PaRot: Patch-Wise Rotation-Invariant Network via Feature Disentanglement and Pose Restoration
Supplementary Material

Dingxin Zhang[*]**, Jianhui Yu**[*]**, Chaoyi Zhang, Weidong Cai**

School of Computer Science, University of Sydney, Australia
dzha2344@uni.sydney.edu.au, {jianhui.yu, chaoyi.zhang, tom.cai}@sydney.edu.au

In the supplementary material, we first introduce the details about the training procedure and the network structures of the PaRot architectures in Section 1. We then discuss the effect of three loss functions proposed in the disentanglement modules in Section 2. Section 3 illustrates the restored pose feature of the inter-scale learning. Finally, we provide additional ablation study results of hyperparameter setting in Section 4 and experimental results related to segmentation in Section 5.

1 Implementation Details

1.1 Experimental Setting

The model is evaluated with PyTorch in Nvidia RTX3090. Settings about generating patches are introduced in the main paper.

Our total loss function $\mathcal{L}_{total}$ is defined as:

$$\begin{aligned}\mathcal{L}_{total} = \mathcal{L}_{cls} + \alpha_\ell \mathcal{L}_{equi\_\ell} + \mathcal{L}_{orth\_\ell} + \beta_\ell \mathcal{L}_{inv\_\ell} \\ \alpha_g \mathcal{L}_{equi\_g} + \mathcal{L}_{orth\_g} + \beta_g \mathcal{L}_{inv\_g},\end{aligned} \quad (1)$$

where $\mathcal{L}_{cls}$ is the cross-entropy classification loss, and the subscripts ℓ and g denote the loss function belonging to local-scale and global-scale disentanglement modules respectively. Moreover, α_ℓ, α_g, β_ℓ, and β_g are the weighting parameters adjusting the contribution of different loss functions from local and global scales. We set α_ℓ, α_g, β_ℓ, and β_g to 0.2, 0.1, 0, and 0, respectively. The reason why we set the invariant loss function L_{inv} to 0 is discussed in Section 2.

For classification, the input point clouds are randomly scaled in the range of [0.67, 1.5] for augmentation during training, and the training epoch is 250 with batch size of 32. Adam optimizer is utilised and the learning rate is initialised to 1e-3, scheduled to 1e-5 with cosine annealing scheduler. The momentum and weight decay are set to 0.9 and 1e-6 respectively. For segmentation, the experimental settings are the same as those of classification, except that N_g is change to 64 and k_{intra} to 16. We concatenate the one-hot class label vector to the last feature layer following the implementation of PointNet++ (Qi et al. 2017b).

1.2 Model Architecture

The overall architecture of the PaRot model is illustrated in Fig. 1. The details about disentanglement module and

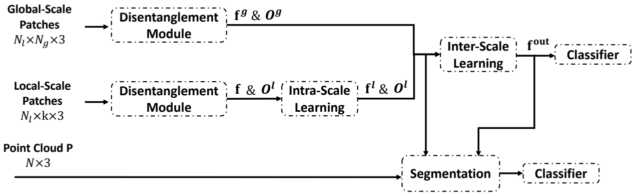


Figure 1: The overall structure of PaRot.

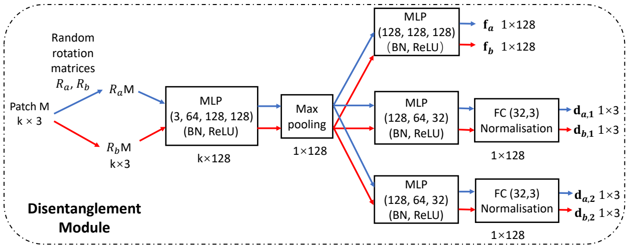


Figure 2: Detailed architecture of the proposed disentanglement module. Two disentanglement modules with similar architecture are assigned to process local-scale patches and global-scale patches independently.

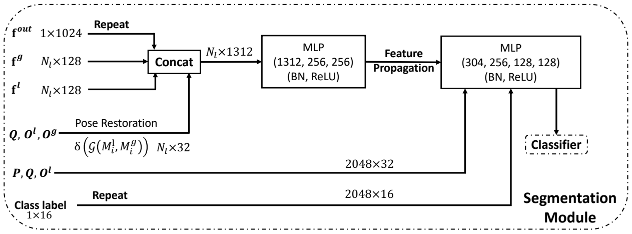


Figure 3: Detailed architecture of the PaRot segmentation module.

