BRep Boundary and Junction Detection for CAD Reverse Engineering

Sk Aziz Ali
German Research Center for Artificial Intelligence (DFKI GmbH)
sk_aziz.ali@dfki.de

Mohammad Sadil Khan
German Research Center for Artificial Intelligence (DFKI GmbH)
Mohammad_Sadil.Khan@dfki.uni-kl.de

Didier Stricker
German Research Center for Artificial Intelligence (DFKI GmbH)
didier.stricker@dfki.de

Abstract—In machining process, 3D reverse engineering of mechanical system is an integral, highly important, and yet time consuming step to obtain parametric CAD models from 3D scans. Therefore, deep learning-based Scan-to-CAD modeling can offer designers enormous editability to quickly modify CAD model, being able to parse all its structural compositions and design steps. In this paper, we propose a supervised boundary representation (BRep) detection network BRepDetNet from 3D scans of CC3D and ABC dataset. We have carefully annotated the ∼50K and ∼45K scans of both the datasets with appropriate topological relations (*e.g.*, next, mate, previous) between the geometrical primitives (*i.e.*, boundaries, junctions, loops, faces) of their BRep data structures. The proposed solution decomposes the Scan-to-CAD problem in Scan-to-BRep ensuring the right step towards feature-based modeling, and therefore, leveraging other existing BRep-to-CAD modeling methods. Our proposed Scan-to-BRep neural network learns to detect BRep boundaries and junctions by minimizing focal-loss and non-maximal suppression (NMS) during training time. Experimental results show that our BRepDetNet with NMS-Loss achieves impressive results.

Index Terms—BRep, Boundary Detection, Junction Detection, Scan-to-CAD, Reverse Engineering, NMS.

I. INTRODUCTION

General purpose computer-aided design (CAD) modeling, using many modern APIs [2], [3], [18], [19], greatly assists in the agile process of Computer-Aided Manufacturing (CAM) and commercial applications [1] at industrial scale. Such a process requires skilled engineers and designers to aid in the creation, modification, analysis, or optimization of a design [17]. The same expertise is needed when the physical object of the design is available. In this context, there has been a huge recent interest in CAD *reverse engineering* [7], which was made possible thanks to the availability of highly accurate 3D scanners[1].

How important is fast track '3D Reverse Engineering (RE)' of mechanical system? While the machining process starts with ready-made CAD models and ends with component manufacturing, RE starts with 3D scanning of either damaged parts, archived components, or any spare parts and ends with deducing its parametric CAD model. However, the purpose of RE is not limited to scanning a physical object to obtain a reliable 3D scan, and then concluding the CAD model. Being able to deduce – (i) the intermediate designing steps, (ii) the history of modifications that are involved in parametric modeling of geometric primitives, (iii) designer's intent to order the CAD primitives (*e.g.,*, plane, cylinders, cones) along with different CAD operation steps (*e.g.,*, chamfer, extrusion, revolution, fillet) are the main stages of RE. Therefore, RE can fix the damaged parts of any mechanical system through fast-track prototyping, mass manufacturing, quality analysis, and other downstream application tasks for analyzing structural compositions of real objects.

Why Boundary Representation (BRep) of CAD is ideal for deep learning-based RE? 3D scans are unstructured representations and are different from CAD models, which are generally stored in their STandard for the Exchange of Product (STEP) [4] file format and represented as Boundary-Representations (BReps) [26]. Consequently, the first step towards successful CAD reverse engineering is to learn/infer BRep [26] representation from its corresponding 3D scan, namely *Scan-to-BRep*. This step is considered as the gateway for numerous downstream CAD modeling applications like – grouping and segmentation of BRep faces [8], [11], [14], deducing machining features [6], retrieving CAD sketches [8], predicting mating components for assembling multiple CAD models [12].

Our proposed method brings the problem of 3D scan to BRep parsing closer to real-world settings. In particular, we annotate ∼50K real 3D scans of CC3D [5] and ∼45K high-resolution meshes of ABC [13] datasets with BRep edge, face, and their topological information. In particular, a per-vertex annotation is considered in terms of BRep boundaries and junctions. To understand the BRep chain complex, in other words, the adjacency systems between BRep *body, face, boundary, junction, co-edge/boundary loop, shell, fin, and region* entities, one must parse it into different '*closed and connected loops of co-edges* (CCLC)'. Therefore, once the boundary loops and junction points are detected in a given scan, the next steps like segmenting and grouping them into their respective BRep entities, fitting parametric curves and surfaces on the segmented boundaries and faces [10], [22], inferring the CAD construction steps and operation types involved at each step [8], and retrieval of base sketches [21] follow on. Parsing BRep from 3D scans can also be seen as geometric computation with its *chain complexes* [9]. Once

---

the BRep representation is learned, we gain complete control BRep chain complex (refer to [10] for the definition).

