Towards Robust Point Cloud Models with Context-Consistency Network and Adaptive Augmentation

Anonymous authors
Paper under double-blind review

Abstract

3D point cloud models based on deep neural networks were proven to be vulnerable to adversarial examples, with a quantity of novel attack techniques proposed by researchers recently. It is of paramount importance to preserve the robustness of 3D models under adversarial environments, considering their broad application in safety- and security-critical tasks. Unfortunately, defenses for 3D models are much less studied compared to 2D image models. In this paper, we design an effective defense methodology, consisting of two innovations. (1) We introduce `CCDGN`, a novel 3D *DNN architecture* which includes robust and light-weight modules to alleviate adversarial examples. (2) We propose `AA-AMS`, a novel *data augmentation* strategy to adaptively balance the model usability and robustness. Extensive evaluations indicate the integration of the two techniques provides much more robustness than existing defense solutions for 3D models.

1 Introduction

A point cloud is a popular representation of 3D objects and shapes. It consists of a set of data points with x, y and z coordinates to describe the external surface of an object. Interpreting point cloud data becomes important in many scenarios, *e.g.*, robotics (Kim et al., 2018), manufacturing (Arsalan Soltani et al., 2017), construction (Macher et al., 2017), *etc*. Recently, researchers designed new models based on Deep Neural Networks (DNN) (*e.g.*, PointNet (Qi et al., 2017a), DGCNN (Wang et al., 2019)) for 3D object classification, segmentation and detection, which achieve remarkable breakthrough over traditional methods.

Unfortunately, DNNs are well known to be vulnerable against Adversarial Examples (AEs) (Szegedy et al., 2014; Goodfellow et al., 2015): where imperceptible perturbations on a normal sample can mislead the model to make wrong predictions. Over the years, a plethora of attacks were designed to efficiently generate AEs (Goodfellow et al., 2015; Carlini & Wagner, 2017b; Madry et al., 2018). New techniques were further proposed to attack point cloud models (Liu et al., 2019; Zheng et al., 2019; Xiang et al., 2019).

Past works have extensively explored methods of defending 2D models against AEs. In contrast, how to enhance the robustness of 3D models is relatively less studied. The unique features of point cloud data and models increase the difficulty of model protection: (1) point clouds usually have irregular formats determined by the sensors for data collection; (2) adversaries have more choices to perform the attacks (*e.g.*, adding or removing points) in addition to changing the coordinate values; (3) 3D point clouds have a larger perturbation space than the 2D image space, resulting in more qualified AEs. These features make existing solutions less effective. For instance, some works performed adversarial training for 3D models (Liu et al., 2019), which fail to defeat all types of attacks, especially the unseen ones. Some works designed more robust 3D network structures (Zhou et al., 2019; Wu et al., 2020). They can be easily bypassed by adaptive attacks (Ma et al., 2020). Hence, it is urgent but challenging to have a general and comprehensive defense mechanism.

In this paper, we propose a new solution to effectively defend point cloud models against AEs with two contributions. First, from the perspective of network architectures, we design a light-weight Context-Consistency Module (CCM), which can smooth the perturbations of adversarial point clouds, and reduce their distance with clean samples in the feature space. Equipped with this module, we design Context-Consistency Dynamic Graph Network (CCDGN), a new 3D network structure with higher robustness against various types of AEs. We leverage the mutual information theory to explain the effectiveness of our new network solution.

Our second contribution is the introduction of a new data augmentation strategy, named Adaptive Augmentation with Adversarial and Mix-up Samples (AA-AMS). Researchers have proposed to train 3D point cloud models with adversarial examples (Liu et al., 2019) or mix-up sampling (Chen et al., 2020; Zhang et al., 2021). However, these methods cannot achieve comprehensive protection due to the variety of techniques in crafting AEs. Hence, we propose to augment the training set with different types of adversarial examples and mix-up samples. Simply incorporating all these data samples could easily affect the model accuracy over clean samples or overfit some specific attack. To balance the trade-off between model usability and robustness, we dynamically monitor the model's behaviors during training, and adaptively select the samples that can best improve the model performance.

We perform comprehensive evaluations over the ModelNet40 against four state-of-the-art white-box attacks and one black-box attack for point cloud models. Experimental results show that our CCDGN has better robustness with different training strategies than other baseline models. The integration of CCDGN and AA-AMS outperforms existing solutions with about 8% on average adversarial accuracy.

2 Background and Related Works

2.1 Point Cloud Models

A point cloud is formally defined as a set $x = \{x_i\}_{i=1}^{N}$, where $x_i \in \mathbb{R}^3$ is a 3D point with (*x, y, z*) coordinates, and N is the number of points. A point cloud model is thus a parameterized function $f_\theta : \mathcal{X} \mapsto \mathcal{Y}$ that predicts the corresponding label from a point cloud. Researchers have proposed different deep learning algorithms and neural networks to realize this classification tasks. We describe three common models. (1) PointNet (Qi et al., 2017a): this network consists of single variable-functions, a max pooling layer, and a function of the max pooled features to handle unordered points with arbitrary dimensions. It converts the point cloud data to feature vectors with fixed length, and then learns the labels. (2) PointNet++ (Qi et al., 2017b): this is a hierarchical neural network, which recursively applies PointNet over partitioned point sets to learn the local structures. Both of PointNet and PointNet++ adopt the coordinates of the points to produce the features. (3) DGCNN (Wang et al., 2019): this Dynamic Graph Convolutional Neural Network integrates a new module EdgeConv to point cloud models. This module captures the local geometric structures by constructing a local graph and learning the embeddings for the edges. Then the integrated model can learn to semantically group the points for more accurate classification and segmentation. Different from PointNet and PointNet++, DGCNN considers the neighbors of the points and adopts high-order features, *i.e.*, distances between adjacent points, to predict the labels. Hence, it gives higher robustness than the other two models. We also validate this conclusion in Section 5.3.