[*] These authors contributed equally.

segmentation module are presented in Fig. 2 and Fig. 3 respectively. The architectures of intra-scale learning module, inter-scale learning module and feature propagation module are relative simple and have been explained in the main work, therefore we only provide the information with written expressions. It is worth mentioning that all geometric relation encoding functions, *i.e.*, $\delta(\cdot)$, consist of a fully connected layer with output channel number of 32, a batch normalisation layer, and an ReLU.

In the intra-scale learning module, we implement EdgeConv (Wang et al. 2019) by performing k-nn search within Euclidean space, where we take as input the rotation-invariant features from two patches as well as the encoded pose feature. The channel numbers of the MLP inside intra-scale learning module are 288, 128, 128, and 128. The inter-scale learning module is responsible for both feature aggregation and channel raising, thus the channel of MLP is set to 288, 256, 512, 1024 with LeakyReLU (0.2) as the activation function. The classifiers for classification and segmentation are borrowed from PointNet++ (Qi et al. 2017b).

2 Loss Function

The proposed disentanglement module contains three loss functions, constraining the learned features being either rotation-invariant or rotation-equivariant. In this section, we implement an ablation study to investigate the effectiveness of three loss functions *i.e.*, Eqs. (1-3) of the main work on our model performance.

As shown in Table 1, when no restrictions are applied, the classification accuracy (F) is 90.4%, which is lower than our best model B with the accuracy of 91.0%. Fig. 4 (c) and (f) show that the combination of $\mathcal{L}_{equi}$ and $\mathcal{L}_{orth}$ speeds up the learning of rotation-equivariant vectors and sufficiently enforces two vectors to be perpendicular to each other, which improves the accuracy by 0.6%. Comparing models C-E with B, it is clear that the single application of the loss function cannot achieve the best result. Moreover, we find that when applying all the loss functions (model A), the model performance drops compared to model B. As shown in Fig. 4 (a), (b), (d) and (e), the $\mathcal{L}_{inv}$ of model A decreases faster than model B and has a better performance in first 30 epochs. However, when the number of epoch is sufficiently large, $\mathcal{L}_{inv}$ will hinder the learning of shape content feature and result in a drop of accuracy.

To further analyse the effect of implementing the combination of $\mathcal{L}_{equi}$ and $\mathcal{L}_{orth}$, we visualise the equivariant loss curve and the orthogonal loss curve of B and F in Fig. 4 (c) and (f). When $\mathcal{L}_{equi}$ and $\mathcal{L}_{orth}$ are not used, the equivariant loss will still decrease slowly but the $\mathcal{L}_{orth}$ will keep increasing, which means the learned two direction vectors are parallel to each other and it will cause some ambiguity problems when restoring the pose information. In addition, it shows that learning orientation matrices for local-scale patches are more difficult than for global-scale patches.

Model	$\mathcal{L}_{inv}$	$\mathcal{L}_{equi}$	$\mathcal{L}_{orth}$	Acc.
A	✓	✓	✓	90.6
B		✓	✓	**91.0**
C	✓			90.6
D		✓		90.3
E			✓	88.6
F				90.4

Table 1: Ablation study on loss functions in our disentanglement module. Results on ModelNet40 under z/SO3 are reported.

searching	radius	k_ℓ	z/z	z/SO3	FLOPs
ball query	0.2	64	90.3	90.4	1431M
ball query	0.3	64	90.3	90.5	1431M
knn	-	32	90.7	90.6	1220M
knn	-	64	**90.9**	**91.0**	1431M
knn	-	128	90.8	90.6	1852M

Table 2: Ablation study on generation of local-scale patches. Results on ModelNet40 under z/z, z/SO3.

