial point clouds. The design of such a feature set is inspired by common techniques in 3D mesh steganalysis. To our best of knowledge, this is the first work that takes into account 3D steganalysis for defending 3D adversarial attacks, and the critical contributions of our work are summarized as follows:

- An adversarial attack and defense model, which is adapted from the celebrated Prisoners' Problem in steganography, is put forward in this work to clarify the roles of all parties in adversarial attack and defense.

- We find a substantial divergence of opinion on the definition of proactive defense and passive defense. For example, wu et al. [25] regarded the detection based method as a passive defense because it does not modify the target model, but Wang et al. [22] classified it as a proactive defense. In our work, we provide a plausible definition for such two types of defenses from the perspective of steganalysis.

- We devise a 3D adversarial point cloud detector through the lens of 3D steganalysis. Our detector is double-blind; that is to say, it does not rely on the exact knowledge of the adversarial attack means and victim models. Simultaneously, we craft a 64-D discriminative feature set tailored for 3D adversarial point clouds.

- The detector we devised only requires a small number of samples for training and can achieve good detection results on multiple types of 3D adversarial point clouds.

2. Adversarial Attack and Defense Model

A celebrated model, Simmons model — Prisoners’ Problem [21], elaborates on the roles of all parties in steganography. Analogously can we design a model, which is called AADM, to assist us in clarifying various roles in adversarial attack and defense? Specifically, there exist four pivotal roles in our model: attacker Alice, spy Bob, consultant Eve, and model holder Wendy. During an attack, Alice manages to communicate with Bob to steal target model information and then creates various adversarial examples to attack the target model for benefit. We should note that these attacks may not arouse suspicion of Wendy because the created adversarial examples resemble benign ones in appearance. During the defense, Wendy will consult Eve to counter adversarial attacks. Due to her identity as the model holder, Wendy can take any defense measure to protect the victim model, such as model retraining and input data pretreatment, yet Eve, as a consultant, has no such privileges but can monitor the input examples. Therefore, Eve will defend adversarial attacks by detection-based methods that are defined as passive defenses in our work. Fig. 2 illustrates these four key roles and their interactions.
3. Related Work

3.1. Point Cloud and its Representation

A point cloud is made up of numerous points that are acquired by 3D scanners, e.g., LiDARs and depth cameras, and each point contains information not only on 3D coordinate, but also on intensity or RGB. In this work, we only consider the former and denote a point cloud with $N$ points by $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^N$ where point $\mathbf{p}_i = \{v_{i,x}, v_{i,y}, v_{i,z}\}$.

3.2. 3D Adversarial Point Cloud Attack

The majority of existing 3D adversarial attack methods are designed for deep learning-based classifiers, such as PointNet, PointNet++, and DGCNN. In this work, we roughly group these methods into four categories, i.e., point shifting based methods, point adding based methods, point dropping based methods, and isometry transformation-based methods. Let $\mathcal{P}^* = \{\mathbf{p}_i^*\}_{i=1}^M$ be the adversarial counterpart of $\mathcal{P}$ with $M$ points where $\mathbf{p}_i^* = \{v_{i,x}, v_{i,y}, v_{i,z}\}$. Note that whether $M$ is equal to $N$ depends on the attack method adopted by Alice.

3.2.1 Point Shifting Based Methods

Xiang et al. [27] proposed a targeted attack against point cloud classifiers by slightly disturbing points in $\mathcal{P}$. To be exact, they managed to seek out 3D adversarial point clouds with minimum perturbation by solving the following C&W optimization problem [1]:

$$\text{minimize } D_{adv}(\mathcal{P}^*) + \lambda D_{dist}(\mathcal{P}, \mathcal{P}^*)$$ (1)

Here $D_{adv}(\mathcal{P}^*)$ is the adversarial loss calculated by $\max(\mathbf{f}(\mathbf{P}^*), \mathbf{f}(\mathcal{P})); \mathbf{f}$ denotes the $i$-th components of the logit vector $\mathbf{F}(\mathcal{P}^*)$, and $D_{dist}(\mathcal{P}, \mathcal{P}^*) = ||\mathcal{P} - \mathcal{P}^*||_p$ in this case. Liu et al. [14] extended adversarial attacks to 3D point clouds by adapting the work of Goodfellow et al. [3], which is also known as fast gradient sign method (FGSM), into three similar methods, i.e., FGSM for point clouds, normalized gradient $L_2$ method, and gradient $L_2$ method. In addition, they provided two original approaches: gradient projection and norm clipping. To generate 3D adversarial point clouds more quickly and further improve the attack success rates, Zhou et al. [32] proposed a label guided adversarial generative network LG-GAN, which takes a target label $t$ and a benign point cloud $\mathcal{P}$ and generate an adversarial counterpart $\mathcal{P}^*$ with small point perturbation. Lately, Kim et al. [8] proposed a minimal 3D adversarial attack aimed at searching for a point set $S$ with the smallest cardinality from $\mathcal{P}$ as perturbed points. Different from the above perturbation attacks, Liu et al. [15] proposed three shape attacks, i.e., perturbation resampling, adversarial sticks, and adversarial sinks, which will deform $\mathcal{P}$ to a large extent to misguide point cloud classifiers.