Contributions. (1) We contribute in preparing annotations (ensuring through quality measurements) for deriving BRep chain complex on industrial-level 3D scans from CC3D [5] and ABC [13] datasets. **(2)** Besides, we propose a new network for *boudary and junction* points detection from a 3D scan. **(3)** For this task, the existing state-of-the-art methods [20], [22] rely upon non-maximal suppression (NMS) as a post-prediction step to heuristically prune large number of incorrect predictions. In contrast, our detection network includes a novel way to marginalize false positives/negatives directly during training time. This results in impressive boundary and junction detection recall gains compared to other methods. **(4)** Source code is available here https://github.com/saali14/Scan-to-BRep.
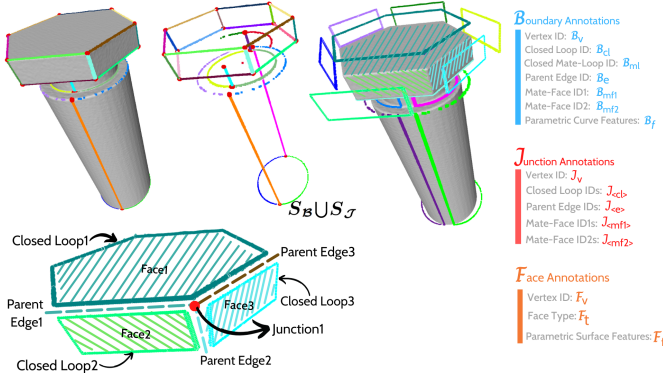
II. BRep Data Annotation Features

Fig. 1. The *left* figures relate a 3D scan with its corresponding BRep topology (color-coded by unique edge and different closed-loop ids). A list of BRep annotations per scan point (on the *right*) covers all attributes under CCLC.

Formally, a BRep topology $\mathcal{T} = (\mathcal{B}, \mathcal{J}, \mathcal{F}, \mathcal{A})$ is defined by sets of faces $\mathcal{F}$, edges $\mathcal{E}$, boundaries $\mathcal{B}$, junctions $\mathcal{J}$, and an adjacency system $\mathcal{A} = (\mathcal{N}, \mathcal{P}, \mathcal{M}) \in \{0, 1\}^{|\mathcal{B}| \times |\mathcal{B}|}$ that ensures directed topological walk along the boundaries to traverse from current edge/face/junction to its *next* $\mathcal{N}$, *previous* $\mathcal{P}$, and *mating* $\mathcal{M}$ counterparts is possible. We redirect readers to [14] and the Figure 1 to understand how different edges form a boundary, how boundaries form closed loops, and how loops form a closed CAD manifold. Each of the BRep topological entities are parametric – *e.g.,* edges are either *spline, arc, lines, ellipse* or other types of curves, whereas faces are *planar, B-Spline, parabolic* or other types of surfaces – with unique ids based on forward/reverse order of the topological walk along the $\mathcal{B}$. The figure 1 illustrates how a 3D scan S and the aforementioned BRep topology is related. For instance, the overlaid scan points $S_{\mathcal{B}}$ and $S_{\mathcal{J}}$, that are annotated as $\mathcal{B}$ and $\mathcal{J}$, are decomposed into closed-loops sharing sharing boundaries. We do not clutter the figure by showing rest of the scan points $S_{\mathcal{F}}$ that are annotated different face Ids from $\mathcal{F}$. The list of attributes in figure 1 are stored for each scan point belonging to $\mathcal{B}$ / $\mathcal{J}$ / $\mathcal{F}$. Among the existing

datsets [5], [13], [24] that are dedicated to feature-based CAD modeling/reconstruction purposes, typical shortfalls are – *(i)* unavailability of 3D scans (often substituted by CAD models), *(ii)* sharp-edge labels. Note that those boundaries which has with small acute angle between its adjacent BRep faces. The main difficulty of detecting boundary/junction points is that they can be smooth and seamless (see the vertical rule along the cylindrical barrel in Figure 1). *(iii)* CAD construction step labels per BRep entity (mainly the faces), and *(iv)* the CAD operations (*e.g.,*extrusion, fillet, cut, chamfer) involved in those construction steps, and *(v)* proper segmentation labels of scan points w.r.t their respective BRep entities (as well as their parameters). The previous variants of CC3D dataset [5], [8] provide annotations mentioned in *(i) – (iv)*. We have complemented annotations in *(v)*. We encourage the community to use our annotations in the future.

