Point Cloud Augmentation with Weighted Local Transformations

Sihyeon Kim[1]*, Sanghyeok Lee[1]*, Dasol Hwang[1]
Jaewon Lee[1], Seong Jae Hwang[2], Hyunwoo J. Kim[1]†
[1]Korea University [2]University of Pittsburgh
{sh_bs15, cat0626, dd_sol, 2j1ejyu, hyunwoojkim}@korea.ac.kr
sjh95@pitt.edu

Abstract

*Despite the extensive usage of point clouds in 3D vision, relatively limited data are available for training deep neural networks. Although data augmentation is a standard approach to compensate for the scarcity of data, it has been less explored in the point cloud literature. In this paper, we propose a simple and effective augmentation method called PointWOLF for point cloud augmentation. The proposed method produces smoothly varying non-rigid deformations by locally weighted transformations centered at multiple anchor points. The smooth deformations allow diverse and realistic augmentations. Furthermore, in order to minimize the manual efforts to search the optimal hyperparameters for augmentation, we present AugTune, which generates augmented samples of desired difficulties producing targeted confidence scores. Our experiments show our framework consistently improves the performance for both shape classification and part segmentation tasks. Particularly, with PointNet++, PointWOLF achieves the state-of-the-art **89.7** accuracy on shape classification with the real-world ScanObjectNN dataset. The code is available at https://github.com/mlvlab/PointWOLF.*

1. Introduction

Modern deep learning techniques, which established their popularity on structured data, began showing success on point clouds. Unlike images with clear lattice structures, each point cloud is an unordered set of points with no inherent structures that globally represent various 3D objects. Recent deep learning efforts have focused on enabling neural networks to operate on point clouds. While several point cloud datasets appeared, a particular dataset of scanned real-world objects [1] required a much greater understanding of the point cloud structures to identify highly complex real-world objects. In response, the approaches have evolved from extracting point-wise information with no structural information [2] to explicitly encoding the *local structure* [3]. These works on network development have been making steady progress despite the scarcity of point cloud data.

Figure 1. **Locally augmented point clouds using PointWOLF.** In each row, the left most sample is the original, and the remaining samples are its locally transformed results (brighter regions indicate stronger local transformations). PointWOLF can *locally* transform objects while preserving the original shape identity.

Our interest lies in data augmentation, which is extensively utilized in other machine learning pipelines for solving the data scarcity issue. Interestingly, despite its prevalence in other data modalities, data augmentation (DA) on point clouds is relatively less explored. For instance, conventional data augmentation (CDA) [2, 3], which consists of global rotation, scaling, translation, and small point-

*Equal contribution. † is the corresponding author.

wise noise, is commonly applied to point cloud datasets. Recently, PointAugment [4] proposes to learn the transformation matrix with an augmentor network to produce augmentations. PointMixup [5] finds linearly interpolated point cloud samples and their classes (*e.g.*, Mixup [6]). Despite their efforts to elevate the previous CDA, they still have apparent limitations. Specifically, PointAugment relies on a single (thus global) transformation matrix/vector for each sample, and PointMixup mixes up the samples globally without explicitly considering each sample's local structures. Thus, the need for a point cloud augmentation approach capable of producing diverse samples that *accurately depict real-world local variability* (*e.g.*, airplanes with varying lengths of wings and body) still remains.

In this work, we propose a novel point cloud augmentation PointWOLF that satisfies such needs. PointWOLF generates diverse and realistic local deformations such as a person with varying postures (see Figure 1). Our approach systematically enables local deformation by first considering multiple local transformations with respect to anchor points and carefully combining them in smoothly varying manners. Furthermore, we present AugTune to adaptively control DA strength in a sample-wise manner. AugTune produces consistent and beneficial samples during training with a single hyperparameter which alleviates the known dependence on hyperparameter selection of augmentation. We believe our method can further resolve this common dependence issue in general data augmentation.

Our **contributions** is fourfold: (**i**) We propose a powerful point cloud transformation approach capable of generating diverse and realistic augmented samples by deforming local structures. (**ii**) Our framework adaptively adjusts the strength of augmentation with only a single hyperparameter. (**iii**) We demonstrate that our framework brings consistent improvements over existing state-of-the-art augmentation methods on both synthetic and real-world datasets in point cloud shape classification and part segmentation tasks. (**iv**) Our framework improves the robustness of models against various local and global corruptions.

2. Related Work

Deep Learning on Point Clouds. Early deep learning works on point cloud have focused on enabling existing CNNs to operate on point clouds. These include multi-view based methods like [7, 8, 9, 10] where they project the 3D point cloud to 2D space through bird's-eye view or multi-view where 2D convolution becomes feasible. Similarly, other works [11, 12, 13, 14, 15] voxelize the point cloud to directly apply 3D convolution on the voxelized point cloud. To preserve the original structure of point clouds, point-based methods have emerged. PointNet [2] considers each point cloud as an unordered set and derives point-wise features with multi-layer perceptron and max pooling. However, a symmetric function such as pooling cannot characterize the local structure of point clouds, thus, PointNet++ [3] appeared which utilizes local information through hierarchical sampling and grouping. Other related studies [16, 17, 18, 19, 20, 21] also rely on grouping to identify the relationship between points and extract local structure. DGCNN [20] explicitly leverages the graph-like structure of the point clouds in the feature space rather than the 3D space. Interestingly, despite these network-wise efforts to exploit the local structure, only a few works have looked for solutions outside the networks, *e.g.*, data augmentation.

