3DeformRS: Certifying Spatial Deformations on Point Clouds

Gabriel Pérez S.*1,2  Juan C. Pérez*,2  Motasem Alfarra*,2  Silvio Giancola2  Bernard Ghanem2

1Universidad Nacional de Colombia, 2King Abdullah University of Science and Technology (KAUST)

gaperezsa@unal.edu.co

1{juan.perezsantamaria, motasem.alfarra, silvio.giancola, bernard.ghanem}@kaust.edu.sa

Abstract

3D computer vision models are commonly used in security-critical applications such as autonomous driving and surgical robotics. Emerging concerns over the robustness of these models against real-world deformations must be addressed practically and reliably. In this work, we propose 3DeformRS, a method to certify the robustness of point cloud Deep Neural Networks (DNNs) against real-world deformations. We developed 3DeformRS by building upon recent work that generalized Randomized Smoothing (RS) from pixel-intensity perturbations to vector-field deformations. In particular, we specialized RS to certify DNNs against parameterized deformations (e.g., rotation, twisting), while enjoying practical computational costs. We leverage the virtues of 3DeformRS to conduct a comprehensive empirical study on the certified robustness of four representative point cloud DNNs on two datasets and against seven different deformations. Compared to previous approaches for certifying point cloud DNNs, 3DeformRS is fast, scales well with point cloud size, and provides comparable-to-better certificates. For instance, when certifying a plain PointNet against a 3° rotation on 1024-point clouds, 3DeformRS grants a certificate 3× larger and 20× faster than previous work.

1. Introduction

Perception of 3D scenes plays a critical role in applications such as autonomous navigation [14] and robotics [32]. The success of autonomous driving cars and surgical robots largely depends on their ability to understand the surrounding scenes. Recent works [36, 37, 46, 51] demonstrated the capability of Deep Neural Networks (DNNs) to process 3D point clouds. These advances allowed DNNs to achieve exceptional performance in challenging 3D tasks, such as shape classification [45, 49], object detection [17] and semantic segmentation [3, 12]. Despite the superior performance of DNNs over traditional approaches, explaining the logic behind their decisions and understanding their limitations is extremely challenging. For instance, earlier works [22, 50] elucidated the limitations of point cloud DNNs to withstand tiny and meaningless perturbations in their input, known as adversarial attacks. The vulnerability of point cloud DNNs against adversarial attacks underscores the importance of considering their robustness for security-critical applications. The study of adversarial robustness in the image domain demonstrated that defenses [20] are often later broken by more sophisticated attacks [4, 8, 11]. This phenomenon poses a difficulty for evaluating robustness [7]. Such difficulty found response in the domain of certified robustness,
wherein a defense’s robustness is theoretically guaranteed and thus independent of the attack’s sophistication. Recent methods for certifying DNNs on images have spurred interest in extending such methods to point cloud DNNs. In particular, the seminal work of Liu et al. [27] certified point cloud DNNs against modification, addition, and deletion of points. Furthermore, Lorenz et al. [28] recently proposed a verification approach to certify against 3D transformations.

In this work, we propose 3DeformRS, a certification approach for point cloud DNNs against spatial and deformable transformations. Figure 1 provides an overview of 3DeformRS’s inner workings. Our approach builds upon the theoretical background of Randomized Smoothing (RS) [10]. Specifically, we build 3DeformRS by leveraging DeformRS [1], an RS reformulation that generalized from pixel-intensity perturbations to vector field deformations, and specializing it to point cloud data. In contrast to previous approaches, our work considers spatial deformations on any point cloud DNN, providing efficiency and practicality. Compared to previous work, we find that 3DeformRS produces comparable-to-better certificates, while requiring significantly less computation time. We thus build on 3DeformRS’ virtues and provide a comprehensive empirical study on the certified robustness of four representative point cloud DNNs (PointNet [36], PointNet++ [37], DGCNN [46], and CurveNet [51]) on two classification datasets (ModelNet40 [49] and ScanObjectNN [45]), and against seven different types of spatial deformations (Rotation, Translation, Affine, Twisting, Tapering, Shearing, and Gaussian Noise).

Contributions. In summary, our contributions are three-fold: (i) We propose 3DeformRS by extending Randomized Smoothing (RS) from certifying image deformations to point cloud transformations. We further show that RS’ classical formulation for certifying input perturbations can be seen as a special case of 3DeformRS. (ii) We conduct a comprehensive empirical study with 3DeformRS, where we assess the certified robustness of four point cloud DNNs, on two classification datasets and against seven spatial deformations. (iii) We compare 3DeformRS with an earlier point cloud certification approach in certification and runtime. We show that 3DeformRS delivers consistent improvements while inheriting RS’ scalability and efficiency.

2. Related Work

3D Computer Vision. Images enjoy a canonical representation as fixed-sized matrices; in contrast, several representations exist for 3D data. Such representations include point clouds [36, 37, 51], meshes [15, 25, 40], voxels [9, 38, 56], multi-view [21, 41, 47] and implicit representations [33–35]. Given the prevalence and practicality of point clouds, we focus our attention on this representation and the associated DNNs that process it. PointNet [36] was the first successful attempt at using DNNs on point clouds. This architecture introduced a point-wise MLP with a global set pooling, and achieved remarkable performance in shape classification and semantic segmentation. PointNet++ [37] then introduced intermediate pooling operations in point clouds for local neighborhood aggregation. Afterwards, DGCNN [46] modeled convolutional operations in point clouds based on dynamically-generated graphs between closest point features. Recently, CurveNet [51] learned point sequences for local aggregation, achieving state-of-the-art performance on 3D computer vision tasks. In this work, we conduct a comprehensive empirical study on certified robustness by analyzing the robustness of these four point cloud DNNs.

