PointCert: Point Cloud Classification with Deterministic Certified Robustness Guarantees

Jinghuai Zhang[1] Jinyuan Jia[2] Hongbin Liu[1] Neil Zhenqiang Gong[1]
Duke University[1] UIUC[2]
{jinghuai.zhang, hongbin.liu, neil.gong}@duke.edu,
{jinyuan}@illinois.edu,

Abstract

Point cloud classification is an essential component in many security-critical applications such as autonomous driving and augmented reality. However, point cloud classifiers are vulnerable to adversarially perturbed point clouds. Existing certified defenses against adversarial point clouds suffer from a key limitation: their certified robustness guarantees are probabilistic, i.e., they produce an incorrect certified robustness guarantee with some probability. In this work, we propose a general framework, namely PointCert, that can transform an arbitrary point cloud classifier to be certifiably robust against adversarial point clouds with deterministic guarantees. PointCert certifiably predicts the same label for a point cloud when the number of arbitrarily added, deleted, and/or modified points is less than a threshold. Moreover, we propose multiple methods to optimize the certified robustness guarantees of PointCert in three application scenarios. We systematically evaluate PointCert on ModelNet and ScanObjectNN benchmark datasets. Our results show that PointCert substantially outperforms state-of-the-art certified defenses even though their robustness guarantees are probabilistic.

1. Introduction

Point cloud classification [11,29,30,38,44,50] has many safety-critical applications, including but not limited to, autonomous driving and augmented reality. However, various studies [12, 16, 22, 32, 40, 43, 46, 51, 52] showed that point cloud classification is vulnerable to *adversarial point clouds*. In particular, an attacker can carefully add, delete, and/or modify a small number of points in a point cloud to make it misclassified by a point cloud classifier.

Existing defenses against adversarial point clouds can be categorized into *empirical defenses* [7,21,33,41,46,49,53] and *certified defenses* [5, 6, 23]. The key limitation of empirical defenses is that they cannot provide formal guarantees, and thus are often broken by advanced, adaptive attacks [34]. Therefore, we focus on certified defenses in this work. Randomized smoothing [5] and PointGuard [23] are two state-of-the-art certified defenses against adversarial point clouds. In particular, randomized smoothing adds random noise (e.g., Gaussian noise) to a point cloud, while PointGuard randomly subsamples a point cloud. Due to the randomness, their certified robustness guarantees are *probabilistic*, i.e., they produce incorrect robustness guarantees with some probability (called *error probability*). For instance, when the error probability is 0.001, they produce incorrect robustness guarantees for 1 out of 1,000 point-cloud classifications on average. Such probabilistic guarantees are insufficient for security-critical applications that frequently classify point clouds.

In this work, we propose PointCert, the first certified defense that has *deterministic* robustness guarantees against adversarial point clouds. PointCert can transform an arbitrary point cloud classifier f (called *base point cloud classifier*) to be certifiably robust against adversarial point clouds. Specifically, given a point cloud and a base point cloud classifier f, PointCert first divides the point cloud into multiple disjoint sub-point clouds using a hash function, then uses f to predict a label for each sub-point cloud, and finally takes a majority vote among the predicted labels as the predicted label for the original point cloud. We prove that PointCert certifiably predicts the same label for a point cloud when the number of arbitrarily added, deleted, and/or modified points is no larger than a threshold, which is known as *certified perturbation size*. Moreover, we also prove that our derived certified perturbation size is *tight*, i.e., without making assumptions on the base point cloud classifier f, it is theoretically impossible to derive a certified perturbation size for PointCert that is larger than ours.

We consider three scenarios about how PointCert could be applied in practice and propose methods to optimize the performance of PointCert in these scenarios. In particular, we consider two parties: *model provider* and *customer*. A model provider (e.g., Google, Meta) has enough labeled

data and computation resource to train a base point cloud classifier f and shares it with customers (e.g., a less resourceful company). Given f, a customer uses PointCert to classify its (adversarial) point clouds. We note that the model provider and customer can be the same entity, e.g., a company trains and uses f itself. We consider three scenarios, in which f is trained by the model provider differently and/or used by a customer differently.

Scenario I represents a naive application of PointCert, in which the base point cloud classifier f is trained using a standard training algorithm and a customer directly applies PointCert to classify its point clouds based on f. PointCert achieves suboptimal performance in Scenario I because f, trained on point clouds, is not accurate at classifying sub-point clouds as they have different distributions. Therefore, in Scenario II, we consider a model provider trains f to optimize the performance of PointCert. In particular, the model provider divides each training point cloud into multiple sub-point clouds following PointCert and trains f based on sub-point clouds. In Scenario III, we consider the model provider has trained f using a standard training algorithm (like Scenario I). However, instead of directly applying f to classify sub-point clouds, a customer prepends a *Point Completion Network (PCN)* [48] to f. Specifically, a PCN takes a sub-point cloud as input and outputs a completed point cloud, which is then classified by f. Moreover, we propose a new loss function to train the PCN such that its completed point clouds are classified by f with higher accuracy, which further improves the performance of PointCert.

We perform systematic evaluation on ModelNet40 dataset [1] and two variants of ScanObjectNN dataset [2]. Our experimental results show that PointCert significantly outperforms the state-of-the-art certified defenses (randomized smoothing [5] and PointGuard [23]) even though their robustness guarantees are probabilistic. For instance, on ModelNet40 dataset, PointCert achieves a *certified accuracy* of 79% when an attacker can arbitrarily perturb at most 50 points in a point cloud, where certified accuracy is a lower bound of testing accuracy. Under the same setting, the certified accuracy of both randomized smoothing and PointGuard is 0. We also extensively evaluate PointCert in the three application scenarios.

In summary, we make the following contributions: (1) We propose PointCert, the first certified defense with deterministic robustness guarantees against adversarial point clouds. (2) We design multiple methods to optimize the performance of PointCert in multiple application scenarios. (3) We extensively evaluate PointCert and compare it with state-of-the-art certified defenses.