2.2 Adversarial Attacks against Point Clouds

The concept of adversarial examples was first proposed in Szegedy et al. (2014), where the adversary tries to identify the imperceptible perturbation with the minimal scale to mislead the 2D image model. Then this attack was extended to the 3D point clouds with more techniques. Generally, these attacks can be classified into three categories, as described below.

Point perturbing. Similar to 2D image adversarial attacks, the adversary can slightly perturb the coordinates of certain points to fool the 3D model. Conventional approaches in 2D image tasks can be applied to 3D point clouds as well. For instance, Xiang et al. (2019) adopted the C&W technique (Carlini & Wagner, 2017a) to identify the optimal shift scale. Liu et al. (2019) adopted the FGSM method (Szegedy et al., 2014) with various perturbation constraints to craft adversarial point clouds.

Point adding. The adversary can inject a small set of new points into the clean point cloud to attack the model. Xiang et al. (2019) designed an initialize-and-shift approach to calculate the added points with their positions. Zhang et al. (2019) proposed a point-wise gradient method to generate the optimal locations for point attachment.

Point dropping. The adversary can also remove some critical points from the original set to alter the classification result. Zheng et al. (2019) constructed the saliency map to identify the critical points and then drop them for attacks. A similar idea was also proposed in Zhang et al. (2019).

2.3 Adversarial Defenses for Point Clouds

A couple of approaches were proposed to defeat adversarial attacks against point clouds. Zhou et al. (2019) designed a new structure DUP-Net, with the SOR operation to drop outliers in the input samples. However, it is only effective for point perturbing attacks, but fails to thwart point adding or dropping attacks. Furthermore, Ma et al. (2020) designed an adaptive attack to completely break this defense. Liu et al. (2019) explored how to train a 3D point cloud model with adversarial examples generated by PGD. They concluded this strategy can beat SOR and salient point removal approaches under certain attacks. Unfortunately, simple adversarial training based on PGD is not robust to cover all types of attacks, which will be demonstrated in our evaluation. Mix-up is a popular technique to augment training data with linear interpolations of feature vectors and labels to defeat 2D adversarial images (Zhang et al., 2018). This idea was then extended to the point cloud scenario, based on which researchers designed PointMixUP (Chen et al., 2020), and PointCutMix (Zhang et al., 2021). Our adaptive augmentation can outperform these purely mix-up strategies from the evaluation.

3 A Robust Cloud Point Model Architecture

We introduce a novel point cloud model architecture as our first defense methodology. We also present some theoretical analysis about the robustness of our architecture.

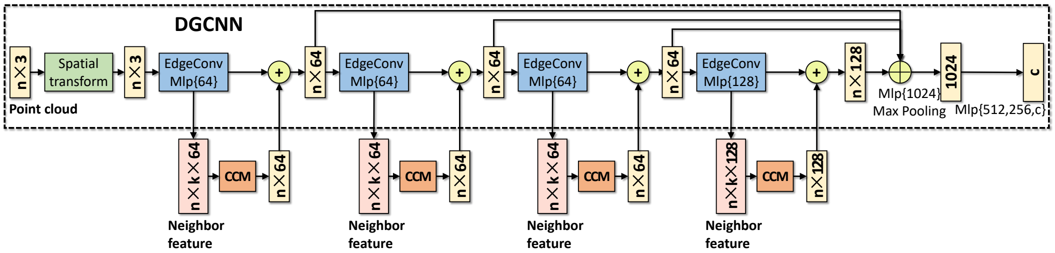


Figure 1: Overview of CCDGN.

3.1 Context-Consistency Dynamic Graph Network

We design Context-Consistency Dynamic Graph Network (CCDGN), a new 3D model structure for robustness enhancement. Figure 1 shows the structure overview. It is mainly built from the DGCNN model with the same spatial transform and EdgeConv layers. We choose DGCNN because it exhibits higher robustness than PointNet and PointNet++, due to the adoption of high-order features.

The key innovation of CCDGN is a Context Consistency Module (CCM). Its goal is to remove the noise from the adversarial samples, which can make features of clean and adversarial samples closer. Specifically, for each EdgeConv layer, we extract the neighbor feature of every point, and feed it to the CCM. In DGCNN, the elements in neighbor feature are coordinates of each point's neighbors. The output of the CCM will be combined with the output of the EdgeConv layer.

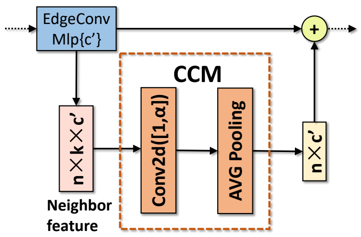
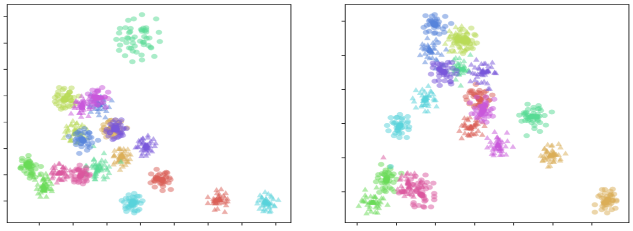


Figure 2: The structure of a CCM.

Figure 2 shows the detailed structure of a CCM in practice. First, it uses a 2D convolutional layer with a receptive field size of $[1, \alpha]$ to process the context information (*i.e.*, coordinates) in the neighbors of each point from the EdgeConv layer. This convolutional layer calculates new coordinates for the neighbors in the scope of α, which automatically learns to smooth the noise in the neighbor features. The sliding window in the convolutional layer can handle all the continuous scopes in the neighbor feature. In this way, the feature distance between adversarial and clean samples can be minimized. Second, an average pooling layer is followed by the convolutional layer to reduce the dimension. This function chooses the proper elements in the features based on their values. It can prevent noise accumulation during the model's forward propagation process by averaging elements in the features. Finally, a residual connection transfers the adaptively selected context information to the output of the EdgeConv. With such operation, the CCM can keep the features between different layers (*i.e.*, contexts extracted from inputs) consistent. As a result, it can prevent the adversarial noise in the features from growing quickly at deeper layers in the model.