3.2.2 Point Adding Based Methods

Xiang et al. [27] designed three adversarial point generation methods: independent point generation, point cluster generation, and meaningful object generation. For the first one, the authors added some independent points to $\mathcal{P}$. To make the added points as close as possible to $\mathcal{P}$, the authors proposed a method named initialize-and-shift and considered two perturbation metrics, namely, Hausdorff distance and Chamfer measurement, which are respectively defined as follows:

$$D_{Hausdorff}(\mathcal{P}, \mathcal{P}^*) = \max_{\mathbf{p}_j \in \mathcal{P}^*} \min_{\mathbf{p}_i \in \mathcal{P}} ||\mathbf{p}_i - \mathbf{p}_j||^2_2$$ (2)

$$D_{Chamfer}(\mathcal{P}, \mathcal{P}^*) = \frac{1}{||\mathcal{P}^*||_0} \sum_{\mathbf{p}_j \in \mathcal{P}^*} \min_{\mathbf{p}_i \in \mathcal{P}} ||\mathbf{p}_i - \mathbf{p}_j||^2_2$$ (3)

For the point cluster generation method, the authors defined a farthest distance by $D_{far}(\mathcal{P}^*) = \max_{\mathbf{p}_j \in \mathcal{P}^*} \min_{\mathbf{p}_i \in \mathcal{P}} ||\mathbf{p}_i - \mathbf{p}_j||_2$. They regarded $D_{far}$ and $D_{Chamfer}$ as $D_{dist}$ in (2). As for the last method, they placed small but meaningful objects, e.g., lamps and airplanes, near $\mathcal{P}$ and then considered $D_{Chamfer}$ and $L_p$ norm as $D_{dist}$ in (2). Yang et al. [28] proposed a point-attach method that is similar to Xiang’s but only considered $D_{Chamfer}$ and gave a Chamfer budget $\epsilon$, i.e., $D_{Chamfer}(\mathcal{P}, \mathcal{P}^*) < \epsilon$. Kim et al. [8] put forward a point addition attack similar to the first point generation based method proposed in [27] by attaching minimum additional points to $\mathcal{P}$.

3.2.3 Point Dropping Based Methods

Zheng et al. [31] proposed a critical point dropping based attack by building saliency maps for $\mathcal{P}$ using existing point cloud DNNs. During the generation of saliency maps, the point dropping is superseded by the point shifting that moves critical points to the centroid of $\mathcal{P}$ due to the former’s non-differentiability. Finally, 3D adversarial point cloud $\mathcal{P}^*$ can be obtained by iteratively dropping points from $\mathcal{P}$ in virtue of dynamic saliency maps or dynamic critical subset. Wicker and Kwiatkowska [24] occluded critical points of $\mathcal{P}$ to different extents to figure out the robustness of 3D DNNs. Yang et al. [28] proposed a point-detach attack similar to [31] to mislead point cloud classifier by iteratively detaching critical points from $\mathcal{P}$.

3.2.4 Isometry Transform Methods

Zhao et al. [30] found that existing 3D DNNs are extremely susceptible to isometry transformations, such as rotation. Motivated by this fact, they designed a black box attack equipped with the Thompson sampling, that is to say attackers are not clear about the attacked model. What’s more, they proposed a method dubbed restricted isometry attack.
against a specific model. This is the first work to generate 3D adversarial point clouds without relying on point manipulation mentioned before.

3.3. 3D Adversarial Point Cloud Defense

Inspired by Prisoners’ Problem in steganalysis, we believe that adversarial example defenders usually demonstrate two defensive attitudes: proactive and passive. In 3D settings, the proactive defenders tend to strengthen victim DNNs forwardly by model modification [2] and adversarial training [6] or manipulating all input point clouds [14, 33] irreversibly before classification tasks to counter adversarial attacks. For passive defenders, Yang et al. [28] proposed the first detection-and-reject defense to 3D adversarial point clouds. Motivated by the fact that adversarial examples would destabilize the output of the victim model (PointNet in their experiments), they used the output statistical features of M-order perturbation set of a sample \( P \) for adversarial point cloud detection. Liu et al. [16] proposed PointGuard that uses the output labels of multiple subsampled versions of \( P \) on the victim model to determine the final predictive label for \( P \). It is worth mentioning that both [28] and [16] rely on the robustness of victim DNNs to downsampled examples and noisy point clouds, which may involve model retraining. Passive defenders often only play a supervisory role, like the consultant Eve in AADM. Their passiveness is mainly due to restricted permissions or simply wanting to fight against adversarial examples at a relatively small cost.

3.4. 3D steganalysis

Yang and Irvissimitzis [29], who are the pioneers in 3D steganalysis, proposed a 208-D steganalytic feature set \( \text{YANG208} \) for detecting 3D mesh steganography or watermarking. Li et al. [10] reduced \( \text{YANG208} \) to a 52-D lite \( \text{LFS52} \) by abandoning some redundant geometrical features. Later, Li and Bors [11] combined mesh features in the spherical coordinate system with \( \text{LFS52} \) to generate a 76-D feature set. In the same period, Kim et al. [7] extended \( \text{LFS52} \) to a 64-D feature set by considering the mean curvature, total curvature, and edge normal as additional features to enhance the discrimination between cover meshes and stego meshes. Li et al. [13] proposed to extract edge vector features both in the Cartesian coordinate system and Laplacian coordinate system to design a 3D steganalyzer with better performance. Lately, Li and Bors [12] applied multi-resolution 3D wavelet analysis to 3D steganalysis and designed a new set of 228-D steganalytic features. Inspired by the tensor voting theory, Zhou et al. [34] proposed a 100-D normal voting tensor-based steganalytic feature set, which is universally acknowledged as the best steganalytic feature set [35].