Robustness. Szegedy et al.’s [43] seminal work exposed the vulnerability of DNNs against small input modifications, now known as adversarial examples. Later works observed the pervasiveness of this phenomenon [8, 18], spurring an arms race between defenses that enhanced DNNs’ adversarial robustness [20, 31, 52] and attacks that could break such defenses [4, 8]. The conflict between ever-more complex defenses and attacks also incited interest towards “certified robustness” [48], wherein defenses enjoy theoretical guarantees about the inexistence of adversarial examples that could fool them. A set of works focused on exact verification [19, 54], while others considered probabilistic certification [24, 26]. Randomized Smoothing (RS) [10] has emerged as a certification approach from the probabilistic paradigm that scales well with models and datasets. Notably, RS has been successfully combined with adversarial training [39], regularization [53], and
smoothing-parameters’ optimization [2, 13]. Recently, DeformRS [1] reformulated RS to consider general parameterized vector-field deformations. In this work, we develop 3DeformRS by specializing and extending DeformRS to certify point cloud DNNs against spatial deformations.

**Certification on 3D Point Clouds.** Since the seminal work of Xiang et al. [50] attacked point cloud DNNs, several works studied the robustness of such DNNs [22, 23, 29, 30, 42, 44, 50, 55]. Despite growing interest in the robustness of point cloud DNNs, only two works have addressed their certification. PointGuard [27] provided tight robustness guarantees against modification, addition, and deletion of points. 3DCertify [28] generalized DeepG [5] to 3D point clouds and proposed 3Certify, a verification approach to certify robustness against common 3D transformations. PointGuard has the benefit of low computational cost (compared to that of exact verification), but does not allow for spatial deformations. Conversely, 3DCertify considers such transformations, but suffers from impractical computational costs. In contrast, our 3DeformRS approach combines the best of both worlds, thus allowing for spatial deformations while enjoying low computational cost.

### 3. Our approach: 3DeformRS

We present 3DeformRS, a probabilistic certification for point cloud DNNs against spatial deformations.

**Preliminaries.** Our methodology builds on Randomized Smoothing (RS) [10], arguably the most scalable DNN-certification method. Given a classifier $f : \mathbb{R}^d \rightarrow \mathcal{P}(\mathcal{Y})$ that maps an input $x \in \mathbb{R}^d$ to the probability simplex $\mathcal{P}(\mathcal{Y})$, RS constructs a smooth classifier $g$ that outputs the most probable class when $f$’s input is subjected to additive perturbations sampled from a distribution $\mathcal{D}$. While RS certified against additive pixel perturbations in images, DeformRS [1] extended this formulation to certify against image deformations by proposing a parametric-domain smooth classifier. Specifically, given image $x$ with coordinates $p$, a parametric deformation function $\nu_\phi$ with parameters $\phi$ (e.g. if $\nu$ is a rotation, then $\phi$ is the rotation angle), and an interpolation function $I_T$, DeformRS defined a parametric-domain smooth classifier

$$g_\phi(x, p) = E_{x \sim \mathcal{D}} [f(I_T(x, p + \nu_\phi))] \tag{1}.$$  

In a nutshell, $g$ outputs $f$’s average prediction over transformed versions of $x$. Note that, in contrast to RS, this formulation deforms pixel locations instead of their intensity. DeformRS showed that parametric-domain smooth classifiers are certifiably robust against perturbations to the deformation function’s parameters via the following corollary.

**Corollary 1** (restated from [1]). Suppose $g$ assigns class $c_A$ to an input $x$, i.e. $c_A = \arg\max_i g_\phi^A(x, p)$ with $p_A = g_\phi^A(x, p)$ and $p_B = \max_{c \neq c_A} g_\phi^c(x, p)$. Then $\arg\max_c g_{\phi+\delta}^c(x, p) = c_A$ for all parametric perturbations $\delta$ satisfying:

$$\|\delta\|_1 \leq \lambda(p_A - p_B) \quad \text{for } \mathcal{D} = \mathcal{U}[-\lambda, \lambda],$$

$$\|\delta\|_2 \leq \frac{\sigma}{2} \left( \Phi^{-1}(p_A) - \Phi^{-1}(p_B) \right) \quad \text{for } \mathcal{D} = \mathcal{N}(0, \sigma^2 I).$$

In short, Corollary (1) states that, as long as the parameters of the deformation function (e.g. rotation angle) are perturbed by a quantity upper bounded by the certified radius, $g_\phi$’s prediction will remain constant. This result allowed certification against various image deformations. In this work, we build upon DeformRS and specialize it to certify against spatial deformations in point clouds.

### 3.1. Parametric Certification for 3D DNNs

We specialize the result in Corollary (1) to point clouds. In this setup, $p \in \mathbb{R}^{N \times 3}$ is a point cloud consisting of $N$ 3-dimensional points. We highlight two key differences between certifying images and point clouds. (i) The interpolation function $I_T$, while essential in images for the pixels’ discrete locations, is irrelevant for 3D coordinates and so can be omitted. (ii) Most recent DNNs neglect the points’ color information and exclusively rely on the points’ location. We combine these observations and modify the parametric domain smooth classifier from Eq. (1) to propose

$$\hat{g}_\phi(p) = E_{x \sim \mathcal{D}} [f(p + \nu_\phi+\epsilon)]. \tag{2}$$

Our $\hat{g}_\phi$, inheriting $g_\phi$’s structure, is certifiable against parametric perturbations via Corollary (1). Note that, since point cloud DNNs exclusively rely on location, our parametric certification is thus equivalent to input certification.

