Common Corruption Robustness of Point Cloud Detectors: Benchmark and Enhancement

Shuangzhi Li, Graduate Student Member, IEEE, Zhijie Wang, Student Member, IEEE, Felix Juefei-Xu, Member, IEEE, Qing Guo, Member, IEEE, Xingyu Li, Member, IEEE, and Lei Ma, Member, IEEE

Abstract—Object detection through LiDAR-based point cloud has recently been important in autonomous driving. Although achieving high accuracy on public benchmarks, the state-of-the-art detectors may still go wrong and cause a heavy loss due to the widespread corruptions in the real world like rain, snow, sensor noise, etc. Nevertheless, there is a lack of a large-scale dataset covering diverse scenes and realistic corruption types with different severities to develop practical and robust point cloud detectors, which is challenging due to the heavy collection costs. To alleviate the challenge and start the first step for robust point cloud detection, we propose the physical-aware simulation methods to generate degraded point clouds under different real-world common corruptions. Then, for the first attempt, we construct a benchmark based on the physical-aware common corruptions for point cloud detectors, which contains a total of 1,122,150 examples covering 7,481 scenes, 25 common corruption types, and 6 severities. With such a novel benchmark, we conduct extensive empirical studies on 12 state-of-the-art detectors that contain 6 different detection frameworks. Thus we get several insight observations revealing the vulnerabilities of the detectors and indicating the enhancement directions. Moreover, we further study the effectiveness of existing robustness enhancement methods based on data augmentation, data denoising, test-time adaptation. The benchmark can potentially be a new platform for evaluating point cloud detectors, opening a door for developing novel robustness enhancement methods.

I. INTRODUCTION

OBJECT detection via LiDAR-based point cloud [1], [2], as a crucial task in 3D computer vision, has been widely used in applications like autonomous driving [3]. Recently, the data-driven methods (i.e., deep neural networks) have significantly improved the performance of 3D point cloud detectors [2], [4], [5] on various public benchmarks, e.g., KITTI [6], NuScenes [7], and Waymo [8]. However, the scenarios covered by these public benchmarks are usually limited. For instance, there is a lack of natural fog effects in these datasets, while fog could affect the reflection of laser beams and corrupt point cloud data with false reflections by droplets [9], [10]. Apart from the external scenarios, the internal noise of sensors can also increase the deviation and variance of ranging measurements [11] and result in corrupted data and detector performance degradation. Given that LiDAR-based point cloud detection is usually used in safety-critical applications (e.g., autonomous driving) and these external and internal corruptions could potentially affect detectors’ robustness [11], [12], [13], it is critical to comprehensively evaluate an object detector under those corruptions before deploying it in real-world environments.

There are some works constructing datasets while considering extreme weather like CADC [14], Boreas [15], SeeThrough-Fog (STF) [10]. Nevertheless, the constructed datasets only consider limited situations in the real world due to the heavy collection costs, which are far from a comprehensive evaluation. For instance, Boreas only covers 4 rainy scenes and 5 snowy scenes. STF only contains foggy point clouds at severity levels of “dense” and “light”. Hence, there is an increasing demand for extending existing benchmarks to conduct a comprehensive evaluation through covering diverse corruptions in the real world. A straightforward way is to synthesize the corrupted point clouds given the success of similar solutions in the image-based tasks [16], [17] and 3D object recognition [11], [18]. However, there is no accessible dataset for the robustness evaluation of

Manuscript received 15 October 2022; revised 21 April 2023; accepted 8 September 2023. Date of publication 16 October 2023; date of current version 21 February 2025. The work of Shuangzhi Li, Zhijie Wang, and Lei Ma was supported in part by Canada CIFAR AI Chairs Program, in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grants RGPIN-2021-02549, RGPAS-2021-00034, and DGCER-2021-00019, and in part by JST-Mirai Program Grants JPMJM20B8, JSPS KAKENHI, JP21H04877, and JP23H03372. The work of Qing Guo was supported in part by the National Research Foundation, Singapore, and DSO National Laboratories under the AI Singapore Programme under A*STAR under Grant A18A10045. This work was done prior to Felix Juefei-Xu joined Meta. Meta is not involved in creating/releasing this benchmark. The Guest Editor coordinating the review of this manuscript and approving it for publication was Prof. Kai Xu. (Corresponding author: Qing Guo.)

Shuangzhi Li and Xingyu Li are with the University of Alberta, Edmonton, AB T6G 2R3, Canada (e-mail: shuangzh@ualberta.ca; xingyu@ualberta.ca).

Zhijie Wang is with the University of Alberta, Edmonton, AB T6G 2R3, Canada, and also with the Alberta Machine Intelligence Institute, Edmonton, AB T6G 2R3, Canada (e-mail: zhijie.wang@ualberta.ca).

Felix Juefei-Xu is with the Meta AI, New York, NY 10001 USA (e-mail: felixu@meta.com).

Qing Guo is with the Institute of High Performance Computing (IHPC) and Centre for Frontier AI Research (CFAR), Agency for Science, Technology and Research (A*STAR), Singapore 138632 (e-mail: tsingguo@iee.org).

Lei Ma is with the University of Alberta, Edmonton, AB T6G 2R3, Canada, and with the Alberta Machine Intelligence Institute, Edmonton, T5J 3B1, Canada, and also with the The University of Tokyo, Tokyo 113-8654, Japan (e-mail: ma.lei@acm.org).

We make this benchmark publicly available on https://github.com/Castiel-Lee/robustness_pc_detector.

This article has supplementary downloadable material available at https://doi.org/10.1109/TMM.2023.3318317, provided by the authors.

Digital Object Identifier 10.1109/TMM.2023.3318317

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
point cloud detectors. Note that, the robustness datasets (e.g., Modelnet40-C [11]) for 3D object recognition cannot be used to evaluate the point cloud detectors, directly: (1) the example in the recognition dataset only contains the points of an object and cannot be adopted for object detection task that aims to localize and classify objects in 3D scene. (2) The latest Modelnet-C [18] and Modelnet40-C [11] only consider 7 corruptions and 15 corruptions, respectively, which is still limited for a comprehensive evaluation in safety-critical environments such as autonomous driving. The main challenge for building a dataset for the robustness evaluation of point cloud detection stems from the huge amount of diverse corruption types with different physical imaging principles. For example, flawed sensors and different object characteristics could lead to noise-like corruptions and affect spherical and Cartesian coordinates of points, respectively. Different weathers like rain and fog might lead to false reflections. These corruptions have different imaging principles and need careful designs of the respective simulation methods.

In this work, for the first attempt, we construct a benchmark to evaluate the robustness of point cloud object detectors based on LiDAR under diverse common corruptions and discuss the effectiveness of existing robustness enhancement methods. Regarding the benchmark construction, we first collect existing simulation methods for common corruptions and improve them based on their affecting ranges and physical mechanisms of formation. Then, we borrow 7,481 raw 3D scenes (i.e., clean point clouds) from [6] and build large-scale corrupted datasets by adding 25 corruptions with 6 different severity levels to each clean point cloud. Finally, we obtain a total of 1,122,150 examples covering 7,481 scenes, 25 common corruption types, and 6 severity levels. Compared with real-world data benchmark (see Table I), the proposed benchmark synthesized more examples for benchmarking robustness. Compared with other synthesized benchmark (see Table I), our benchmark provides more types of corruption patterns to specifically support benchmarking object detection. Note that, we conduct extensive experiments to quantitatively validate the effectiveness of simulation methods by evaluating the naturalness of synthesized data.