2. Related Work

Many works [12, 16, 20, 22, 27, 40, 43, 46, 51, 52] developed attacks to point cloud classification. Next, we discuss empirical and certified defenses against these attacks.

Empirical defenses: Many empirical defenses [7, 19, 21, 35, 41, 46, 49, 53] have been proposed to defend against adversarial point clouds. However, those empirical defenses do not have formal robustness guarantees and thus can often be broken by advanced, adaptive attacks. For instance, Sun et al. [34] designed adaptive attacks with 100% attack success rate to adversarial training based defenses [7, 53].

Certified defenses: Randomized smoothing [3, 5, 17, 18, 25, 31] can turn an arbitrary classifier into a certifiably robust one via adding random noise to an input. When generalized to point cloud, randomized smoothing can only certify robustness against point modification attacks [23]. PointGuard [23] creates multiple sub-point clouds from a point cloud and takes a majority vote among them to predict the label of the point cloud. However, unlike PointCert, each sub-point cloud is sampled from the point cloud uniformly at random. Due to the inherent randomness, both randomized smoothing and PointGuard only have probabilistic guarantees. [26, 28] proposed 3DCertify and 3DeformRS to certify robustness of point cloud classification against common 3D transformations, e.g., rotations. However, both methods are not applicable to point addition (or deletion or modification or perturbation) attacks, which can *arbitrarily* manipulate points. Fischer et al. [9] generalized randomized smoothing [5] to certify robustness of point cloud segmentation, which is different from our work since we focus on point cloud classification.

3. Problem Definition

In point cloud classification, a point cloud classifier g predicts a point cloud P into one of c classes (denoted as $\{1, 2, \cdots, c\}$). Formally, we have $g : P \longrightarrow \{1, 2, \cdots, c\}$. A point cloud P is a *set* of points. For simplicity, we denote $P = \{\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_n\}$, where n is the number of points. Each point $\mathbf{e}_i = (e_{i1}, e_{i2}, \cdots, e_{io})$ is a vector that specifies the three coordinates of the point in the three-dimensional space and (optionally) the point's other information such as RGB values that describe the color features.

3.1. Adversarial Point Clouds

Existing attacks to point cloud classification can be categorized into *point addition attacks* [16, 43, 46], *point deletion attacks* [40, 46, 52], *point modification attacks* [12, 16, 43, 46], and *point perturbation attacks* [22, 46, 51]. Specifically, in point addition (or deletion or modification) attacks, an attacker can *arbitrarily* add new points (or delete or modify existing points) to a point cloud. Note that modifying a point is equivalent to deleting an existing point and adding a new point. In point perturbation attacks, an attacker can use any combination of the three operations (i.e., addition, deletion, and modification) to perturb a point cloud.

Given a point cloud P, we use P' to denote its adversarially perturbed version. We use $d(P, P')$ to denote the *perturbation size*, i.e., the minimum number of perturbed (i.e., added, deleted, and/or modified) points that can turn P to P'. Formally, we have $d(P, P') = \max(|P|, |P'|) - |P \cap P'|$, where $|\cdot|$ measures the number of points in a point cloud and $\cap$ represents the intersection between two sets. Suppose we are given a perturbation size t. We use $\mathcal{S}(P, t)$ to denote the set of all possible adversarial point clouds whose perturbation sizes are at most t. Formally, we have $\mathcal{S}(P, t) = \{P' | d(P, P') \leq t\}$.

3.2. Certifiably Robust Point Cloud Classifier

Certified perturbation size: We say a point cloud classifier is certifiably robust if it certifiably predicts the same label for a point cloud when the number of points arbitrarily added, deleted, and/or modified by an attacker is less than a threshold, called *certified perturbation size*. Formally, given a point cloud P and a point cloud classifier g, we say g is certifiably robust for P with a certified perturbation size $t(P)$ if g predicts the same label for the point cloud P and any adversarial point cloud with perturbation size at most $t(P)$, i.e., $g(P') = g(P)$ for $\forall P' \in \mathcal{S}(P, t(P))$.

Probabilistic vs. deterministic guarantees: We say a point cloud classifier g produces an *incorrect* certified perturbation size $t(P)$ for a point cloud P if there exists an adversarial point cloud P' with perturbation size at most $t(P)$ such that g predicts different labels for P' and P, i.e., $\exists P' \in \mathcal{S}(P, t(P))$, $g(P') \neq g(P)$. A certifiably robust point cloud classifier has *probabilistic* guarantees if it produces an incorrect certified perturbation size for a point cloud with an error probability α. A certifiably robust point cloud classifier has *deterministic* guarantees if its produced certified perturbation sizes are always correct.

4. Our PointCert

We first describe our PointCert framework, which builds an ensemble point cloud classifier from an arbitrary point cloud classifier (called *base point cloud classifier*). Then, we derive the certified perturbation size of our ensemble point cloud classifier.

4.1. Building an Ensemble Point Cloud Classifier

Dividing a point cloud into multiple disjoint sub-point clouds: Suppose we have a point cloud $P = \{\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_n\}$, where n is the number of points and $\mathbf{e}_i = (e_{i1}, e_{i2}, \cdots, e_{io})$ $(i = 1, 2, \cdots, n)$ is a point. Our idea is to divide the point cloud P into m sub-point clouds. In particular, our division aims to achieve three goals. The first goal is that an adversarially perturbed point should influence a small number of sub-point clouds. In other words, most sub-point clouds are not influenced when the number of adversarially perturbed points is small. The second goal is that a point should be assigned into a sub-point cloud deterministically. As we will see in the next subsection, the first two goals enable us to derive a deterministic certified perturbation size of PointCert for a point cloud. The third goal is that the sub-point clouds should contain similar number of points. In particular, if some sub-point clouds contain (much) less number of points, then the base point cloud classifier may be more likely to misclassify them. As a result, our ensemble point cloud classifier is less accurate. As we will see in our experiments, the third goal enables PointCert to produce larger certified perturbation sizes.

