LEARN THE SOLID, NOT THE FILE: CANONICAL INPUTS FOR NEURAL NETWORKS ON CAD BOUNDARY REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Boundary representation (B-rep) is the standard format used by modern CAD systems for parametric 3D models. It turns out, the exact same solid can be represented by different B-reps: for example, two engineers using different operations, a geometry kernel rebuilding the file, and an export setting repartitioning faces will lead to different B-reps even though the underlying solid remains the same.

We show that existing B-rep encoders are not robust to variation in the B-rep with the same solid on perturbations applied to standard benchmarks, naturally occurring variations inherent to CAD software, and differences in how designers model the same part via a human dataset we created in FreeCAD. The performance of popular B-rep encoders often collapses catastrophically.

We propose the *canonical region graph*, an input representation whose nodes, features and coordinate frame are derived from the solid itself and show theoretical invariance guarantees on repartitioning and rigid motions. It matches the strongest baseline on standard benchmarks, and is stable under every perturbation we test.

1 INTRODUCTION

Computer Aided Design (CAD) is a long-standing technology that has had a profound influence on how just about any object was designed, from everyday consumer goods to architectural infrastructure and aerospace vehicles (Groover & Zimmers, 1983). Boundary representations (B-reps) are the most popular and standard format for most modern CAD software (Weiler, 1986). A B-rep describes a solid by its bounding surface patches (faces), the curves where patches meet (edges), and their connectivity. It gives the precise mathematical instructions to create the object (Stroud, 2006).

A growing body of work trains neural networks directly on B-reps and have been used for wide range of applications such as machine feature recognition (Yao et al., 2026), semantic segmentation (Lou et al., 2023), part classification (Li et al., 2026), CAD retrieval (Usama et al., 2026), generative design (Jayaraman et al., 2022), CAD synthesis (Xu et al., 2024), CAD sequence reconstruction (Zhang et al., 2024), manufacturability and cost estimation (Ballegeer et al., 2026), assembly and joint prediction (Willis et al., 2022) and engineering simulation (Heidari & Iosifidis, 2025).

It turns out that the B-rep is only one of many valid ways to represent the same solid. There are many different decompositions of the same surface: the same geometric boundary can admit arbitrarily many valid topological decompositions into faces and edges (Tierney et al., 2017), placed in any coordinate frame with rigid motions, and its surfaces written as analytic primitives or as splines (ISO 10303-42, 2025). An export setting in the CAD software itself can re-partition a solid's boundary. Moreover, the final B-rep often is decided by factors no mechanical engineer thinks are conscious design decisions (e.g. the order of modeling operations, whether a profile was sketched as one curve or two, a cleanup checkbox, and which geometry kernel produced or last translated the file) (Park et al., 2021; Zhang et al., 2020). Two engineers modeling the same part, or one file passing through two CAD systems, routinely yield different drawings of an identical solid. As such, every solid admits an *equivalence class* of B-reps.

Therefore, one should naturally expect that a B-rep neural network, should be *robust* to the equivalence class of B-reps. That is, any valid B-rep of the solid should produce the same output, or at

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

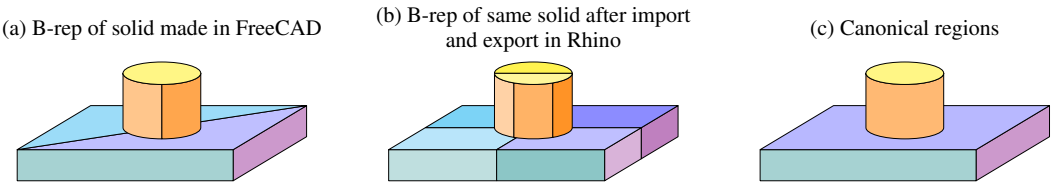


Figure 1: Canonical Regions. We show the face partitionings of the same solid that can naturally arise and how the canonical regions re-partitions the solid. (a) A B-rep that was made in FreeCAD. (b) The same solid after importing it and exporting it in Rhino 3D CAD software. (c) Canonical regions of this solid which is used to construct the canonical region graph. Both (a) and (b) therefore would lead to the same graph using our method, while for the baselines, the graph depends on the original partitioning.

least approximately so. To our knowledge, none of the existing preprocessing methods enforce such robustness; and moreover, we show on standard benchmark datasets that these models often significantly break down in performance when such valid perturbations of the B-rep are applied. Despite the widespread use and the amount of research interest in these B-rep neural networks, such a lack of robustness surprisingly remains.

We present a new graph-based B-rep input representation called *canonical region graph*. There are two key components. The first is *canonical regions*: undo the arbitrary cuts: wherever several faces are pieces of the same underlying surface (e.g. multiple arcs of one cylinder, two halves of one plane) and merge them, so that a node of our graph is a whole surface region, not whichever fragment of it the file happened to contain; two nodes are connected when their corresponding regions touch. The second is *canonical frames*: orient coordinate system in a frame from the solid itself, that is, the origin at its center of mass, and axes along its principal directions. Thus, the frame is invariant to rotations, rigid motions, and coordinate system of the file. Then, the features computed along these canonical regions and boundaries become provably invariant to repartitioning and rigid motions.