Particularly, the size of the receptive field α in the CCM can impact the model robustness when the order of inputs changes. This is because in a point cloud, the correlation between neighbor points is less tight than the correlation between neighbor pixels in a 2D image. Visiting too many points with a big receptive field can make the noise unacceptably large. On the other hand, using a small receptive field to visit very few points can make the information from points useless to calculate the correct coordinates. Currently, there are no theoretical guidelines for determining this hyperparameter, and we figure out this optimal value empirically in Section 5.1.

Figure 3 visualizes the effects of CCM. We use the t-SNE method to show the feature map of DGCNN and CCDGN. We randomly choose 10 classes (represented with different colors), and each class contains 50 point clouds. Circles and triangles denote the clean and perturbed point clouds, respectively. From Figure 3a, we can see that in DGCNN, some perturbed data are far from the clean data in the same class, or even overlapped with data from other classes. This implies misclassification for those data. In contrast, for CCDGN (Figure 3b), the perturbed and clean data in the same class are more close, and there are less overlap among different classes. This indicates that CCM can effectively remove the noise, making the perturbed and clean data more close in the feature space.



(a) DGCNN (b) CCDGN

Figure 3: Feature map visualization with t-SNE.

3.2 Robustness Analysis of Point Cloud

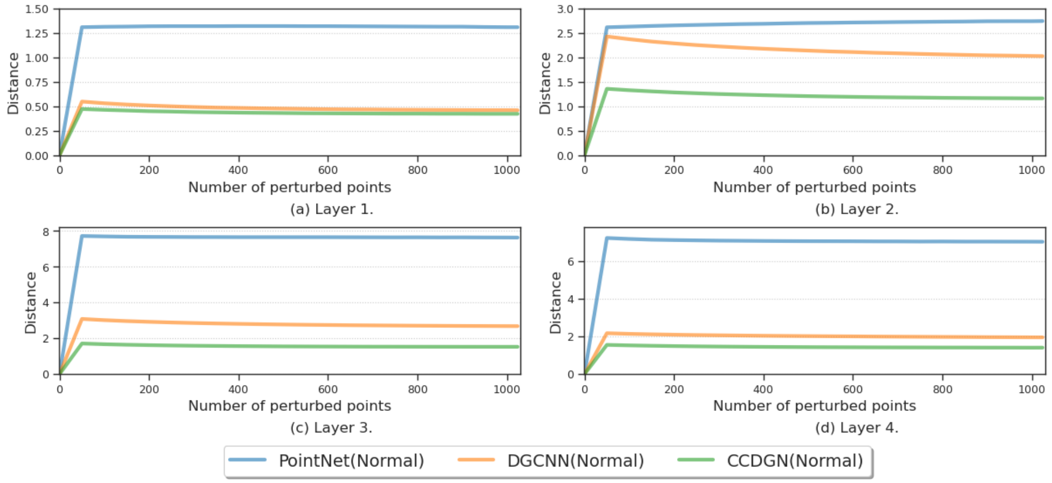
We present some theoretical analysis about the robustness of CCDGN. Past works have developed frameworks to study the vulnerability of adversarial examples for 2D image models based on the mutual information theory (Hjelm et al., 2018; Zhu et al., 2020). Inspired by those works, we aim to disclose the factors that can affect the robustness of point cloud models.

Specifically, we apply mutual information to calculate the correlation between perturbed point clouds and the features of clean point clouds. A high correlation indicates the feature context of noisy data is more consistent with that of clean data. Hence, the model is more robust to predict correct labels from noisy samples. However, it is computationally infeasible to calculate such mutual information, due to the high dimension of the input space and feature space. Alternatively, we propose to estimate the mutual information with a substitute measurement, *i.e.*, the k-Measurement M_k. This measurement is based on the cosine distance, which can represent both the direction and magnitude of a distance in a high dimension at the same time. Formally, we have the following definition:

Definition 1 (k-Measurement)**.** *Let f be a function that maps a point cloud to the feature space: $\{x_i|x_i \in R^d, i \in [N]\} \mapsto \{x_i|x_i \in R^D, i \in [N]\}$. $S = \{X_i|X_i = (x_i, y_i, z_i), i \in [N]\}$ is a clean point cloud. S_k is a perturbed point cloud with k different points compared with S,* i.e., *$S_k = \{X_{j_i}|X_{j_i} = (x_{j_i}, y_{j_i}, z_{j_i}) \in S, j_i \in [N-k]\} \cup \{X_{h_i} + \epsilon_{h_i}|X_{h_i} = (x_{h_i}, y_{h_i}, z_{h_i}) \in S, \epsilon_{h_i} = (\epsilon_{0,h_i}, \epsilon_{1,h_i}, \epsilon_{2,h_i}), h_i \in [k]\}$. Then the k-Measurement M_k for f, S and S_k is defined as:*

$$M_k(f, S, S_k) = 1 - \frac{f(S) \cdot f(S_k)}{\|f(S)\|\|f(S_k)\|}$$

We introduce a general theorem to prove that under the same k, a small $M_k(f, S, S_k)$ implies a large mutual information $I(S_K, f(S))$. The proof can be found in the appendix.



(a) Layer 1.

(b) Layer 2.

(c) Layer 3.

(d) Layer 4.

Figure 4: Cosine distance of features between clean and adversarial samples from different layers.