To reach the first goal, we propose to assign each point into one sub-point cloud. Therefore, an adversarially added or deleted point only influences one sub-point cloud, i.e., adding one point only influences the sub-point cloud which the added point is assigned to while deleting one point only influences the sub-point cloud from which the point is deleted. An adversarially modified point influences at most two sub-point clouds, i.e., the sub-point clouds which the point belongs to before and after modification. To reach the second goal, we propose to use the coordinates $e_{i1}, e_{i2}, \cdots, e_{io}$ to determine which sub-point cloud that the point $\mathbf{e}_i$ belongs to. Note that we cannot use the index of a point since the point cloud contains a *set* of points. To reach the third goal, we propose to use a hash function to assign a point $\mathbf{e}_i$ into a sub-point cloud. While PointCert is applicable with any hash function, we use a *cryptographic hash function* (e.g., MD5) in our experiments because it is designed to have uniformly random output. In particular, a cryptographic hash function takes any string as input and outputs a large integer that is roughly uniformly at random in the output space of the cryptographic hash function.

Combining the above three ideas, we first transform each value e_{ij} $(j = 1, 2, \cdots, o)$ into a string s_{ij}, then concatenate s_{ij}'s of a point $\mathbf{e}_i$ into S_i (i.e., $S_i = s_{i1} \oplus s_{i2} \oplus \cdots \oplus s_o$, where $\oplus$ represents string concatenation), and finally use a hash function (denoted as $Hash$) to compute the hash value of S_i (denoted as $Hash(S_i)$). We assign the point $\mathbf{e}_i$ to the r_ith sub-point cloud, where $r_i = Hash(S_i) \bmod m$, where mod is the modulo operation. For simplicity, we use $P_1, P_2, \cdots, P_m$ to denote the m sub-point clouds created from P. Note that some sub-point clouds may be empty, i.e., include no points.

Building an ensemble point cloud classifier: Given the m sub-point clouds $P_1, P_2, \cdots, P_m$ created from the point cloud P and a base point cloud classifier f, we build an ensemble point cloud classifier h. In particular, we first use f to predict a label for each non-empty sub-point cloud. Note that we do not consider those empty sub-point clouds. Then, we compute the number (denoted as $M_l(P)$) of non-empty sub-point clouds that are predicted to have label l by f. Formally, we define $M_l(P) = \sum_{i=1}^{m} \mathbb{I}(f(P_i) = l) \cdot \mathbb{I}(|P_i| > 0)$, where $l = 1, 2, \cdots, c$, $\mathbb{I}$ is an indicator function, and $|\cdot|$

measures the number of points in a sub-point cloud. For simplicity, we call $M_l(P)$ *label frequency* for label l. Our ensemble point cloud classifier h predicts the label whose label frequency is the largest for the point cloud P. Formally, we denote $h(P)$ as the label predicted for P by h and we have $h(P) = \text{argmax}_{l=1,2,\cdots,c} M_l(P)$.

We note that there may exist multiple labels with tied largest label frequencies. Usually, we break such ties uniformly at random, i.e., we predict a label among the tied ones uniformly at random. However, such random tie breaking introduces randomness and makes it hard to derive deterministic guarantees. To address the challenge, we break ties using the label indices deterministically. In particular, we order the c labels as $1, 2, \cdots, c$ and we predict the "smallest" label among the tied ones. For example, suppose labels 1 and 2 have tied largest label frequencies, i.e., $M_1(P) = M_2(P) > M_l(P)$, where $l \neq 1, 2$. Our h predicts label 1 for P. More formally, our h predicts label y for a point cloud P if $M_y(P) \geq \max_{l \neq y}(M_l(P) + \mathbb{I}(y > l))$.

4.2. Deriving Certified Perturbation Size

Derivation goal: Suppose our ensemble point cloud classifier predicts a label y for a point cloud P. P' is an adversarially perturbed version of P. Our goal is to derive the largest certified perturbation size $t(P)$ such that our ensemble point cloud classifier is guaranteed to predict label y for any P' with perturbation size at most $t(P)$. Formally, we aim to find the largest $t(P)$ such that we have $M_y(P') \geq \max_{l \neq y}(M_l(P') + \mathbb{I}(y > l))$ for any $P' \in \mathcal{S}(P, t(P))$. Our idea is to first derive a lower bound of $M_y(P')$ and an upper bound of $\max_{l \neq y}(M_l(P') + \mathbb{I}(y > l))$, and then find the largest $t(P)$ such that the lower bound is no smaller than the upper bound. Next, we first describe how we derive the lower/upper bounds and then how we find the largest certified perturbation size $t(P)$.

Deriving a lower bound of $M_y(P')$ and an upper bound of $\max_{l \neq y}(M_l(P') + \mathbb{I}(y > l))$: Recall that we divide a (adversarial) point cloud into m sub-point clouds. Since each point only appears in one sub-point cloud, an adversarially added or deleted point only impacts one sub-point cloud and may change the label predicted by the base point cloud classifier for the impacted sub-point cloud. Moreover, a modified point only impacts two sub-point clouds at most and thus impacts the predicted labels for two sub-point clouds at most. For simplicity, we define an *impact factor* τ for an operation (i.e., addition, deletion, modification) as the largest number of sub-point clouds that are impacted when the operation is applied to one point. The *impact factor* is 1 for addition/deletion and 2 for modification.

If an attacker can arbitrarily add (or delete or modify) at most t points to P, then at most $\tau \cdot t$ sub-point clouds in $P_1, P_2, \cdots, P_m$ are impacted. Therefore, we have $M_y(P') \geq M_y(P) - \tau \cdot t$ and $M_l(P') \leq M_l(P) + \tau \cdot t$ for $\forall l \neq y$ and $\forall P' \in \mathcal{S}(P, t)$. We treat $M_y(P) - \tau \cdot t(P)$ as a lower bound of $M_y(P')$ and $\max_{l \neq y}(M_l(P) + \tau \cdot t + \mathbb{I}(y > l))$ as an upper bound of $\max_{l \neq y}(M_l(P') + \mathbb{I}(y > l))$.

