We appreciate positive and constructive comments and address the main concerns raised by the reviewers below.

Novelty (All) The proposed algorithm is a unique combination of a GCN and a novel rotation-invariant local descriptor for object recognition on 3D point cloud data. (It is rare to use GCN for rotation-invariant 3D point cloud recognition.) We build a hierarchical graph structure on top of the learned rotation-invariant *local* descriptors to extract *global* representations. Our descriptor based on the stochastic local reference frame is more effective in handling rotation transformations than SpecGCN [4] and PointNet++ [3], which do not consider rotation-invariance although they adopt similar hierarchical network structures. Moreover, our GCN models are robust to noise and outliers as illustrated in Figure 3 of the main paper while MLP, which can be regarded as the PointNet++ implementation with our rotation-invariant descriptors, suffers from substantial accuracy loss with the challenges.

Discussion about RIConv [12] and ClusterNet [11] (R2, R4) Both methods start with local feature extraction steps, which require the manual descriptor design using distances, angles, and others while our approach learns local features directly from 3D points based on both shallow and deep learning algorithms, *i.e.,* PCA + MLP, before they are fed into a GCN. Note that our training procedure takes the original 3D points and, consequently, is free from information loss. The manual feature extraction steps in RIConv and ClusterNet may incur the loss and lead to performance degradation. In particular, a local triangle and a descriptor in [12] have many-to-one correspondences because a point is expressed in terms of distances and angles with respect to two reference points, which means the point defines a circular trace.

Test on another challenging dataset (R2, R3) We conduct an additional experiment of part segmentation on ShapeNet. Table 4 presents that the proposed method is also effective to rotation-invariant part segmentation and outperforms the state-of-the-art, especially in the z/SO(3) scenario. The results from other methods are copied from [12].

Table 4: Results on ShapeNet in terms of mIoU.

Method	Input	SO(3)/SO(3)	z/SO(3)
PointNet	xyz	74.4	37.8
PointNet++	xyz+normal	76.7	48.2
RIConv	xyz	75.5	75.3
Ours	xyz	**77.3**	**77.2**

Missing reference and discussion (R1, R4) The main reason for so-called the local reference frame in A-CNN [A1] is not for rotation-invariant recognition but for the definition of a convolution operation with unordered data such as point cloud. Therefore, the accuracy of A-CNN on z/SO(3) is as low as 35.8% according to our experiment based on the official code. It is difficult to discuss RS-CNN [A2] rigorously because the detailed information about the local reference frame is missing. However, RS-CNN extracts hand-crafted features as in RIConv and ClusterNet, and employs additional information, normal vectors, for the extension for the local coordinate system. Moreover, it is not clear if the extension leads to invariance to rotation; although it is tested only for two rotation angles, its accuracy is not impressive at all, especially, considering the very high performance of PointNet++. Both MeshCNN [A3] and Geodesic CNN (G-CNN) [A4] are designed for meshes and their target tasks are different from ours. MeshCNN takes advantage of local hand-crafted features but does not use local reference frames. G-CNN defines a local system of discretized polar coordinates built upon triangular meshes, which is not suitable for the recognition on 3D point cloud data because the surface information is missing in our task.

Clarification (R1, R4) Since there are not many points in the higher level of the graph, the dilated k-NN search on the sparse points is not suitable. We compute PCAs at the lowest level only since it is sufficient to project points onto the (most) local reference frame for rotation-invariant recognition. Meanwhile, GCN is effective to learn global structures in a progressive manner. In practice, computing PCAs at every level does not affect the overall accuracy at all. Although PCA is sometimes unstable, the topology of the constructed graph is identical while the learned descriptors on the unstable node are different. Such a moderate discrepancy can be handled by GCN and may have a regularization effect. This can be a reason for the good performance of our algorithm in the presence of noise and outliers.

Variance of results and computational cost (R1, R3) The variance of our accuracy based on 5 trials with random initializations is 0.036, which implies statistical superiority to other methods. Our algorithm requires 0.56 ms for inference while [12] takes 0.22 ms. The difference is mainly because PCAs are computed on CPU many times. It would be possible to parallelize their computations on GPU and improve speed.

Others [All] We will add missing details (including axis visualizations) and release our source code for reproduction if our paper is accepted.

References

[A1] Komarichev, A., et al.: A-CNN: Annularly convolutional neural networks on point clouds. In CVPR. (2019)

[A2] Liu, Y., et al.: Relation-shape convolutional neural network for point cloud analysis. In CVPR. (2019)

[A3] Hanocka, R., et al.: MeshCNN: A network with an edge. In SIGGRAPH. (2019)

[A4] Masci, J., et al.: Geodesic convolutional neural networks on riemannian manifolds. In ICCV workshops. (2015)