We thank all the reviewers for their diligence, appreciation of our work, and valuable comments / suggestions. Given the tight space constraint, we have done our best to address a majority of each reviewer's questions / comments.

<u>Reviewer 1:</u> A1: (The number of parameters..) We decided upon this architecture to be model-agnostic to downstream tasks and follow existing point cloud learning models (e.g. Wang, Yue, et al. "Dynamic graph CNN for learning on point clouds". ACM TOG 2019.) that similarly go from lower to higher number of parameters, progressively. **A2:** (The self-supervised regression..) We were inspired by the seminal work: C. Doersch, A. Gupta, and A. A. Efros. "Unsupervised visual representation learning by context prediction", ICCV 2015, which proposed a SSL task of *predicting the relative location of random image patches*. Your suggestion of rotations is well received and in fact we can even introduce different "local rotations" of the cover tree balls to make the pretext task even harder and more interesting. **A3:** (l.32-35 The text suggests..) Yes, this is a typo that distorts the meaning. We will rephrase it to convey that any supervised learning on BOTH voxel and raw point cloud representations require massive labeled point clouds, but voxelization has additional burdens of the memory overhead and introducing quantization artifacts. Thanks for pointing this out. **A4:** (l.38-40, is there a..) GANs and AEs can also work on raw PCs. We will omit this misleading statement. **A5:** (briefly explain specific terms...) We will add the explanation of *sample complexity*. Query set Q is defined on l.$110-112$. **A6:** (most importantly l.13 point..) We will rewrite it as "the pre-trained point embeddings are input to the downstream network". **A7:** (l.56 'non-parametric'..) Parametric solid models like *Non-uniform rational basis spline* (NURBS) that best-fits a point cloud. Our intuition was that the surrogate labels generated from a non-parametric cover-tree like approach that could hierarchically data-partition the 3D input point cloud, by closely fitting to the point cloud's distribution, would result in: (i) a much more memory-efficient data structure (as opposed to voxel grids whose memory requirements grow cubically with resolution) and (ii) the labels would be more appropriate as they would avoid regions of "dead-space" (no points). **A8:** (l.142 quadrant ...) The sphere is in the 3D input space ($d = 3$) and quadrants are just the sphere divided by xy and yz-planes passing through the sphere's center. **A9:** (How are the "ball vectors"..) The ball vector is the *barycenter* of the points present in the ball. The blue box in Fig. 2 just maps the point's indices to balls they are part of, to further compute the ball vectors.

<u>Reviewer 2:</u> A1: (The cover tree is not uniquely..) True, the cover tree is not unique. Finding a canonical and "optimal" covering of data points with balls of fixed diameter, where optimality can be defined as "least overlapping balls" or "least dead space of overall covering" etc., is NP-hard. The covertree was a good option as it uses a *fast and greedy construction approach* which takes *sub-linear time* and enjoys strong theoretical approximation guarantees. We also tried picking centers using a *kernel density estimation* (KDE) approach and found that it did not give us any improvements. An ablation study can certainly be added. **A2:** (for ShapeNet part segmentation:) For segmentation, we followed the experiment setup mentioned in Qi, Charles R., et al. "Pointnet: Deep learning on point sets for 3D classification and segmentation". CVPR 2017. Unlike images, our point cloud data had no background. We computed the IoU for each object over all parts occurring in that object and the mean IoU (mIoU) for each of the 16 categories separately. For each object category evaluation, we consider that shape category for query set Q for testing and pick K' classes from the remaining set of classes. We average our results over 10 query sets.

<u>Reviewer 3:</u> A1: (The evaluation is done on few-shot..) Given the scarcity of "FSL on point clouds" papers and that neither PointNet nor PointNet++ are developed with FSL in mind, it is not obvious that PointNet++ will always improve upon PointNet in the FSL setup. **A2:** (It is not clear whether the proposed..) Our SSL method is agnostic to the downstream network. We conducted a limited-setting experiment by SSL pre-training on ModelNet40 using our method and carrying out the classification task with Pointnet++ on ModelNet for a 10-way 10-shot FSL setting. We observed a boost in *classification accuracy* in Pointnet++ (without SSL) from 23.05% to 38.15% (with OurSSL). **A3:** (how good the pre-trained..) Given the limited time, we performed a classification task by pre-training with OurSSL on 4.2K (subset of ShapeNet) point clouds and then using DGCNN for the downstream full supervision task on ModelNet40. *Accuracy*: DGCNN (without SSL + Full supervision) = 91.73%, while DGCNN (with OurSSL + Full supervision) = 92.84%. We also conducted a similar classification task, by pretraining with OurSSL on the entire Sydney10 (sparse) dataset, followed by DGCNN in full supervision on the entire Sydney10 dataset. *Accuracy*: DGCNN (without SSL + Full supervision) = 92.3%, while DGCNN (with OurSSL + Full supervision) = 96.15%.

<u>Reviewer 4:</u> A1: (few-shot learning is somewhat uninteresting..) On the contrary, we find this work very interesting. Our aim is to build a hierarchical model-agnostic SSL model, so that downstream tasks like classification, segmentation (part and semantic) can use the pre-trained point cloud embeddings and adaptively learn extremely fast, with much fewer point cloud examples, thus improving the *transfer learning abilities* of FSL models. It is also widely accepted that FSL is a much more challenging setting because the downstream learner performs poorly, unless the data best covers the distribution of samples per class. FSL is a promising learning paradigm due to its ability to learn out of order distributions quickly with only a few samples. Please see our reply to Reviwer 3 A3 for more experimental results. **A2:** (ln 34: Whether a method ...) Please see our response to Reviewer 1 A3. **A3:** (Table 1.: Please indicate in the table..) Thanks, we will add this in cam-ready. **A4:** (ln 216: VoxelNet[a]..) Thanks for pointing this out. We will replace "VoxelNet" by "VoxelSSL" throughout our paper, as this is a self-supervised method on point clouds.