With such a novel benchmark, we investigate the robustness of current point cloud detectors by conducting extensive empirical studies on 12 existing detectors, covering 3 different representations and 2 different proposal architectures. In particular, we study the following four research questions to identify the challenges and potential opportunities for building robust point cloud detectors:

- **How do the common corruption patterns affect the point cloud detector’s performance?** Given overall common corruptions, an accuracy drop of 10.94% (on average) on all detectors anticipates a noticeable accuracy drop of detectors against diverse corruption patterns.

- **How does the design of a point cloud detector affect its robustness against corruption patterns?** Compared with two-stage detectors, one-stage detectors perform more robust against overall corruptions. Compared with point-based detectors, voxel-involving detectors perform more robust against a majority of corruptions.

- **What kind of detection bugs exist in point cloud detectors against common corruption patterns?** Followed by the decrease in the rate of true detection, common corruptions widely trigger a number of false detections on all point cloud detectors.

- **How do the robustness enhancement techniques improve point cloud detectors against common corruption patterns?** Even with the help of data augmentation, denoising, and test-time adaptation, common corruptions still cause a severe accuracy drop of over 10%.

In summary, this work makes the following contributions:

- **We design the first robustness benchmark of point cloud detection covering 25 common corruptions related to natural weather, noise disturbance, density change, and object transformations at the object and scene level.**

- **Based on the benchmark, we conduct extensive empirical studies to evaluate the robustness of 12 existing detectors to reveal the vulnerabilities of the detectors under common corruptions.**

### Table I

**Summary of Datasets Used for LiDAR-Based Point Cloud Object Detection**

| Dataset     | Year | Real/Simulated | Frames | BBboxes | Classes | Corruptions                                                                 | Corruption | Robustness Metric |
|-------------|------|----------------|--------|---------|---------|------------------------------------------------------------------------------|------------|-------------------|
| KITTI [6]   | 2012 | real           | 15K    | 300K    | 4       | cutout, noise                                                                | 2          | -                 |
| NuScene [7] | 2019 | real           | 400K   | 1.4M    | 23      | rain, snow, clouds, cutout, various vehicle types, noise                    | 2          | -                 |
| Waymo [8]   | 2019 | real           | 200K   | 12M     | 4       | rain, fog, cutout, dust, various vehicle types, noise                       | 2          | -                 |
| Boreas [15] | 2022 | real           | 7.1K   | 300K    | 4       | snow, rain, min, clouds, cutout, noise                                       | 2          | -                 |
| STF [10]    | 2020 | real           | 13.5K  | 100K    | 4       | fog, rain, snow, cutout, noise                                              | 3          | -                 |
| CADC [14]   | 2020 | real           | 7K     | 334K    | 10      | snow, bright light, cutout, noise                                            | 5          | -                 |
| ONCE [19]   | 2021 | real           | 1M(15K labeled) | 417K | 5       | rain, clouds, cutout, noise                                                  | 2          | -                 |
| ModelNet40-C [11] | 2022 | real/simulated | 185K   | -       | 40      | occlusion, LiDAR, local_density_indec, cutout, uniform, Gaussian, impulse, upsample, background, rotation, shear, PFD, RBF, inv_RBF | 6 ✓        | ✓                 |
| ModelNet-C [18] | 2022 | real/simulated | 185K   | -       | 40      | scale, rotate, jitter, drop, global/local, add, global/local                 | 6 ✓        | ✓                 |
| Argoverse [20] | 2019 | real           | 445K   | 993K    | 15      | rain, cutout, dust, noise                                                    | 2          | -                 |
| Lyft Level 5 [21] | 2020 | real           | 30K    | 1.3M    | 9       | rain, cutout, noise                                                         | 2          | -                 |
| Ours        | 2022 | real/simulated | 1.1M   | 15M     | 8       | Scen: rain, snow, fog, uniform, rad, gaussian, rad, impulse, upsample, background, cutout, beam_det, local_decay, layer_det, Object: uniform, gaussian, impulse, upsample, cutout, local, decay, shear, scale, rotation, PFD, translation | 6 ✓        | ✓                 |
We study the existing methods of data augmentation, denoising, and test-time adaptation and explore their performance on robustness enhancement for point cloud detection and further discuss their limitations.

II. RELATED WORK

A. LiDAR Perception

LiDAR perception is sensitive to both internal and external factors that could result in different corruptions. Adversarial weather [9] (e.g., snow, rain, and fog) can dim or even block transmissions of lasers by dense liquid or solid droplets. Regarding noise characteristics of point clouds, strong illumination [23] affects the signal transmission by lowering Signal-to-Noise Ratio (SNR), increasing the noise level of LiDAR ranging [24]. Besides, the intrinsically inaccurately ranging and the sensor vibration [25], [26] potentially trigger noisy observations during LiDAR scanning. Environmental floating particles (e.g., dust [27]) could perturb point cloud with the background noise. Density distribution of LiDAR-based point clouds can also easily affect autonomous driving. For instance, common object-object occlusions block LiDAR scanning on objects in the scene [13]. Besides, the dark-color cover and rough surface [22] could affect LiDAR’s reflection and thus reduce local point density when sensing such objects. Moreover, the malfunction of (fixed or rotary) lasers [13] globally loses points or layers of points in point clouds. For 3D tasks, various shapes [29], [30], locations and poses [28] of objects can also influence the context perception in the scene.

Apart from these natural corruptions, LiDAR perception is also sensitive to adversarial attack. Adversarial attacks [31] pose significant security issues and vulnerability on 3D point cloud tasks (e.g., classification [32], detection [33], and segmentation [34]).

B. Point Cloud Detectors

Based on the different representations acquired from point clouds, point cloud detectors can be categorized into 2D-view-based detectors (e.g., VeloFCN [35] and PIXOR [36]), voxel-based detectors (e.g., SECOND [37] and VoTr [38]), point-based detectors (e.g., PointRCNN [39] and 3D-SSD [40]), and point-voxel-based detectors (e.g., PVRCNN [41] and SA-SSD [42]). On the other hand, based on the different proposal architectures, point cloud detectors can also be divided into one-stage detectors (e.g., 3D-SSD [40] and SA-SSD [42]) and two-stage detectors (e.g., PointRCNN [39] and PVRCNN [41]). In this article, we select 12 representative methods covering all these categories.

C. Robustness Benchmarks Against Common Corruptions

Several attempts have been made to benchmark robustness for different data domains. Based on ImageNet [43], ImageNet-C simulates real-world corruptions to test image classifiers’ robustness. ObjectNet [17] illustrates the performance degradation of 2D recognition models considering object backgrounds, rotations, and imaging viewpoints. Inspired by 2D works, several benchmarks were built for 3D tasks. Modelnet40-C [11] corrupts ModelNet40 [44] with 15 simulated common corruptions affecting point clouds’ noise, density, and transformations, to evaluate the robustness of point cloud recognition. Targeting 7 fundamental corruptions (i.e., “Jitter”, “Drop Global/Local”, “Add Global/Local”, “Scale”, and “Rotate”), ModelNet-C reveals the vulnerability of different components of 9 existing point cloud classifiers. Regarding point cloud detection, NuScenes [7], Waymo [8], and STF [10] collect LiDAR scans under adversarial rainy, snowy, and foggy conditions, where the accuracy of 3D detectors is tested. However, to the best of our knowledge, a lack of benchmark of point cloud detection’s robustness comprehensively against various common corruptions is still remaining.