3 Restored Pose Feature Visualisation

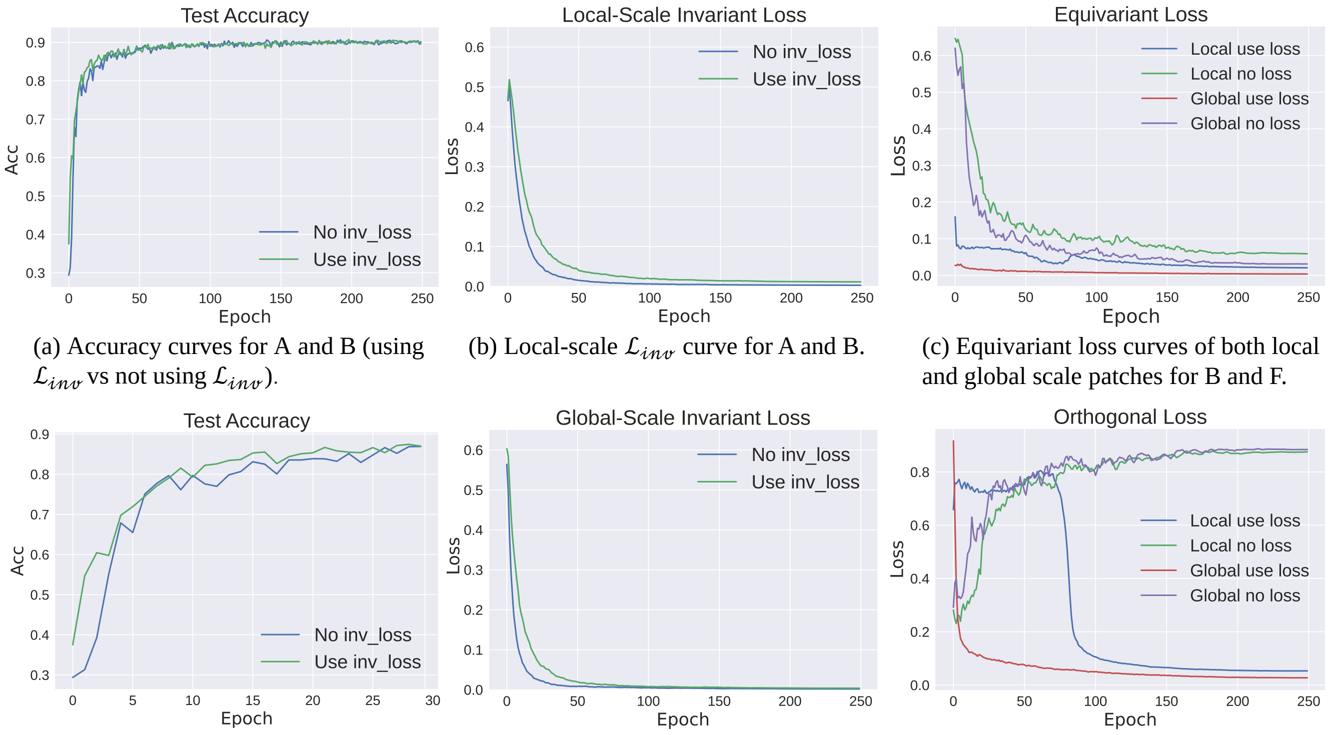
To examine the rotation invariance and effectiveness of our restored pose information, we visualise the restored pose features of inter-scale learning module in ShapeNet part segmentation task. We follow the same procedure of visualising disentangled feature in the main paper, selecting three channels as the RGB values and choose three objects from aeroplane, guitar, and pistol class rotating around z-axis for visualisation. We set N_ℓ to 2048 to provide dense results with saved models.

As it has been discussed in the main paper, the pose restoration module for inter-scale learning aims to explore the relationship between the patch-wise features and the global context. The learned features need to be rotation-invariant and contain both relative positional information and patch-wise orientation information. As illustrated in Fig. 5, restored features are consistent under different orientations. Besides, areas close to centers of objects are generally painted with green, while farther areas are painted in pink. In addition, effected by the orientation of specific patches, some marginal areas are presented in blue and some complicated patches (i.e. the wing-fuselage connection joint and the wheel of pistol) are shown in red.

4 Ablation Study

4.1 Generation of Local-scale Patches

There are two neighbor search methods investigated for generating local-scale patches: k-NN search and ball query. In Table 2, we examine both methods and report their corresponding results with different numbers of neighbors extracted, where we find that models utilising k-nn outperform models employing ball query method. This might because the ball query method strictly constrains the size of generated patches, and k-nn method is more flexible and would generate better patches from sparse and dense parts of point clouds. When setting N_ℓ to 64, the k-nn based model can achieve the best performance, and the computational cost is also moderate.