4. Our 3D Adversarial Point Cloud Detector

4.1. Motivation

The above defenses are single-blind at most. In other words, they rely on the exact knowledge of either attack means and victim models extremely. However, for one thing, defenders, like Eve in AADM, may know the exact information neither on attacks from Alice nor on Wendy’s models, and they do not have the right to fiddle with models and input data. For another, model holder Wendy may only want to defend against Alice’s attacks at the least cost. To adapt to the scenarios described above, we devise a double-blind 3D adversarial point cloud detector, abbreviated as \( 3D-\text{APCD} \).

4.2. Framework of 3D-APCD

We follow the design idea of existing 3D steganalyzers. Specifically, \( 3D-\text{APCD} \) consists of three basic steps: calibration, calibrated feature extraction, and classification, as illustrated in Fig. 3. Given a point cloud \( P \), unknown whether it is benign or malicious. In the first step, we carry out the principal component analysis on \( P \) and then obtain a new coordinate system composed of three principal component vectors. We transform \( P \) by aligning the original coordinate system with the newly obtained one. Then we normalize the transformed \( P \) into a unit cube whose center point is \((0.5, 0.5, 0.5)\). The above operations can alleviate the impact of affine transformation on the subsequent feature extraction and classification for \( P \). Finally, a reference point cloud \( \bar{P} \) is produced by smoothing \( P \). In this work, we apply the Laplacian smoothing with one iteration [34], which modifies \( P \) to \( \bar{P} \) as below:

\[
\bar{p}_i = p_i + \tau \sum_{p_j \in N_k(p_i)} w_{i,j}(p_j - p_i)
\]

such that \( \sum_{p_j \in N_k(p_i)} w_{i,j} = 1 \) \hspace{1cm} (4)

Here \( N_k(p_i) \) denotes the \( k \) nearest neighbors of \( p_i \), \( \tau \in (0, 1) \) is a scale factor, and \( w_{i,j} \) is calculated by

\[
w_{i,j} = \frac{||p_j - p_i||^q_p}{\sum_{p_j \in N_k(p_i)} ||p_j - p_i||^q_p} \hspace{1cm} (5)
\]

where \( p \) and \( q \) are set to 2 and \(-1\) respectively in our experiments. Note that the neighbors of \( p_i \) and \( \tau \) are critical to the Laplacian smoothing, and their settings will be discussed in Section 5. In the second step, we can obtain the calibrated features of \( P \) by calculating the norm difference between feature vectors of \( P \) and \( \bar{P} \). However, existing 3D steganalysis techniques are only applicable for polygon meshes but not suitable for point clouds. Consequently, we put forward a feature extraction method tailored for point clouds, which
Figure 3. Framework of 3D-APCD based on learning from the statistics of residual features and classification via Fisher linear ensemble classifier.

will be described in the next section. Ultimately, the extracted calibrated features will be thrown into the classifier to determine whether \(P\) is malicious or not. We use the off-the-shelf Fisher linear discriminant (FLD) ensemble classifier proposed in [9] due to its fast training and reliable classification accuracy.

4.3. Feature Design for 3D Adversarial Point Clouds

Unlike polygon meshes, point clouds are made up of numerous unorganized points so that we know neither their topology nor their geometry. Therefore, the feature extraction in 3D-APCD is to be based on estimation. For simplicity, let \(\Phi = \{\phi_1, \cdots, \phi_n\}\) be the calibrated feature set we expect to extract.

**Surface normal based features.** We obtain the surface normal at each point of \(P\) by using the \(k\)-neighborhood least-squares normal estimation [5], which was originally applied to the surface reconstruction from point clouds. Specifically, we perform the PCA on \(N_k(p_i)\) to estimate the surface normal \(n_i\). To this end, we first compute the covariance matrix \(CM_i\) for each \(N_k(p_i)\), which is defined by

\[
CM_i = \sum_{p_j \in N_k(p_i)} (p_j - \bar{p}_i)(p_j - \bar{p}_i)^T
\]

where \(p_j = [v_{j,x}, v_{j,y}, v_{j,z}]^T\) and \(\bar{p}_i = \frac{1}{N} \sum_{p_j \in N_k(p_i)} p_j\).