D. Robustness Enhancement for Point Cloud Detection

Recently, improving the robustness of point cloud detection has also received significant concerns. Zhang et al. propose PointCutMix [45] as a single way to generate new training data by replacing the points in one sample with their optimal assigned pairs in another sample. Lee et al. [46] propose a rigid subset mix (RSMix) augmentation to get a virtual mixed sample by replacing part of the sample with shape-preserved subsets from another sample. Specifically for 3D object detection, there are several ways to improve detectors’ robustness. Choi et al. [47] propose a part-aware data augmentation that stochastically augments the partitions of objects by 5 basic augmentation methods. LiDAR-Aug [48] presents a rendering-based LiDAR augmentation framework to improve the robustness of 3D object detectors. LiDAR light scattering augmentation [12] and LiDAR fog stimulation [49] utilize physics-based simulators to generate data corrupted by fog/snow/rain and then augment object detectors. Lehner et al. [50] improve the generalization of 3D object detectors to bad-shape objects by means of adversarial vector fields. Self-supervised pre-training [51], [52] can also endow the model with resistance to augmentation-related transformations. Besides, denoising methods [53], [54], [55] can remove the outliers in point clouds and thus potentially improve detectors’ robustness. Also, test-time BN [56] adapt the statistics of BN layers to models during testing for generalization to diverse test-time domains. Regarding module design, there are also some detectors specialized for resisting corruptions, e.g., BtcDet [13] with the occupancy estimator for estimating occluded regions and Centerpoint [57] with key-point detector for a flexible orientation regression. In this article, we evaluate data augmentation, denoising, test-time adaptation methods for improving point cloud detectors against diverse common corruption patterns.

III. BACKGROUND

A. Point Cloud Detection

Point clouds detectors aim to detect objects of interest in point clouds in the format of bounding boxes (BBoxes). Suppose a frame of point cloud data \( P \) is a set of point \( p = [x^p, y^p, z^p, r^p] \), where \( (x^p, y^p, z^p) \) denotes its 3D location and \( r^p \) denotes...
reflective intensity. Thus we can formulate the point cloud detection as:

$$\text{Det}(\mathbf{P}) = \{\mathbf{b}_i\}^N_i$$

$$\mathbf{b}_i = [x_i, y_i, z_i, w_i, h_i, l_i, \theta_i, c_i, s_i]$$

where $\text{Det}(\cdot)$ represents the detector; $N$ is the number of detected BBoxes in $\mathbf{P}$; $\mathbf{b}_i$ denotes $i_{th}$ detected BBox in $\mathbf{P}$, where $i = 1, 2, \ldots, N$; $(x_i, y_i, z_i)$ is the Cartesian coordinate of the center of $\mathbf{b}_i$, $(w_i, h_i, l_i)$ is its dimensions, $\theta_i$ is its heading angle, $c_i$ is its classification label, and $s_i$ is its prediction confidence score.

**Point cloud feature representation:** Representation for features used in point cloud detection includes 2D-view images, voxels, and raw points. By projecting point clouds into a 2D bird’s eye view or front view, 2D-view-based 3D detectors can intuitively fit into a 2D image detection pipeline [35], [36]. However, 2D-view images could lose depth information [2], where the localization accuracy of the detector is affected. To efficiently acquire 3D spatial knowledge in large-scale point clouds, the “voxelization” operation is leveraged to partition unordered points into spatially and evenly distributed voxels [37], [58]. After pooling interior features, those voxels are fed into a sparse 3D convolution backbone [37] for feature abstraction. Given an appropriate voxelization scale, voxel-based representation is computationally efficient, but the quantization loss by voxelization is also inevitable [2]. Different from the above methods, PointNet [59] and PointNet++ [60] directly extract abstract features from raw points, which keeps the integrity of spatial context in point clouds. However, the point-based detectors are not cost-efficient for large-scale data [2]. As a trade-off between the voxel-based and point-based methods, Point-voxel-based representations [41], [42] possess the potential of fusing the high-efficient voxels and accurate-abstract points in feature abstraction.

**Proposal architecture:** One-stage detectors [37], [52] directly generate candidate BBoxes from the abstracted features. To improve candidate BBoxes’ precision, two-stage detectors [13], [41] refine those BBoxes by region proposal network (RPN) and tailor them into unified size by region of interest (RoI) pooling before predicting output BBoxes. Compared with one-stage detectors, two-stage ones [2] usually present more accurate localization but intuitively, are more computationally time-consuming.

**B. Robustness Enhancement Solutions**

Several attempts have been made to enhance the robustness of point cloud detectors. In this article, we explore data augmentation, denoising, and test-time adaptation methods and study their effects on improving point cloud detectors’ robustness against common corruptions. Data augmentation [61] effectively increases the amount of relevant data by slightly modifying existing data or newly creating synthetic data from existing data. Data augmentation on the point cloud [47], [50], [62] provides detectors with a way to be trained with a larger dataset and thus potentially obtain more robust detectors. Considering most detectors commonly adopt the global/scene-level augmentations (e.g., \{random_world_flip, random_world_rotate, random_world_scale\}), we comparatively choose the local/object-level part-aware data augmentation [47] which mixes up 5 basic object-level augmentation methods (i.e., \{datout, swap, mix, sparse, noise\}) in the partitions of an object. We also explore Cutout [63] and Cut-Mix [64] in the 3D point cloud detection, which have been widely applied as 2D image augmentation methods for robustness enhancement. As the data augmentation by extreme samples generated through adversarial attacks has drawn increasing attention, we selected the adversarial shape-deformation augmentation 3D-VField [50] and study its robustness enhancement on point cloud detection.

Different from data augmentation, denoising [53], [54] serves as a pre-process during testing stage to detect and remove spatial outliers in point clouds, which can reduce the effects of noisy point cloud data. Considering not severely degrading the efficiency of detector inference, we choose the common-K-nearest-neighbors-based outlier removing (KNN-OR) [55] of high efficiency (around 0.05 s on each sample), which simply removes the outlier points of over 3 times the standard deviation of distance distribution within the cluster of 50 points. Besides, test-time adaptation methods [56], [65] try to tackle data distribution shifts between training and testing data by adapting models to testing samples during testing time. Considering the test-time adaptation is relatively unexplored in the 3D point cloud detection, we adapt the test-time batch normalization (TT-BN) [56] well-explored in the image domain to point cloud detection, which utilizes testing data to update the statistics of BN layers during testing time.

**IV. PHYSICAL-AWARE ROBUSTNESS BENCHMARK FOR POINT CLOUD DETECTION**

We propose the first robustness benchmark of point cloud detectors against common corruption patterns. We first introduce different corruption patterns collected for this benchmark and dataset in Section IV-A. Then we propose the evaluation metrics used in our benchmark in Section IV-B. Finally, we introduce the subject-object detection methods and robustness enhancement methods selected for this benchmark in Section IV-C.

**A. Physical-Aware Corrupted Dataset Construction**

After the literature investigation in Section II-A, we summarize 25 corruption patterns in Table II and categorize them into 4 categories based on presentations of common corruptions: weather, noise, density, and transformation. On the other hand, we also divide common corruption patterns into the scene-level and the object-level. As an initial effort, the dataset covers representative but not all corruptions, and we encourage continuous work with more diverse corruptions considered in the future.

The simulation of corruptions implemented in the article mainly operates on the spatial locations and the reflection intensity of points in the point cloud. Those point-targeting operations are equivalent to the perturbations of the real-world corruptions on the LiDAR point cloud and have been widely utilized in the simulation-related studies, as in noise-related [11], [18], [24], [25], [47], density-related [11], [13], [18], [47], [66], [66].
and transformation-related [11], [18], [28], [30], [66]. Next, we briefly introduce each corruption pattern in the following (refer to Supplementary C for detailed implementations and visualizations).

**Weather corruptions:** LiDAR is sensitive to adversarial weather conditions, such as rainy, snowy, and foggy [9]. Dense droplets of liquid or solid water dim the reflection intensity and reduce the signal-to-noise ratio (SNR) of received lights. Floating droplets can also reflect and fool sensors with false alarms. Both effects, in some cases, can significantly affect the detectors. To simulate three weather corruptions: \{rain, snow, fog\}, we adopt LiDAR light scattering augmentation (LISA) [12] as a simulator for rain and snow and LiDAR fog stimulation (LFS) [49] as a fog simulator.