Data Augmentation. Data augmentation (DA) has become a necessity for modern machine learning model training to improve the generalization power. For point clouds, global similarity transformations such as rotation, scaling, and translation with point-wise jittering [2, 3] are conventionally used. However, such conventional DAs (CDA) do not augment *local* structures which many successful works mentioned above find beneficial and explicitly leverage. In light of this, only a few recent studies proposed more advanced DAs on point cloud. PointAugment [4] learns an augmentor network to augment samples to be more difficult to classify than the original sample. PointMixup [5], enables Mixup [6, 22] technique to point cloud, specifically by interpolating between two point cloud samples and predicting the ratio of the mixed classes with a soft label. While these works enable augmentations beyond simple similarity transformations, the transformations are fundamentally *global*: PointAugment learns a sample-wise global transformations matrix and PointMixup globally interpolates between samples. Thus, they often do not produce augmentations that are truly local and realistic. In response to this need, we propose a novel augmentation method PointWOLF which *locally* transforms samples as in Figure 1.

Searching for Optimal DA. In practice, identifying strong candidate transformations and optimal parameters for DA lacks intuitive conventions and heuristics thus requires extensive searching process. Several works address this, for instance, AutoAugment [23] and Fast AutoAugment [24] dynamically search for the best transformation policy via costly solvers such as reinforcement learning or bayesian optimization. RandAug [25] has drastically reduced the search space by binding multiple augmentation parameters as a single hyperparameter. In this paper, we present AugTune that efficiently controls the sample-wise DA strength with a single parameter using the target confidence score.

3. Method

We first briefly describe the conventional DA for point clouds. Then, we describe *PointWOLF*, which aims to generate augmented point clouds. Unlike previous works that perform a global transformation and point-wise jittering, our framework augments the point clouds by locally

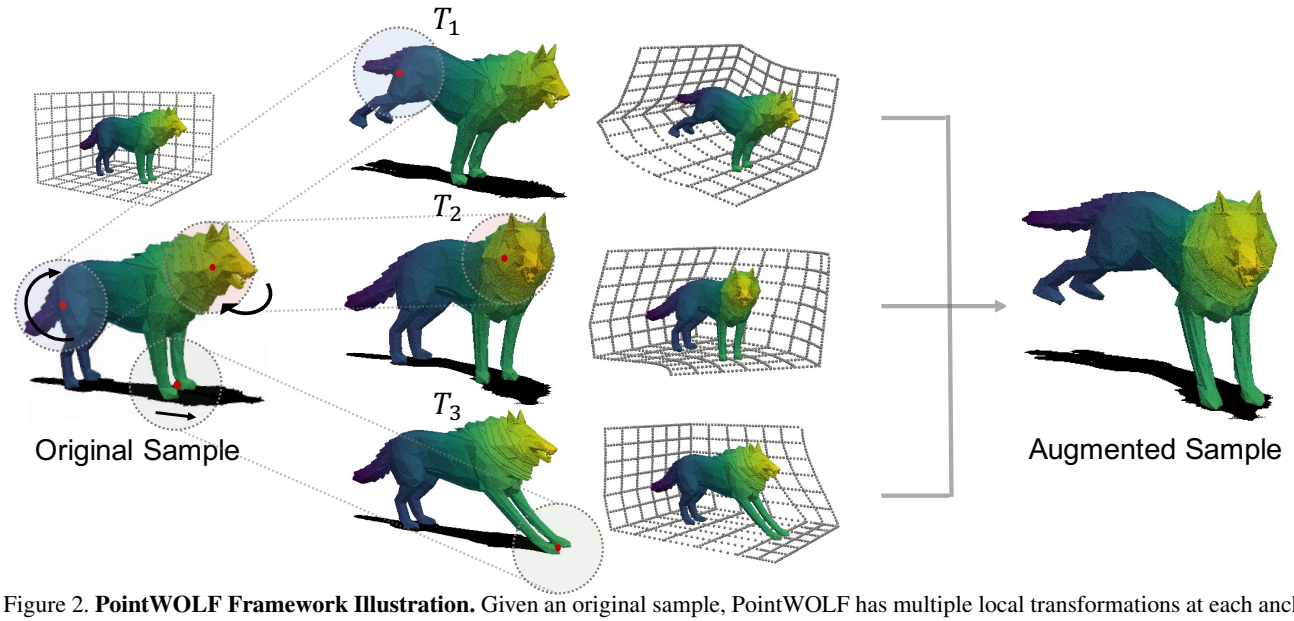


Figure 2. **PointWOLF Framework Illustration.** Given an original sample, PointWOLF has multiple local transformations at each anchor point (red). PointWOLF produces smoothly varying non-rigid deformations based on the weighted local transformations.

weighted multiple transformations. We generate diverse and realistic augmented samples by deforming local structures. Also, to reduce the dependence on optimal hyperparameters of DA frameworks, we present *AugTune*, adaptively modulates the augmentation strength with a single parameter.

3.1. Preliminaries

A point cloud $\mathcal{P} \in \mathbb{R}^{(3+C)\times N}$ in 3D is a set of points $\{\mathbf{p}_1, \cdots, \mathbf{p}_N\}$. Each point is represented as a vector $\mathbf{p}_i \in \mathbb{R}^{3+C}$ which is a concatenation of its coordinates (*i.e.*, [x,y,z]) and C dimensional input features such as color and a surface normal. Since the problem of our interest only focuses on the point cloud structure, we assume that no additional input features are given, *i.e.*, $\mathcal{P} \in \mathbb{R}^{3\times N}$. A conventional data augmentation (CDA) [2, 3] for point clouds applies a *global* similarity transformation (*e.g.*, scaling, rotation and translation) and point-wise jittering. A resulting augmented point cloud $\mathcal{P}' \in \mathbb{R}^{3\times N}$ is given as follows:

$$\mathcal{P}' = s\mathbf{R}\mathcal{P} + \mathbf{B}, \tag{1}$$

where $s > 0$ is a scaling factor, $\mathbf{R}$ is a 3D rotation matrix, and $\mathbf{B} \in \mathbb{R}^{3\times N}$ is a translation matrix with global translation and point-wise jittering. Typically, $\mathbf{R}$ is an extrinsic rotation parameterized by a uniformly drawn Euler angle for up-axis orientation. Scaling and translation factors are uniformly drawn from an interval, and point-wise jittering vectors are sampled from a truncated Gaussian distribution.