The determination of \(N_k(p_i)\) in this case will be discussed in Section 5. k-D tree, a special case of spatial binary search tree, is used here for acquiring the adjacency of \(p_i\). By computing the eigenvector corresponding to the minimum eigenvalue of \(CM_i\), we can obtain the surface normal \(n_i\) at \(p_i\). Similar operations are performed on \(\bar{p}_i\) simultaneously so that we can get the smoothed version \(\tilde{n}_i\) of \(n_i\). The angle between \(n_i\) and \(\tilde{n}_i\) is computed as follows:

\[
\theta_i = \arccos \frac{n_i \cdot \tilde{n}_i}{||n_i|| \cdot ||\tilde{n}_i||}, i = 1, \cdots, N.
\]

So far, a \(N \times 1\) calibrated feature matrix \(CFM\) composed of \(N\) angels in this case is generated. We perform a logarithmic mapping on \(CFM\) and get

\[
CFM_{nl} = \ln(CFM + \sigma)
\]

where \(\sigma\) is a small offset constant used for constraining feature distribution and circumventing numerical calculation errors. Note that the introduced non-linear mapping is of significance to 3D-APCD since it can mitigate the impact of excessive value difference of components in \(CFM\) on the classifier. Then, we calculate four moments, i.e., mean, variance, skewness, and kurtosis on each column vector of \(CFM_{nl}\) and obtain the first four feature values of \(\Phi\) by

\[
\Phi_{1\sim4} = \{\text{mean}(CFM_{nl}), \text{var}(CFM_{nl}), \text{skewness}(CFM_{nl}), \text{kurtosis}(CFM_{nl})\}
\]

**Normal voting tensor based features.** Apart from the first-order local description of point clouds, such as point coordinates and point normals, a second-order one is considered in 3D-APCD. As we know, a symmetric tensor \(T\) can be easily diagonalized as follows:

\[
T = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \begin{bmatrix} q_1^T \\ q_2^T \\ q_3^T \end{bmatrix}
\]

where \(q_i\) is the eigenvector corresponding to the eigenvalue \(\lambda_i\). Since \(T\) is symmetrical, without loss of generality, we...
set $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$. In tensor voting theory [17], $T$ can be depicted by an ellipsoid and be rewritten by

$$
T = (\lambda_1 - \lambda_2)q_1^Tq_1 + (\lambda_2 - \lambda_3)(q_1^Tq_2 + q_2^Tq_2) + \lambda_3(q_1^Tq_3 + q_2^Tq_3 + q_3^Tq_3)
$$

(11)

Here $q_1^Tq_1$, $q_1^Tq_2 + q_2^Tq_2$, and $q_1^Tq_3 + q_2^Tq_3 + q_3^Tq_3$ describe a stick, a plate, and a ball respectively, and their salient values are $\lambda_1 - \lambda_2$, $\lambda_2 - \lambda_3$, and $\lambda_3$ respectively. Unfortunately, the off-the-shelf tensor communication methods [34] are tailored for polygon meshes, which are not suitable for point clouds. [18] proposed a point-based tensor voting model, but through experiments, we find features extracted by such models are not efficacious at distinguishing adversarial examples from benign ones. In order to effectively apply tensor voting to 3D-APCD, we construct $T$ with the surface normals estimated above like this:

$$
T_i = \sum_{p_j \in N_k(p_i)} \mu_{ij} n_j \cdot n_j^T, \quad i = 1, \ldots, N
$$

(12)

Here $\mu_{ij}$ is a voting weight of $p_j$ which is defined like (5). What’s more, we provide two additional voting tensors with different neighbors:

$$
T_i = \sum_{p_j \in N_{2k}(p_i)} \mu_{ij} n_j \cdot n_j^T
$$

$$
T_i = \sum_{p_j \in N_{3k}(p_i)} \mu_{ij} n_j \cdot n_j^T
$$

(13)

The value of $k$ in Eqs. (12) and (13) is to be discussed in Section 5. After the voting process, a 4-D vector $\lambda_i = [\lambda_1 - \lambda_2, \lambda_2 - \lambda_3, \lambda_3, \frac{\lambda_1 + \lambda_2 + \lambda_3}{3}]$ of each $T_i$ are taken as features, and we can also obtain its smoothed version $\bar{\lambda}_i$ in the same way. The $L_2$ norms of $\lambda_i - \bar{\lambda}_i$ under three different neighbor scales compose a $N \times 12$ CFM. Finally, we can determine the calibrated feature $\Phi_{53\sim64}$ through (8) and (9).

**Curvature based features.** To extract curvature information at each point, we take a simple approach that estimate the principal curvatures of $p_i$ by computing the minimum and maximum eigenvalues of the covariance matrix associated with $k$ nearest neighbors of $p_i$. Firstly, we construct a projection matrix with normal $n_i$ as follows:

$$
PM_i = I - n_i \cdot n_i^T
$$

(14)

Then we project all surface normals of $k$ nearest neighbors of $p_i$ into the tangent plane determined by $n_i$ by

$$
n_j^{proj} = PM_i n_j
$$

(15)

Here the neighbors of $p_i$ will be determined in Section 5. Finally, the PCA is performed on these projected surface normals $n_j^{proj}$, and then the maximum and the minimum eigenvalues, denoted by $K_1$ and $K_2$ respectively, are taken as the principal curvatures of $p_i$. We consider three curvatures as features, i.e., Gaussian curvature, mean curvature, and curvature ratio, which are respectively defined by

$$
GC_i = K_1 \times K_2
$$

$$
MC_i = K_1 + K_2
$$

$$
CR_i = \frac{\min\{|K_1|, |K_2|\}}{\max\{|K_1|, |K_2|\}}
$$

(16)

Similar operations are performed on $\tilde{P}$ to generate $\tilde{CF}_i$, $\tilde{MC}_i$, and $\tilde{CR}_i$. $|CF_i - \tilde{CF}_i|$, $|MC_i - \tilde{MC}_i|$, and $|CR_i - \tilde{CR}_i|$ are calculated for all points $P$ to generate a $N \times 3$ CFM. By (8) and (9), we can obtain $\Phi_{53\sim64}$.