(a) Accuracy curves for A and B (using $\mathcal{L}_{inv}$ vs not using $\mathcal{L}_{inv}$).

(b) Local-scale $\mathcal{L}_{inv}$ curve for A and B.

(c) Equivariant loss curves of both local and global scale patches for B and F.

(d) Accuracy curves of first 30 epochs for A and B (using $\boldsymbol{\mathcal{L}_{inv}}$ vs not using $\boldsymbol{\mathcal{L}_{inv}}$).

(e) Global-scale $\boldsymbol{\mathcal{L}_{inv}}$ curve for A and B.

(f) Orthogonal loss curves of both local and global scale patches for B and F.

Figure 4: Accuracy and loss curves for model A, B, and F in Table 1.

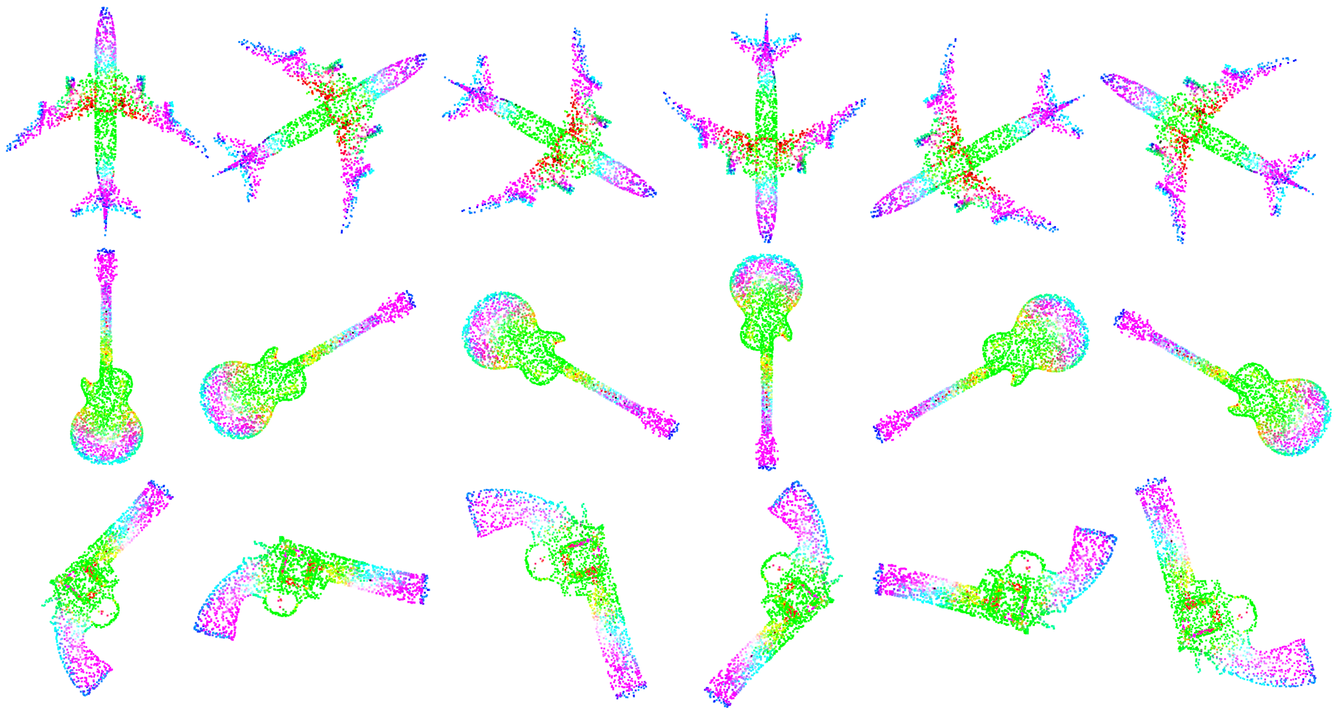


Figure 5: Visualisation of restored pose features in inter-scale learning. From left to right, each sample is rotated 60° around z-axis.