Thus, CDA is simply a similarity transformation with small jittering that cannot simulate diverse shapes and deformable objects. Unlike synthetic datasets like ModelNet [14] and ShapeNet [26], a real-world dataset like ScanObjectNN [1] further necessitates the generation of sophisticated deformations such as a mixture of local transformations. These are exemplified in Figure 1: airplanes with varying lengths and directions of wings and body, guitars in varying sizes and aspect ratios, and people with different heights and postures (e.g., crossing legs).

3.2. PointWOLF

We present a simple yet effective **point** cloud augmentation with **w**eighted **l**oc**a**l trans**f**ormations (PointWOLF). Our method generates deformation for point clouds by a convex combination of multiple transformations with smoothly varying weights. PointWOLF first selects several anchor points and locates random local transformations (*e.g.*, similarity transformations) at the anchor points. Based on the distance from a point in the input to the anchor points, our method differentially applies the local transformations. The smoothly varying weights based on the distance to the anchor points allow spatially continuous augmentation and generate realistic samples. Our framework can be viewed as a kernel regression with transformations.

Sampling anchor points is the first step of our framework to locate multiple local transformations. To minimize the redundancy between local transformations, the anchor points $\mathcal{P}^{\mathcal{A}} \subset \mathcal{P}$ are selected by the Farthest Point Sampling (FPS) algorithm. FPS randomly chooses the first point and then sequentially chooses the farthest points from previous points. This maximizes the coverage of anchor points and allows diverse transformations.

Local transformations in our framework are centered at the anchor points. At each anchor point, we randomly sample a local transformation that includes scaling from the anchor point, changing aspect ratios, translation, and rotation

around the anchor point. This subsumes the global transformation in (1). Given an anchor point $\mathbf{p}_j^{\mathcal{A}}$ in $\mathcal{P}^{\mathcal{A}}$, the local transformation for an input point $\mathbf{p}_i$ can be written as:

$$\mathbf{p}_i^j = \mathbf{S}_j \mathbf{R}_j (\mathbf{p}_i - \mathbf{p}_j^{\mathcal{A}}) + \mathbf{b}_j + \mathbf{p}_j^{\mathcal{A}}, \tag{2}$$

where $\mathbf{R}_j$, $\mathbf{S}_j$ and $\mathbf{b}_j$ are rotation matrix, scaling matrix and translation vector $\mathbf{b}_j$ respectively which specifically correspond to $\mathbf{p}_j^{\mathcal{A}}$. $\mathbf{S}$ is a diagonal matrix with three positive real values, *i.e.*, $\mathbf{S} = \text{diag}(s_x, s_y, s_z)$ to allow different scaling factors for different axes. In order to change the aspect ratios along arbitrary directions, a randomly rotated scaling matrix, *e.g.*, $\mathbf{S}'_j = \mathbf{R}^{-1}\mathbf{S}_j\mathbf{R}$, can be used. Since many commonly used datasets are pre-aligned in a standard space (*e.g.*, airplanes facing the same direction), we may assume sensible object orientations. In practice, we see that scaling with reasonable rotations as in (2) is sufficient.

Smooth deformations are key to generate realistic and locally transformed samples. A naïve application of a random local transformation within its finite neighborhood may result in a discontinuous shape and an overlap of different parts. It has a high chance to lose discriminative structures. Instead, we employ the Nadaraya-Watson kernel regression [27, 28] to smoothly interpolate the local transformations in the 3D space. Given M local transformations $\{T_j\}_{j=1}^M$, our smoothly varying transformation at an arbitrary point $\mathbf{p}_i$ is given as:

$$\hat{T}(\mathbf{p}_i) = \frac{\sum_{j=1}^M K_h(\mathbf{p}_i, \mathbf{p}_j^{\mathcal{A}}) T_j}{\sum_{k=1}^M K_h(\mathbf{p}_i, \mathbf{p}_k^{\mathcal{A}})}, \tag{3}$$

where $K_h(\cdot,\cdot)$ is a kernel function with bandwidth h, and T_j is the local transformation in (2) centered at $\mathbf{p}_j^{\mathcal{A}}$. To define $\hat{T}(\mathbf{p}_i)$ at any point in the 3D space, we use a kernel function that has a strictly positive value for any pair of points, *i.e.*, $K_h(\mathbf{p}_i, \mathbf{p}_j) > 0$ for $\forall \mathbf{p}_i, \forall \mathbf{p}_j$. The following proposition theoretically guarantees that our augmentation is a smooth transformation under mild conditions. The proof is in the supplement.

Proposition 1 *If a kernel function $K_h(\cdot,\cdot)$ and all local transformations $\{T_j\}_{j=1}^M$ are smooth, then the locally weighted transformation $\hat{T}(\cdot)$ in (3) is a smooth transformation.*

In our experiments, we use the Gaussian kernel with Euclidean distance after projection. Our kernel function is

$$K_h(\mathbf{p}_i, \mathbf{p}_j^{\mathcal{A}}; \Pi_j) = \exp\left(\frac{-\|\Pi_j(\mathbf{p}_i - \mathbf{p}_j^{\mathcal{A}})\|_2^2}{2h^2}\right), \tag{4}$$

where h is the bandwidth and $\Pi_j \in \mathbb{R}^{3\times 3}$ is a projection matrix. $\Pi = \text{diag}(\pi_x, \pi_y, \pi_z)$ is constructed with $\pi_x, \pi_y, \pi_z \sim \text{Bernoulli}(\beta)$ for $\beta \in (0,1)$[1], which acts as a "binary mask" to measure distances with respect to a random subset of the coordinates. For instance, a kernel function with $\Pi_j = \text{diag}(0,0,1)$ attenuates the influence of local transformation T_j based on the distance from $\mathbf{p}_j^{\mathcal{A}}$ along the z-axis, and this allows more diverse and realistic transformations such as shearing and torsion (Section 4.4) by a combination of local similarity transformations. Similar to the scaling matrix $\mathbf{S}$ in (2), in our experiments we use the projections onto the canonical axes/planes instead of an arbitrary subspace. Our preliminary experiments show that they are sufficient for pre-aligned point clouds.