We provide a study showing that popular B-rep encoders are not invariant to this equivalence class of B-rep models for the same solid. We demonstrate this on two kinds of perturbations. The first is automatic: to every benchmark test solid, we apply re-partitioning of faces by axis-aligned planes, random rigid motion, re-expression of every analytic surface as its exact NURBS equivalent the operation kernel translators perform, and a plain import/export round-trip through a commercial NURBS-native kernel on the entire test split. The second is human: a dataset we created and labeled by CAD experts in FreeCAD. The dataset was labeled independently by two CAD experts, with one of the experts giving three different ways of modeling the same solid so that we capture both variations possible from one labeler and variations across labels. In all cases, the popular B-rep encoders collapse in performance when such perturbations are introduced, while our method is not only robust but matches or exceeds the compared methods in performance on the standard benchmarks.

2 Related Work

Learning on B-reps. There has been a lot of interest in encoder models on B-reps (Liu et al., 2025; Dai et al., 2025; Liang et al., 2025; Kim et al., 2026). A native B-rep cannot be used directly as input for a standard neural network: it must be first preprocessed into a tractable input format. The field has consolidated to a handful of approaches to turn a B-rep into a neural network input and each inherits specific properties of the B-rep that may not be relevant the underlying solid itself. UV-Net (Jayaraman et al., 2021) attaches to every face a small image grid sampled in the face's own UV parameterization (and to every edge a curve grid), encodes both convolutionally, and passes messages over the face-adjacency graph: the features therefore inherit the parameterization of the faces. BRepNet (Lambourne et al., 2021) defines convolution directly on the coedge structure with learned kernels expressed as topological walks. The topology is from the B-rep, so splitting or merging a face rewrites the very walks the kernels are defined over. AAGNet (Wu et al., 2024) combines per-face UV grids with face and edge attributes for semantic and instance machining-feature recognition. All three express geometry in the file's coordinate frame, so a rigid motion

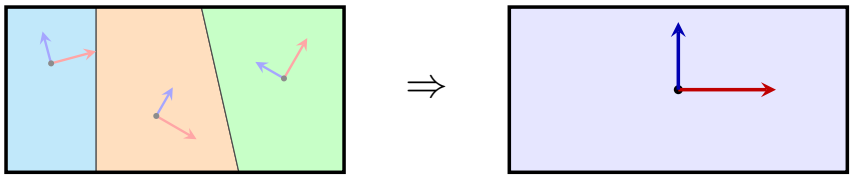


Figure 2: Canonical Frame. Left: Each stored face carries its own local frame (and parameterization), all of which change when the file is re-cut or re-expressed. Right: we use a single canonical frame computed from the whole which cannot change, because the whole does not.

moves every feature. Other approaches include tokenizing the hierarchical tree of the B-rep and using the sequence as the input (Zhang et al., 2024), which directly depends on the hierarchical tree of the B-rep, and reducing the B-rep down to a list of primitives (Wu et al., 2021); however, this can only be used on very primitive CADs. Our canonical graph region method is a new input that focuses on the information about the solid and avoids using information that's particular to how the B-rep was constructed.

Robustness for B-rep Neural Networks. The UV-Net work observed a piece of the problem (Jayaraman et al., 2021), that a face's UV grid can be sampled in several equivalent orders and plain grid convolutions are sensitive to the choice, for which they propose to fix the sampling of a *fixed* face on a fixed partition of the solid's boundary into faces; but it does not address changes to the partitioning, and both the UV-Net and AAGNet inherit this. BRepGAT (Lee et al., 2023) mentions a large gap in performance on between their own dataset and on externally authored models, and attributes it to topological structure differing across CAD systems; however, they don't provide any solutions. Some recent work improve B-rep output validity and error reduction (Hafez & Rashid, 2023; Shen et al., 2025; Liu et al., 2026; Qin et al., 2026b; Qi et al., 2026; Qin et al., 2026a). Lee et al. (2025a) addresses one source of B-rep representation variability by replacing NURBS surfaces with analytic primitives and Jones et al. (2023) learns entity correspondence under topology changes. Sun et al. (2010) simplifies B-reps by removing unwanted features. Ballegeer & Benoit (2026) proposes a rotation-invariant method.

3 The Canonical Region Graph

Like many B-rep encoding methods, our model consumes a graph and is trained with a graph neural network. Existing methods typically use a graph whose nodes are the faces that are stored in the file, with features sampled from that face's parameterization in the file's coordinate frame. Our graph replaces all of this with quantities that can be computed from the solid itself: one node per *canonical region* and coordinate system computed relative to the solid's own frame rather than the file's. The full detailed construction of what we will discuss here is defined end-to-end, along with keys, tolerances, frame, grid, and every constant in Appendix E.