To consider input deformations, let us study the general case where $\nu_\phi = \phi \in \mathbb{R}^{N \times 3}$. Under this setup, and by setting $p' = p + \phi$, our smooth classifier becomes:

$$\check{g}_\phi(p) = E_{x \sim \mathcal{D}} [f(p')]. \tag{3}$$

The classifier in Eq. (3) is a general case of both the domain smooth classifier [1] and the input smooth classifier proposed earlier in [10]. Note that this expression elucidates how parametric certification of point cloud DNNs is equivalent to input certification. At last, we note that the smooth classifier in Eq. (3) is also certifiable via Corollary 1. We highlight here that directly certifying $\check{g}$ against parametric transformations (e.g. rotation) will perform poorly, as empirically observed in [27]. However, such deficient performance is not an inherent weakness of RS for certifying point cloud transformations, but rather due to a sub-optimal formulation of RS for spatial deformations. We argue that our formulation, i.e. the parametric domain smooth classifier in Eq. (2), is more suitable for modeling spatial deformations than the one presented in [27].

Next, we outline various spatial deformations we consider for assessing the robustness point cloud DNNs.
3.2. Modeling Spatial Deformations

We now detail the spatial deformations we consider to assess the robustness of point cloud DNNs. Our formulation from Eq. (3) requires modeling spatial deformations as additive perturbations on the points’ coordinates. Thus, given a point cloud \( \mathbf{p} \) that is transformed to yield \( \mathbf{p}' \), we define the flow that additively perturbs \( \mathbf{p} \) as \( \mathbf{p}' = \mathbf{p} + \mathbf{p}' \), where \( \mathbf{p}' \in \mathbb{R}^{N \times 3} \). Modeling transformations is then equivalent to modeling the induced per-point flow. Thus, we are required to model each deformation via a parametric flow, whose parameters are those of the corresponding transformation.

We consider six parameterizable flows, corresponding to four linear and two nonlinear transformations. In particular, we consider four linear, rigid and deformable transformations: (1) rotation, (2) translation, (3) shearing, and (4) the general affine transformation. Furthermore, we follow prior work [28] and consider two nonlinear transformations: (5) tapering and (6) twisting. As a summary, we report all the deformation flows and their corresponding parameters in Table 1, and visualize some of their effects in Figure 2.

Table 1. Per-point Deformation Flows for semantically-viable spatial deformations. Convention: \( e_\alpha = \cos(\alpha) \) and \( s_\alpha = \sin(\alpha) \). Without loss of generality, we only show the rotation around the \( z- \) axis, and leave the formulation of other rotations to the Appendix.

<table>
<thead>
<tr>
<th>Name</th>
<th>Flow (( \mathbf{p} ))</th>
<th>( \phi )</th>
<th>Name</th>
<th>Flow (( \mathbf{p} ))</th>
<th>( \phi )</th>
<th>Name</th>
<th>Flow (( \mathbf{p} ))</th>
<th>( \phi )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Translation</td>
<td>( \hat{x} = t_x )</td>
<td>([t_x])</td>
<td>Rotation</td>
<td>( \hat{x} = (c_{\gamma} - 1)x - s_{\gamma}y )</td>
<td>([\gamma])</td>
<td>Aﬃne</td>
<td>( \hat{x} = ax + by + cz + d )</td>
<td>([a])</td>
</tr>
<tr>
<td></td>
<td>( \hat{y} = s_{\gamma}x + (c_{\gamma} - 1)y )</td>
<td>([b] )</td>
<td></td>
<td>( \hat{y} = ex + fy + gz + h )</td>
<td>([c])</td>
<td></td>
<td>( \hat{z} = ix + jy + kz + l )</td>
<td>([d])</td>
</tr>
<tr>
<td></td>
<td>( \hat{z} = l_z )</td>
<td>([l_z])</td>
<td></td>
<td>( \hat{z} = 0 )</td>
<td></td>
<td></td>
<td>( \hat{z} = 0 )</td>
<td></td>
</tr>
<tr>
<td>Shearing</td>
<td>( \hat{x} = az )</td>
<td>([a])</td>
<td>Twisting</td>
<td>( \hat{x} = (c_{\gamma} - 1)x - s_{\gamma}y )</td>
<td>([\gamma])</td>
<td>Tapering</td>
<td>( \hat{x} = (\frac{1}{2}a^2 + b)x )</td>
<td>([a])</td>
</tr>
<tr>
<td></td>
<td>( \hat{y} = bz )</td>
<td>([b])</td>
<td></td>
<td>( \hat{y} = s_{\gamma}x + (c_{\gamma} - 1)y )</td>
<td>([\gamma])</td>
<td></td>
<td>( \hat{y} = (\frac{1}{2}a^2 + b)yz )</td>
<td>([b])</td>
</tr>
<tr>
<td></td>
<td>( \hat{z} = 0 )</td>
<td></td>
<td></td>
<td>( \hat{z} = 0 )</td>
<td></td>
<td></td>
<td>( \hat{z} = 0 )</td>
<td></td>
</tr>
</tbody>
</table>

3.2. Modeling Spatial Deformations

We now detail the spatial deformations we consider to assess the robustness of point cloud DNNs. Our formulation from Eq. (3) requires modeling spatial deformations as additive perturbations on the points’ coordinates. Thus, given a point cloud \( \mathbf{p} \) that is transformed to yield \( \mathbf{p}' \), we define the flow that additively perturbs \( \mathbf{p} \) as \( \mathbf{p}' = \mathbf{p} + \mathbf{p}' \), where \( \mathbf{p}' \in \mathbb{R}^{N \times 3} \). Modeling transformations is then equivalent to modeling the induced per-point flow. Thus, we are required to model each deformation via a parametric flow, whose parameters are those of the corresponding transformation.