[1]To prevent the projection matrix from zero-matrix, we resample Π if (0,0,0) is selected.

Algorithm 1 PointWOLF

Input: original point cloud $\mathcal{P} \in \mathbb{R}^{3\times N}$
Input: # points N, # anchor points M, kernel bandwidth h
Input: range for scaling ρ_s, range for rotation ρ_r, range for translation ρ_t, axis dropout probability β
Output: augmented point cloud $\mathcal{P}' \in \mathbb{R}^{3\times N}$

$\mathcal{P}^{\mathcal{A}} \leftarrow \text{FPS}(\mathcal{P}, M)$ ▷ $\mathcal{P}^{\mathcal{A}} \in \mathbb{R}^{3\times M}$
for $j = 1$ to M **do**
  $\mathbf{S}_j \leftarrow \text{diag}(s_x, s_y, s_z)$ ▷ $s \sim \text{U}_{[1,\rho_s]}$
  $\mathbf{R}_j \leftarrow \text{RotationMatrix}(\theta_x, \theta_y, \theta_z)$ ▷ $\theta \sim \text{U}_{[-\rho_r,\rho_r]}$
  $\mathbf{b}_j \leftarrow (b_x, b_y, b_z)$ ▷ $b \sim \text{U}_{[-\rho_t,\rho_t]}$
  $\mathbf{\Pi}_j \leftarrow (\pi_x, \pi_y, \pi_z)$ ▷ $\pi \sim \text{Bernoulli}(\beta)$
end for
for $i = 1$ to N **do**
  for $j = 1$ to M **do**
    $\mathbf{p}_i^j \leftarrow \mathbf{S}_j\mathbf{R}_j(\mathbf{p}_i - \mathbf{p}_j^{\mathcal{A}}) + \mathbf{b}_j + \mathbf{p}_j^{\mathcal{A}}$ ▷ Eq. (2)
    $w_i^j \leftarrow K_h(\mathbf{p}_i, \mathbf{p}_j^{\mathcal{A}}; \Pi_j)$ ▷ Eq. (4)
  end for
  $\mathbf{p}'_i \leftarrow \frac{\sum_{j=1}^M w_i^j \mathbf{p}_i^j}{\sum_{k=1}^M w_i^k}$ ▷ Eq. (3)
end for
$\mathcal{P}' \leftarrow \{\mathbf{p}'_i\}_{i=1}^N$

We have introduced our framework from a *kernel regression* perspective. Figure 2 shows a pipeline of our framework where the Augmented Sample is obtained by combining local transformations as a smooth transformation $\hat{T}$ and applying it on the Original Sample. Interestingly, at a high-level, we may also view our framework as an adaptive *interpolation of multiple globally transformed point clouds* resulting from applying different (local) transformations (*e.g.*, T_1, T_2, T_3 in Figure 2) on the Original Sample. Thus, our framework can be implemented in two ways: **(1)** Transforming each point once by a smoothly varying transformation $\hat{T}$ in Eq. (3) and **(2)** Transforming each point M times by the local transformations $\{T_j\}_{j=1}^M$ and interpolate these M augmented points by the adaptive weights $K(\mathbf{p}, \mathbf{p}_j^{\mathcal{A}}) / \sum_k K(\mathbf{p}, \mathbf{p}_k^{\mathcal{A}})$. Although both approaches require $O(MN)$ complexity if we mainly consider M anchor points and N points, the second approach is slightly more efficient in practice since this only involves operations on points (vector) while the first approach involves operations on transformation matrices. Thus, we show the pseudocode of the second approach in Algorithm 1 and show the first approach's pseudocode in the supplement.

Algorithm 2 AugTune

Input: original point cloud $\mathcal{P} \in \mathbb{R}^{3\times N}$, ground truth $\mathbf{y}$
Input: classifier $f(\cdot;\mathbf{w})$, difficulty coefficient $\lambda \in (0, 1]$
Output: Final augmented point cloud $\mathcal{P}^* \in \mathbb{R}^{3\times N}$

$\mathcal{P}' \leftarrow$ PointWOLF($\mathcal{P}$) ▷ Algorithm 1
$\hat{\mathbf{y}}_{\mathcal{P}} \leftarrow f(\mathcal{P};\mathbf{w})$, $\hat{\mathbf{y}}_{\mathcal{P}'} \leftarrow f(\mathcal{P}';\mathbf{w})$
$c_{\mathcal{P}} \leftarrow \hat{\mathbf{y}}_{\mathcal{P}}^\top \mathbf{y}$, $c_{\mathcal{P}'} \leftarrow \hat{\mathbf{y}}_{\mathcal{P}'}^\top \mathbf{y}$ ▷ confidence scores
$c \leftarrow \max(c_{\mathcal{P}'}, (1-\lambda)c_{\mathcal{P}})$ ▷ target confidence score
$\tilde{\alpha} \leftarrow \dfrac{c - c_{\mathcal{P}'}}{c_{\mathcal{P}} - c_{\mathcal{P}'}}$ ▷ approximate α^* by Eq. (7)
$\mathcal{P}^* \leftarrow \tilde{\alpha}\mathcal{P} + (1-\tilde{\alpha})\mathcal{P}'$ ▷ interpolate $\mathcal{P}$ and $\mathcal{P}'$

3.3. AugTune: Effective DA Tuning Method

The keys to effective data augmentation are strong candidate transformations and the optimal strength of the augmentation. We introduced PointWOLF that generates more diverse and smooth nonlinear transformations. Now, we present an efficient scheme to adaptively adjust the strength of data augmentation during training with a **single** hyperparameter. We believe that our scheme benefits not only our framework but also any classical data augmentation methods that heavily rely on an exhaustive grid search with a huge number of hyperparameters.