5. Experiments

5.1. Setup of Experiments

**Dataset**. Our experiments are reported on ModelNet40 which contains 12,311 CAD models that belong to 40 common object categories. We sample 1024 points from each model uniformly to form a new point cloud and split these point clouds into two groups, where 9,843 for point clouds for training and 2,468 point clouds for testing.

**Evaluation Metric.** We adopt the FLD ensemble classifier for example classification due to its fast training and reliable classification accuracy. The performance of 3D-APCD is evaluated by the “out-of-bag” error $E_{OOB}$, which is an unbiased estimate of test error defined by

$$
E_{OOB} = \frac{1}{2N_{data}} \sum_{i=1}^{N_{data}} \left(B^{(l)}(\Phi_i) + 1 - B^{(l)}(\Phi_i^*) \right)
$$

(17)

where $N_{data}$ is the number of examples; $\Phi_i$ and $\Phi_i^*$ are features of benign examples and malicious examples respectively; $l$ denotes the number of base learners used in the FLD ensemble classifier and $B^{(l)}(\cdot) \in \{0, 1\}$.

5.2. Training Set Size

To figure out whether 3D-APCD is sensitive to changes in the training set size, five subsets with different sizes, i.e., $\{150, 250, 350, 450, 550\}$, are generated from the ModelNet40 training set as benign examples. Simultaneously, 30 different subsets are created by randomly selecting 200 point clouds from the ModelNet40 testing set 30 times. We select three 3D adversarial attack methods, i.e., iterative gradient $L_2$ attack [14], adversarial independent points [27] (AIP), and point-detach attack [28] (PD), which are based

---

1 http://modelnet.cs.princeton.edu
Table 1. Average OOB errors and median OOB errors of three different types of 3D adversarial point clouds under five training set sizes

| $L_2$ attack [14] | 300 | 500 | 700 | 900 | 1100 |
|-------------------|-----|-----|-----|-----|-----|
| $E_{oob}$(Average) | 0.0510 | 0.0465 | 0.0452 | 0.0439 | 0.0431 |
| $E_{oob}$(Median)  | 0.0499 | 0.0453 | 0.0478 | 0.0463 | 0.0443 |
| AIP attack [27]    | 300 | 500 | 700 | 900 | 1100 |
| $E_{oob}$(Average) | 0.0413 | 0.0401 | 0.0399 | 0.0385 | 0.0387 |
| $E_{oob}$(Median)  | 0.0415 | 0.0413 | 0.0406 | 0.0392 | 0.0400 |
| PD attack [28]     | 300 | 500 | 700 | 900 | 1100 |
| $E_{oob}$(Average) | 0.2956 | 0.2913 | 0.2899 | 0.2901 | 0.2872 |
| $E_{oob}$(Median)  | 0.3000 | 0.2932 | 0.2912 | 0.2920 | 0.2917 |

Table 2. Average OOB errors and median OOB errors of three different types of 3D adversarial point clouds with $\tau = 0.1, 0.3, 0.5, 0.7, 0.9$

| $L_2$ attack [14] | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
|-------------------|-----|-----|-----|-----|-----|
| $E_{oob}$(Average) | 0.0521 | 0.0450 | 0.0479 | 0.0407 | 0.0421 |
| $E_{oob}$(Median)  | 0.0502 | 0.0437 | 0.0500 | 0.0417 | 0.0430 |
| AIP attack [27]    | 0.0422 | 0.0410 | 0.0399 | 0.0392 | 0.0400 |
| $E_{oob}$(Average) | 0.0431 | 0.0402 | 0.0407 | 0.0399 | 0.0398 |
| $E_{oob}$(Median)  | 0.0431 | 0.0402 | 0.0407 | 0.0399 | 0.0398 |
| PD attack [28]     | 0.2956 | 0.2972 | 0.2711 | 0.2472 | 0.2694 |
| $E_{oob}$(Average) | 0.3000 | 0.3000 | 0.2667 | 0.2500 | 0.2583 |
| $E_{oob}$(Median)  | 0.3000 | 0.3000 | 0.2667 | 0.2500 | 0.2583 |

5.3. Parameter Analysis

As we mentioned in Section 4, the smoothing coefficient $\tau$ and neighbors of $p_i$ involved in four different steps of $3D$-$APCD$ are five pivotal parameters that may affect the detection performance. Without loss of generality, let $k_{ls}$, $k_{nor}$, $k_{cur}$, and $k_{tot}$ denote the number of neighbors of $p_i$ in the Laplacian smoothing and three feature extraction processes, respectively. Figuring out their settings affect the detection performance of $3D$-$APCD$ allows us to make bet-