Theorem 1. *Let f be a function that maps a point cloud to the feature space, and Q be the distribution of clean point clouds. S is sampled from Q. $Q^k(S, \epsilon)$ is the distribution of noisy point clouds, in which each element S_k is perturbed from S with an additional noise ϵ, and the difference of numbers of points between S and S_k is smaller than a constant k,* i.e.*, $-k \leq |S_k| - |S| \leq k$. Then for every $S \sim Q$ and $S_k \sim Q^k(S, \epsilon)$, the mutual information $I(S_k, f(S))$ has a lower bound, which is negatively correlated with the k-measurement $M_k(f, S, S_k)$.*

Theorem 1 establishes a connection between $M_k(f, S, S_k)$ and $I(S_k, f(S))$. A small k-measurement $M_k(f, S, S_k)$ could increase the mutual information value $I(S_k, f(S))$. With a large $I(S_k, f(S))$, the corresponding point cloud model is more robust, as the confidence of correctly predicting $f(S)$ from the perturbed sample S_k is higher. Hence, *one possible strategy of improving the model's robustness is to decrease the k-measurement $M_k(f, S, S_k)$.*

We use this metric to demonstrate the superiority of CCDGN benefiting from the CCM. For comparisons, we consider three network architectures: PointNet has five convolutional layers, and the first four are used to extract features; both DGCNN and CCDGN have four EdgeConv layers for feature extraction. We compute the cosine distances between the features of clean and adversarial point clouds at these four layers[1]. Figure 4 shows the distance of each case versus the number of perturbed points in AEs. We observe that CCDGN always gives the lowest distances, as the CCMs can efficiently increase the mutual information and enhance the robustness. This indicates the effectiveness of CCDGN in handling adversarial perturbations in the point clouds.

4 An Adaptive Augmentation Strategy

In addition to CCDGN, we also introduce a novel data augmentation strategy to enhance the robustness of a point cloud model. In the conventional 2D image tasks, there are general two types of training strategies for defeating adversarial examples. Unfortunately, they cannot achieve satisfactory performance when extended to 3D point cloud models. The first strategy is adversarial training, which augments the training set with adversarial examples crafted by the PGD technique. However, there are essentially different types of AE generation methods for point clouds. Adversarial training with one type of AEs cannot provide comprehensive protection for other types of attacks (Liu et al., 2019; Zhang et al., 2021), while simply incorporating all these sorts of AEs can significantly harm the model accuracy for clean samples. The second strategy is to mix up clean samples with different labels for model training (Zhang et al., 2018). This strategy is applied to the point cloud classification (Chen et al., 2020; Zhang et al., 2021), which have limited robustness improvement.

[1] For CCDGN, we choose $\alpha = 4$, which is identified in Section 5.1.

Algorithm 1: Adaptive Augmentation with Adversarial and Mix-up Samples (AA-AMS).

```
Input  : $Q$: point cloud training set
Output: $M$: robust point cloud model
Initialize($M$);
foreach training epoch do
    $acc = []$;
    foreach batch $(X, Y) \sim Q$ do
        $X_{\text{perturb}} = \texttt{AE-GEN}^{\texttt{perturb}}(X, Y, M)$, $X_{\text{drop}} = \texttt{AE-GEN}^{\texttt{drop}}(X, Y, M)$,
         $(X_{\text{mix}}, Y_{\text{mix}}) = \texttt{MS-GEN}(X, Y)$;
        calculate accuracy $acc_x$, $acc_{\text{perturb}}$ and $acc_{\text{drop}}$ for $X$, $X_{\text{perturb}}$ and $X_{\text{drop}}$;
        $acc.\text{append}(acc_x)$, $acc_{\min} = \min(acc_{\text{perturb}}, acc_{\text{drop}})$;
        if $acc_{\min} > T * \text{mean}(acc)$ then
            train $M$ with $(X, Y)$ and $(X_{\text{mix}}, Y_{\text{mix}})$;
        else
            train $M$ with $(X_{\text{perturb}}, Y)$, $(X_{\text{drop}}, Y)$ and $(X, Y)$;
return $M$
```

Our Adaptive Augmentation strategy (AA-AMS) considers the adversarial examples (of different types), mix-up samples as well as clean samples for model training. However, it is difficult to decide the type and quantity of samples to be used before the training task, as the training process is dynamic and relatively random. To overcome this challenge, AA-AMS adaptively selects the desired samples in each epoch based on the current model. This dynamic selection can efficiently balance the model robustness and accuracy over clean samples for the complex 3D point cloud tasks.

Our training algorithm is shown in Algorithm 1. At every training epoch, for each batch (X, Y) from the training set Q, we first generate three types of batches from each sample in the batch[2]: (1) X_{drop} is a batch of AEs with the point dropping technique using the function $\texttt{AE-Gen}^{\texttt{drop}}$; (2) X_{perturb} is a batch of AEs with the point perturbing technique using the function $\texttt{AE-Gen}^{\texttt{perturb}}$; (3) X_{mix} is a batch of mix-up samples with the corresponding mix-up labels Y_{mix} using the function MS-Gen. Second, we compute the accuracy of clean and AE batches from the current model, as acc_x, acc_{drop} and acc_{perturb}, respectively. We compute $acc_{\min} = \min(acc_{\text{drop}}, acc_{\text{perturb}})$, and compare it with the weighted mean accuracy of the clean batches $acc_{\text{avg}} = T * \text{mean}(acc)$, where acc is a collection of clean accuracy acc_x at the current training epoch. If $acc_{\min}$ is higher than acc_{avg}, then this model is regarded as robust enough to defend against different types of AEs. So we perform *mix-up augmentation* to improve the model's generalization and utility, *i.e.*, training the model with the clean batch (X, Y) and mix-up batch $(X_{\text{mix}}, Y_{\text{mix}})$. Otherwise, we perform *adversarial augmentation* to improve the model's robustness, *i.e.*, training it with the clean batch (X, Y) and two types of adversarial batches (X_{drop}, Y), (X_{perturb}, Y).

In practice, we implement MS-Gen with the PointCutMix approach (Zhang et al., 2021). For $\texttt{AE-Gen}^{\texttt{drop}}$, we adopt the Saliency Map Attack (Zheng et al., 2019). For $\texttt{AE-Gen}^{\texttt{perturb}}$, we utilize the 3D L_∞-BIM technique (Kurakin et al., 2017), which is a basic version of L_∞-PGD (Xiang et al., 2019)[3]. Besides, we calculate the averaged accuracy of acc_x for clean samples to avoid overfitting of T on a specific model and make the algorithm better generalize to other models. The optimal value of T needs to be empirically determined, as shown in Section 5.2. Our experiments in Section 5.3 indicate that AA-AMS can help the model obtain higher robustness than conventional adversarial training methods under the same computational complexity limitation.