A. Annotation Quality Assessment

(a) Boundary and Junction Annotation "Quality Measure" (on CC3D Dataset)

(b) CC3D vs ABC Dataset Statistics

Fig. 2.
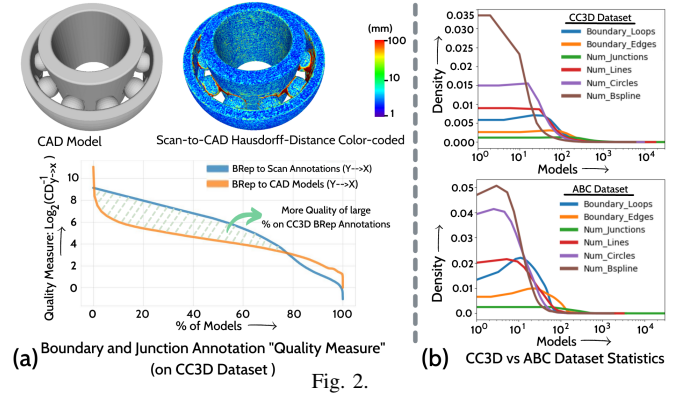
We perform a crucial annotation quality assessment over all samples of *CC3D* dataset. For this, we compute inverse of uni-directional and density-aware Chamfer-distance [25] from parametric BRep Boundary $S_{\mathcal{B}}$ to 3D real scan S in logarithmic scale as:

$$\log_2 \left(CD^{-1} \right) = \log_2 \left(\frac{1}{|S_{\mathcal{B}}|} \sum_{s_i \in S_{\mathcal{B}}} \min_{p \in S} \left(1 - e^{\|s_i - p\|_2} \right) \right)^{-1},$$
(1)

and compare the same measurement when computed from the parametric BRep Boundary $S_{\mathcal{B}}$ to the parametric CAD model of the scan. The figure 2-(a) illustrates a larger percentage of models has significantly higher quality scores when annotations are compared against real scans. This re-affirms that our scan-to-BRep annotations are closer to the real-world setting. It is common that real scans are partial and noisy than their CAD versions which are regular/smooth and complete (see the color-coded Hausdorff distances from scan-to-CAD in figure 2-(a)). Therefore, it is necessary to not have annotations that are overly-smoothed like parametric BRep entities. Besides, the figure 2-(b) also shows a comprehensive statistics on both *CC3D* and *ABC* models in terms of overall model complexities. For instance, it is clear that area covered

under density plots for different sets parametric curves per model is several times more in *CC3D* than *ABC*.
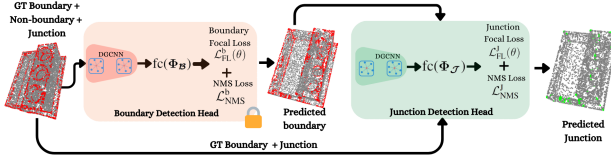
III. B-Rep Boundary and Junction Detection

Fig. 3. Our network architecture.

A. Network Architecture of BRepDetNet

Figure 3 illustrates that our neural network is comprised of separate boundary detection and junction detection heads. Both network heads use DGCNN [23] as point feature encoder that maps $\mathcal{D} : \boldsymbol{S} \mapsto \boldsymbol{\Phi} \in \mathbb{R}^{N \times 128}$ resulting into 128 dimensional point-level deep feature vectors. The two separate DGCNN encoders output boundary and junction embedding vectors $\boldsymbol{\Phi}_{\mathcal{B}}$ and $\boldsymbol{\Phi}_{\mathcal{J}}$ respectively. Next, we apply fully-connected layer to resize fc($\boldsymbol{\Phi}_{\mathcal{B}}$) and fc($\boldsymbol{\Phi}_{\mathcal{J}}$) with final dimensions in $\mathbf{R}^{N \times 1}$.