Unlike other types of corruptions which are relatively simply implemented and widely applied in LiDAR-based point cloud studies, the mechanism of weather simulation on point clouds complicately involved interaction between LiDAR lights and dense droplets. Since there is a lack of the realness verification study in [12], [49], we conduct experiments to further verify the realness/naturalness of weather simulators for a convincing benchmarking under weather-relevant corruptions. Specifically, we train weather-oriented PointNet-based classifiers with datasets collected in real snowy and foggy weather. Then, we leverage the classification accuracy of those trained classifiers testing on simulated data to measure the similarity of simulated data to real data. As shown in Table S12 in Supplementary B, the testing accuracy 97.13% and 92.60% of trained weather classifiers on simulated snow data and fog data show that the simulated snow and fog weather conditions as a fog simulator.

**Density corruptions:**

**Noise corruptions:**

**Transformation corruptions:**

**Object-level factors, dark-color covers (e.g., glasses and plastics) of objects can affect the point density of objects. Hence, at the object level, we also propose 5 corruptions: \{local_dec, local_inc\} locally decrease or increase the density of points; \{beam_del, layer_del\} randomly delete points or layers of points in point clouds. In terms of object-level factors, dark-color cover [22] and transparent materials (e.g., glasses and plastics) of objects can affect the point density of objects. Hence, at the object level, we also propose a set of corruptions: \{cutout, local_dec, local_inc\}, affecting the point density of objects.

**Transformation corruptions:** In the scenario of autonomous driving, shapes of objects within one class could be various (e.g.,

---

**TABLE II**

**Taxonomy of Collected Common Corruption Patterns**

| Corruption Category | Sub-categorization | Potential Reasons |
|---------------------|-------------------|------------------|
| Weather             | rain, snow, fog   | Environment: natural weather [9]; |
| Noise               | uniform_rad, gaussian_rad, impulse_rad, upsample | Environment: strong illumination [23]; |
| Density             | cutout, local_dec, local_inc, beam_del, layer_del | Sensor: different scanning layers, object occlusion [13], and random signal loss [27] |
| Transformation      | translation, rotation, shear, FFD, scale | Object deformation: bending or moving pedestrians [29], different styles of vehicles [30]; |

---

**Noise corruptions:** Noise commonly exists in point cloud data due to strong illumination [23], limited ranging accuracy of sensors, and sensor vibration [25], [26]; could increase the variance of ranging or extend the positioning bias. Floating particles, e.g., dust [27], could cause the background noise in point clouds. Hence, we collect 5 scene-level noise corruptions: \{uniform_rad, gaussian_rad, impulse_rad\} add uniform, Gaussian, impulse noise on the spherical coordinates of points; \{upsample\} randomly samples points nearby original points; \{background\} uniformly randomly samples points within the spatial range of point clouds. Besides scene-level effects, object-related factors could cause noise in LiDAR points, e.g., dark color [22] and coarse surface. Thus, we formulate 4 object-level corruptions: \{uniform, gaussian, impulse\} add uniform, Gaussian, impulse noise on the Cartesian coordinates of points of objects; \{upsample\} upsample points nearby original points of objects.

**Density corruptions:** The density-related corruptions refer to the corruption patterns that change the global or local density distribution of LiDAR point clouds. For instance, the global static density of points in LiDAR varies due to different amounts of scanning layer (e.g., 32 or 64). Besides, inter-object occlusion and random signal loss [13] could remove points randomly. We hence propose 5 corruptions: \{cutout\} cuts out the sets of locally gathering points; \{local_dec, local_inc\} locally decrease or increase the density of points; \{beam_del, layer_del\} randomly delete points or layers of points in point clouds. In terms of object-level factors, dark-color cover [22] and transparent materials (e.g., glasses and plastics) of objects can affect the point density of objects. Hence, at the object level, we also propose a set of corruptions: \{cutout, local_dec, local_inc\}, affecting the point density of objects.

---

**AUTHOR:** IEEE TRANSACTIONS ON MULTIMEDIA, VOL. 27, 2025
flat sports cars and round vintage cars [30], bending and walking pedestrians [29]). Those long-tail data could potentially be recognized wrong. Besides, dynamic changes in heading directions and locations of objects [28] could potentially affect the positioning accuracy of detectors. Hence, we formulate 5 corruptions: {translation, rotation} change locations and heading directions of objects to a milder degree, i.e., < 1 m and < 10°; {shear} [68] and {scale}, as linear deformations, slant and scale points of objects; {FFD} adopts free-form deformation (FFD) [69] to distort the point shape of an object in a nonlinear manner.

**Dataset selection:** As one of the most popular benchmarks in autonomous driving, KITT [6] contains 7481 training samples covering 8 object classes. Unlike other large-scale datasets including weather and other corruptions in Table I, the data in KITT are mostly collected under clean conditions and also have a relatively simple annotation format, which makes it a good option for conducting comparative experiments. We also encourage the future extension to other real or synthesized datasets. To simulate various levels of severity in the real world, we set 6 severity levels for each corruption (considering “clean” as level 0).

### B. Evaluation Metrics

To quantify the robustness performance of detectors, we design the following evaluation metrics from two perspectives: 1) detection accuracy and 2) number of bugs triggered.

**Overall accuracy.** For each test, we use the overall accuracy (OA), by taking the average of APs (average precision) at three difficulty levels (i.e., “Easy”, “Moderate”, and “Hard”). And we follow the common settings of IoU thresholds (Car: 0.7, Pedestrian: 0.5, Cyclist: 0.5) to search for the true positive detections in AP and recall calculation. For every corruption, we calculate corruption error (CE) to measure performance degradation according to OA by:

$$\text{CE}_{c,s}^m = \text{OA}_{c,s}^m - \text{OA}_{c,s}^\text{clean}$$

where $\text{OA}_{c,s}^m$ is the overall accuracy of detector $m$ under corruption $c$ of severity level $s$ (exclude “clean”, i.e., severity level 0) and $\text{clean}$ represent the clean data. For detection $m$, we can calculate the mean CE (mCE) over $C$ corruptions by:

$$\text{mCE}_c^m = \frac{\sum_{s=1}^5 \sum_{c=1}^C \text{CE}_{c,s}^m}{5C}$$

**Detection bug:** There are various bugs existing in the pipeline of point cloud detection, such as annotation errors, run-time errors, detection bugs. In this article, we focus on the bugs in detection results. Specifically, we’re interested in false detection, false classification, and missed detection:

- False detection (FD) on detection BBoxes: maximum IoU $> 0$ with correct classification w.r.t. ground-truth BBoxes;
- False classification (FC) on detection BBoxes: maximum IoU $> 0$ with false classification w.r.t. ground-truth BBoxes;
- Missed detection (MD) on detection BBoxes: maximum IoU $= 0$ w.r.t. ground-truth BBoxes.

Correspondingly, the bug rates (BRs) are calculated by:

$$\text{BR}_s = \frac{N_s}{N_{\text{det}}}$$

where $s$ stands for FD, FC, and MD; $N_s$ is the number of objects of $s$; $N_{\text{det}}$ is the number of detected objects.

To measure the increase of BR after being affected by common corruptions, we calculate corruption risk (CR) and the mean CR (mCR) for detector $m$ by

$$\text{CR}_{c,s}^m = \text{BR}_{c,s}^m - \text{BR}_{c,s}^\text{clean}$$

$$\text{mCR}_c^m = \frac{\sum_{s=1}^5 \sum_{c=1}^C \text{CR}_{c,s}^m}{5C}$$

where $\text{BR}_{c,s}^m$ is the BR of detector $m$ under corruption $c$ of severity level $s$ and $C$ for the number of corruptions.