We consider six parameterizable flows, corresponding to four linear and two nonlinear transformations. In particular, we consider four linear, rigid and deformable transformations: (1) rotation, (2) translation, (3) shearing, and (4) the general affine transformation. Furthermore, we follow prior work [28] and consider two nonlinear transformations: (5) tapering and (6) twisting. As a summary, we report all the deformation flows and their corresponding parameters in Table 1, and visualize some of their effects in Figure 2.

Note that the affine transformation is the most general transformation we consider, capable of modeling any linear transformation. Presumably, a DNN enjoying certified robustness against affine transformations would also enjoy robustness against combinations of the other spatial deformations. We leave formulations of the aforementioned transformations in homogeneous coordinates to the Appendix.

3.2. Modeling Spatial Deformations

We now detail the spatial deformations we consider to assess the robustness of point cloud DNNs. Our formulation from Eq. (3) requires modeling spatial deformations as additive perturbations on the points’ coordinates. Thus, given a point cloud \( \mathbf{p} \) that is transformed to yield \( \mathbf{p}' \), we define the flow that additively perturbs \( \mathbf{p} \) as \( \mathbf{p}' = \mathbf{p} + \mathbf{p}' \), where \( \mathbf{p}' \in \mathbb{R}^{N \times 3} \). Modeling transformations is then equivalent to modeling the induced per-point flow. Thus, we are required to model each deformation via a parametric flow, whose parameters are those of the corresponding transformation.

We consider six parameterizable flows, corresponding to four linear and two nonlinear transformations. In particular, we consider four linear, rigid and deformable transformations: (1) rotation, (2) translation, (3) shearing, and (4) the general affine transformation. Furthermore, we follow prior work [28] and consider two nonlinear transformations: (5) tapering and (6) twisting. As a summary, we report all the deformation flows and their corresponding parameters in Table 1, and visualize some of their effects in Figure 2.

Note that the affine transformation is the most general transformation we consider, capable of modeling any linear transformation. Presumably, a DNN enjoying certified robustness against affine transformations would also enjoy robustness against combinations of the other spatial deformations. We leave formulations of the aforementioned transformations in homogeneous coordinates to the Appendix.

Gaussian Noise. In addition to the above transformations, we also consider Gaussian noise perturbations. Notably, this noise deforms the underlying vector field and, thus, is the most general perturbation. To certify against Gaussian noise, we construct the smooth classifier in Eq. (3) with \( \epsilon \in \mathbb{R}^{N \times 3} \) and \( D = \mathcal{N}(0, \sigma^2 I) \). While this deformation is very general, the high dimensionality of the certified parameters limits its applicability to imperceptible perturbations. Nevertheless, as adversaries may take such form [50], we consider this perturbation in our experiments.

Design Choices. For each spatial deformation and point cloud DNN, we assess certified robustness by constructing the smooth classifier from Eq. (2). In deformations whose parameter space is bounded (e.g. rotations, where angles beyond \( \pm \pi \) radians are redundant), we smooth with a uniform distribution and thus obtain an \( \ell_1 \) certificate. For the remaining deformations, we employ Gaussian smoothing and thus obtain an \( \ell_2 \) certificate.

4. Experiments

4.1. Setup

Datasets. We experiment on ModelNet40 [49] and ScanObjectNN [45]. ModelNet40 comprises 12,311 3D CAD models from 40 classes. ScanObjectNN has 2,902 real-world 3D scans of 15 classes. While ModelNet40 has synthetic objects, ScanObjectNN used 3D cameras and so contains natural and challenging self-occlusions. We use the ScanObjectNN variant that omits background data. For both datasets, the shapes are dimension-normalized, origin-centered and sampled with 1024 points.

Models. We study four point cloud DNNs: PointNet [36], PointNet++ [37], DGCNN [46] and CurveNet [51]. For PointNet, we used 3DCertify’s implementation, which uses \( z- \) rotation augmentation on ModelNet40. For CurveNet, we used the official implementation’s weights, trained on ModelNet40 with axis-independent \([0.66, 1.5]\) scaling and \( \pm 0.2 \) translation. On ScanObjectNN, PointNet and CurveNet are trained without augmentation. For PointNet++ and DGCNN, we used the PyTorch Geometric library [16] with default hyper-parameters and without augmentations.