B. BRep Boundary Detection

To formalize the problem of classifying each point as either a boundary point or an interior point of a 3D scan $\boldsymbol{S} \in \mathbb{R}^{N \times 3}$, we learn a function $f_\theta : \mathbb{R}^3 \to \{0, 1\}$ that maps each point $\mathbf{p}_i$ to a label y_i. Here $y_i = 0$ indicates an interior point, and $y_i = 1$ indicates a boundary point. The function f_θ is parameterized by a neural network weights θ. We use focal loss (FL) [15] as an extension of the binary cross-entropy loss and is designed to address class imbalance. It adds a modulating factor $(1 - p_t)^\gamma$ to the cross-entropy loss, where p_t is the true class probability and γ is the focusing parameter as:

$$
\begin{aligned}
\mathcal{L}_{\text{FL}}^{\text{b}}(\theta) = - \frac{1}{N} \sum_{i=1}^{N} & (1 - f_\theta(\mathbf{p}_i))^\gamma \cdot y_i \log(f_\theta(\mathbf{p}_i)) \\
& + (1 - (1 - f_\theta(\mathbf{p}_i))^\gamma) \cdot (1 - y_i) \log(1 - f_\theta(\mathbf{p}_i)).
\end{aligned}
\tag{2}
$$

Non-Maximal Suppression (NMS) [16] technique is primarily used in object detection tasks to eliminate multiple bounding boxes that are close to each other and keep only the most probable one. We have introduced NMS loss for boundary detection $\mathcal{L}_{\text{NMS}}^b(\theta)$ during the training time as a differentiable approximation to the actual NMS operation:

$$
\mathcal{L}_{\text{NMS}}^{\text{b}} = - \frac{1}{N} \sum_{i=1}^{N} (\alpha t_i \log(p_i) + (1 - \alpha)(1 - t_i) \log(1 - p_i))
\tag{3}
$$

$L_{\text{NMS}}^{\text{b}}(\theta)$ represents the Non-Maximal Suppression Classification loss for a particular prediction $p_i = f_\theta(\mathbf{p}_i)$ and target label t_i. N is the total number of predictions, where p_i is the predicted probability of a point being at the boundary of $\boldsymbol{S}$ and α is a hyperparameter that controls the balance between the two terms in the loss. It's typically set between 0.25 and 0.75. $L_{\text{NMS}}^{\text{b}}(\theta)$ penalizes both false positives and negatives.

This encourages the model to produce more accurate boundary scores. Therefore our total loss for boundary detection is

$$
\mathcal{L}_{\text{total}}^{\text{b}} = \mathcal{L}_{\text{FL}}^{\text{b}} + \mathcal{L}_{\text{NMS}}^{\text{b}}
\tag{4}
$$

C. B-Rep Junction Detection

Detecting the B-Rep junction points is considered as a subordinate task after boundary detection. Let $\mathcal{B}$ be the set of points detected as boundary points (note, this includes *false positives* but not the *false negative* points). A second neural network $g_\phi(\mathbf{x})$ is trained to classify these as either corner points or not. The loss function for this second task is also the binary cross-entropy but restricted to the boundary points:

$$
\begin{aligned}
\mathcal{L}_{\text{FL}}^{\text{J}}(\theta) = - \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x}_i \in \mathcal{B}} & (1 - g_\phi(\mathbf{x}_i))^\gamma \cdot z_i \log(g_\phi(\mathbf{x}_i)) \\
& + (1 - (1 - g_\phi(\mathbf{x}_i))^\gamma) \cdot (1 - z_i) \log(1 - g_\phi(\mathbf{x}_i)).
\end{aligned}
\tag{5}
$$

where z_i denotes label values for $\mathbf{x}_i$ being a junction or not. Similarly, the NMS loss $\mathcal{L}_{\text{NMS}}^{\text{J}}$ and total loss $\mathcal{L}_{\text{total}}^{\text{J}}$ for junction detection are equivalent to the Eq. (3) and (4) respectively.

IV. Experimental Results

We have used 10K randomly sampled scan points as input to out boundary detection head and 4K randomly sampled points from the detected boundaries to the junction detection head. Suppose, during inference time, if the resulting number of boundary points is below the required number of 4K input points, they are randomly up-sampled with duplicated matches. We train our network with batch size 10 and learning rate 0.001 on a 48GB NVIDIA GPU.