AugTune. We present AugTune described in Algorithm 2 to control the strength of augmentation. AugTune adjusts the strength of data augmentation by mixing the augmentation proposal $\mathcal{P}'$ from PointWOLF and the original sample $\mathcal{P}$. Given a classifier $f(\cdot;\mathbf{w})$ and a sample $\mathcal{P}$, let $\hat{\mathbf{y}}_{\mathcal{P}}$ and $c_{\mathcal{P}}$ denote its prediction and confidence score, *i.e.*, $\hat{\mathbf{y}}_{\mathcal{P}} = f(\mathcal{P},\mathbf{w})$ and $c_{\mathcal{P}} = \hat{\mathbf{y}}_{\mathcal{P}}^\top \mathbf{y}$, where $\mathbf{y}$ is the ground truth label represented in one-hot encoding. $\hat{\mathbf{y}}_{\mathcal{P}'}$ and $c_{\mathcal{P}'}$ are similarly defined for $\mathcal{P}'$. Note that all the confidence scores $c_{\mathcal{P}}$, $c_{\mathcal{P}'}$ are obtained on the fly while training the model, *i.e.*, an extra pretrained classifier is not required. To adjust the strength of augmentation, given a *difficulty coefficient* $\lambda \in (0, 1]$, AugTune first computes a target confidence score c for each sample by

$$c = \max(c_{\mathcal{P}'}, (1-\lambda)c_{\mathcal{P}}). \tag{5}$$

Assuming the augmented $\mathcal{P}'$ is difficult than the original $\mathcal{P}$, *i.e.*, $c_{\mathcal{P}'} < c_{\mathcal{P}}$, as λ gets close to 0, it implies that AugTune generates samples similar to the original sample $\mathcal{P}$. Conversely, when $\lambda = 1$, $c = c_{\mathcal{P}'}$, AugTune uses the augmentation proposal $\mathcal{P}'$ without any adjustment. To generate an augmented sample with the target confidence score, we use the linear interpolation of two samples $\mathcal{P}$ and $\mathcal{P}'$. Then, the problem is reduced to finding α^* defined by

$$\alpha^* = \underset{\alpha}{\operatorname{argmin}} \parallel c - f(\alpha\mathcal{P} + (1-\alpha)\mathcal{P}') \parallel^2 . \tag{6}$$

However, solving (6) directly by optimization algorithms or grid search is still computationally expensive. Thus, we approximate α^* by $\tilde{\alpha} = \frac{c - c_{\mathcal{P}'}}{c_{\mathcal{P}} - c_{\mathcal{P}'}}$ which is the solution to

$$\alpha c_{\mathcal{P}} + (1-\alpha)c_{\mathcal{P}'} = c. \tag{7}$$

Our experiments show this approximation does not cause degradation in the target tasks (see the supplement). The final augmented sample $\mathcal{P}^*$ is a convex combination of $\mathcal{P}$ and $\mathcal{P}'$ with $\tilde{\alpha}$, *i.e.*, $\mathcal{P}^* = \tilde{\alpha}\mathcal{P} + (1-\tilde{\alpha})\mathcal{P}'$, then the model parameter $\mathbf{w}$ is updated as $\mathbf{w} \leftarrow \mathbf{w} - \gamma\nabla_{\mathbf{w}}\mathcal{L}(f(\mathcal{P}^*,\mathbf{w}),\mathbf{y})$, where γ is a learning rate. Note that since the correspondence between $\mathcal{P}$ and $\mathcal{P}'$ are known by construction, the interpolation of two point clouds can be obtained by a simple point-wise interpolation given as $\mathbf{p}^* = \tilde{\alpha}\mathbf{p} + (1-\tilde{\alpha})\mathbf{p}'$. Moreover, AugTune works as a safeguard to preserve the shape identity for the final $\mathcal{P}^*$. So, we rarely observed unrealistic augmented samples with reasonable hyperparameters (see the supplement for visualizations).

Remarks. As we viewed our framework as the kernel regression on local transformations, AugTune is directly applicable to the transformation (parameter) space. In other words, instead of the point-wise interpolation, we may simply interpolate the local transformation parameters: scaling matrix $\mathbf{S}'_j = \alpha\mathbf{I} + (1-\alpha)\mathbf{S}_j$, rotation $\theta'_j = (1-\alpha)\theta_j$, and translation $\mathbf{b}'_j = (1-\alpha)\mathbf{b}_j$. However, due to its slightly higher computational cost and inferior performance, we applied AugTune on the input data space, see Section 4.2.

4. Experiments

In this section, we demonstrate the effectiveness of our PointWOLF on both synthetic and real-world datasets. We begin by describing the datasets, baselines, and experimental setup. Then, we evaluate our framework for shape classification and part segmentation (Section 4.1), followed by ablation studies and analyses (Section 4.2). We conduct the experiments to show whether our method improves the robustness of models against both local and global corruptions leveraging diverse locally-augmented samples (Section 4.3). Lastly, we provide qualitative analysis of the augmented samples by PointWOLF (Section 4.4).