Certification. In our experiments, following [1, 39, 53], we construct the hard version of the smooth classifier in Eq. (2) for each deformation. Moreover, we follow common practice and adapt the public implementation from [10] for estimating the certified radius via Monte Carlo sampling. In particular, we use 1,000 samples to estimate the certified radius with a probability of failure of \( 10^{-3} \). For all experiments, we provide envelope certified accuracy curves cross validated at several values of smoothing parameters, detailed in the Appendix. Since all deformations used Gaussian smoothing, the certificates we find are in the \( \ell_2 \) sense (except for rotation, which used uniform smoothing and so its certificate is in the \( \ell_1 \) sense).
4.2. Benchmarking 3D Networks

We present certification curves in Figures 3 and 4 for ModelNet40 and ScanObjectNN, respectively. We also report each curve’s associated Average Certification Radius (ACR) in Table 2 as a summary metric. We further highlight nine main observations from these results, and leave a detailed analysis, together with ablations on RS’ hyperparameters, to the Appendix.

Vulnerability against Rigid Transformations. Our analysis considers, among others, rotation and translation. Remarkably, we find that DNNs are significantly vulnerable even against these rigid perturbations. For most DNNs, the certified accuracy drops dramatically as the perturbation’s magnitude increases. This phenomenon supports further research on increasing the robustness of point cloud DNNs against simple transformations that could happen in the real world.

Deformation Complexity. Each spatial deformation is parameterized by a certain number of values. The number of parameters can be associated with the deformation’s complexity: “simple” deformations, i.e., rotation around the x axis, require few parameters, while “complex” deformations, i.e., affine, require several parameters. Under these notions, we notice that as the deformation’s complexity increases, the DNNs’ certified accuracies drop more rapidly. This observation agrees with intuition: complex transformations should be harder to resist than simple ones.

Gaussian Noise. This deformation is arguably the most general and information-destroying, as it may not preserve distances, angles nor parallelism. Experimentally, we indeed observe that DNNs are rather brittle against this noise: even small magnitudes can break the DNNs’ performance.
Over PointNet of about 7% and 10% in ModelNet40 and
performer is PointNet. Notably, this fact agrees with each
finds that the best performer is CurveNet, while the worst
Certified in the next subsection.
it enjoyed, and so we further study this phenomenon
attribute PointNet's robustness to the training-time augmen-
it was
Net's dominance is
independent of whether they are rotated or twisted around
rotation regime (from
its robust accuracy is mostly maintained across the entire
dermodel.
Training-time Augmentations Boost Certified Robust-
ment. Indeed, we find that, when PointNet is
augmentations. Indeed, we find that, when PointNet is
ModelNet40, CurveNet is the best per-
displaying the best robustness for seven of the 10
deformations we consider, while never scoring last.
PointNet Performs Poorly. On ModelNet40, reported in
Table 2 (top), PointNet achieves the lowest ACR values for
nine of the 10 deformations, while only achieving the best
score in two deformations (z–Rot and Twisting). Similarly,
for ScanObjectNN, in Table 2 (bottom), PointNet consist-
tsently scores last across all transformations, holding the last
position in eight out of 10 transformations, and only holding
the first position in only one (Gaussian noise).
Training-time Augmentations Boost Certified Robust-
ness. On ModelNet40, PointNet was trained with z-rotation
augmentations. Indeed, we find that, when PointNet is
evaluated on ModelNet40, it displays superior performance
against rotation-based transformations along the z axis, i.e.
z–Rot and Twisting. In particular, its ACR is > 0.8 more
than the runner-up in z–Rot and ~ 0.4 in Twisting, while
its robust accuracy is mostly maintained across the entire
rotation regime (from \(-\pi to +\pi\) radians, i.e. all possible
rotations). That is, PointNet correctly classifies most objects,
independent of whether they are rotated or twisted around
the z axis. More interestingly, we also remark how Point-
Net’s dominance is not observed in ScanObjectNN, where
it was not trained with z-rotation augmentations. We thus
attribute PointNet’s robustness to the training-time augmenta-
tions it enjoyed, and so we further study this phenomenon
in the next subsection.
Certified vs. Regular Accuracy. Overall, our analysis
finds that the best performer is CurveNet, while the worst
performer is PointNet. Notably, this fact agrees with each
DNN's plain test performance: CurveNet has an advantage
over PointNet of about 7% and 10% in ModelNet40 and
Table 2. Certified Robustness Assessment on ModelNet40 and ScanObjectNN. We report the Average Certification Radius (ACR) of all DNNs against 10 deformations, on each dataset. For each deformation, we embolden the best performance and underline the worst.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PointNet [36]</td>
<td>2.64</td>
<td>0.31</td>
<td>0.21</td>
<td>0.42</td>
<td>2.48</td>
<td>0.50</td>
<td>0.07</td>
<td>0.06</td>
<td>0.13</td>
<td>0.09</td>
</tr>
<tr>
<td>PointNet++ [37]</td>
<td>1.45</td>
<td>0.34</td>
<td>0.25</td>
<td>0.55</td>
<td>1.66</td>
<td>0.81</td>
<td>0.64</td>
<td>0.16</td>
<td>0.17</td>
<td>0.07</td>
</tr>
<tr>
<td>DGCNN [46]</td>
<td>1.29</td>
<td>0.33</td>
<td>0.26</td>
<td>0.54</td>
<td>1.78</td>
<td>0.90</td>
<td>0.18</td>
<td>0.14</td>
<td>0.20</td>
<td>0.10</td>
</tr>
<tr>
<td>CurveNet [51]</td>
<td>0.76</td>
<td>0.39</td>
<td>0.34</td>
<td>0.56</td>
<td>1.51</td>
<td>1.09</td>
<td>1.32</td>
<td>0.26</td>
<td>0.26</td>
<td>0.08</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ScanObjectNN</th>
<th>z–Rot.</th>
<th>xz–Rot.</th>
<th>xyz–Rot.</th>
<th>Shearing</th>
<th>Twisting</th>
<th>Tapering</th>
<th>Translation</th>
<th>Affine</th>
<th>Affine (NT)</th>
<th>Gaussian Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>PointNet [36]</td>
<td>0.30</td>
<td>0.16</td>
<td>0.15</td>
<td>0.35</td>
<td>0.90</td>
<td>0.72</td>
<td>0.06</td>
<td>0.05</td>
<td>0.11</td>
<td>0.10</td>
</tr>
<tr>
<td>PointNet++ [37]</td>
<td>0.38</td>
<td>0.23</td>
<td>0.22</td>
<td>0.45</td>
<td>0.89</td>
<td>0.93</td>
<td>0.67</td>
<td>0.16</td>
<td>0.17</td>
<td>0.05</td>
</tr>
<tr>
<td>DGCNN [46]</td>
<td>0.32</td>
<td>0.20</td>
<td>0.19</td>
<td>0.37</td>
<td>0.92</td>
<td>0.81</td>
<td>0.10</td>
<td>0.08</td>
<td>0.14</td>
<td>0.05</td>
</tr>
<tr>
<td>CurveNet [51]</td>
<td>0.51</td>
<td>0.31</td>
<td>0.29</td>
<td>0.51</td>
<td>1.24</td>
<td>1.02</td>
<td>0.19</td>
<td>0.14</td>
<td>0.20</td>
<td>0.08</td>
</tr>
</tbody>
</table>