A. Baseline Models

We evaluate our proposed approach on ComplexGen [10] and PieNet [22]. Both the baseline models and our proposed BRepDetNet are trained on ABC [13] dataset and tested on ABC [13] and CC3D [5] dataset for a consistent comparison. For PieNet, we only show the results for boundary prediction due to its unstable training and crashing during inference for junction detection.

B. Comparison with The Baseline Models

Table I provides the quantitative results on boundary and junction prediction tasks for ABC [13] and CC3D [5] dataset. We measure *recall* and *precision* metrics as

$$
\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}
$$

$$
\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}
$$

to evaluate the performance of the baseline methods. As shown in Table I, our method outperforms the baselines in recall and precision for both boundary and junction detection. Our method shows significantly higher accuracy in terms of *precision-recall* combined measure. For instance, ComplexGen method detects fewer boundary points and miss to detect a large amount. Although, it does not detect false

Model	ABC				CC3D			
	Boundary		Junction		Boundary		Junction	
	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision
ComplexGen [10]	0.104	0.334	0.045	0.232	0.056	0.461	0.013	0.097
PieNet [22]	0.338	0.391	-	-	0.430	0.529	-	-
BRepDetNet	**0.615**	**0.523**	**0.361**	**0.273**	**0.747**	**0.631**	**0.427**	**0.364**

TABLE I
RECALL AND PRECISION FOR THE BOUNDARY AND JUNCTION DETECTION ON ABC [13] (LEFT) AND CC3D [5] (RIGHT) DATASET. ALL THE MODELS ARE TRAINED ON ABC [13] DATASET.

Model	Model Trained on ABC dataset				Model Trained on CC3D dataset			
	ABC		CC3D		ABC		CC3D	
	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision
Ours w/o NMS Loss	0.454	0.237	0.622	0.383	**0.502**	0.380	0.692	0.646
Ours	**0.615**	**0.523**	**0.747**	**0.631**	0.370	**0.510**	**0.697**	**0.652**

TABLE II
QUANTITATIVE RESULTS FOR BREPDETNET WITH OR WITHOUT NMS LOSS. RECALL AND PRECISION FOR BOUNDARY DETECTION ON ABC [13] AND CC3D [5] DATASET ARE REPORTED. WE ALSO SHOWCASE CROSS-DATASET GENERALIZATION ABILITY OF OUR MODEL.
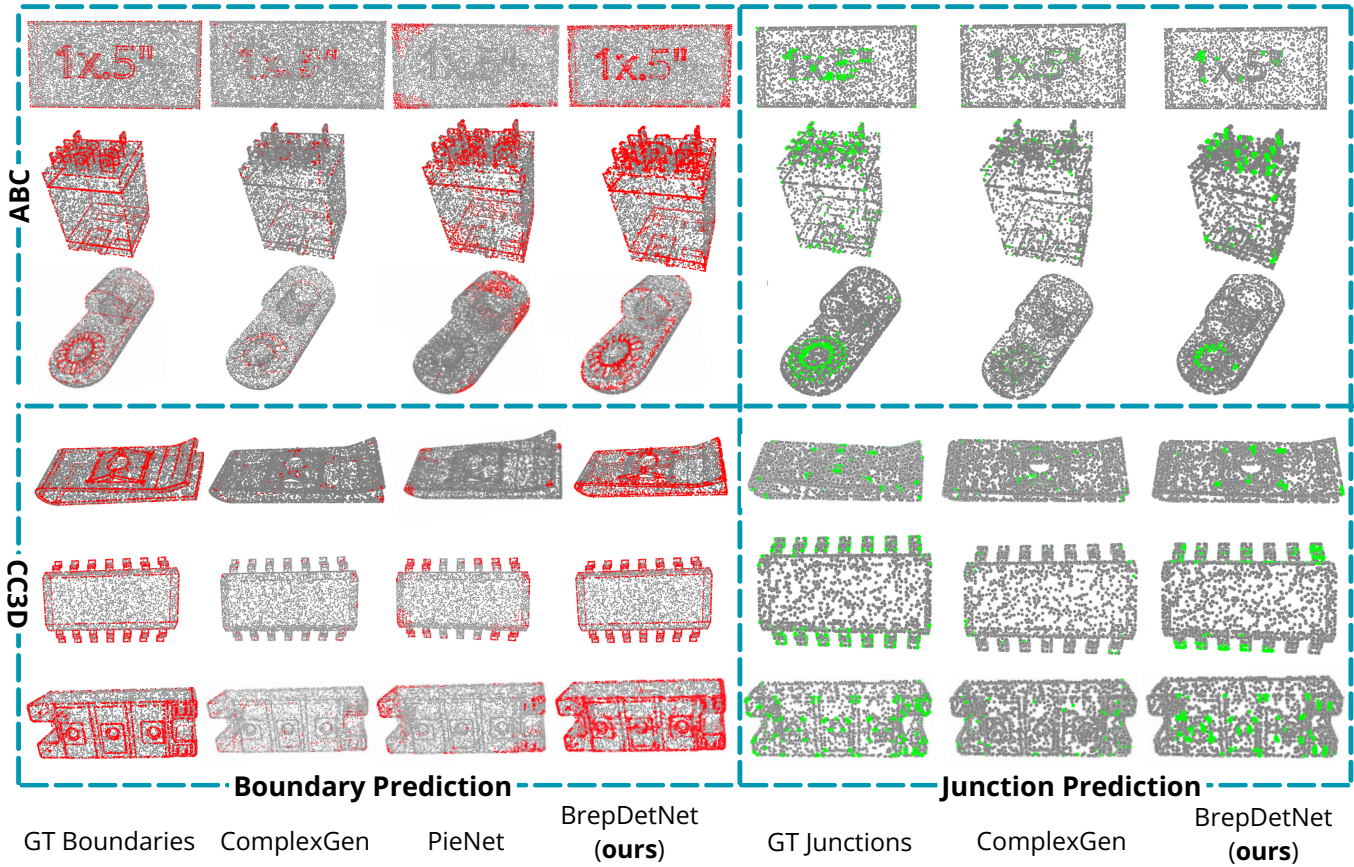