Table 3. Average OOB errors and median OOB errors of three different types of 3D adversarial point clouds with $k_{ls} = 5, 10, 15, 20, and 25$

| $L_2$ attack [14] | 5 | 10 | 15 | 20 | 25 |
|-------------------|---|----|----|----|----|
| $E_{oob}$(Average) | 0.0565 | 0.0672 | 0.0692 | 0.0676 | 0.0701 |
| $E_{oob}$(Median)  | 0.0542 | 0.0646 | 0.0708 | 0.0646 | 0.0698 |
| AIP attack [27]    | 5 | 10 | 15 | 20 | 25 |
| $E_{oob}$(Average) | 0.0427 | 0.0492 | 0.0481 | 0.0517 | 0.0531 |
| $E_{oob}$(Median)  | 0.0422 | 0.0501 | 0.0467 | 0.0525 | 0.0519 |
| PD attack [28]     | 5 | 10 | 15 | 20 | 25 |
| $E_{oob}$(Average) | 0.2958 | 0.3118 | 0.3667 | 0.3861 | 0.3910 |
| $E_{oob}$(Median)  | 0.3001 | 0.3167 | 0.3667 | 0.3583 | 0.3617 |

Table 4. Average OOB errors and median OOB errors of three different types of 3D adversarial point clouds with $k_{nor} = 5, 10, 15, 20, and 25$

| $L_2$ attack [14] | 5 | 10 | 15 | 20 | 25 |
|-------------------|---|----|----|----|----|
| $E_{oob}$(Average) | 0.0571 | 0.0600 | 0.0494 | 0.0507 | 0.0513 |
| $E_{oob}$(Median)  | 0.0563 | 0.0583 | 0.0479 | 0.0500 | 0.0532 |
| AIP attack [27]    | 5 | 10 | 15 | 20 | 25 |
| $E_{oob}$(Average) | 0.0412 | 0.0456 | 0.0461 | 0.0432 | 0.0433 |
| $E_{oob}$(Median)  | 0.0425 | 0.0470 | 0.0472 | 0.0427 | 0.0441 |
| PD attack [28]     | 5 | 10 | 15 | 20 | 25 |
| $E_{oob}$(Average) | 0.2996 | 0.3611 | 0.3894 | 0.3956 | 0.4001 |
| $E_{oob}$(Median)  | 0.3012 | 0.4001 | 0.3970 | 0.4007 | 0.4213 |

Table 5. Average OOB errors and median OOB errors of three different types of 3D adversarial point clouds with $k_{cur} = 5, 10, 15, 20, and 25$

| $L_2$ attack [14] | 5 | 10 | 15 | 20 | 25 |
|-------------------|---|----|----|----|----|
| $E_{oob}$(Average) | 0.0542 | 0.0560 | 0.0536 | 0.0537 | 0.0563 |
| $E_{oob}$(Median)  | 0.0537 | 0.0542 | 0.0500 | 0.0521 | 0.0553 |
| AIP attack [27]    | 5 | 10 | 15 | 20 | 25 |
| $E_{oob}$(Average) | 0.0427 | 0.0443 | 0.0434 | 0.0429 | 0.0430 |
| $E_{oob}$(Median)  | 0.0417 | 0.0457 | 0.0452 | 0.0421 | 0.0400 |
| PD attack [28]     | 5 | 10 | 15 | 20 | 25 |
| $E_{oob}$(Average) | 0.2899 | 0.2917 | 0.3988 | 0.3856 | 0.3970 |
| $E_{oob}$(Median)  | 0.3027 | 0.3133 | 0.3833 | 0.38347 | 0.3921 |
Table 6. Average OOB errors and median OOB errors of three different types of 3D adversarial point clouds with $k_{v}$, $t$, and $\sigma$.

| $L_2$ attack [14] | 5 | 10 | 15 | 20 | 25 |
|-------------------|---|----|----|----|----|
| $E_{ooob}$(Average) | 0.0537 | 0.0807 | 0.0791 | 0.0776 | 0.0794 |
| $E_{ooob}$(Median)  | 0.0558 | 0.0812 | 0.0771 | 0.0771 | 0.0801 |
| AIP attack [27]    | 5 | 10 | 15 | 20 | 25 |
| $E_{ooob}$(Average) | 0.0437 | 0.0512 | 0.0557 | 0.0512 | 0.0537 |
| $E_{ooob}$(Median)  | 0.0428 | 0.0520 | 0.0543 | 0.0533 | 0.0521 |
| PD attack [28]     | 5 | 10 | 15 | 20 | 25 |
| $E_{ooob}$(Average) | 0.2954 | 0.3733 | 0.3800 | 0.3886 | 0.4000 |
| $E_{ooob}$(Median)  | 0.3012 | 0.3832 | 0.3827 | 0.3833 | 0.3911 |

ter parameter selection, and thus we correspondingly conduct five groups of experiments. Based on the observation in Section 5.2, we randomly select 250 point clouds from the ModelNet training set as benign examples to train 3D-APCD. Here we also select the three attack methods considered in Section 5.2 with the same configuration. In each group of experiments, the values of $\tau$, $k_{l}$, $k_{n}$, and $k_{c}$ are initially set to 0.1, 5, 5, and 5 respectively. Relevant experimental results are shown in Tables 2, 3, 4, 5, 6. For $\tau$, it is clear to see that appropriately increasing its value is conducive to improve 3D-APCD generally speaking, their increase has less effect on methods based on point shifting and points adding than that based on point dropping. Therefore, considering more neighbors of $p_i$ is not necessarily beneficial to our 3D-APCD. In fact, small $k_{l}$, $k_{n}$, $k_{c}$, and $k_{v}$ can achieve better detection results in our experiments.