Table 3. Test Set Accuracy on ModelNet40 and ScanObjectNN.

<table>
<thead>
<tr>
<th>ModelNet40</th>
<th>ScanObjectNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PointNet [36]</td>
<td>85.94%</td>
</tr>
<tr>
<td>PointNet++ [37]</td>
<td>90.03%</td>
</tr>
<tr>
<td>DGCNN [46]</td>
<td>90.03%</td>
</tr>
<tr>
<td>CurveNet [51]</td>
<td>93.84%</td>
</tr>
</tbody>
</table>

CurveNet Performs Remarkably. In terms of ACR, Cur-
veNet displays larger robustness than competitors across
the board. In particular, Table 2 (top), shows that, on
ModelNet40, CurveNet achieves the best ACR for seven of
the 10 deformations we present, while scoring last only in
two deformations (z–Rot and Twisting). Analogously, for
ScanObjectNN (Table 2, bottom), CurveNet is the best per-
derformer, displaying the best robustness for seven of the 10
deformations we consider, while never scoring last.

CurveNet Performs Poorly. On ModelNet40, reported in
Table 2 (top), PointNet achieves the lowest ACR values for
nine of the 10 deformations, while only achieving the best
score in two deformations (z–Rot and Twisting). Similarly,
for ScanObjectNN, in Table 2 (bottom), PointNet consist-
tsently scores last across all transformations, holding the last
position in eight out of 10 transformations, and only holding
the first position in only one (Gaussian noise).

Training-time Augmentations Boost Certified Robust-
ness. On ModelNet40, PointNet was trained with z-rotation
augmentations. Indeed, we find that, when PointNet is
evaluated on ModelNet40, it displays superior performance
against rotation-based transformations along the z axis, i.e.
z–Rot and Twisting. In particular, its ACR is > 0.8 more
than the runner-up in z–Rot and ~ 0.4 in Twisting, while
its robust accuracy is mostly maintained across the entire
rotation regime (from \(-\pi to +\pi\) radians, i.e. all possible
rotations). That is, PointNet correctly classifies most objects,
independent of whether they are rotated or twisted around
the z axis. More interestingly, we also remark how Point-
Net’s dominance is not observed in ScanObjectNN, where
it was not trained with z-rotation augmentations. We thus
attribute PointNet’s robustness to the training-time augmenta-
tions it enjoyed, and so we further study this phenomenon
in the next subsection.

Certified vs. Regular Accuracy. Overall, our analysis
finds that the best performer is CurveNet, while the worst
performer is PointNet. Notably, this fact agrees with each
DNN’s plain test performance: CurveNet has an advantage
over PointNet of about 7% and 10% in ModelNet40 and
ScanObjectNN, respectively. Thus, our results find a corre-
lation between regular and certified performance.

ModelNet40 and ScanObjectNN. CurveNet is, arguably,
the best performer on both datasets in terms of ACR. How-
ever, we note that all of CurveNet’s certified accuracies drop
significantly from ModelNet40 to ScanObjectNN. Thus,
our results agree with how ModelNet40’s synthetic nature
(compared to ScanObjectNN’s realistic nature) implies that
ModelNet40 is a “simpler” dataset than ScanObjectNN.

Sizable Variations in Robustness. For a given deformation
and dataset, we note sizable robustness differences across
DNNs. Specifically, we observe that one DNN may ob-
tain an ACR even 10× larger than other DNN. This phe-
nomenon happens even when the plain accuracy of the
DNNs being certified is mostly comparable, as reported in
Table 3. Thus, we argue that certified robustness should be
a design consideration when developing DNNs. That is, our
results suggest that (i) plain accuracy may not provide the
full picture into a DNN’s performance, and (ii) certified ac-
curacy may be effective and efficient for assessing DNNs.