Method	C.mIoU	aero	bag	cap	car	chair	earph.	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate	table
PointNet (Qi et al. 2017a)	74.4	81.6	68.7	74.0	70.3	87.6	68.5	88.9	80.0	74.9	83.6	56.5	77.6	75.2	53.9	69.4	79.9
PointNet++ (Qi et al. 2017b)	76.7	79.5	71.6	**87.7**	70.7	88.8	64.9	88.8	78.1	79.2	**94.9**	54.3	92.0	76.4	50.3	68.4	81.0
DGCNN (Wang et al. 2019)	73.3	77.7	71.8	77.7	55.2	87.3	68.7	88.7	85.5	**81.8**	81.3	36.2	86.0	77.3	51.6	65.3	80.2
RI-Conv(Zhang et al. 2019)	75.3	80.6	80.2	70.7	68.8	86.8	70.4	87.2	84.3	78.0	80.1	57.3	91.2	71.3	52.1	66.6	78.5
GCANet (Zhang et al. 2020)	77.3	81.2	**82.6**	81.6	70.2	88.6	70.6	86.2	**86.6**	81.6	79.6	58.9	90.8	76.8	53.2	67.2	**81.6**
TFN (Poulenard and Guibas 2021)	78.4	80.3	77.3	82.6	74.7	88.8	**76.3**	90.7	81.7	77.4	82.4	**60.7**	93.2	79.4	54.3	**74.7**	79.6
Li et al. (2021)	74.1	81.9	58.2	77.0	71.8	**89.6**	64.2	89.1	85.9	80.7	84.7	46.8	89.1	73.2	45.6	66.5	81.0
VN-DGCNN⋆ (Deng et al. 2021)	75.4	81.0	76.1	76.0	71.4	88.1	59.4	91.3	85.0	80.4	85.5	44.7	92.3	74.5	52.4	68.7	78.9
PaRot	**79.5**	**82.9**	82.1	83.2	**75.7**	89.4	76.1	**91.5**	86.1	81.4	80.3	59.3	**94.3**	**79.7**	**57.0**	73.3	79.2

Table 3: Segmentation per class results and averaged class mIoU on ShapeNet Part dataset under SO3/SO3. ⋆ indicates our reproduced results based on official implementations.