[2] We do not consider the point adding technique as the generation complexity is extremely high. Our experiments show the incorporation of the other two AEs can defeat the point adding AEs as well.

[3] We do not use L_∞-PGD because when we randomly project the point cloud to an initialization position, the model has a high chance to give a wrong prediction initially, and the adversary will obtain less useful information than starting from the original position.

5 Evaluations

Dataset and Models. We perform comprehensive experiments to validate the effectiveness of CCDGN, AA-AMS, and their combination. The dataset we adopt is ModelNet40 (Wu et al., 2015), which contains 12,311 CAD objects from 40 different classes. These objects are split into a training set of 9,843 samples and a testing set of 2,468 samples. For the training process, all models are trained for 250 epochs with a learning rate of 0.001 and the Adam optimizer (Kingma & Ba, 2015). The size of an input point cloud is 1024 * 3, *i.e.*, there are 1024 points in each point cloud with three coordinates. We mainly consider the PointNet and DGCNN models in this paper. The evaluation results for PointNet++ give the same conclusion, and can be found in the appendix.

Attacks. To demonstrate the comprehensive protection of our methodology, we implement five state-of-the-art adversarial attacks for testing (four white-box attacks and one black-box attack). All of them are implemented as untargeted attacks. Specifically,

- SMA$-k$ (Zheng et al., 2019) is a point dropping attack which drops 5 $\times k$ points in k iterations based on the saliency map.
- APP (Xiang et al., 2019) is a point perturbing attack which shifts points with 10 binary searches and 100 iterations for each search.
- AIC (Xiang et al., 2019) is a point adding attack which conducts 10 binary searches and 100 iterations for each search to add 512 points to the point cloud. Chamfer distance is adopted to measure the point locations.
- AIH (Xiang et al., 2019) is similar as AIC with the Hausdorff distance.
- AdvPC (Hamdi et al., 2020) is a state-of-the-art black-box attack with higher transferability than others. We follow the same hyperparameters in the original paper, and use a larger number of iterations (500) to improve its performance.

Baselines. We select a couple of baseline methods to compare with our solution. (1) For the ablation study of CCDGN, we choose the conventional PointNet and DGCNN as the baselines. (2) For the ablation study of AA-AMS, we compare it with normal training, adversarial training and mix-up training. For adversarial training, we consider two strategies: AT-BIM trains the model using the 3D L_∞-BIM point perturbing technique (Kurakin et al., 2017), with the configurations of 20 iterations, $\epsilon = 0.02$ and step $= 0.005$; AT-SMA trains the model using the point dropping technique (Zheng et al., 2019), with the configurations of 20 iterations and 5 points dropped in each iteration. For mix-up training, we select PointCutMix-K (Zhang et al., 2021), as it achieves the highest robustness in the white-box scenario. (3) For evaluating the integration of the two techniques, we consider the following state-of-the-art solutions: adversarial training (AT-BIM and AT-SMA); mix-up training (PointCutMix-R and PointCutMix-R (Zhang et al., 2021)), SRS (Zhang et al., 2019), SOR with the configuration of k=2 and α=1.1 (Rusu et al., 2008) and DUP-Net (Zhou et al., 2019). Since these solutions have different targets, we will also consider some of their combinations for evaluation.

Metrics. We measure the model accuracy over clean samples and different types of adversarial examples to represent its usability and robustness, respectively. We further introduce two metrics to quantify the overall robustness of a methodology: (1) **AAUA** measures the Average Accuracy Under Attacks in our consideration; (2) **LAUA** measures the Lowest Accuracy Under Attacks, which represents the worst situation.

5.1 Ablation Study of CCDGN

As discussed in Section 3.1, the size α of the receptive field in the CCM can affect the model's robustness against different types of attacks. We first perform ablation studies on the hyperparameter α. We compare the performance of our CCDGN for different α values with PointNet and DGCNN under four white-box attacks. Each model is trained with PointCutMix-K, which gives the best results compared to other training strategies. Table 1 presents the results. First, we observe that PointNet has better robustness against the point perturbing attack (APP) and adding attack (AIH), as it only uses individual points to generate features, avoiding the noise accumulation. However, it has very bad performance for the point dropping attack (SMA$-$40). Second, our CCDGN with the CCMs provide better accuracy for both clean and adversarial examples. The accuracy values for different AEs change with the hyperparameters. We find that $\alpha = 4$ can give the best trade-off considering all the point adding, dropping and perturbing attacks. It gives the highest **AAUA** and second highest **LAUA**. The clean sample accuracy is also higher than PointNet and DGCNN. In the following experiments, we will fix α as 4.

Network	Clean	Adversarial Examples					
Structure	Sample	SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet	90.83	63.11	**82.55**	76.14	**69.56**	72.84	63.11
DGCNN	91.88	79.91	74.88	76.01	68.47	74.82	**68.47**
CCDGN (α=20)	92.71	**82.04**	80.21	73.78	66.44	75.62	66.44
CCDGN (α=16)	92.53	80.80	78.21	73.70	64.41	74.28	64.41
CCDGN (α=12)	**92.74**	80.64	79.55	75.04	66.44	75.42	66.44
CCDGN (α=8)	92.05	81.66	78.81	71.14	63.92	73.88	63.92
CCDGN (α=4)	92.25	81.17	79.46	**76.46**	68.30	**76.35**	**68.30**
CCDGN (α=1)	92.37	80.60	78.45	74.88	66.60	75.13	66.60

Table 1: Model accuracy for different network architectures and hyperparameters (%).