Computing certified perturbation size: Our goal is to find the largest t such that the lower bound of $M_y(P')$ is no smaller than the upper bound of $\max_{l \neq y}(M_l(P') + \mathbb{I}(y > l))$. In other words, we aim to find the largest t such that $M_y(P) - \tau \cdot t \geq \max_{l \neq y}(M_l(P) + \tau \cdot t + \mathbb{I}(y > l))$. Therefore, we have $t \leq \frac{M_y(P) - \max_{l \neq y}(M_l(P) + \mathbb{I}(y > l))}{2 \cdot \tau}$. Since the number of points that an attacker can add (or delete or modify) should be an integer, we have the certified perturbation size as $t(P) = \lfloor \frac{M_y(P) - \max_{l \neq y}(M_l(P) + \mathbb{I}(y > l))}{2 \cdot \tau} \rfloor$, where $\lfloor \cdot \rfloor$ is the floor function. In summary, we have the following:

Theorem 4.1 (Certified Perturbation Size)**.** *Suppose we have a point cloud P, a hash function to divide P into m disjoint sub-point clouds, a base point cloud classifier f, and label frequency $M_l(P)$, where $l = 1, 2, \cdots, c$. Our ensemble point cloud classifier h predicts the same label for P and its adversarially perturbed version P' once the perturbation size is at most $t(P)$. Formally, we have:*

$$h(P') = h(P) = y, \forall P' \in \mathcal{S}(P, t(P)), \tag{1}$$

where $t(P) = \lfloor \frac{M_y(P) - \max_{l \neq y}(M_l(P) + \mathbb{I}(y > l))}{2 \cdot \tau} \rfloor$. The impact factor τ is 1 for point addition and deletion attacks, while it is 2 for point modification and perturbation attacks since a point perturbation attack can use any combination of the three operations.

We also prove that our derived certified perturbation size is tight, i.e., without making any assumptions on the base point cloud classifier, it is theoretically impossible to derive a certified perturbation size that is larger than ours. Formally, we have the following theorem:

Theorem 4.2 (Tightness)**.** *Given a point cloud P and a hash function to divide P into m disjoint sub-point clouds, there exists an adversarial point cloud $P' \in \mathcal{S}(P, t(P) + 1)$ and a base point cloud classifier f' such that our ensemble classifier predicts different labels for P and P'. Formally, we have $h'(P') \neq h'(P)$, where h' is the ensemble point cloud classifier built based on f'.*

Proof. See Appendix A. □

5. Applications in Three Scenarios

Scenario I: This scenario is a naive application of PointCert. Suppose a model provider has trained a base point cloud classifier f using the standard training algorithm, and shares it with customers in a black-box or white-box setting. In the black-box setting, the model provider only provides a prediction API for a customer, who can send a point cloud to the model provider and obtain its prediction made by f. In the white-box setting, the model provider

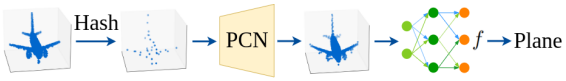


Figure 1. Composition of PCN and f in Scenario III.

shares the model parameters with a customer, who can use f to classify point clouds locally.

Given a black-box or white-box access to f, a customer directly uses PointCert to classify its point clouds. Specifically, given a point cloud, the customer first divides it into sub-point clouds, then uses f to predict a label for each non-empty sub-point cloud, and finally takes a majority vote among the predicted labels of the sub-point clouds as the predicted label for the point cloud.

Scenario II: In Scenario I, f is trained on point clouds, and thus may be inaccurate to classify sub-point clouds as they have different distributions with point clouds. As a result, PointCert is less accurate. In Scenario II, we consider that the model provider trains its f to optimize the performance of PointCert. In particular, to make f more accurate in classifying sub-point clouds, we propose that the model provider trains f on sub-point clouds. In particular, the model provider divides each training point cloud into m disjoint sub-point clouds following PointCert and uses the label of the training point cloud as the label of each sub-point cloud. Then, the model provider trains f on those labeled sub-point clouds. Similar to Scenario I, the model provider can share f with a customer in a black-box or white-box setting, and a customer can directly use PointCert to classify point clouds based on f.

Scenario III: Similar to Scenario I, we consider the model provider has trained f using a standard training algorithm. However, instead of directly using f to classify sub-point clouds, a customer adds a *Point Completion Network (PCN)* [48] before f to improve its accuracy for sub-point clouds. In particular, the PCN takes a sub-point cloud as input and outputs a completed point cloud, which is then classified by f, as shown in Figure 1.

Formulating PCN learning as an optimization problem. Suppose a customer has a set of unlabeled point clouds and (optionally) a small amount of labeled ones. The customer constructs a training dataset $\mathcal{D}_u$. Specifically, the customer divides each unlabeled point cloud into m disjoint sub-point clouds following PointCert. $\mathcal{D}_u$ consists of a set of pairs (P_s, P_p), where P_s is a sub-point cloud and P_p is the corresponding point cloud. A PCN takes P_s as input and aims to output P_p. In existing point completion methods [14, 24, 37, 39, 45, 47, 48], learning a PCN $\mathcal{C}$ essentially formulates a loss term $L_p(\mathcal{D}_u, \mathcal{C})$ over the training dataset $\mathcal{D}_u$ and then uses stochastic gradient descent (SGD) to minimize the loss. We adopt the popular Chamfer Distance proposed in Fan et al. [8] as the loss term in our experiments (the details can be found in Appendix).