### C. Benchmark Subjects

**Point cloud detectors:** For benchmarking point cloud detection, we select 12 representative detectors: SECOND [37], PointRCNN [39], PartA2 [70] PVRCNN [41], PVRCNN++ [71], BtcDet [13], VoTr-SSD, VoTr-TSD [38], Centerpoint [57], Centerpoint_RCNN [57], Centerformer [72], and SE-SSD [73], to cover different kinds of feature representations and proposal architectures. We show the detailed taxonomy in Table V. For a more fair comparison, based on the robust Centerpoint detector and its two-stage version Centerpoint_RCNN, we construct 5 versions of Centerpoint detectors for ablative comparison, as in Table VI, covering 3 types of feature representations and 2 types of proposal architectures. Note that, as in [41], [71], the point-voxel-based feature extraction needs the two-stage proposal structure, where the multi-scale voxel-based features are extracted in the first-stage detection, and the point-voxel-based features combining raw points and multi-scale voxel-based features (from the first stage) are extracted for the second-stage refinement.

**Data augmentation, denoising, and test-time adaptation methods:** In this article, we study the effectiveness of data augmentation, denoising, and test-time adaptation methods for improving detectors’ robustness against corruption. As discussed in Section III-B, for data augmentation, we choose methods of part-aware data augmentation (PA-DA) [47], Cutout [63], CutMix [64], and 3D-VField [50]. For denoising, we adopt K-nearest-neighbors-based outlier removing (KNN-OR) [55] to remove the outliers out with 3 times the standard deviation of distance distribution within the cluster of 50 points. For test-time adaptation, we select the test-time batch normalization (TT-BN) [56] to update statistics of BN layers during testing time. Apart from them, we also augment train data with different corruption categories (Weather, Noise, Density, Transformation) by means of our physical-aware simulation tools to explore the robustness enhancement of the category-oriented augmentation.
TABLE III
AP(%) of ALL DETECTORS UNDER CLEAN OBSERVATIONS (AT THE SEVERITY LEVEL OF 0)

| Category     | PVR-CNN | PointRCNN | PartA2 | SECOND | BicDet | VoTr-SSD | VoTr-SSD | SSRDSS | Centerpoint | PVR-CNN++ | Centerpoint | RCNN |
|--------------|---------|-----------|--------|--------|--------|----------|----------|--------|-------------|-----------|-------------|------|
| Car          | 86.77   | 82.82     | 85.36  | 83.67  | 87.32  | 81.04    | 86.39    | 86.44  | 82.14        | 85.18    | 86.07       | 79.80|
| Pedestrian   | 60.61   | 52.34     | 59.68  | 52.15  | -      | -        | -        | -      | -            | -        | -           | -    |
| Cyclist      | 76.42   | 77.60     | 80.09  | 68.51  | -      | -        | -        | -      | -            | -        | -           | -    |

TABLE IV
CE-AP(%) of DIFFERENT DETECTORS UNDER DIFFERENT CORRUPTIONS ON CAR DETECTION (THE GREEN CELL STANDS FOR THE LOWEST CE-AP AMONG DETECTORS GIVEN A CERTAIN CORRUPTION AND THE YELLOW CELL FOR THE AVERAGE mCE-AP)

| Corruption   | Point-voxel | Point | Voxel | PartA2 | PointRCNN | SECOND | BicDet | VoTr-SSD | VoTr-SSD | SSRDSS | Centerpoint | PVR-CNN++ | Centerpoint | RCNN |
|--------------|-------------|-------|-------|--------|-----------|--------|--------|----------|----------|--------|-------------|-----------|-------------|------|
| Weather      | 25.11       | 26.05 | 25.13 | 23.31  | 24.44     | 21.81  | 31.07  | 28.17    | 26.77    | 29.51  | 25.83       | 26.65     | 25.98       | 21.9  |
| Noise        | 10.19       | 6.69  | 8.26  | 8.32   | 10.45     | 9.51   | 9.13   | 3.79     | 4.11     | 9.34   | 8.15        | 5.98      | 7.83        | 2.19 |
| Density      | 3.75        | 3.77  | 3.58  | 3.97   | 3.66      | 4.27   | 3.99   | 4.51     | 3.59     | 4.26   | 4.11        | 3.98      | 3.98        | 0.75 |

V. EXPERIMENTS AND ANALYSIS

A. Experimental Set-Ups

For a fair comparison, each detector in Table V is trained with the clean training set of KITTI, following the training strategy in each article, and evaluated with corrupted validation sets of KITTI. All detectors go through the training of 80 epochs among which the best checkpoint is selected by metrics of mAP. All detectors are executed based on the open-source codes released on GitHub [74], [75], where the configuration files and pre-trained checkpoints can be found. The experiments are executed on the NVIDIA RTX A6000 GPU with a memory of 48 GB. The batch size of each detector is optimized to reach the limit of GPU memory. In robustness enhancement experiments, we first follow the recommended setting “dropout_p02_swap_p02_mix_p02_sparse40_p01_noise 10_p01” in [47] to adopt PA-DA to augment the clean train set. For CutMix, we follow the settings (i.e., “swap_p10”) in [47] to implement the augmentation. For Cutout, we utilize the cutout in our corruption simulation toolkit with the randomly selected severity level for each sample. For 3D-VField augmentation, we follow the settings in [50]. The “augmented + clean” set (2x3712 samples) is used for all augmentation methods. For a fair comparison, the category-oriented data augmentation also builds “augmented + clean” set from the clean train set. For each basic corruption category {Weather, Noise, Density, Transformation} or overall corruptions, we augment every sample of KITTI train set with a randomly selected corruption at a randomly selected severity level. By means of the GPU-accelerated tool remove_statistical_outlier of the package “open3d”, KNN-RO is implemented to denoise the val data during the detection inference. By modifying the parameters “running_mean” and “running_val” of PyTorch-based BN layers during testing, we implement the TT-BN to update the statistics of BN based on testing data.

Note that, since only detection of “Car” is available for all detectors, as shown in Table III, the following evaluation will mainly focus on detected results in the “Car” category. We encourage readers to refer to Supplementary A for complete evaluation results, e.g., about “Pedestrian”.

B. Effects of Common Corruptions to Point Cloud Detectors

How do different corruptions affect detectors’ overall accuracy? As shown in the yellow cell in Table IV, the average mCE-AP of 10.94% anticipates a noticeable accuracy drop of detectors against diverse corruption patterns. It suggests an urgent
need of addressing the point cloud detector’s robustness issue. Specifically, \{rain, snow\} and \{shear, FFD\} corruptions have the AP loss of more than 20% (last column in Table IV), which presents a serious degradation of detection accuracy. By contrast, scene-level and object-level \{upsample, scene-level beam_del, object-level rotation\} show less effects on detectors (\(CE_{\text{AP}}\) less than 1%). It demonstrates that upsampling noise, sparse beam loss, and slight rotation affect detectors’ accuracy slightly.

Besides, as shown in Table S6 in Supplementary A, the recall metric performs similarly to AP, as the serious recall loss of over 22% on \{rain, snow, shear\}. In addition, object-level \{cutout, local_dec, FFD\} present an unignorable drop of recall within [18%, 22%].