5.2 Ablation Study of AA−AMS

Next, we focus on the evaluation of our adaptive augmentation strategy. One important hyperparameter in AA−AMS is T, which determines the kind of batch samples for training. We perform an ablation study to select the optimal T value. We use the PointNet model, which is simple and easy to obtain the results. We generate X_{drop} using the Saliency Map Attack (10 iterations, 10 points dropped in each iteration) and X_{perturb} using the 3D L_∞-BIM attack (10 iterations, $\epsilon = 0.02$ and step $= 0.005$). $(X_{\mathrm{mix}}, Y_{\mathrm{mix}})$ are generated by PointCutMix-K. Four white-box attacks are used for evaluation. Table 2 presents the accuracy of models trained with different strategies. *The computation complexities of AT-BIM, AT-SMA and AA−AMS are similar, as the number of iterations for AE generation in AA−AMS is half of the others.*

Training	Clean	Adversarial Examples					
Strategy	Sample	SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
Normal	88.76	41.88	55.64	49.68	43.43	47.66	41.88
PointCutMix-K	**90.83**	63.11	82.55	76.14	69.56	72.84	63.11
AT-BIM	88.23	45.41	85.39	84.98	86.36	75.54	45.41
AT-SMA	87.38	**67.37**	79.79	75.73	74.92	74.45	**67.37**
AA−AMS (T=0.7)	88.64	51.30	86.69	85.31	85.96	77.32	51.30
AA−AMS (T=0.5)	89.45	48.99	87.01	**86.49**	**87.26**	**77.44**	48.99
AA−AMS (T=0.3)	89.65	46.02	**87.30**	86.00	86.77	76.52	46.02
AA−AMS (T=0.1)	89.20	42.98	80.24	79.87	81.01	71.03	42.98

Table 2: Model accuracy for different training strategies and hyperparameters (%).

From Table 2, we observe that PointCutMix-K can achieve high accuracy over clean samples and AEs with SMA−40. However, it behaves much worse under the other three attacks. For AA−AMS, the value of T can affect the model accuracy over different types of samples. With $T = 0.5$, the model has the highest robustness against AIC and AIH attacks. Although **LAUA** in this configuration is lower than PointCutMix-K and AT-SMA (due to the bad performance in SMA−40), the average accuracy **AAUA** is still the highest. This validates the advantage of AA−AMS, and we will adopt $T = 0.5$ for the following experiments.

5.3 More Comprehensive Evaluations

After identifying the optimal hyperparameters, we compare our methodology with existing works of different network architectures (PointNet, DGCNN, DGCNN with SOR, SRS and DUP-Net) and training strategies (Normal, PointCutMix-R, PointCutMix-K, AT-BIM, AT-SMA). Table 3 shows the comparison results. There can be a lot of combinations with these solutions. Since PointNet has the least robustness among these architectures, we mainly compare the DGCNN architecture. First, we observe that our solution achieves the highest accuracy over clean samples. Second, for adversarial attacks, our solution also gives the best result for APP and AIC attacks. For SMA−40, our solution is worse than DGCNN+AT-SMA; for AIH, our solution is slightly worse than PointNet + AT-BIM. Nevertheless, it still gives the highest **AAUA** and **LAUA**, due to its comprehensive robustness.

Defense Solutions	Clean Sample	Adversarial Examples					
		SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet + Normal	88.76	41.88	55.64	49.68	43.43	47.66	41.88
PointNet + PointCutMix-K	90.83	63.11	82.55	76.14	69.56	72.84	63.11
PointNet + AT-BIM	88.23	45.41	85.39	84.98	**86.36**	75.54	45.41
PointNet + AT-SMA	87.38	67.37	79.79	75.73	74.92	74.45	67.37
DGCNN + Normal	91.03	65.87	46.10	54.06	48.78	53.70	46.10
DGCNN + PointCutMix-R	90.91	72.65	71.63	62.26	56.53	65.77	56.53
DGCNN + PointCutMix-K	91.88	79.91	74.88	76.01	68.47	74.82	68.47
DGCNN + AT-BIM	91.27	66.68	89.98	81.37	76.99	78.76	66.68
DGCNN + AT-SMA	91.80	**84.66**	72.00	71.75	64.25	73.17	64.25
DGCNN + SOR + Normal	91.00	66.00	86.83	51.82	54.38	64.76	51.82
DGCNN + SOR + AT-BIM	91.77	65.52	84.97	58.59	58.27	66.84	58.27
DGCNN + SOR + AT-SMA	91.05	80.59	86.91	59.85	60.25	71.90	59.85
SRS*	83.00	35.10	64.70	59.50	58.80	54.53	35.10
DUP-Net*	86.30	43.70	84.50	61.40	62.70	63.08	43.70
CCDGN + AA-AMS	**92.41**	77.72	**90.50**	**86.09**	84.05	**84.74**	**77.72**

Table 3: Model accuracy for different solutions under the white-box attacks (%).*Data of SRS and DUP-Net are adopted from (Zhou et al., 2019).

We further evaluate our methodology against a black-box attack (AdvPC). The adversary crafts AEs from a different source model, and then leverages the transferability to attack the target victim model. We consider two constraints to generate AEs for testing. The results are shown in Table 4. We observe that for the source model of DGCNN with $\epsilon = 0.18$, our solution (CCDGN + AA-AMS) is slightly worse than DGCNN + AT-BIM. For the source model of DGCNN with $\epsilon = 0.45$, our solution is slightly worse than DGCNN + AA-AMS. For the rest of cases, it gives the highest accuracy. This indicates the effectiveness of our proposed solution under the black-box attack.

Target Model	Training Strategy	Source Model					
		PointNet		DGCNN		CCDGN	
		$\epsilon = 0.18$	$\epsilon = 0.45$	$\epsilon = 0.18$	$\epsilon = 0.45$	$\epsilon = 0.18$	$\epsilon = 0.45$
PointNet	Normal	84.50	84.50	86.27	86.27	86.98	85.56
	AT-BIM	86.11	84.70	87.88	87.88	85.76	86.82
	AA-AMS	86.59	85.51	88.02	86.95	88.02	88.02
DGCNN	Normal	89.21	89.21	87.02	86.30	83.38	85.57
	AT-BIM	89.08	90.54	**89.81**	89.08	88.71	89.08
	AA-AMS	88.89	89.26	89.63	89.63	90.00	**90.73**
CCDGN	Normal	89.42	88.33	89.05	88.33	85.42	85.42
	AT-BIM	88.61	88.61	88.61	88.25	88.25	88.97
	AA-AMS	**90.93**	**90.56**	89.45	**90.19**	**90.56**	90.19

Table 4: Model accuracy for different solutions under the black-box attacks (%).