Datasets. We use both synthetic and real-world datasets for shape classification to evaluate our framework. ModelNet40 (**MN40**) [14] is a widely used synthetic benchmark dataset containing 9,840 CAD models in the training set and 2,468 CAD models in the validation set with total 40 classes of common object categories. As in [5], we also use the reduced version of MN40 (**ReducedMN40**) to simulate data scarcity. ScanObjectNN (SONN) [1] is a recent point cloud object dataset constructed from the real-world indoor datasets such as SceneNN [29] and ScanNet [30]. We use the following versions of SONN: (1) **OBJ_ONLY** which has 2,309 and 581 scanned objects for the training and validation sets respectively and (2) **PB_T50_RS** which is a perturbed version with 11,416 and 2,882 scanned ob-

Table 1. **Overall accuracy on ModelNet40.**

Dataset	Model	CDA	CDA (w/o R)	PointAugment [4]	PointMixup [5]	**PointWOLF**
MN40	PointNet	89.2	89.7	90.8	89.9	**91.1**
	PointNet++	91.3	92.5	92.4	92.7	**93.2**
	DGCNN	91.7	92.7	92.9	93.1	**93.2**
ReducedMN40	PointNet	81.9	82.7	84.1	83.4	**85.7**
	PointNet++	85.9	87.8	87.0	88.6	**88.7**
	DGCNN	87.5	88.8	88.3	89.0	**89.3**

Table 2. **Overall accuracy on ScanObjectNN.**

Dataset	Model	CDA	PointAugment [4]	PointMixup [5]	**PointWOLF**
OBJ_ONLY	PointNet	76.1	74.4	-	**78.7**
	PointNet++	86.6	85.4	88.5	**89.7**
	DGCNN	85.7	83.1	-	**88.8**
PB_T50_RS	PointNet	64.0	57.0	-	**67.1**
	PointNet++	79.4	77.9	80.6	**84.1**
	DGCNN	77.3	76.8	-	**81.6**

jects for the training and validation sets respectively. Both have 15 classes. For part segmentation we adopt **ShapeNet-Part** [26] which is a synthetic dataset contains 14,007 and 2,874 samples for training and validation sets. ShapeNet-Part consists of 16 classes with 50 part labels. Each class has 2 to 6 parts.

Implementation Details. All models and experiments are implemented in PyTorch. For PointNet and PointNet++, the PyTorch implementation by [31] was used with a minimum modification. With DGCNN, we use the official PyTorch code by the authors. We train each model with a batch size of 32 for 250 epochs. Note that for maximal fairness and consistency, we reproduced the numbers for every baselines except for PointMixup [5] and followed the evaluation protocol of [5] for every case. For our framework, the augmentation strength of PointWOLF was controlled by AugTune. Indeed in our framework, the *difficulty coefficient* λ is the only hyperparameter to tune. We used $\lambda = 0.1$ for synthetic datasets and $\lambda = 0.3$ for all real-world datasets. For more details, see the supplement.

Baselines. We compare our framework (PointWOLF with AugTune) with the following data augmentation methods: **(1)** A conventional DA (**CDA**) that performs the global similarity transformation (*e.g.*, rotation along the up-axis, scaling, and translation) with point-wise jittering as [3]. **(2) PointAugment** [4] performs shape-wise transformation and point-wise displacement by learning an augmentor network. For datasets on which the models have not been evaluated in the literature, we use the authors' official implementation of [4]. **(3) PointMixup** [5] uses the interpolated sample between two point clouds.

4.1. Shape Classification and Part Segmentation

We evaluate our methods on shape classification using a synthetic dataset (MN40) and a real-world dataset (SONN). Also we conduct experiments on a synthetic dataset (ShapeNetPart) for part segmentation.

Shape Classification. Table 1 shows that our PointWOLF achieves consistent improvements in overall accuracy on *both MN40 and ReducedMN40 with all three models* compared to other augmentation methods (CDA, PointAugment, and PointMixup). Observe that MN40 and ReducedMN40 are pre-aligned synthetic datasets and interestingly CDA without rotation denoted by CDA (w/o R) outperforms CDA. Despite the saturated datasets, PointWOLF improves overall accuracy by 1.6 % compared to the best performing baseline on ReducedMN40 with PointNet.

Next, Table 2 shows the experimental results on SONN that is a more challenging and diverse real-world dataset. As expected, diverse and realistic augmented samples from PointWOLF significantly improve the performance on *both OBJ_ONLY and PB_T50_RS with all three models*. Specifically, on PB_T50_RS with PointNet++, the performance gains are 4.7%, 6.2%, and 3.5% compared to CDA, PointAugment, and PointMixup, respectively. Our PointWOLF benefits the models more on the challenging cases with real-world data.

Part Segmentation. Given a point cloud $\mathcal{P}$, part segmentation is a point-wise classification where a model predicts a label for each point $\mathbf{p}_i$. In part segmentation, to derive the mixing ratio $\tilde{\alpha}$ in (7) at the object-level, we simply used the average of the pixel-wise confidence scores for our AugTune, *i.e.*, $c_{\mathcal{P}} = \sum_i c_{\mathbf{p}_i} / |\mathcal{P}|$. Our experiments in Table 3

Table 3. **Overall mean IoU ($mIoU$) on ShapeNetPart.**

Method	air plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	labtop	motor bike	mug	pistol	rocket	skate board	table	mIoU
PointNet	81.8	**74.7**	**80.2**	71.9	89.6	71.5	90.3	84.9	79.5	95.2	**65.2**	91.1	**81.1**	55.1	72.8	82.2	83.5
+PointWOLF	**82.5**	73.3	78.8	**73.2**	89.6	**72.2**	**91.2**	**86.2**	**79.7**	95.2	64.6	**92.5**	80.2	**56.6**	**73.1**	82.2	**83.8**
PointNet++	81.9	83.4	86.4	**78.6**	90.5	64.7	**91.4**	83.1	83.4	95.1	**69.6**	**94.7**	**82.8**	56.9	**76.0**	**82.3**	84.8
+PointWOLF	**82.0**	**83.9**	**87.3**	77.6	**90.6**	**78.4**	91.1	**87.6**	**84.7**	**95.2**	62.0	94.5	81.3	**62.5**	75.7	83.2	**85.2**
DGCNN	82.2	**75.1**	81.3	**78.2**	90.6	73.6	90.8	87.8	84.4	95.6	**57.8**	92.8	**80.6**	51.5	73.9	82.8	84.8
+PointWOLF	**82.9**	73.3	**83.5**	76.7	**90.8**	**76.7**	**91.4**	**89.2**	**85.2**	**95.8**	53.7	**94.0**	80.1	**54.9**	**74.3**	**83.4**	**85.2**

Table 4. **PointWOLF Ablation.** R: rotate, S: scale, T: translate.