How do corruption severity levels affect detectors’ overall accuracy? We find almost all common corruptions have a predictable trend, i.e., each corruption’s \(CE_{\text{AP}}\) increases as its severity level increases (see Table S1 in the Supplementary A). The only exception is rain, \(CE_{\text{AP}}\) of which remain around 26% regardless of the severity level. There is the plausible explanation: (1) by statistics, we find, due to the unpromising laser reflections of car surfaces, 58.94% of points of “Car” in KITTI have zero-value reflection intensity, and those points are easily filtered out by the rain droplets, causing a noticeable AP drop, and (2) noise points reflected by rain droplets at different severities are sparse (see Figure S3 in the Supplementary B) so that the slight effect of noise points on detection is covered by the randomness of severe influence of removing points with zero-value reflection, which makes the AP drop seemingly unaffected by the rain severity. Interestingly, as for “Pedestrian”, only 10.01% of points have zero-value reflection intensity, causing a slight influence on detection. Hence, the effect of noise points by rain appears and shows a normal trend as the severity level increases (see Table S15 in the https://sites.google.com/ualberta.ca/robustness1pc2detector/Supplementary Website).

C. Reacts of Detector Designing to Common Corruptions

How do different representations affect detectors? As shown in Fig. 1, voxel-based Centerformer and BtcDet record the lowest and highest \(CE_{\text{AP}}\). For most detectors (i.e., except for PointRCNN), \(mCE_{\text{AP}}\) approximately increases as AP increases, indicating an potential trade-off between accuracy and robustness against common corruptions.

We also find that, as shown in Table IV, for most Weather, Noise, and Density-related corruptions, voxel-based methods are generally more robust against corruption patterns. For a more fair comparison between different input representations, we conduct the ablative experiments on the robust Centerpoint. As shown in Table VI, under any given proposal structure, the voxel-based Centerpoint detectors perform more robustly against most Weather, Noise, and Density corruptions and overall corruptions by presenting a lower \(CE_{\text{AP}}\) or \(mCE_{\text{AP}}\).
w.r.t. other detectors. One plausible explanation is that the spatial quantization of a group of neighbor points by voxelization mitigates the local randomness and the absence of points caused by noise and density-related corruptions.

Specifically, for severe corruptions (e.g., shear, FFD in the Transformation), the point-voxel-based methods are more robust. The point-based PointRCNN and PartA2 don’t have performance superiority against most corruptions (except \{scale\}), suggesting potential limitations.

**How do different proposal architectures affect detectors?**

As shown in Fig. 1, two-stage detectors perform less robustly against overall corruptions compared to one-stage detectors, showing a higher \(mCE_{AP}\). Also, as shown in Table VI, under a given representation, the one-stage detectors perform more robustly under overall corruptions, presenting a lower \(mCE_{AP}\). One possible cause is that corrupted data could affect the proposal generation of stage 1 (for two-stage detectors and one-stage ones), and the low-quality proposals significantly affect the BBox regression of stage 2 (only for two-stage detectors).

Specially, as shown in Table S2 in the Supplementary A, one-stage detectors perform more robustly against corruptions of scene-level and object-level Noise, object-level Density, and Transformation, while two-stage detectors are mainly more robust against scene-level Density. As for Weather corruptions, one-stage detectors present better robustness on \{snow, fog\} and two-stage detectors work better under corruptions of \(rain\).

**D. Detection Bugs in Detectors Under Common Corruptions**

**How do different corrupted inputs trigger bugs in detectors?**

We find that the rate of false classification (FC) against common corruption patterns is relatively small, where the average \(CR_{FC}\) is only 0.26\% (refer to Table S9 in Supplementary A). By contrast, the increase of false detection (FD) rate is relatively obvious, by the average \(CR_{FD}\) of 3.19\% (refer to Table S10 in Supplementary A). Regarding missed detection (MD) (refer to Table S11 in Supplementary A), scene-level \{background\} and object-level \{cutout, local dec\} result in an increase of MD rate of more than 4\%.

Surprisingly, according to Table S4 in the Supplementary A, \{rain, snow\} and scene-level \{uniform_rad, gaussian_rad\} even reduce the rate of missing objects. One plausible explanation for this observation is that milder noise points offer a better knowledge of the shape of some objects to detectors, but positioning on those objects is not accurate since the rate of false detection increases (more details in Table S10 and S11 in Supplementary A). Also, we find that, as shown in Fig. S1 in Supplementary A, compared to clean observations, TD rates under corrupted observations are always lower at any distance of objects to LiDAR.

**How do corrupted inputs trigger bugs in different detectors?**

In general, as shown in Table S3 in Supplementary A, most of the detectors perform relatively stable in terms of false classification rates and missed detection rates against corruptions. In contrast, affected by corruptions, all detectors have increasing false detection rates (see Table S3), revealing a serious bias in BBox localization.

**E. Robustness Enhancement Evaluation**

**How do PA-DA and KNN-based outlier-removing affect detectors’ robustness against different corruptions?**

Shown by Table S7 in Supplementary A, the average \(C_{AP}\) with PA-DA slightly decreased to 10.75\% compared to the average \(C_{AP}\) without PA-DA, which still poses serious robustness issues for point cloud detectors. As shown in Tables VII and S7, PA-DA shows a better but still limited robustness improvement in the object-level Noise and Density. The existence of improvement is reasonable since PA-DA involves exactly object-level noise and density-related simulation during augmentation. However, since PA-DA only involves 5 limited basic augmentations at only one severity level, its robustness improvements on detectors, even under object-level noise and density corruptions, are limited. Regarding the denoising strategy, the average \(C_{AP}\) after adopting KNN-RO increases to 13.45\% without PA-DA and 13.22\% with PA-DA (refer to Table S7). These results indicate that KNN-RO might not be capable of enhancing point cloud detectors’ robustness in \(Car\) detection. However, we find that KNN-RO slightly improves the robustness of \(Pedestrian\) detection by decreasing the \(C_{AP}\) by 0.37\% without PA-DA and 0.89\% with PA-DA (Table S8 in Supplementary A). The robustness improvement on \(Pedestrian\) is reasonable due to KNN-RO’s removing spatial outliers on the background and objects. But what causes the performance drop on \(Car\) detection? By investigating KITTI, we found, 15.32\% of \(Car\) BBoxes have scanning points of less than 20, higher than 7.14\% of \(Pedestrian\) BBoxes. Such relatively few points within a \(Car\) BBox (larger than \(Pedestrian\)) are distributed more sparsely and thus easier removed by KNN-RO, causing a missing of some parts of the car or even the whole car (refer to Figure S9 in the https://sites.google.com/ualberta.ca/robustness1pc2detector/Supplementary). It significantly facilitates KNN-RO’s damage on the point imaging of cars and further the detection accuracy on \(Car\) detection. Actually, after investigation, 72.14\% of objects affected by KNN-RO.
are cars while only 0.94% of them are pedestrians, which verifies much more serious effects of KNN-RO on cars.

**How do PA-DA and KNN-based outlier-removing affect different detectors’ robustness against corruption?** Except for PVRCNN, SE-SSD, Centerpoint, and Centerformer, other detectors perform more robustly against corruption patterns after adopting PA-DA (refer to Fig. S2 in Supplementary A). Moreover, PointRCNN and VoTr-SSD increase their AP by 1.16% and 2.46% after adopting PA-DA, respectively. According to Fig. S2, KNN-RO degrades AP metric for all detectors, presenting no improvement on the robustness of any detector on Car detection. However, adopting KNN-RO slightly improves the AP by 0.37% without PA-DA and 0.89% with PA-DA on Pedestrian detection, respectively (refer to Table S5 in Supplementary A). It illustrates again that compared with on Car, KNN-RO is more effective in removing perturbations caused by corruptions on Pedestrian.