However, f is still likely to misclassify the point clouds completed by such a PCN. The reason is that existing point completion methods did not aim to complete point clouds that can be classified by f with high accuracy, since it is not their goal. To bridge this gap, we propose another loss term, which is smaller if the completed point clouds can be classified by f with higher accuracy. Formally, we define the following loss term: $L_c(\mathcal{D}_l, \mathcal{C}, f) = \frac{1}{|\mathcal{D}_l|}\sum_{(P,y)\in\mathcal{D}_l}\mathcal{L}(f(\mathcal{C}(P)), y)$, where $\mathcal{D}_l$ is the set of labeled point clouds and $\mathcal{L}$ is the loss function for classification such as cross-entropy loss. Combining the two loss terms, our final loss used to train a PCN is as follows:

$$L_p(\mathcal{D}_u, \mathcal{C}) + \lambda \cdot L_c(\mathcal{D}_l, \mathcal{C}, f), \tag{2}$$

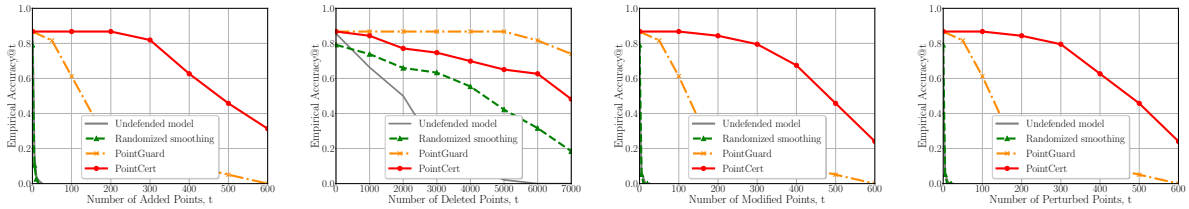
where λ is a hyperparameter used to balance the two loss terms. We note that Scenario III is not applicable to the customer without any unlabeled or labeled point clouds. Moreover, when a customer only has unlabeled point clouds, the customer can only use a standard point completion method to learn a PCN, i.e., $\lambda = 0$ in Equation 2.

Solving the optimization problem in white-box and black-box settings. In the white-box setting, a customer has access to the model parameters of f. Therefore, the customer can solve the optimization problem in Equation (2) using the standard SGD to learn a PCN. In the black-box setting, the customer only has access to the prediction API of f, and thus cannot solve the optimization problem using SGD. The customer could use zeroth-order optimization methods [10] to solve the optimization problem. However, such method often incurs a large number of queries to the prediction API. To address the challenge, we propose that the customer learns a student model using *knowledge distillation* [13] by viewing f as a teacher model. Roughly speaking, the customer can first query f using his/her unlabeled and labeled point clouds (excluding labels) to obtain their output logits predicted by f, then divide the logits by T which is a temperature parameter in knowledge distillation, and finally train a student model. Given the student model, the customer can use it to replace f in Equation (2) and train a PCN using SGD. We note that the customer essentially treats the composition of the PCN and the student model (or the teacher model) as a new base point cloud classifier in our PointCert framework.

6. Experiments

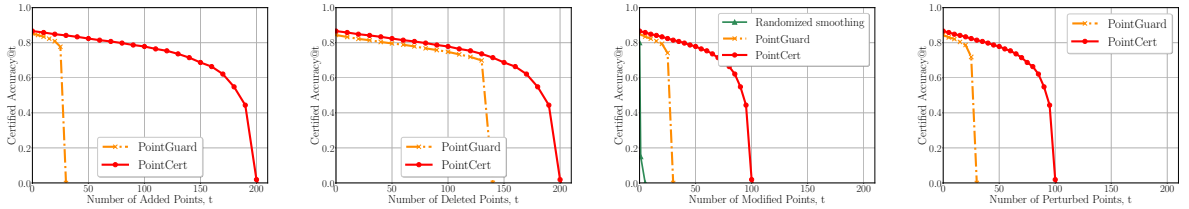
6.1. Experimental Setup

Datasets and models: We adopt two publicly available benchmark datasets, namely ModelNet40 [42] and two variants of ScanObjectNN [36], in our evaluation. Each point cloud of ModelNet40 has 10,000 points and we also keep at most 10,000 points in each point cloud of ScanObjectNN. We do not reduce the size of a point cloud by sub-sampling



(a) Point addition attacks (b) Point deletion attacks (c) Point modification attacks (d) Point perturbation attacks

Figure 2. Comparing the empirical accuracy of different defenses. Scenario II is considered.



(a) Point addition attacks (b) Point deletion attacks (c) Point modification attacks (d) Point perturbation attacks

Figure 3. Comparing the certified accuracy of randomized smoothing, PointGuard, and PointCert. Randomized smoothing can only provide certified robustness guarantees against point modification attacks. Scenario II is considered.

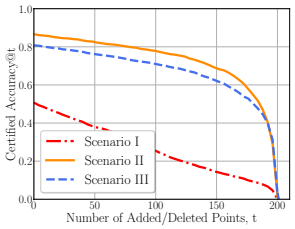
its points to simulate real-world attack scenarios. Our method and compared baselines are evaluated using same number of points. The detailed dataset description is shown in Appendix C. We evenly split the training point clouds in each dataset into two **balanced halves**. One half is used for the **model provider** to train base point cloud classifier in the three scenarios, and the other is used for a **customer** to train a PCN in Scenario III. We consider PointNet [29] and DGCNN [38], which are frequently used by the community, as the base point cloud classifiers.

Compared methods: We compare PointCert with undefended model, randomized smoothing [5], and PointGuard [23]. Both randomized smoothing and PointGuard only have probabilistic robustness guarantees. Details of these methods can be found in Appendix D.

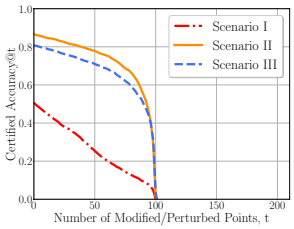
Evaluation metrics: We use *Empirical Accuracy*@t and *Certified Accuracy*@t as evaluation metrics. In particular, Certified Accuracy@t is the fraction of testing point clouds in a testing dataset whose certified perturbation sizes are at least t and whose labels are correctly predicted. The Empirical Accuracy@t is the testing accuracy of each model under the empirical attacks with perturbation size t. We note that the Certified Accuracy@t is a *lower bound* of testing accuracy that a defense can achieve when the perturbation size is at most t, no matter how the perturbation is crafted. Empirical Accuracy@t is an *upper bound* of testing accuracy that each model can achieve under attacks with perturbation size at most t. For undefended model, we only report Empirical Accuracy@t because it does not have certified robustness guarantees. Besides, randomized smoothing can only provide certified robustness guarantees for point modification attacks, though we can report its Empirical Accuracy@t against other attacks.