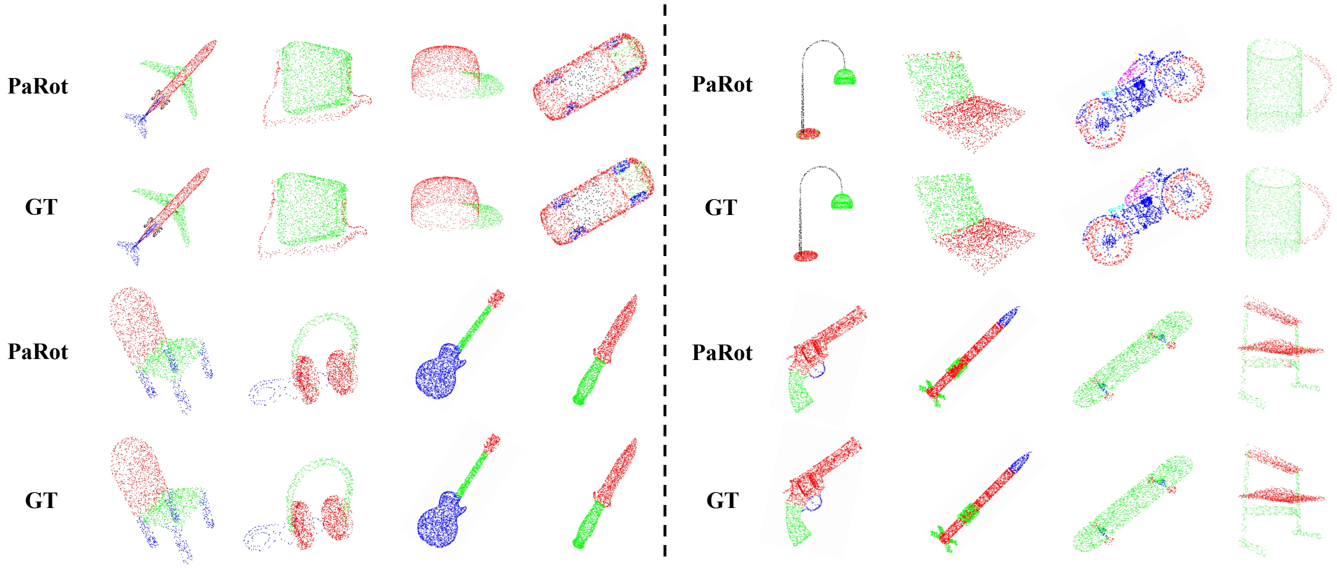


Figure 6: Comparisons between ground truth (GT) annotations and the outputs generated by PaRot for z/SO3 ShapeNet Part segmentation task.

N_ℓ	k_{intra}	N_g	z/z	z/SO3	FLOPs
64	32	32	90.1	90.0	358M
128	32	32	90.8	90.5	716M
256	32	32	**90.9**	**91.0**	1431M
512	32	32	90.6	90.5	2861M
256	8	32	90.3	90.3	995M
256	16	32	90.5	90.5	1140M
256	64	32	90.6	90.5	2012M
256	32	8	90.7	90.7	1273M
256	32	16	90.7	**91.0**	1325M
256	32	64	90.6	90.5	1641M

Table 4: Ablation studies on N_ℓ, N_g, and k_{intra}. Experiments are conducted on ModelNet40 under z/z, z/SO3.

4.2 Hyperparameter Selection

We have investigated k_ℓ in Section 4.1 and there are three other hyperparameters, *i.e.*, the number of patches to generate N_ℓ, the number of points in global-scale patches N_g, and the numbers of neighbours to query in intra-learning k_{intra}. To analyse the impact of those three hyperparameters, we conduct more experiments on ModelNet40 and results are shown in Table 4.

For the number of patches to generate, if we set N_l to be a small value, the generated patches will not be able to cover all the parts of point cloud and results in the reduction of accuracy. However, setting N_l to a very large value will not only significantly increase the computational cost, but also reduce the receptive field of intra-scale learning and harm the performance. The value of k_{intra} also has a high influence to the computational cost, and we found that when $N_l = 256$, setting k_{intra} to 32 will achieve the best performance. Ablation studies on the N_g (number of points sampled for global scale patches) show that the proposed methods can restore efficient inter-scale pose information when only using 8 points for global patches and it can substantially reduce the computational cost. Besides, Table 4 also shows that we can further reduce the computational cost of PaRot by modifying hyperparameters while maintaining a high accuracy.

5 Additional Segmentation Results

We report the results of ShapeNet Part segmentation SO3/SO3 in terms of the per-category mIoU in Table 3. It is shown that typical rotation-sensitive models perform much better in SO3/SO3 than in z/SO3. By augmenting the training samples with rotations, typical models can outperform some rotation-robust methods in segmentation tasks, especially in class cap, lamp, and laptop. However, the proposed PaRot method still outperforms these methods in terms of averaged class mIoU and achieves balance performance among all 16 classes. We also visualise one sample from each object class with our trained z/SO3 model in Fig. 6. Although we can detect some segmentation errors when comparing the ground truths and predicted samples, PaRot provides accurate predictions in most classes.

References

Deng, C.; Litany, O.; Duan, Y.; Poulenard, A.; Tagliasacchi, A.; and Guibas, L. J. 2021. Vector Neurons: A General Framework for SO(3)-Equivariant Networks. In *ICCV*.

Li, F.; Fujiwara, K.; Okura, F.; and Matsushita, Y. 2021. A Closer Look at Rotation-invariant Deep Point Cloud Analysis. In *ICCV*.

Poulenard, A.; and Guibas, L. J. 2021. A Functional Approach to Rotation Equivariant Non-Linearities for Tensor Field Networks. In *CVPR*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *CVPR*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NeurIPS*.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic Graph CNN for Learning on Point Clouds. In *ACM ToG*.

Zhang, Z.; Hua, B.; Chen, W.; Tian, Y.; and Yeung, S. 2020. Global Context Aware Convolutions for 3D Point Cloud Understanding. In *3DV*.

Zhang, Z.; Hua, B.; Rosen, D. W.; and Yeung, S. 2019. Rotation Invariant Convolutions for 3D Point Clouds Deep Learning. In *3DV*.