Fig. 4. Visual results for boundary (left) and junction (right) prediction on ABC [13] (top) and CC3D [5] (bottom) dataset. The red points are the boundary points and the green points are the junction points. All the models have been trained on ABC [13] dataset.
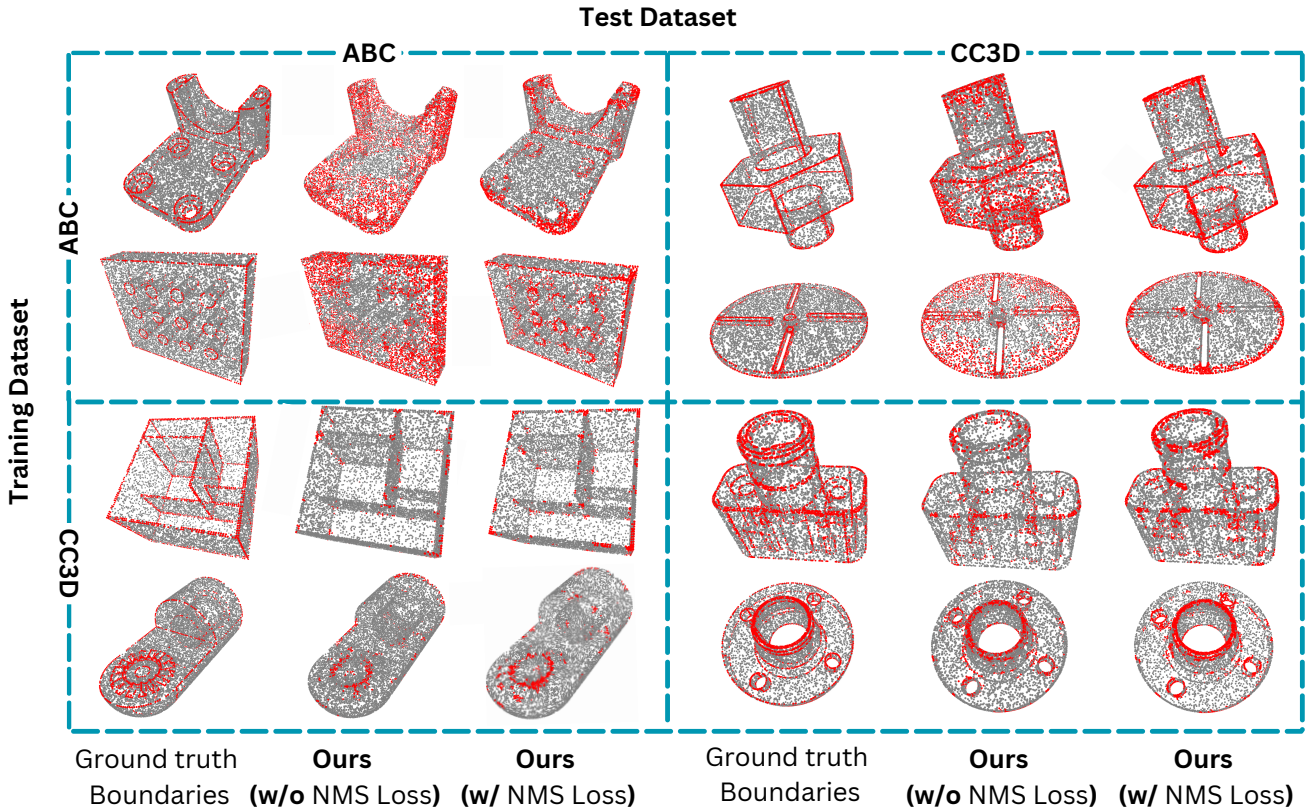
Fig. 5. Visual results for boundary detection for BRepDetNet with and without NMS Loss. The red points are the boundary points. We observe that using NMS loss significantly reduces the number of false positives leading to higher precision in boundary detection.