In experiments, we use the attacks developed by [43] for point addition, modification, and perturbation attacks and [40] for point deletion attack. We note that there are no existing adversarial point cloud attacks tailored to randomized smoothing, PointGuard, and PointCert. To bridge this gap, we generalize existing attacks to these ensemble models and compute their Empirical Accuracy@t. The key idea of our attacks to ensemble models is to identify a set of critical points to add (or delete) such that the classification losses of the base point cloud classifier on different groups of point clouds (e.g., sub-point clouds in PointCert) are maximized. Details of our empirical attacks can be found in Appendix E.

Parameter setting: Our PointCert has a parameter m and a hash function to divide a point cloud into m sub-point clouds. By default, we set $m = 400$ and use MD5 as the hash function. Despite the large m, the inference time per testing point cloud of PointCert is less than 0.51s because the sub-point cloud sizes are small. Moreover, we set PointNet as the default base point cloud classifier. In Scenario III, we set the default value of λ to be 5×10^{-4} to balance the two loss terms. By default, we assume 25% of the customer's point clouds are labeled while the remaining is unlabeled in Scenario III. For point cloud completion, we use coarse output of PCN [48] since we do not require the fine-grained output. In Scenario I and II, white-box and black-box settings have no difference. In Scenario III, we assume the white-box setting by default; and in the black-box setting, we learn a student model using knowledge distillation with a temperature $T = 20$. Due to space constraint, we show the results on ScanObjectNN in Appendix.

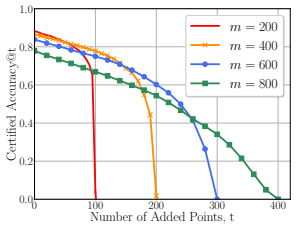


(a) Point addition/deletion attacks

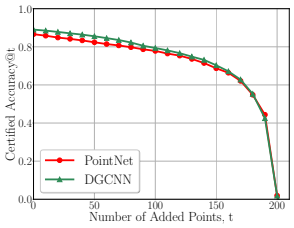


(b) Point modification/perturbation attacks

Figure 4. Comparing the certified accuracy of PointCert in the three application scenarios under different attacks.



(a)



(b)

Figure 5. (a) Impact of m on PointCert. (b) Comparing different base point cloud classifiers. Scenario II is considered.

6.2. Experimental Results

6.2.1 Comparing Different Defenses

Figure 2 compares the empirical accuracy of all methods under empirical attacks, while Figure 3 compares the certified accuracy of randomized smoothing, PointGuard, and PointCert in Scenario II. We note that these defenses have accuracy-robustness tradeoffs, which are controlled by their parameters, e.g., m in PointCert. Therefore, to fairly compare PointCert with randomized smoothing and PointGuard, we make them have similar certified accuracy under no attacks (i.e., $t = 0$) by following previous work [23]. In particular, we use the default m for PointCert and respectively search σ and k for randomized smoothing and PointGuard. Our searched parameters are $\sigma = 0.25$ and $k = 256$. We use the settings in [23] for other parameters of randomized smoothing and PointGuard.

We have the following observations from the experimental results. First, an undefended model is not robust against adversarial point clouds. For instance, adding or modifying only 1 out of 10,000 points can substantially reduce its empirical accuracy. Second, PointCert achieves larger empirical accuracy and certified accuracy than randomized smoothing and PointGuard even though their certified robustness guarantees are probabilistic. The reasons are that 1) randomized smoothing adds Gaussian noise to every point in a point cloud, making its classification less accurate, and 2) an adversarially added, deleted, and/or modified point impacts multiple subsampled point clouds in PointGuard. In contrast, PointCert does not add noise to points in a point cloud and each perturbed point impacts at most 1 or 2 sub-point clouds.

An interesting exception is that PointGuard achieves better empirical accuracy than PointCert under our empirical point deletion attacks. The reason is that each sub-point cloud in PointCert contains much less number of points after thousands of points are deleted, and thus the base point cloud classifier in PointCert is less accurate. In contrast, each subsampled point cloud in PointGuard still contains k points even if thousands of points are deleted. Third, every method achieves much higher empirical accuracy against point deletion attacks than against other attacks when the perturbation size is the same, which indicates that state-of-the-art point deletion attack is not powerful enough.

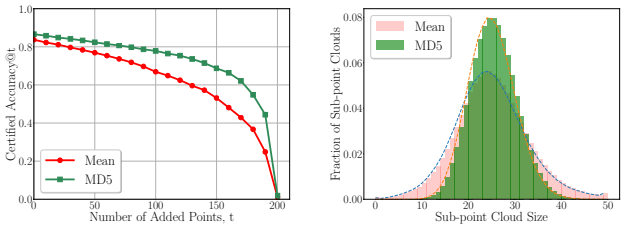
6.2.2 Comparing the Three Scenarios

Figure 4 compares the three application scenarios of PointCert under attacks. In each scenario, the certified accuracy of PointCert is the same for point addition and deletion attacks, and is the same for point modification and perturbation attacks. Thus, both Figures 4a and 4b showcase the certified accuracy of PointCert under two attacks.

First, PointCert achieves the best certified accuracy in Scenario II as the base point cloud classifier in Scenario II is trained on sub-point clouds and is more accurate in classifying them. Second, PointCert achieves better certified accuracy in Scenario III than in Scenario I. The reason is that, in Scenario III, a customer trains a PCN to turn a sub-point cloud into a completed point cloud, which can be well classified by the base point cloud classifier trained using a standard algorithm. Third, in each scenario, given the same certified accuracy, the perturbation size that PointCert can tolerate under point addition/deletion attacks is twice of that under point modification/perturbation attacks. The reason is that modifying a point is equivalent to adding a point and deleting a point, which could impact two sub-point clouds in the worst case. Due to such relationship, we compare results on point addition attacks in the following section.