**How does the category-oriented data augmentation against different corruption categories?** As shown in Table VII, as augmentation methods relevant to the object-level Density, Cutout and CutMix present the limited AP increases of 1.42% and 0.44% on the object-level Density, respectively. It is mainly because Cutout involves one limited corruption and CutMix only augments data at the fixed severity level. Surprisingly, compared with the AP increase of 0.05% on Transformation, shape-deformation-related 3D-VField presents a significant AP increase of 4.68% on object-level Noise, revealing a high relevance of its adversarial deformation to Noise-related corruptions (refer to Figure S10 in [https://sites.google.com/ualberta.ca/robustness1pc2detector/Supplementary]). Compared with original detectors, detectors with TT-BN perform worse under all corruption categories. One plausible explanation is, due to the large-scale input samples under the autonomous driving scenarios, the relatively small batch size (less than 64 samples per batch) causes unstable statistics (i.e., the mean and standard deviation) for BN layers and thus cause unstable and biased detection.

As shown in Table VII, we found that the augmentation by all corruptions reaches the lowest $CE_{AP}$ of 7.69% (i.e., an AP increase of 3.24% than Original) under overall corruptions. For each corruption category, the detectors augmented by the corresponding basic corruption category show a significant improvement (i.e., an obvious $CE_{AP}$ drop) compared with the other robustness enhancement methods. By a simple ensemble of the best results under each category, the category-oriented data augmentation significantly decreases the overall $CE_{AP}$ to 6.16%, revealing the great potential for corruption-oriented augmentations on robustness enhancement.

**VI. DISCUSSIONS**

According to detailed findings in the empirical study of Section V, we summarize the following insights as guidelines for future robustness enhancement studies:

**Insight 1.** Common corruptions related to natural weather and shape transformation significantly challenge the point cloud detectors.

**Insight 2.** Regarding the input feature representation of detectors, the voxel-based detectors perform more robustly against common corruptions than point-based detectors, especially for most Weather, Noise, and Density corruptions.

**Insight 3.** Regarding the proposal structures of detectors, one-stage detectors comprehensively perform more robustly against overall corruptions than two-stage ones.

**Insight 4.** As for the detection results, corruptions more commonly cause a severe bias in BBox localization rather than erroneous object classification.

**Insight 5.** For the existing robustness enhancement methods, the explored augmentation methods only work on limited corruption categories; KNN-RO causes precision damage on Car detection; TT-BN easily degrades detection accuracy due to the limitation of batch size. To contrast, corruption-category-oriented augmentation shows great potential in robustness enhancement.

**VII. CONCLUSION**

In this article, we propose the first physical-aware robustness benchmark of point cloud detection against common corruption patterns, which contains a total of 1,122,150 examples covering 25 common corruption types and 6 severity levels. Based on the benchmark, we conduct extensive empirical studies on 12 detectors covering 6 different detection frameworks and reveal the vulnerabilities of the detectors. Moreover, we further study the effectiveness of existing robustness enhancement methods of data augmentation, denoising, and test-time adaptation and find them limited, calling for more research on robustness enhancement. We hope this benchmark and empirical study results can guide future research toward building more robust and reliable point cloud detectors.

In the future, we plan to extend the corruption simulation on more large-scale datasets (e.g., NuScenes [7], Waymo [8], and ONCE [19]) for a more extensive and comprehensive benchmark on robust point cloud detection. However, large-scale datasets with a wider range of locations and times of collection contain more LiDAR observations under corrupted conditions (e.g., rain, snow, rare cars, and other corruptions) as shown in Table I. Thus, one challenge of extending into large-scale datasets is to erase the effects of original corruptions to make data more controllable to simulate corruptions individually. Also, we plan to extend our work into a robustness benchmark involving multi-modal point cloud detection (including e.g., images or videos). One of the main challenges is to design the physical-aware corruption models consistent in both point clouds and images for the high-fidelity simulation of diverse common corruptions.

**REFERENCES**

[1] Y. Guo et al., “Deep learning for 3D point clouds: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 12, pp. 4338–4364, Dec. 2021.

[2] R. Qian, X. Lai, and X. Li, “3D object detection for autonomous driving: A survey,” Pattern Recognit., vol. 130, p. 108796, 2022.

[3] C. Badue et al., “Self-driving cars: A survey,” Expert Syst. Appl., vol. 165, 2021, Art. no. 113816.

[4] E. Arnold et al., “A survey on 3D object detection methods for autonomous driving applications,” IEEE Trans. Intell. Transp. Syst., vol. 20, no. 10, pp. 3782–3795, Oct. 2019.
[5] D. Fernandes et al., “Point-cloud based 3D object detection and classification methods for self-driving applications: A survey and taxonomy,” Inf. Fusion, vol. 68, pp. 161–191, 2021.

[6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” Int. J. Robot. Res., vol. 32, no. 11, pp. 1231–1237, 2013.

[7] H. Caesar et al., “nuScenes: A multimodal dataset for autonomous driving,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11621–11631.

[8] P. Sun et al., “Scalability in perception for autonomous driving: Waymo open dataset,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11682–11692.

[9] R. H. Rasshofer, M. Spies, and H. Spies, “Influences of weather phenomena on automotive laser radar systems,” Adv. Radio Sci., vol. 9, pp. 49–60, 2011.

[10] M. Bielcic et al., “Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11682–11692.

[11] J. Sun et al., “Benchmarking robustness of 3D point cloud recognition against common corruptions,” 2022, arXiv:2201.12296.

[12] V. Kilic et al., “LiDAR light scattering augmentation (LISA): Physics-based simulation of adverse weather conditions for 3D object detection,” 2021, arXiv:2107.07004.

[13] X. Yu, Y. Zhong, and U. Neumann, “Behind the curtain: Learning occluded shapes for 3D object detection,” Proc. AAAI Conf. Artif. Intell., vol. 36, no. 3, pp. 2893–2901, 2022.

[14] M. Pitropov et al., “Canadian adverse driving conditions dataset,” Int. J. Robot. Res., vol. 40, no. 4/5, pp. 681–690, 2021.

[15] K. Burnett et al., “Boreas: A multi-season autonomous driving dataset,” Int. J. Robot. Res., vol. 42, no. 1/2, pp. 33–42, 2023.

[16] D. Hendrycks and T. Dietterich, “Benchmarking neural network robustness to common corruptions and perturbations,” in Proc. Int. Conf. Learn. Representations, 2019. [Online]. Available: https://openreview.net/forum?id=H2zf6rCqYm

[17] A. Barbu et al., “ObjectNet: A large-scale bias- controlled dataset for pushing the limits of object recognition models,” in Proc. Adv. Neural Inf. Process. Systems, 2021.

[18] J. Ren, L. Pan, and Z. Liu, “Benchmarking and analyzing point cloud classification under corruptions,” in Proc. Int. Conf. Mach. Learn., 2022, pp. 18559–18575.

[19] J. Mao et al., “One million scenes for autonomous driving: Once dataset,” in Proc. NeurIPS Datasets Benchmarks Track, 2021.

[20] M.-F. Chang et al., “Argoverse: 3D tracking and forecasting with rich maps,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 8748–8757.

[21] R. Kesten et al., “Level 5 perception dataset 2020,” 2019. [Online]. Available: https://level-5-global/leval5/data/

[22] D. Bolkas and A. Martinez, “Effect of target color and scanning geometry on point cloud-noise and plane fitting,” J. Appl. Geodesy, vol. 12, no. 1, pp. 109–127, 2018.

[23] F. Villa, F. Severini, F. Madonini, and F. Zappa, “SPADs and siPMs arrays: A novel point cloud data augmentation method,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 11025–11038.

[24] Y. Zhu et al., “Adversarial attacks against LiDAR semantic segmentation in autonomous driving,” in Proc. 19th ACM Conf. Embedded Networked Sensor Syst., 2021, pp. 329–342.

[25] B. Li, T. Zhang, and T. Xia, “Vehicle detection from 3D LiDAR using fully convolutional network,” in Proc. Robot.: Sci. Syst. XII, 2016.

[26] B. Yang, W. Luo, and R. Urtasun, “PIXOR: Real-time 3D object detection from point clouds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 7652–7660.