Local Transformation	R	S	T	Accuracy
None				86.6
+R	✓			88.1
+S		✓		88.6
+T			✓	89.5
+R, S, T	✓	✓	✓	**89.7**

show that on ShapeNetPart [26], PointWOLF consistently improves mean IoU (mIoU) over baselines (0.3% over PointNet, 0.4% over PointNet++ and DGCNN), demonstrating the applicability of PointWOLF to point-wise tasks.

4.2. Analyses on PointWOLF and AugTune

We conduct ablation studies and analyses on SONN (OBJ_ONLY) dataset with PointNet++ to analyze the significance of each component of PointWOLF and AugTune.

Local Transformation Ablations. Table 4 reports the ablations on three types of local transformations in PointWOLF: rotation (R), scaling (S), and translation (T). PointWOLF with no local transformations is equivalent to PointNet++ [3] with CDA. All three types of local transformations contribute to the accuracy gain. The best performance is obtained by +RST which utilizes all three local transformations, providing 3.5% improvement over the baseline with no local transformations denoted by 'None'.

AugTune Ablations. We evaluate how effectively AugTune controls the augmentation strengths given suboptimal augmentation ranges. We set the augmentation ranges S = (ρ_r=15°, ρ_s=2, ρ_t=1) and use the multiples of the augmentation ranges: kS=($k\rho_r$, $k\rho_s$, $k\rho_t$). Table 5 shows that PointWOLF **w/ AugTune** outperforms PointWOLF **w/o AugTune** by 0.4 % $\sim$ 1.9 %. Our AugTune simplifies and accelerates the augmentation strength tuning with one *difficulty coefficient* λ. Our AugTune also benefits other augmentation methods, e.g., CDA, (see the supplement).

Interpolation Space for AugTune. Two interpolation spaces can be considered for AugTune: the input data space and the transformation (parameter) space. Although directly tuning the transformation parameters seems natural, we have experimentally shown that AugTune in the input data space is a sensible choice. Table 6 shows the superiority of AugTune in the input data space regarding both performance and computational efficiency. For N points and M anchor points, AugTune in the transformation (parameter) space requires computing a new transformation for each point and each anchor point in $O(MN)$. Contrarily, AugTune in the input data space simply interpolates the points (*i.e.*, $\alpha\mathbf{p} + (1-\alpha)\mathbf{p}'$ for each $\mathbf{p}$) in $O(N)$.

Table 5. **Search Space Robustness Comparison.**

Search Space	w/o AugTune	w/ AugTune
S	88.8	**89.2**
$2S$	87.6	**88.6**
$3S$	86.1	**88.0**

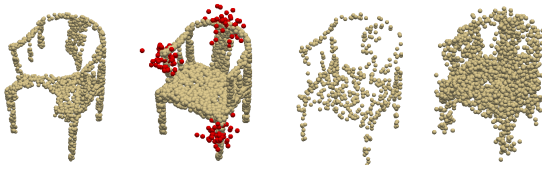
Table 6. **Interpolation space for AugTune.**

Space	Accuracy	Complexity
Transformation Space	88.1	$O(MN)$
Input Data Space	89.7	$O(N)$

4.3. Robustness to Corruption

Additional studies demonstrate our PointWOLF improves the robustness of models against various corruptions as shown in Figure 3. First, we consider two *local* corruptions: (1) **LocalDrop** drops $\mathcal{C}$ local clusters and (2) **LocalAdd** adds $\mathcal{C}$ local clusters where a cluster consists of K nearest points from a randomly selected cluster center point. We used $K = 50$ in both cases. Second, to examine the general robustness to global corruption, we perform random point-wise (3) **Dropout** with a dropout rate $r \in \{0.25, 0.5, 0.75\}$ and (4) **Noise** perturbation by offsets drawn from a Gaussian distribution with standard deviation $\sigma \in \{0.01, 0.03, 0.05\}$.

We trained PointNet++ with CDA (baseline) and PointWOLF and evaluated them on corrupted samples by the local and global corruptions above. Experimental results on MN40 in Table 7 show that compared to CDA, PointWOLF consistently and significantly improves the robustness against various corruptions. Importantly, the gain over the baseline significantly increases as the amount of corrup-



(a) LocalDrop (b) LocalAdd (c) Dropout (d) Noise

Figure 3. **Illustration of Local and Global Corruption.** (a) and (b) are local corruptions while (c) and (d) are global corruptions.

Table 7. **Robustness to Corruption.**

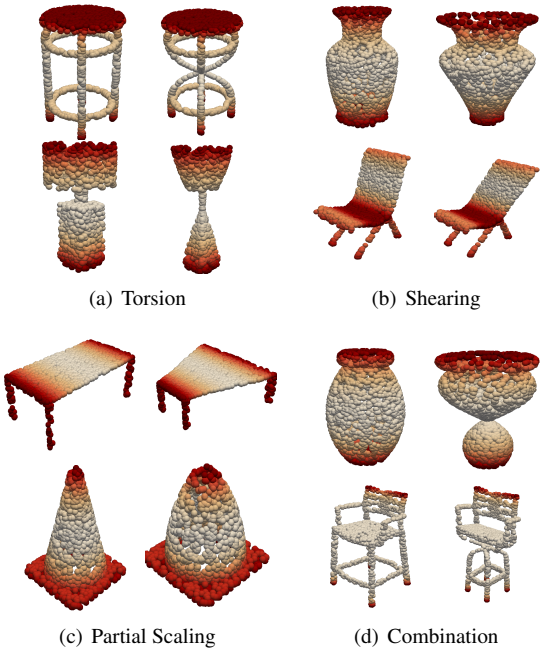
Corruption		CDA	PointWOLF
LocalDrop	$\mathcal{C}$=3	67.0	68.8 (1.8↑)
	$\mathcal{C}$=5	63.2	66.5 (3.3↑)
	$\mathcal{C}$=7	52.8	60.0 (7.2↑)
LocalAdd	$\mathcal{C}$=3	73.9	77.2 (3.3↑)
	$\mathcal{C}$=5	63.5	69.4 (5.9↑)
	$\mathcal{C}$=7	51.5	64.6 (13.1↑)
Dropout	r=0.25	91.2	92.2 (1.0↑)
	r=0.5	84.0	90.4 (6.4↑)
	r=0.75	29.5	60.8 (31.3↑)
Noise	σ=0.01	91.5	93.0 (1.5↑)
	σ=0.03	78.8	87.6 (8.8↑)
	σ=0.05	22.9	45.1 (22.2↑)

