We thank all the reviewers for their insightful comments and encouraging remarks. Our rebuttal addresses the main questions, but not all, due to space limitations; the minor questions will all be dealt with in the revision.

ML contribution (R1 & R3). Our paper is positioned as an *application* paper, which constitutes a novel application of machine learning techniques to one of the most classic problems in 3D shape perception. Specifically, "PIE-Net is the first deep method that directly estimates the parametric curves (R1)", and produces very promising results.

Our innovation is not intended to be the development of new ML techniques, but lends itself to the application setting. Also, as a first attempt, we would rather showcase a relatively simple approach, rather than a complex one. Our work shows that even a simple learning paradigm such as a region proposal network can already significantly outperform the state of the art. Hence, the value of our contribution is not as a "final say", but in setting up a strong baseline to entice and stimulate future work on a fundamental and frequently encountered task in shape understanding.

"Only evaluated on one dataset, limiting applications (R3)." Sorry, this is not quite true. Please refer to Fig. 8 in the paper (and more results in the supplementary), where we applied PIE-Net trained on ABC to test shapes belonging to *novel* categories, i.e., categories *not found* in the training set. These results are exactly meant to demonstrate the generality and superiority of PIE-Net, even on non-CAD models such as the vases in the last two rows of Fig. 8. In terms of *quantitative* results, R3 is correct in that we only tested on ABC as the ground truth is available.

Paper should be more self-contained (R1 & R2). This is quite easy to fix. Clearly, the issue was limited space, as we wanted to show more experimental and comparison results in the paper. In the revision, we can make some space and move condensed coverage of necessary technical details from the supplementary material to the main paper.

"The only important thing I am really missing is an evaluation of the runtime of the method (R2)." For all the results shown in the paper, the average running time is about 0.5 second for point classification and 3 seconds for curve generation, per point cloud. In comparison, the average running times for point classification by VCM, EAR, and EC-Net are 5.5, 4.0, and 0.8 seconds, respectively — they are all slower than PIE-Net.

Training times for the point classification, the open and closed curve proposal networks were about 23, 12, and 8 hours, respectively, for 100 epochs, on an NVIDIA TITIAN X GPU. We will add all these numbers to the revision.

"τ_c, τ_e: In classification one would expect the threshold be 0.5, not hand crafted (R1)." These thresholds are not learned (hence no "overlearning"); they are set by the user. NMF was employed to select among points that all passed the threshold. While 0.5 is a typical classification threshold, it is far from a "fit-all" choice; it is often unsuitable in the case of *imbalanced classification*, which is our case here. In general, choices of thresholds are problem-dependent.

Performance on random/non-uniform sampling (R1). We sampled 100K points uniformly over each CAD shape, and then sub-sampled 8,096 points non-uniformly via random sampling. We re-trained and re-tested PIE-Net on the new point clouds, keeping all other settings unchanged. The following shows the performance numbers on metrics listed in Fig. 7 in the submission; the original numbers from Fig. 7 are provided in brackets for reference:

ECD: 0.0137 (0.0088); IOU: 0.5976 (0.6223); precision: 0.6816 (0.6918); recall: 0.8319 (0.8584).

As we can see, while there is a slight performance degradation, the new numbers are quite comparable to the original and still outperform all performance statistics obtained by VCM, EAR, and EC-Net, except for one case, VCM with $\tau = 0.12$, which yielded a recall of 0.8385, but it is paired with a very low precision of only 0.3063.

"Point segmentation and edge proposal modules are trained separately ... why not end-to-end (R4)?" We assume that R4 meant that our point *classification* and curve proposal networks were trained separately, which is correct. We will make sure to clearly state that in the revision as requested by the reviewer.

While end-to-end learning holds many merits, there is also a "flip side", e.g., see recent discussions on the limits to end-to-end learning [1, 2]. In our case, while it could be possible to design an end-to-end trained network to perform PIE-Net's tasks, we believe that it will likely be an inefficient approach. The network may be overly complex with a higher-than-necessary capacity, hence prone to overfitting. Our view is that point classification and parametric curve generation are standalone and clearly delineated modules, where attaining the best results for each task individually does not hinder the final outcome when the two modules are executed sequentially. Perhaps another perspective of our design decision is that we are utilizing the inductive bias arising from the pursuit of model simplicity.

References

[1] Tobias Glasmachers. Limits of end-to-end learning. *Proceedings of Machine Learning Research*, 77:17–32, 2017.

[2] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *ICML*, 2019.