positives/negatives as compared to PieNet. In contrast to boundary prediction, detecting junction points is more difficult as they are often less than $\sim 1\%$ of the total boundary points. As in Table I, BRepDetNet shows significant improvement in recall compared to the state-of-the-art methods for junction detection tasks. We also observe that despite being trained on a clean and noise free meshes of ABC [13] dataset, our model generalizes well on real 3D scans. The result surpasses that of baseline models in cross-dataset experiments on CC3D [5].

Figure 4 shows some qualitative results between the baseline models and BrepDetNet. As shown, our model predicts more accurate boundary points (red points) compared to other methods. Furthermore, it can be seen that most of the false positive predictions for BRepDetNet are close to the ground-truth boundary points. Thanks to the NMS loss, furthest non-boundary points are not misclassified like PieNet [22].

C. Ablation Study

In this section, we showcase the effectiveness of NMS loss and . Table II reports the precision and recall for the boundary prediction using BRepDetNet with and without NMS loss. In Figure 5, some visual results are provided. To evaluate the performance, we trained both models on ABC [13] and tested on both ABC [13] and CC3D [5] datasets. We use the same strategy with CC3D [5] dataset. This results in four different experimental scenarios as shown in Table II. We observe that

using NMS Loss significantly improves the precision in all four scenarios. This can be explained by the reduction of misclassified boundary points. As demonstrated in Figure 5, NMS loss leads to a lower number of false positives.

V. LIMITATION AND CONCLUSION

In this paper, we propose BRepDetNet, a supervised boundary and junction detection network as the Scan-to-BRep step for 3D reverse engineering. We offer both topological and geometrical annotations that relates 3D real scans with its BRep chain complex. Our proposed approach outperforms the baseline models on both ABC [13] and CC3D [5] datasets. As part of future extension, we plan to make our network end-to-end trainable and resilient even when the annotation quality is poor. We also notice some degree of mis-classification of boundary points directly affect the subsequent junction prediction. Therefore, improving a joint learning of boundary and junction detection tasks, followed by segmentation, and grouping of BRep entities will also be part of our future work.

VI. ACKNOWLEDGEMENTS

REFERENCES

[1] 3D Systems Inc. 2021. geomagic Design X. https://www.3dsystems.com/software/geomagic-design-x, 2021.

[2] AutoCAD. AutoCAD: 2D and 3D CAD. https://www.autodesk.com/products/autocad/.

[3] Autodesk. Autodesk Fusion 360 API . https://autodeskfusion360.github.io/, 2014.

[4] Mangesh P. Bhandarkar and Rakesh Nagi. Step-based feature extraction from step geometry for agile manufacturing. *Computers in Industry*, 41(1):3–24, 2000.

[5] K. Cherenkova, D. Aouada, and G. Gusev. Pvdeconv: Point-voxel deconvolution for autoencoding cad construction in 3d. In *International Conference on Image Processing (ICIP)*, 2020.