tions increases: 7.2% for LocalDrop ($\mathcal{C} = 7$), 13.1% for LocalAdd ($\mathcal{C} = 7$), 31.3% for Dropout ($r = 0.75$), and 22.2% for Noise ($\sigma = 0.05$). We believe that the diverse samples augmented by locally weighted transformations in PointWOLF help models to learn more robust features against both 'local' and 'global' corruptions.

4.4. Qualitative Analysis

Although PointWOLF essentially makes use of simple transformations such as rotation, scaling, and translation, we interestingly find that PointWOLF often mimics highly advanced yet realistic global deformations like torsion and shearing which *cannot* trivially be applied to point clouds. We achieve this by (1) projecting the transformations to random subsets of the axes and (2) allowing AugTune to identify "beneficial" cases which interestingly turn out to be a set of realistic deformations. Figure 4 displays several such examples. For instance, when two anchor points are located at the top and bottom of the stool in Figure 4(a), a *torsion* occurs when it rotates only along the up-axis while preserving the near-anchor shapes of bright regions.

Similarly, a combination of local scaling and translation produce *shearing* or *partial scaling*. In fact, many advanced deformations that naturally preserve the shape identity are commonly defined by combinations of simpler transformations. In this sense, PointWOLF can adaptively allow a set of local transformations that often mimic advanced deformations. Importantly, seeing how these visually explainable augmentations from local transformations also bring empirical benefits, understanding and exploiting local structures are crucial for successful DA on point cloud.



(a) Torsion (b) Shearing

(c) Partial Scaling (d) Combination

Figure 4. **Advanced Deformations by PointWOLF.** In each transformations, the locally transformed samples (right) are generated from original samples (left).

5. Conclusion

We propose a novel point cloud augmentation method, PointWOLF, which augments point clouds by weighted local transformations. Our method generates diverse and realistic augmented samples with smoothly varying deformations formulated as a kernel regression and brings significant improvements on point cloud tasks across several datasets. Moreover, to find an optimal augmentation in an expansive search space, our AugTune adaptively controls the strength of augmentation during training with a single hyperparameter. Our findings show that the augmentations we produce are not only visually realistic but also beneficial to the models, further validating the importance of understanding the local structure of point clouds.

Acknowledgments. This work was supported by ICT Creative Consilience program(IITP-2021-2020-0-01819) supervised by the IITP, Research on CPU vulnerability detection and validation (No. 2019-0-00533), the National Research Council of Science & Technology (NST) grant by the Korea government (MSIT) (No. CAP-18-03-ETRI), and Samsung Electronics.

References

[1] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *ICCV*, pages 1588–1597, 2019. 1, 3, 5

[2] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 77–85, 2017. 1, 2, 3

[3] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, pages 5099–5108, 2017. 1, 2, 3, 6, 7

[4] Ruihui Li, Xianzhi Li, Pheng-Ann Heng, and Chi-Wing Fu. Pointaugment: An auto-augmentation framework for point cloud classification. In *CVPR*, pages 6377–6386, 2020. 2, 6

[5] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees G. M. Snoek. Pointmixup: Augmentation for point clouds. In *ECCV*, pages 330–345, 2020. 2, 5, 6

[6] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2

[7] Maximilian Jaritz, Jiayuan Gu, and Hao Su. Multi-view pointnet for 3d scene understanding. In *ICCV*, pages 3995–4003, 2019. 2

[8] Charles Ruizhongtai Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, pages 5648–5656, 2016. 2

[9] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, pages 945–953, 2015. 2

[10] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *CVPR*, pages 186–194, 2018. 2

[11] Christopher B. Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pages 3075–3084, 2019. 2

[12] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel CNN for efficient 3d deep learning. In *NeurIPS*, pages 963–973, 2019. 2

[13] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *ICCV*, pages 8957–8965, 2019. 2

[14] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 2, 3, 5

[15] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, pages 4490–4499, 2018. 2

[16] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *CVPR*, pages 984–993, 2018. 2

[17] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *NeurIPS*, pages 828–838, 2018. 2

[18] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, pages 8895–8904, 2019. 2

[19] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, pages 6410–6419, 2019. 2

[20] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph CNN for learning on point clouds. *ACM Trans. Graph.*, (5):146:1–146:12, 2019. 2

[21] Wenxuan Wu, Zhongang Qi, and Fuxin Li. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, pages 9621–9630, 2019. 2

[22] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, pages 6438–6447, 2019. 2

[23] Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, pages 113–123, 2019. 2

[24] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. In *NeurIPS*, pages 6662–6672, 2019. 2

[25] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In *NeurIPS*, pages 18613–18624, 2020. 2

[26] Li Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas J. Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Trans. Graph.*, (6):210:1–210:12, 2016. 3, 6, 7

[27] Elizbar A Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, (1):141–142, 1964. 4

[28] Geoffrey S Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 359–372, 1964. 4

[29] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. Scenenn: A scene meshes dataset with annotations. In *3DV*, pages 92–101, 2016. 5

[30] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 2432–2443, 2017. 5

[31] yanx27. Pytorch implementation of pointnet and pointnet++. https://github.com/yanx27/Pointnet_Pointnet2_pytorch, 2020. 6