We further compare our methodology with more baselines, model architectures and attack configurations. The results can be found in the appendix. All the results confirm that our proposed CCDGN trained with AA-AMS has the best robustness against different types of AEs.

6 Conclusion

Numerous research works have been done to increase our understanding about the inherent features of adversarial examples and model robustness in 2D image tasks. However, studies of adversarial defenses in the point cloud domain are still at an early stage. We advance this research direction with two contributions. For network architecture, we propose CCDGN, which can denoise the adversarial point clouds and smooth the perturbations in the feature space. For model training, we propose AA-AMS, which can adaptively select clean, mix-up or adversarial samples to balance the model utility and robustness. Comprehensive evaluations show that our solution outperforms a variety of baselines under different types of white-box and black-box attacks.

References

Amir Arsalan Soltani, Haibin Huang, Jiajun Wu, Tejas D Kulkarni, and Joshua B Tenenbaum. Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In *Proc. of the CVPR*, pp. 1511–1519, 2017.

Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *Proc. of the S&P*, pp. 39–57, 2017a.

Nicholas Carlini and David A. Wagner. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In *Proc. of the 10th {ACM} Workshop on Artificial Intelligence and Security*, pp. 3–14, 2017b.

Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees G. M. Snoek. PointMixup: Augmentation for Point Clouds. In *Proc. of the ECCV*, pp. 330–345, 2020.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *Proc. of the ICLR*, 2015.

Abdullah Hamdi, Sara Rojas, Ali K. Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *Proc. of the ECCV*, volume 12357, pp. 241–257, 2020.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

Pileun Kim, Jingdao Chen, and Yong K Cho. Slam-driven robotic mapping and registration of 3d point clouds. *Automation in Construction*, 89:38–48, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the ICLR*, 2015.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Proc. of the ICLR (Workshop)*, 2017.

Daniel Liu, Ronald Yu, and Hao Su. Extending Adversarial Attacks and Defenses to Deep 3D Point Cloud Classifiers. In *Proc. of the ICIP*, 2019.

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 *Proc. of the MM*, pp. 1819–1827, 2020.

Hélène Macher, Tania Landes, and Pierre Grussenmeyer. From point clouds to building information models: 3d semi-automatic reconstruction of indoors of existing buildings. *Applied Sciences*, 7(10):1030, 2017.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *Proc. of the ICLR*, 2018.

Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. of the CVPR*, 2017a.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Proc. of the NIPS*, pp. 5099–5108, 2017b.

Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Emanuel Dolha, and Michael Beetz. Towards 3D Point cloud based object maps for household environments. *Robotics Auton. Syst.*, 56(11):927–941, 2008.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing Properties of Neural Networks. In *Proc. of the ICLR*, 2014.

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 Trans. Graph.*, 38(5):146:1–146:12, 2019.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A deep representation for volumetric shapes. In *Proc. of the CVPR*, 2015.

Ziyi Wu, Yueqi Duan, He Wang, Qingnan Fan, and Leonidas J. Guibas. If-defense: 3d adversarial point cloud defense via implicit function based restoration. *CoRR*, abs/2010.05272, 2020.

Chong Xiang, Charles R. Qi, and Bo Li. Generating 3D Adversarial Point Clouds. In *Proc. of the CVPR*, 2019.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond Empirical Risk Minimization. In *Proc. of the ICLR*, 2018.

Jinlai Zhang, Lyujie Chen, Bo Ouyang, Binbin Liu, Jihong Zhu, Yujing Chen, Yanmei Meng, and Danfeng Wu. PointCutMix: Regularization Strategy for Point Cloud Classification. *CoRR*, abs/2101.01461, 2021.

Qiang Zhang, Jiancheng Yang, Rongyao Fang, Bingbing Ni, Jinxian Liu, and Qi Tian. Adversarial attack and defense on point sets. *CoRR*, abs/1902.10899, 2019.

Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. PointCloud Saliency Maps. In *Proc. of the ICCV*, 2019.

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 *Proc. of the ICCV*, 2019.

Sicheng Zhu, Xiao Zhang, and David Evans. Learning Adversarially Robust Representations via Worst-Case Mutual Information Maximization. In *Proc. of the ICML*, pp. 11609–11618, 2020.

A Mutual Information Maximization

Theorem 1. *Let f be a function that maps a point cloud to the feature space, and Q be the distribution of clean point clouds. S is sampled from Q. $Q^k(S, \epsilon)$ is the distribution of noisy point clouds, in which each element S_k is perturbed from S with an additional noise ϵ, and the difference of numbers of points between S and S_k is smaller than a constant k,* i.e.*, $-k \leq |S_k| - |S| \leq k$. Then for every $S \sim Q$ and $S_k \sim Q^k(S, \epsilon)$, the mutual information $I(S_k, f(S))$ has a lower bound, which is negatively correlated with the k-measurement $M_k(f, S, S_k)$.*

Proof. According to the definition of mutual information, we have the following equation:

$$I(S_k, f(S)) = H(f(S)) - H(f(S)|S_k) = -\sum_{1}^{|Q|} \frac{1}{|Q|} \log f(S) - H(f(S)|S_k).$$

We use $B(S, \epsilon)$ to denote a hyper-sphere whose center is S and radius is $||\epsilon||$. So the second term of the above expression can be rewritten as follows:

$$\begin{aligned} -H(f(S)|S_k) &= \sum_{k} \sum_{S_k \sim Q^k(S,\epsilon)} Pr[f(S), S_k] \log Pr[f(S)|S_k] \\ &\geq \sum_{k} \int_{\epsilon} \int_{S_k \sim B(S,\epsilon)} \frac{1}{M_k(f, S, S_k)} \log \frac{|Q|}{M_k(f, S, S_k)}. \end{aligned}$$

The mutual information can be further derived as follows:

$$I(S_k, f(S)) \geq -\sum_{1}^{|Q|} \frac{1}{|Q|} \log f(S) + \sum_{k} \int_{\epsilon} \int_{S_k \sim B(S,\epsilon)} \frac{1}{M_k(f, S, S_k)} \log \frac{|Q|}{M_k(f, S, S_k)}.$$