[6] Andrew R. Colligan, Trevor T. Robinson, Declan C. Nolan, Yang Hua, and Weijuan Cao. Hierarchical cadnet: Learning from b-reps for machining feature recognition. *Computer-Aided Design*, 147:103226, 2022.

[7] Michal Dúbravčík and Štefan Kender. Application of reverse engineering techniques in mechanics system services. *Procedia Engineering*, 48:96–104, 2012.

[8] Elona Dupont, Kseniya Cherenkova, Anis Kacem, Sk Aziz Ali, Ilya Aryhannikov, Gleb Gusev, and Djamila Aouada. Cadops-net: Jointly learning cad operation types and steps from boundary-representations. In *2022 International Conference on 3D Vision (3DV)*, 2022.

[9] Francesco Furiani, Giulio Martella, and Alberto Paoluzzi. Geometric computing with chain complexes: Design and features of a julia package. *arXiv preprint arXiv:1710.07819*, 2017.

[10] Haoxiang Guo, Shilin Liu, Hao Pan, Yang Liu, Xin Tong, and Baining Guo. Complexgen: Cad reconstruction by b-rep chain complex generation. *ACM Transactions on Graphics (TOG)*, 41(4):1–18, 2022.

[11] Pradeep Kumar Jayaraman, Aditya Sanghi, Joseph G. Lambourne, Karl D.D. Willis, Thomas Davies, Hooman Shayani, and Nigel Morris. Uv-net: Learning from boundary representations. In *CVPR*, 2021.

[12] Benjamin T. Jones, Dalton Hildreth, Duowen Chen, Ilya Baran, Vova Kim, and Adriana Schulz. SB-GCN: structured BREP graph convolutional network for automatic mating of CAD assemblies. *CoRR*, abs/2105.12238, 2021.

[13] S. Koch, A. Matveev, Y. Jiang, F. Williams, A. Artemov, E. Burnaev, M. Alexa, D. Zorin, and D. Panozzo. Abc: A big cad model dataset for geometric deep learning. In *In Computer Vision and Pattern Recognition (CVPR)*, 2019.

[14] Joseph G. Lambourne, Karl D.D. Willis, Pradeep Kumar Jayaraman, Aditya Sanghi, Peter Meltzer, and Hooman Shayani. Brepnet: A topological message passing system for solid models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12773–12782, June 2021.

[15] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[16] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *International conference on pattern recognition (ICPR)*, volume 3, pages 850–855. IEEE, 2006.

[17] MMM Sarcar, K Mallikarjuna Rao, and K Lalit Narayan. *Computer aided design and manufacturing*. PHI Learning Pvt. Ltd., 2008.

[18] Siemens. ParaSolid. https://www.autodesk.com/products/autocad/.

[19] Solidworks. 3D CAD Design Software. https://www.solidworks.com/. Online: accessed 02-June-2022.

[20] Xuefeng Tan, Dejun Zhang, Long Tian, Yiqi Wu, and Yilin Chen. Coarse-to-fine pipeline for 3d wireframe reconstruction from point cloud. *Computers & Graphics*, 106:288–298, 2022.

[21] Mikaela Angelina Uy, Yen-Yu Chang, Minhyuk Sung, Purvi Goel, Joseph Lambourne, Tolga Birdal, and Leonidas Guibas. Point2cyl: Reverse engineering 3d objects from point clouds to extrusion cylinders. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[22] Xiaogang Wang, Yuelang Xu, Kai Xu, Andrea Tagliasacchi, Bin Zhou, Ali Mahdavi-Amiri, and Hao Zhang. Pie-net: Parametric inference of point cloud edges. *Advances in neural information processing systems*, 33:20167–20178, 2020.

[23] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 2019.

[24] Karl D. D. Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G. Lambourne, Armando Solar-Lezama, and Wojciech Matusik. Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences. *ACM Trans. Graph.*, 40(4), jul 2021.

[25] T. Wu, L. Pan, J. Zhang T. WANG, Z. Liu, and D. Lin. Density-aware chamfer distance as a comprehensive metric for point cloud completion. In *In Advances in Neural Information Processing Systems (NeurIPS), 2021*, 2021.

[26] Xianghao Xu, Wenzhe Peng, Chin-Yi Cheng, Karl DD Willis, and Daniel Ritchie. Inferring cad modeling sequences using zone graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6062–6070, 2021.