[27] Y. Yan, Y. Mao, and B. Li, “Second: Sparsely embedded convolutional sensors,” Sensors, vol. 18, no. 10, 2018, Art. no. 3337.

[28] J. Mao et al., “Voxelformer for 3D object detection,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 3164–3173.

[29] S. Shi, X. Wang, and H. Li, “PointRCNN: 3D object proposal generation and detection from point cloud,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 770–777.

[30] Z. Yang, Y. Sun, S. Liu, and J. Jia, “3DSSD: Point-based 3D single stage object detector,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11040–11048.

[31] S. Shi et al., “PV-RCNN: Point-voxel feature set abstraction for 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 10529–10538.

[32] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, “Structure aware single-stage 3D object detection from point cloud,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11873–11882.

[33] J. Deng et al., “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2009, pp. 248–255.

[34] Z. Wu et al., “3D ShapeNets: A deep representation for volumetric shapes,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 1912–1920.

[35] J. Zhang et al., “Pointcutmix: Regularization strategy for point cloud classification,” Neurocomputing, vol. 505, pp. 58–67, 2022.

[36] D. Lee et al., “Regularization strategy for point cloud via rigidly mixed sample,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 15900–15909.

[37] J. Choi, Y. Song, and N. Kwak, “Part-aware data augmentation for 3D object detection in point cloud,” in Proc. IEEE/CVF Int. Conf. Intell. Robots Syst., 2021, pp. 3391–3397.

[38] J. Fang et al., “LiDAR-Aug: A general rendering-based augmentation framework for 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 4710–4720.

[39] M. Hahner, C. Sakaridis, D. Dai, and L. Van Gool, “Fog simulation on real LiDAR point clouds for 3D object detection in adverse weather,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Pattern Recognit., 2021, pp. 15283–15292.

[40] A. Lehner et al., “3D-VField: Adversarial augmentation of point clouds for domain generalization in 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 17305–17304.

[41] X. Yu et al., “Point-BERT: Pre-training 3D point cloud transformers with masked point modeling,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 1913–1932.

[42] Z. Zhang, R. Girdhar, A. Joulin, and I. Misra, “Self-supervised pretraining of 3D features on any point cloud,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 10252–10263.

[43] Y. Duan, C. Yang, H. Chen, W. Yan, and H. Li, “Low-complexity point cloud denoising for LiDAR by PCA-based dimension reduction,” Opt. Commun., vol. 482, 2021, Art. no. 126567.

[44] X. Ning, F. Li, G. Tian, and Y. Wang, “An efficient outlier removal method for scattered point cloud data,” Flows One, vol. 13, no. 8, 2018, Art. no. e0201280.

[45] A. Carrilho, M. Golo, and R. Santos, “Statistical outlier detection method for airborne LiDAR data,” Int. Arch. Photogrammetry Remote Sens. Spatial Inf. Sci., vol. 42, pp. 87–92, 2018.

[46] S. Schneider et al., “Improving robustness against common corruptions by covariate shift adaptation,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 11559–11567.

[47] T. Yin, X. Zhou, and P. Krahenbuhl, “Center-based 3D object detection and tracking,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 11784–11793.

[48] M. Ye et al., “HVNet: Hybrid voxel network for LiDAR based 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 5049–5058.

[49] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 652–660.
C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep hierarchical feature learning on point sets in a metric space,” in Proc. Adv. Neural Inf. Process. Syst., vol. 30, 2017, pp. 5105–5114.

D. A. V. Dyk and X.-L. Meng, “The art of data augmentation,” J. Comput. Graphical Statist., vol. 10, no. 1, pp. 1–50, 2001.

Y. Chen et al., “PointMixup: Augmentation for point clouds,” in Proc. 16th Eur. Conf. Comput. Vis., 2020, pp. 330–345.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” J. Mach. Learn. Res., vol. 15, no. 1, pp. 1929–1958, 2014.

S. Yun et al., “CutMix: Regularization strategy to train strong classifiers with localizable features,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 6023–6032.

D. Chen, D. Wang, T. Darrell, and S. Ebrahimi, “Contrastive test-time adaptation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 295–305.

S. Cheng et al., “Improving 3D object detection through progressive population based augmentation,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 279–294.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, “A kernel method for the two-sample-problem,” in Proc. Adv. Neural Inf. Process. Syst., vol. 19, 2006, pp. 513–520.

B. Chen and A. Kaufman, “3D volume rotation using shear transformations,” Graphical Models, vol. 62, no. 4, pp. 308–322, 2000.

T. W. Sederberg and S. R. Parry, “Free-form deformation of solid geometric models,” in Proc. 13th Annul. Conf. Comput. Graph. Interactive Techn., 1986, pp. 151–160.

S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, “From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 8, pp. 2647–2664, Aug. 2021.

S. Shi et al., “PV-RCNN: Point-voxel feature set abstraction with local vector representation for 3D object detection,” Int. J. Comput. Vis., vol. 131, no. 2, pp. 531–551, 2023.

Z. Zhou, X. Zhao, Y. Wang, P. Wang, and H. Foroosh, “Centerformer: Center-based transformer for 3D object detection,” in Proc. Comput. Vis.: 17th Eur. Conf., 2022, pp. 496–513.

W. Zheng, W. Tang, L. Jiang, and C.-W. Fu, “Se-SSD: Self-ensembling single-stage object detector from point cloud,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 14494–14503.

O. D. Team, “OpenPCDet: An open-source toolbox for 3D object detection from point clouds,” 2020. [Online]. Available: https://github.com/openmmlab/OpenPCDet

B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu, “Class-balanced grouping and sampling for point cloud 3D object detection,” 2019, arXiv:1908.09492.

L. V. D. Maaten and G. Hinton, “Visualizing data using t-SNE,” J. Mach. Learn. Res., vol. 9, no. 11, pp. 2579–2605, 2008.

Shuangzhi Li (Graduate Student Member, IEEE) received the B.S. degree in electronic science and technology and the M.S. degree in electronic science and technology from the Beijing University of Posts and Telecommunications, Beijing, China, in 2018 and 2021, respectively. He is currently working toward the Ph.D. degree in software engineering and intelligent system with the University of Alberta, Edmonton, AB, Canada. His research interests include the robustness of AI systems and 3D computer vision.

Zhijie Wang (Student Member, IEEE) received the B.E. degree in telecommunication engineering from the Beijing University of Posts and Telecommunications, Beijing, China, and the M.E. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada. He is currently working toward the Ph.D. degree in software engineering with the University of Alberta, Edmonton, AB, Canada. He is also affiliated with Alberta Machine Intelligence Institute (AMII). His research interests include quality assurance of AI systems and human-computer interaction in AI software development.

Xingyu Li (Member, IEEE) received the B.Sc. degree from Peking University, Beijing, China, the M.Sc. degree from the University of Alberta, Edmonton, AB, Canada, and the Ph.D. from the University of Toronto, Toronto, ON, Canada. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering with the University of Alberta, and an Alberta Machine Intelligence Institute (AMII) Fellow. Her research interests include machine learning, computer vision, AI for health, and multidimensional data analytics and understanding.

Lei Ma (Member, IEEE) received the B.E. degree from Shanghai Jiao Tong University, Shanghai, China, and the M.E. and Ph.D. degrees from The University of Tokyo, Tokyo, Japan. He is currently an Associate Professor with The University of Tokyo, and the University of Alberta, Edmonton, AB, Canada. He was honorably selected as a Canada CIFAR AI Chair and Fellow with Alberta Machine Intelligence Institute (AMII). His recent research interests include the interdisciplinary fields of software engineering (SE) and trustworthy artificial intelligence with a special focus on the quality, reliability, safety, and security aspects of AI Systems.