5.4. Evaluation on 3D Adversarial Attack

We test 3D-APCD with $k_{l}$, $k_{n}$, $k_{c}$, and $k_{v}$ are initially set to 0.1, 5, 5, and 5 and $\tau$ = 0.5 against various untargeted 3D adversarial attacks except for LG-GAN. Relevant experimental results are shown in Table 7, where $\epsilon$, $N_{iter}$, $\sigma_{stick}$, $\sigma_{sink}$, $N_{aip}$, and $N_{ac}$ denote the perturbation size, number of iterations, target label, number of sticks, number of sink points, number of added independent points, number of added clusters respectively. It can be seen from Table 7 that 3D-APCD can achieve high detection accuracy on the attack methods based on small point perturbation. It should be emphasized that while the detection accuracy of 3D-APCD will be greatly reduced once faced with 3D adversarial attacks based on point dropping and shape attack, it can still act as the role of Eve and offer useful detection information for Wandy in our AADM.

Table 7. Average OOB errors and median OOB errors of different types of 3D adversarial point clouds with varying settings.

| $L_2$ attack [14] | $\epsilon = 1$ | $\epsilon = 2$ | $\epsilon = 3$ |
|-------------------|----------------|----------------|----------------|
| $E_{ooob}$(Average) | 0.0899 | 0.0432 | 0.0342 |
| $E_{ooob}$(Median)  | 0.1001 | 0.0411 | 0.0351 |
| Chamfer [15]       | $N_{iter} = 10$ | $N_{iter} = 20$ | $N_{iter} = 30$ |
| $E_{ooob}$(Average) | 0.0383 | 0.0403 | 0.0463 |
| $E_{ooob}$(Median)  | 0.0396 | 0.0416 | 0.0476 |
| Gradient.Proj [15] | $\epsilon = 1$ | $\epsilon = 2$ | $\epsilon = 3$ |
| $E_{ooob}$(Average) | 0.1401 | 0.0954 | 0.0588 |
| $E_{ooob}$(Median)  | 0.1532 | 0.1012 | 0.0591 |
| Pertb. resample [15] | $\epsilon = 1$ | $\epsilon = 2$ | $\epsilon = 3$ |
| $E_{ooob}$(Average) | 0.1981 | 0.1254 | 0.0988 |
| $E_{ooob}$(Median)  | 0.1932 | 0.1412 | 0.1001 |
| LG-GAN [32]        | $t = 1$ | $t = 8$ | $t = 22$ |
| $E_{ooob}$(Average) | 0.0531 | 0.0566 | 0.0598 |
| $E_{ooob}$(Median)  | 0.0425 | 0.0578 | 0.0572 |
| Adv.sticks [15]    | $\sigma_{stick} = 50$ | $\sigma_{stick} = 100$ | $\sigma_{stick} = 150$ |
| $E_{ooob}$(Average) | 0.4156 | 0.3412 | 0.2912 |
| $E_{ooob}$(Median)  | 0.4431 | 0.3534 | 0.3978 |
| Adv.sinks [15]     | $\sigma_{sink} = 50$ | $\sigma_{sink} = 100$ | $\sigma_{sink} = 150$ |
| $E_{ooob}$(Average) | 0.4414 | 0.3912 | 0.3413 |
| $E_{ooob}$(Median)  | 0.4569 | 0.3988 | 0.3281 |
| AIP [27]           | $N_{aip} = 100$ | $N_{aip} = 150$ | $N_{aip} = 200$ |
| $E_{ooob}$(Average) | 0.1921 | 0.1342 | 0.0832 |
| $E_{ooob}$(Median)  | 0.2169 | 0.1487 | 0.0981 |
| ACP [27]           | $N_{ac} = 1$ | $N_{ac} = 2$ | $N_{ac} = 3$ |
| $E_{ooob}$(Average) | 0.2747 | 0.2218 | 0.1931 |
| $E_{ooob}$(Median)  | 0.2839 | 0.2424 | 0.2104 |
| AOP [27]           | plane | chair | bed |
| $E_{ooob}$(Average) | 0.4001 | 0.3812 | 0.3944 |
| $E_{ooob}$(Median)  | 0.3941 | 0.3910 | 0.4123 |
| PD attack [28]     | $N_{d} = 40$ | $N_{d} = 60$ | $N_{d} = 80$ |
| $E_{ooob}$(Average) | 0.3432 | 0.2914 | 0.2341 |
| $E_{ooob}$(Median)  | 0.3369 | 0.3028 | 0.2402 |

to help us comprehend adversarial attack and defense more generally. Then we rethink two significant but vague concepts, i.e., active defense and passive defense, through the lens of steganalysis. Most importantly, we design a detector tailored for 3D adversarial point clouds in virtue of different geometrical features of point clouds. While our detector is double-blind, extensive experimental results demonstrate that the proposed 3D adversarial point cloud detector can achieve fairly good detection performance on multiple types of 3D adversarial point clouds.

6. Conclusion

In this work, we apply 3D steganalysis techniques to 3D adversarial point cloud detection. We first introduce AADM...
References

[1] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In Proceedings of IEEE Symposium on Security and Privacy, pages 39–57, 2017.

[2] Xiaoyi Dong, Dongdong Chen, Hang Zhou, Gang Hua, Weiming Zhang, and Nenghai Yu. Self-robust 3d point recognition via gather-vector guidance. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11513–11521, 2020.

[3] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2015.

[4] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(12):4338–4364, 2021.