4.3. Simple but Effective: Augmented Training

Previously, in Figure 3, we observed PointNet’s domi-
nant performance on z–axis rotation and twisting. We at-
tributed this superiority to augmentations PointNet enjoyed
during training. Here we investigate the effect of other train-
ing augmentations on certified robustness. In particular, we
train two DNNs with four different deformations and then
assess their certified robustness against such deformations.
We report the results of this experiment in Figure 5.

15174
We conduct augmentations during training and record relative improvements on certified accuracy. Most training augmentations improve certification across the board.

<table>
<thead>
<tr>
<th></th>
<th>Plain</th>
<th>z-Rot.</th>
<th>xyz-Rot.</th>
<th>Trans.</th>
<th>Twist.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PointNet++</td>
<td>90.1%</td>
<td>+0.2%</td>
<td>-0.3%</td>
<td>-0.4%</td>
<td>-0.1%</td>
</tr>
<tr>
<td>DGCNN</td>
<td>89.8%</td>
<td>+1.3%</td>
<td>+0.8%</td>
<td>+1.3%</td>
<td>-0.2%</td>
</tr>
</tbody>
</table>

Figure 5. **Relative effects of training augmentations on Average Certified Accuracy.** We conduct augmentations during training and record relative improvements on certified accuracy. Most training augmentations improve certification across the board.

We further compare 3DeformRS and 3DCertify [28], which is, to the best of our knowledge, the current state-of-the-art certification approach against spatial deformations. 3DCertify’s analysis focuses exclusively on PointNet and ModelNet40. We used their official implementation and pre-trained weights, and note that 3DCertify certifies a 100-instance subset of the test set. We note that the certified accuracies reported in [28] are w.r.t. the correctly-classified samples from such subset. Moreover, we underscore that 3DCertify’s analysis focuses exclusively on PointNet trained on ModelNet40.

**Technical Details.** We first discuss the intricacies of providing a fair comparison between 3DCertify (an exact verification approach) and 3DeformRS (a probabilistic certification approach). We underscore fundamental differences regarding the input each approach receives when certifying: (i) a given DNN, e.g. PointNet, (ii) w.r.t. a desired transformation, e.g. Rotation, on (iii) a selected point cloud input, e.g. a plane. In particular, 3DCertify receives as additional input the exact magnitude of the transformation. 3DCertify then uses the transformation and its magnitude, e.g. $3^\circ$, to return a boolean value stating if the given DNN’s output is certifiable against a transformation of such magnitude. On the other hand, 3DeformRS mechanism is fundamentally different: it computes a tight certification radius for the same (DNN-transformation-point cloud) tuple, but leveraging a random distribution with which the input was smoothed. The two approaches can be compared if the certified radius provided by 3DeformRS is evaluated against the transformation magnitude received by 3DCertify.

However, we note that the certification radius computed by 3DeformRS depends on the hyper-parameter $\sigma$ (or $\lambda$ for uniform smoothing) from Corollary 1. That is, 3DeformRS will compute a different radius for each $\sigma$ considered. While there exists a $\sigma^*$ with which 3DeformRS provides the largest certified radius for the transformation being considered, $\sigma^*$ is not known a priori. Thus, we are required to run a grid search of $\sigma$ values to compute each instance’s largest certified radius. While this procedure is computationally intensive, the output of each experiment—considering one $\sigma$ over the whole dataset—provides an entire certified accuracy curve, in contrast to a single point in this curve, as provided by exact verification approaches.

We underscore how this curve can provide insights into a DNN’s robustness. In practice, when comparing with 3DCertify on specific transformation magnitudes, we run a grid search of at most 18 $\sigma$ values to obtain each instance’s largest certified radius. Hence, we report certified accuracy envelope curves, i.e. certified accuracy curves that consider each instance’s largest certified radius. Additionally, to circumvent the problem arisen from considering different norms (3DCertify’s $\ell_\infty$ norm vs. 3DeformRS’s $\ell_1$ or $\ell_2$), we limit our comparison to single-parameter transformations.

**Results.** We compare against 3DCertify, in their reported experimental setup, along three dimensions: (i) certificate magnitude, (ii) point cloud cardinality and (iii) speed. Overall, we find that 3DeformRS provides comparable-to-better certificates, scales well with point cloud size, and has manageable computational cost.

**Certificate magnitude.** We consider rotations w.r.t. each axis, and evaluate 3DCertify’s official 64-points DNN against rotations of $\gamma \in \{1, 2, 3, 4, 5, 6, 7, 8, 10, 15\}$ degrees. Whenever possible, we used 3DCertify’s reported results, otherwise we used their public implementation to run certification. We find that 3DeformRS achieves comparable-to-better certificates, while providing the additional benefit of full certified accuracy curves, instead of
is, certifying small perturbations with 3DeformRS attains a necessary amount, still requires only a case of certifying with of the same implementation can leverage GPUs for accelerating certification on forward passes and against an of perturbations. For example, certifying PointNet with 3DCertify’s computational cost scales with the perturbations. Unlike 3DCertify, 3DeformRS enjoys a virtually-speed boost, while large perturbations can even enjoy an even better speed boost. Moreover, accelerating 3DeformRS via an Nvidia V100 GPU provides a 5× boost over its CPU counterpart, further improving runtime over 3DCertify.