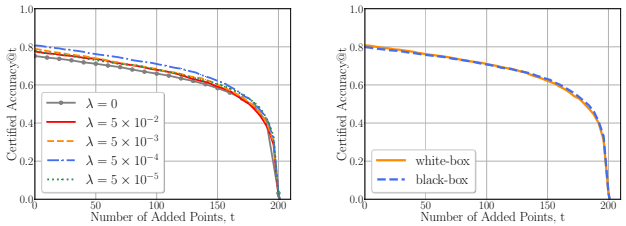
6.2.3 Scenario II

Impact of m: Figure 5a shows the impact of m on certified accuracy of PointCert. As the results show, m achieves a tradeoff between accuracy without attacks (i.e., $t = 0$) and robustness. In particular, when m is smaller, PointCert can achieve a higher accuracy without attacks, but is less robust (i.e., certified accuracy drops to 0 more quickly). The reason is that a smaller m means each sub-point cloud includes more points and thus is more likely to be classified correctly, but each adversarially perturbed point impacts a larger fraction of the m sub-point clouds.



(a) (b)

Figure 6. (a) Comparing the certified accuracy of PointCert with two hash functions in Scenario II. (b) The distribution of the sub-point cloud sizes for the two hash functions.



(a) (b)

Figure 7. (a) Impact of λ on PointCert in Scenario III. (b) Comparing certified accuracy of PointCert in the white-box and black-box settings in Scenario III.

Impact of different base point cloud classifiers: Figure 5b compares the certified accuracy of PointCert for different base point cloud classifiers. The experimental results demonstrate that PointCert achieves nearly similar certified accuracy with different base point cloud classifiers.

Impact of hash function: PointCert uses a hash function to divide a point cloud into sub-point clouds. We compare the cryptographic hash function MD5 with a *mean* based one. In the mean based hash function, given a point $\mathbf{e}_i$, we first compute the mean value of the coordinates of the point (i.e., $\frac{1}{o}\sum_{j=1,2,\cdots,o} e_{ij}$), then take the first four digits (denoted as d_1, d_2, d_3, d_4) of the mean, and finally assign the point $\mathbf{e}_i$ to the r_ith sub-point cloud, where $r_i = \sum_{j=1}^{4} d_j \cdot 10^{4-j} \mod m$. Figure 6a compares the certified accuracy of PointCert with the two hash functions. Our result indicates that PointCert achieves higher certified accuracy when using MD5. This is because MD5 generates sub-point clouds with more similar sizes. In particular, Figure 6b shows the distribution of the number of points in sub-point clouds for the two hash functions. We observe that the sub-point cloud sizes in MD5 are more concentrated than those in mean. The reason is that cryptographic hash function aims to produce uniformly random hash values in its output space.

6.2.4 Scenario III

Impact of λ: Figure 7a shows the impact of λ. The certified accuracy first increases and then decreases as λ increases. The reasons are as follows. When λ is too small, the base point cloud classifier is less accurate in classifying the point clouds completed by the PCN. When λ is too large, the PCN completes point clouds with low fidelity, as shown in Figure 16 in Appendix. The fact that $\lambda > 0$ outperforms $\lambda = 0$ indicates that our new loss term $L_c(\mathcal{D}_l, \mathcal{C}, f)$ for training PCN improves PointCert.

White-box vs. black-box: In Scenario III, a customer uses different methods to train a PCN in the white-box and black-box settings. Figure 7b compares the certified accuracy of PointCert in the two settings. The results show that PointCert can achieve similar certified accuracy in both settings, which means that the distilled student model approximates the teacher model well. We also found that when the student model and teacher model have different architectures in the black-box setting, our PointCert still achieves high certified accuracy. Due to limited space, we show the results in Figure 8a in Appendix.

Pre-trained PCN improves certified accuracy: In our previous experiments, we assume a customer trains a PCN from scratch. However, when a customer has a small amount of point clouds, it may be hard to train a good PCN from scratch. To address the issue, the customer could fine-tune a pre-trained PCN instead of training from scratch. We pretrain a PCN using 8-class of ShapeNet [4] (used in [24]) and adopt it in our experiment. Figure 8b in Appendix shows our results, which indicates that pre-trained PCN can improve the certified accuracy of PointCert.

Impact of label ratio: Figure 8c in Appendix shows the impact of the fraction of a customer's point clouds that are labeled on the certified accuracy of PointCert. We observe PointCert achieves higher certified accuracy when a customer has more labeled point clouds. This is because with more labeled point clouds, the learnt PCN outputs completed point clouds that are classified by the base point cloud classifier with a higher accuracy.

7. Conclusion

In this paper, we propose the first certified defense, namely PointCert, that has deterministic robustness guarantees against point addition (or deletion or modification or perturbation) attacks to point cloud classification. Moreover, we propose methods to optimize the performance of PointCert in multiple application scenarios. Interesting future work includes: 1) exploring/designing hash functions to further improve the robustness guarantees of PointCert, and 2) generalizing PointCert to other domains, e.g., graph.

Acknowledgements: We thank the anonymous reviewers for their constructive comments. This work was supported by NSF under grant No. 2112562, 2112562, 1937786, and 1937787, ARO grant No. W911NF2110182, and Facebook Research Award.

References

[1] ModelNet40. https://modelnet.cs.princeton.edu/, 2015. Accessed: 2022-06. 2

[2] ScanObjectNN. https://github.com/hkust-vgd/scanobjectnn, 2019. Accessed: 2022-06. 2

[3] Xiaoyu Cao and Neil Zhenqiang Gong. Mitigating evasion attacks to deep neural networks via region-based classification. In *Annual Computer Security Applications Conference*, 2017. 2

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 8

[5] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, 2019. 1, 2, 6, 11, 12

[6] Dishanika Dewani Denipitiyage, Thalaiyasingam Ajanthan, Parameswaran Kamalaruban, and Adrian Weller. Provable defense against clustering attacks on 3d point clouds. In *The AAAI-22 Workshop on Adversarial Machine Learning and Beyond*, 2021. 1

[7] Xiaoyi Dong, Dongdong Chen, Hang Zhou, Gang Hua, Weiming Zhang, and Nenghai Yu. Self-robust 3d point recognition via gather-vector guidance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2