[5] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. SIGGRAPH Comput. Graph., 26(2):71–78, July 1992.

[6] Yulong Cao Qi Alfred Chen Jiachen Sun, Karl Koenig and Zhuoqing Mao. On the adversarial robustness of 3d point cloud classification. 2021.

[7] Dongkyu Kim, Han-Ul Jang, Hak-Yeol Choi, Jeongho Son, In-Jae Yu, and Heung-Kyu Lee. Improved 3d mesh steganalysis using homogeneous kernel map. In Proc. Int. Conf. inf. Sci. Appl., Macau, China, 2017, pages 358 – 365.

[8] Jaeyeon Kim, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Minimal adversarial examples for deep learning on 3d point clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 7797–7806, 2021.

[9] Jan Kodovsky, Jessica Fridrich, and Vojtˇech Holub. Ensemble classifiers for steganalysis of digital media. IEEE Trans. Inf. Forensics Secur., 7(2):432–444, Apr. 2012.

[10] Zhenyu Li and Adrian G. Bors. 3D mesh steganalysis using local shape features. In Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Barcelona, Spain, 2016, pages 2144–2148.

[11] Zhenyu Li and Adrian G. Bors. Steganalysis of 3D objects using statistics of local feature sets. Inf. Sci., 415-416:85–99, Nov. 2017.

[12] Zhenyu Li and Adrian G. Bors. Steganalysis of meshes based on 3D wavelet multiresolution analysis. Inf. Sci., 522:164–179, 2020.

[13] Zhenyu Li, Daofu Gong, Fenlin Liu, and Adrian G Bors. 3D steganalysis using the extended local feature set. In Proc. IEEE Int. Conf. Image Process., Athens, Greece, 2018, pages 1683–1687.

[14] Daniel Liu, Ronald Yu, and Hao Su. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. In Proceedings of IEEE International Conference on Image Processing, pages 2279–2283, 2019.

[15] Daniel Liu, Ronald Yu, and Hao Su. Adversarial shape perturbations on 3d point clouds. In Proceedings of the European Conference on Computer Vision, pages 88–104, 2020.

[16] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6186–6195, 2021.

[17] Gérard Medioni, Chi-Keung Tang, and Mi-Suen Lee. Tensor voting: Theory and applications. In Proceedings of 12th Congres Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle, 2000.

[18] Min Ki Park, Seung Joo Lee, and Kwan H Lee. Multi-scale tensor voting for feature extraction from unstructured point clouds. Graphical Models, 74(4):197–208, 2012.

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

[20] 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.

[21] Gustavus J. Simmons. The Prisoners’ Problem and the Subliminal Channel, pages 51–67. Springer US, Boston, MA, 1984.

[22] Jinwei Wang, Junjie Zhao, Qilin Yin, Xiangyang Luo, Yuhui Zheng, Yun Qing Shi, and Sunil K. Jha. Sm2net: A new deep convolutional neural network model for adversarial example detection. IEEE Transactions on Multimedia, 2021.

[23] Yue Wang and Justin Solomon. Object dgcnn: 3d object detection using dynamic graphs. arXiv preprint arXiv:2110.06923, 2021.

[24] Matthew Wicker and Marta Kwiatkowska. Robustness of 3d deep learning in an adversarial setting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11759–11767, 2019.

[25] Haibin Wu, Andy T. Liu, and Hung yi Lee. Defense for black-box attacks on anti-spoofing models by self-supervised learning. arXiv preprint arXiv:2006.03214, 2020.

[26] Ziyi Wu, Yueqi Duan, He Wang, Qingnan Fan, and Leonidas J. Guibas. If-defense: 3d adversarial point cloud defense via implicit function based restoration. arXiv preprint arXiv:2010.05272, 2021.

[27] Chong Xiang, Charles R. Qi, and Bo Li. Generating 3d adversarial point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9128–9136, 2019.

[28] Jiancheng Yang, Qiang Zhang, Rongyao Fang, Bingbing Ni, Chong Xiang, Charles R. Qi, and Bo Li. Generating 3d adversarial point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7797–7806, 2021.

[29] Fangbin Xiong, Zhongming Gu, and Fan Yang. Pointguard: Provably robust 3d point cloud classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6186–6195, 2021.

[30] Yue Zhao, Yuwei Wu, Caihua Chen, and Andrew Lim. On isometry robustness of deep 3d point cloud models under adversarial attacks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1198–1207, 2020.
[31] Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. Pointcloud saliency maps. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1598–1606, 2019.

[32] Hang Zhou, Dongdong Chen, Jing Liao, Kejiang Chen, Xiaoyi Dong, Kunlin Liu, Weiming Zhang, Gang Hua, and Nenghai Yu. Lg-gan: Label guided adversarial network for flexible targeted attack of point cloud based deep networks. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10353–10362, 2020.

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

[34] H. Zhou, K. Chen, W. Zhang, C. Qin, and N. Yu. Feature-preserving tensor voting model for mesh steganalysis. IEEE Trans. Vis. Comput. Graph., 27(1):57–67, Jan. 2021.

[35] Hang Zhou, Weiming Zhang, Kejiang Chen, Weixiang Li, and Nenghai Yu. Three-dimensional mesh steganography and steganalysis: A review. IEEE Trans. Vis. Comput. Gr., 2021.