4.5. Ablations

The stochastic nature of 3DeformRS requires a Monte Carlo method followed by a statistical test to bound the probability of returning an incorrect prediction. This statistical test is parameterized by a failure ratio $\alpha$ which, throughout our experiments, was set to the default value of $10^{-3}$, following [10]. Here, we analyze the sensitivity of 3DeformRS to other failure probabilities $\alpha$. We show in the Appendix the certified accuracy curves for failure probabilities $\alpha \in \{10^{-2}, 10^{-4}, 10^{-5}\}$. We underscore that, in our experiments, 3DeformRS shows negligible variation in certified accuracy w.r.t. changes in $\alpha$. In particular, we notice that all ACRs are $\sim 0.22$ for all the $\alpha$ values we considered.

5. Conclusions and Limitations

In this work, we propose 3DeformRS, a method for certifying point cloud DNNs against spatial deformations. Our method provides comparable-to-better certificates than earlier works while scaling better to large point clouds and enjoying practical computation times. These virtues of 3DeformRS allow us to conduct a comprehensive empirical study of the certified robustness of point cloud DNNs against semantically-viable deformations. Furthermore, 3DeformRS’s practical runtimes may enable its usage in real-world applications. While our stochastic approach is practical with its faster and top-performing certification, its stochasticity may also raise concerns when comparing against exact verification approaches. Moreover, we note that our work solely focused on assessing the 3D robustness of point cloud DNNs against input deformations, disregarding other types of perturbations. Possible avenues for future work include incorporating better training algorithms such as MACER [53] and SmoothAdv [39] for further robustness improvements.

Acknowledgements. This publication is based upon work supported by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under Award No. OSR-CRG2019-4033. We would also like to thank Jesús Zarzar for the help and discussions.

<table>
<thead>
<tr>
<th>Points</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boopathy et al. [6]</td>
<td>3.7</td>
<td>3.6</td>
<td>3.3</td>
<td>2.2</td>
<td>4.4</td>
<td>5.6</td>
<td>6.7</td>
</tr>
<tr>
<td>DeepPoly</td>
<td>95.1</td>
<td>94.0</td>
<td>91.3</td>
<td>72.2</td>
<td>51.1</td>
<td>39.3</td>
<td>28.1</td>
</tr>
<tr>
<td>3DCertify (Taylor 3D)</td>
<td>97.5</td>
<td>94.0</td>
<td>93.5</td>
<td>81.1</td>
<td>66.7</td>
<td>49.4</td>
<td>37.1</td>
</tr>
<tr>
<td>3DeformRS (ours)</td>
<td>98.8</td>
<td>97.6</td>
<td>97.8</td>
<td>100</td>
<td>100</td>
<td>97.8</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 5. Rotation Certificates when varying Point Cloud Cardinality. The three previous methods consider linear relaxations and use the DeepPoly verifier, while 3DeformRS is based on Randomized Smoothing. Baselines taken from [28].

<table>
<thead>
<tr>
<th>Device</th>
<th>$\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>38.8</td>
</tr>
<tr>
<td>GPU</td>
<td>7.3</td>
</tr>
</tbody>
</table>

Table 6. Runtimes for 3DeformRS. We compare certification runtime for a single $\sigma$ value both in CPU and GPU (values in seconds). 3DeformRS’s CPU version enjoys reasonable certification times, and leveraging a GPU lowers the runtime by $\sim 5 \times$. 

individual points in these plots. Refer to the Appendix for such certified accuracy curves.

Point Cloud Cardinality. 3DCertify provides exact verification on point clouds of limited size (64 points reported in [28]). However, DNNs may deal with point clouds of at least 1024 points in practice. We experiment on this setup and compare with previous approaches by varying the point clouds’ size and certifying against a $3^\circ z-$rotation. For this experiment, we follow 3DCertify’s setup [28] and use the same DNN weights for each certification method. Table 5 shows that our approach provides a better certificate for DNNs when trained and tested on large point clouds. In particular, we mark three main observations. First, 3DeformRS performs up to 60% better on the realistic setup of 1024 points. Second, 3DeformRS provides a certified accuracy of over 97% across the board; while large robustness is expected from a model augmented with $z-$rotations, our approach shows that providing such certificates is possible. Third, 3DeformRS breaks the trend of decreasing certificates from which other approaches suffered when handling larger point clouds. Thus, we find that 3DeformRS enjoys scalability and invariance to larger input size.

Speed. Certification via 3DCertify [28] is computationally expensive, and its current official implementation cannot benefit from GPU hardware accelerators. Furthermore, 3DCertify’s computational cost scales with the perturbation’s magnitude, hindering certification against large perturbations. For example, certifying PointNet with 64 points against an $z-$rotation on a CPU requires $\sim 13k$ seconds for $1^\circ$, and up to $\sim 89k$ seconds for $10^5$.

Unlike 3DCertify, 3DeformRS enjoys a virtually-constant computation cost, as it requires a fixed number of forward passes and $\sigma$ values. In addition, our native implementation can leverage GPUs for accelerating certification. Certifying against $z-$rotation with 3DeformRS on the same CPU requires only $\sim 40$ seconds per $\sigma$ (independent of $\sigma$’s magnitude, as reported in Table 6). Even the extreme case of certifying with 100 values for $\sigma$, arguably an unnecessary amount, still requires only $\sim 4k$ seconds. That is, certifying small perturbations with 3DeformRS attains a $\sim 3 \times$ speed boost, while large perturbations can even enjoy a $\sim 20 \times$ boost. Moreover, accelerating 3DeformRS via an
References


[29] Chengcheng Ma, Weiliang Meng, Baoyuan Wu, Shibiao Xu, and Xiaopeng Zhang. Efficient joint gradient based attack


[54] Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neu-