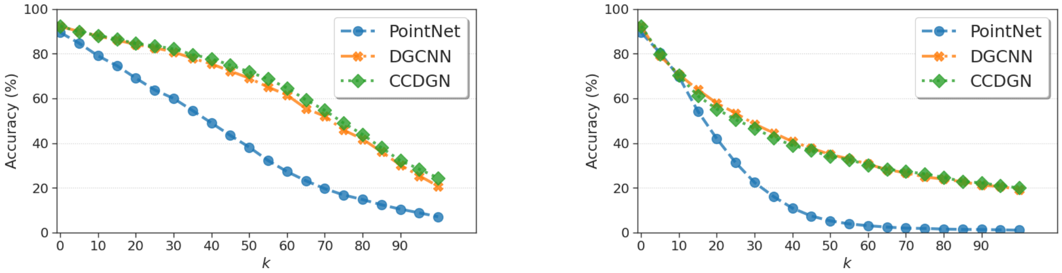
This means the lower bound of $I(S_k, f(S))$ is increased when $M_k(f, S, S_k)$ is smaller. □

B PointNet++ under attacks

Network Structure	Training Strategy	Clean	Attacks					
			SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet++ (MSG)	Normal	89.77	62.18	0.00	0.00	0.00	15.55	0.00
	PointCutMix-R	92.57	85.92	0.00	0.16	2.15	22.06	0.00
	AT-BIM	88.47	64.00	0.00	0.00	0.00	16.00	0.00
PointNet++ (SSG)	Normal	89.85	56.45	0.00	0.00	0.00	14.11	0.00
	PointCutMix-R	92.78	86.57	0.00	1.83	2.88	22.82	0.00
	AT-BIM	89.53	71.51	0.00	0.00	0.00	17.88	0.00

Table 1: Accuracy of PointNet++ under untargeted attacks.

We further compare the accuracy of two types of PointNet++ under attacks. The results are shown in Table 1. For both types of PointNet++, training models with mix-up samples can significantly improve the accuracy under the dropping point attack, as mix-up samples can be seen as clean points dropped a lot of original points. However, PointNet++ cannot defend against adding perturbation attacks and adding additional points attacks. As we analyzed before, PointNet++ uses each point and its neighbors sampled based on distances coordinates to generate local features directly. When sampling neighbors on perturbed point clouds or point clouds with additional points, PointNet++ will use more noisy points to generate local features causing noise accumulating. Comparing with previous works, we find when using targeted attacks to attack PointNet++, the accuracy under attacks are significantly higher than results in Table 1. It is easy to understand that untargeted attacks are more powerful, and PointNet++ does not always predict adversarial examples as labels the adversary wants. For an adversary who wants the model to give wrong labels instead of specific labels, attacking PointNet++ is uncomplicated. Since the structure of PointNet++ is fragile under attacks, we do not apply our AA−AMS on it.



(a) Accuracy under SMA attack with different k

(b) Accuracy under BIM attack with different k

Figure 1: Accuracy of models under SMA$-k$ and BIM$-k$. All models are trained with our AA$-$AMS.

C Experiments of AA$-$AMS

When comparing normally trained models and models trained with AT-BIM, we find that DGCNN achieves the highest clean accuracy. However, the DGCNN does not outperform our CCDGN under all four white-box attacks. On the other hand, the PointNet shows the worst performance. When comparing models trained with AT-BIM and our AA$-$AMS, we can clearly notice that the AA$-$AMS generalizes well to other model structures. Our CCDGN outperforms other baselines on clean accuracy and many white-box attacks. It achieves not only the highest **AAUA** but also the highest **LAUA**. It means that our CCDGN can work with the AA$-$AMS together in harmony. In summary, for each architecture, AA$-$AMS gives the best performance compared to Normal or AT-BIM training. To sum up, the integration of CCDGN and AA$-$AMS is the most robust solution.

Network Structure	**Training Strategy**	**Clean Sample**	**Adversarial Examples**					
			SMA$-$40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet	Normal	88.76	41.88	55.64	49.68	43.43	47.66	41.88
	AT-BIM	88.23	45.41	85.39	84.98	86.36	75.54	45.41
	AA$-$AMS	89.45	48.99	87.01	**86.49**	**87.26**	77.44	48.99
DGCNN	Normal	91.03	65.87	46.10	54.06	48.78	53.70	46.10
	AT-BIM	91.27	66.68	89.98	81.37	76.99	78.76	66.68
	AA$-$AMS	**92.21**	75.41	**90.83**	85.47	83.93	83.91	75.41
CCDGN	Normal	90.87	67.94	57.47	61.04	53.37	59.96	53.37
	AT-BIM	90.05	67.37	88.80	83.77	79.75	79.92	67.37
	AA$-$AMS	**92.41**	**77.72**	**90.50**	**86.09**	84.05	**84.74**	**77.72**

Table 2: Model accuracy for different solutions under the white-box attacks (%).

D Experiments under SMA and BIM

BIM$-k$ is a point perturbing attack using the L_∞ basic iterative method: each sample is generated with k iterations, $\epsilon = 0.03$ and step size $= 0.0005$. We do not adopt the PGD attack as the adversarial point cloud will get disrupted at the beginning of sample generation.

Furthermore, we show the accuracy of models trained with our AA$-$AMS under SMA$-k$ and BIM$-k$ with different k in Figure 1. When models are attacked by SMA$-k$, the PointNet is more fragile than other two models, resulting it has the lowest accuracy. The CCDGN outperforms the DGCNN with the k increasing becoming more clearly. When we attack models with BIM$-k$, we find that when the k is small, the accuracy of three models are very close. With the k increasing, the accuracy of the PointNet drops very quickly. As for CCDGN and DGCNN, the accuracy of DGCNN is higher than the accuracy of CCDGN at the start. However, when the k is higher than 60, the CCDGN starts to outperform the DGCNN. Both of them achieve higher accuracy than the PointNet. Overall, our CCDGN is the most robust one.