[8] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017. 5, 11

[9] Marc Fischer, Maximilian Baader, and Martin Vechev. Scalable certified segmentation via randomized smoothing. In *International Conference on Machine Learning*, 2021. 2

[10] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 2013. 5

[11] Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. Mvtn: Multi-view transformation network for 3d shape recognition. In *IEEE/CVF International Conference on Computer Vision*, 2021. 1

[12] Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *European Conference on Computer Vision*, 2020. 1, 2

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 5

[14] Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. Pf-net: Point fractal network for 3d point cloud completion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 5

[15] Jinyuan Jia, Binghui Wang, Xiaoyu Cao, Hongbin Liu, and Neil Zhenqiang Gong. Almost tight l0-norm certified robustness of top-k predictions against adversarial perturbations. In *International Conference on Learning Representations*, 2022. 13

[16] Jaeyeon Kim, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Minimal adversarial examples for deep learning on 3d point clouds. In *IEEE/CVF International Conference on Computer Vision*, 2021. 1, 2

[17] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy*, 2019. 2

[18] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Second-order adversarial attack and certifiable robustness. 2018. 2

[19] Kaidong Li, Ziming Zhang, Cuncong Zhong, and Guanghui Wang. Robust structured declarative classifiers for 3d point clouds: Defending adversarial attacks with implicit gradients. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2

[20] Daizong Liu and Wei Hu. Imperceptible transfer attack and defense on 3d point cloud classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 2

[21] Daniel Liu, Ronald Yu, and Hao Su. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. In *IEEE International Conference on Image Processing*, 2019. 1, 2

[22] Daniel Liu, Ronald Yu, and Hao Su. Adversarial shape perturbations on 3d point clouds. In *European Conference on Computer Vision*, 2020. 1, 2

[23] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 6, 7, 11, 12

[24] Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and sampling network for dense point cloud completion. In *AAAI Conference on Artificial Intelligence*, 2020. 5, 8

[25] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *European Conference on Computer Vision*, 2018. 2

[26] Tobias Lorenz, Anian Ruoss, Mislav Balunović, Gagandeep Singh, and Martin Vechev. Robustness certification for point cloud models. In *IEEE/CVF International Conference on Computer Vision*, 2021. 2

[27] Chengcheng Ma, Weiliang Meng, Baoyuan Wu, Shibiao Xu, and Xiaopeng Zhang. Efficient joint gradient based attack against sor defense for 3d point cloud classification. In *ACM International Conference on Multimedia*, 2020. 2

[28] Juan C Pérez, Motasem Alfarra, Silvio Giancola, Bernard Ghanem, et al. 3deformrs: Certifying spatial deformations on point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2

[29] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017. 1, 6

[30] Shi Qiu, Saeed Anwar, and Nick Barnes. Geometric back-projection network for point cloud classification. *IEEE Transactions on Multimedia*, 2021. 1

[31] Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. *Advances in Neural Information Processing Systems*, 2019. 2

[32] Wen Shen, Qihan Ren, Dongrui Liu, and Quanshi Zhang. Interpreting representation quality of dnns for 3d point cloud processing. *Advances in Neural Information Processing Systems*, 2021. 1

[33] Jiachen Sun, Yulong Cao, Christopher B Choy, Zhiding Yu, Anima Anandkumar, Zhuoqing Morley Mao, and Chaowei Xiao. Adversarially robust 3d point cloud recognition using self-supervisions. *Advances in Neural Information Processing Systems*, 2021. 1

[34] Jiachen Sun, Karl Koenig, Yulong Cao, Qi Alfred Chen, and Z Morley Mao. On adversarial robustness of 3d point cloud classification under adaptive attacks. *arXiv preprint arXiv:2011.11922*, 2020. 1, 2

[35] Jiachen Sun, Qingzhao Zhang, Bhavya Kailkhura, Zhiding Yu, Chaowei Xiao, and Z Morley Mao. Benchmarking robustness of 3d point cloud recognition against common corruptions. *arXiv preprint arXiv:2201.12296*, 2022. 2

[36] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *IEEE/CVF International Conference on Computer Vision*, 2019. 5, 11

[37] Xiaogang Wang, Marcelo H Ang Jr, and Gim Hee Lee. Cascaded refinement network for point cloud completion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 5

[38] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 2019. 1, 6

[39] Xin Wen, Tianyang Li, Zhizhong Han, and Yu-Shen Liu. Point cloud completion by skip-attention network with hierarchical folding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 5

[40] Matthew Wicker and Marta Kwiatkowska. Robustness of 3d deep learning in an adversarial setting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2, 6, 12

[41] 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*, 2020. 1, 2

[42] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2015. 5, 11

[43] Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2, 6, 12

[44] Tiange Xiang, Chaoyi Zhang, Yang Song, Jianhui Yu, and Weidong Cai. Walk in the cloud: Learning curves for point clouds shape analysis. *arXiv preprint arXiv:2105.01288*, 2021. 1

[45] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In *European Conference on Computer Vision*, 2020. 5

[46] Jiancheng Yang, Qiang Zhang, Rongyao Fang, Bingbing Ni, Jinxian Liu, and Qi Tian. Adversarial attack and defense on point sets. *arXiv preprint arXiv:1902.10899*, 2019. 1, 2

[47] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *IEEE/CVF International Conference on Computer Vision*, 2021. 5

[48] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *International Conference on 3D Vision (3DV)*, 2018. 2, 5, 6

[49] Jinlai Zhang, Lyujie Chen, Bo Ouyang, Binbin Liu, Jihong Zhu, Yujin Chen, Yanmei Meng, and Danfeng Wu. Pointcutmix: Regularization strategy for point cloud classification. *Neurocomputing*, 2022. 1, 2

[50] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *IEEE/CVF International Conference on Computer Vision*, 2021. 1

[51] Yue Zhao, Yuwei Wu, Caihua Chen, and Andrew Lim. On isometry robustness of deep 3d point cloud models under adversarial attacks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2

[52] Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. Pointcloud saliency maps. In *IEEE/CVF International Conference on Computer Vision*, 2019. 1, 2

[53] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. In *IEEE/CVF International Conference on Computer Vision*, 2019. 1, 2