Nodes and Edges: In the canonical region graph, the nodes are the maximally connected regions after undoing the arbitrary cuts (i.e. merging faces that are pieces of one underlying surface). In order to do this, each face gets an exact key identifying its surface in the following way: if the face is an analytic surface (a plane, cylinder, sphere, cone or torus), then the key is based on the surface type and parameters defining it. Next, if the face is saved as a spline (i.e. in NURBS, CAD's generic freeform format that defines a surface using a grid of control points rather than an equation) but actually an analytic surface up to a tolerance, the key is based on the analytic surface. The same applies when the file describes the surface procedurally rather than by shape (e.g. a straight line revolved about an axis, or a circle dragged along a direction), which is a third way a kernel can write down the identical cylinder or plane. Recovering analytic surfaces from freeform representations, and preferring the simplest primitive that fits, is standard practice in reverse engineering (Várady et al., 1997); ours is a deterministic least-squares variant with the selection rule and its failure cases in Appendix E. If it's genuinely a complex NURBs surface, the key is based on the control grid, read

from the underlying *untrimmed* surface. In a B-rep, a face in this case is stored as a bounded portion of a larger underlying surface, and splitting a face only redraws the bounds. Both pieces still point at the same underlying surface, so the key survives any cutting. It is important to note here that the key is computed based on the underlying *untrimmed* surface the face inherits from (as the B-rep contains this information), rather than only the bounded portion defining the face. Two faces merge exactly when their keys are equal *and* they share an edge. The shared-edge condition is required because sometimes two separate features are machined to the same surface (e.g. two identical sub-parts) must still remain two separate regions, and not merge into one. These maximally connected faces become the canonical region and becomes a node in the canonical region graph. Two nodes share an edge if the corresponding canonical regions share a boundary curve.

The Canonical Frame. Several features to be consumed by the graph neural network depend on positions and directions. A coordinate is only meaningful relative to a *frame*. The B-rep's frame is arbitrary because it depends on how the designer oriented the part. We compute one from the solid instead: the origin is the boundary's centroid, and the axes are its principal directions. Each axis's sign is chosen by which side of the perpendicular plane through the centroid carries more boundary area (i.e. a third-moment "lopsidedness"); when a part is too symmetric for this to decide, the two choices are smoothly blended by the lopsidedness weight in order to have continuity. In cases where the principal axis directions themselves are ambiguous, we average feature values across the ambiguous rotations, in closed form. At non-degenerate configurations, the frame transforms equivariantly under rigid motions. At symmetric configurations, where no unique frame exists, we instead construct averaged features that are independent of the ambiguous basis. Keeping mirror images distinct requires some additional care in the frame construction; see Appendix D. Lastly, scale: before any feature is computed, the solid is uniformly rescaled so that its total boundary area is 1. As a consequence, our method is also scale-invariant, a property the popular baselines already have (Jayaraman et al., 2021; Wu et al., 2024).

Node and Edge Features. Each node carries 115 features summarizing its region and each edge has 10 features summarizing the shared boundary between two regions, with every position and direction expressed in the canonical frame. The features are largely the attributes the field has always used such as surface type, area and convexity (Joshi & Chang, 1988) and overlap with that of BRepNet almost item for item (Lambourne et al., 2021). What differs is where they are computed: regions in the canonical frame and not the stored faces in the B-rep's frame. Thus, every feature inherits the desired invariance properties from its region and frame. In particular, there are no UV grids or features that read a parameterization.

Learning on the graph. The network is a standard graph transformer: standardized features pass through an MLP stem, 8 rounds of edge-conditioned multi-head attention with residual feed-forward blocks (width 512, 8 heads, 17.5M parameters), and a linear head producing per-region logits. On tasks that require us to make a prediction based on a B-rep face, the face simply inherits its region's prediction, so any two faces of one region agree by construction. Exact details in Appendix E.

4 Invariance guarantees

We show a number of guarantees for the canonical region graph. Throughout, S denotes the solid and ∂S its boundary surface. We call a partition of ∂S into faces *valid* if every face is contained in a single supporting parametric surface (i.e. patches do not straddle distinct surfaces) and distinct faces meet only along piecewise-regular curves and isolated vertices (i.e. finitely many smooth arcs, excluding degenerate intersections). These conditions hold for the valid B-reps: each B-rep face is a trimmed region of a single supporting surface, and face adjacencies are represented by finitely many topological edges supported by curves. Inputs violating the required B-rep validity conditions are rejected during intake (ISO 10303-42, 2025).

The first result shows that valid B-reps composed of the supported analytic surfaces of the same solid yield the same decomposition into canonical regions. All proofs are in Appendix A.

Proposition 1 (Canonicality)**.** *∂S decomposes uniquely into maximal connected patches of analytic surfaces. For* every *valid partition P, whose faces lie on analytic surfaces (natively or recognized*

Table 1: Invariance under perturbations on MFInstSeg test solids. For each class of perturbations, we compare the perturbed version with the original in whether the region graph structure match and by how much the largest feature differs as a multiple of the standard deviation of that feature in the training set. We show the 50 and 99 percentiles and worst number among the solids.

		exact-feature residual (σ)		
perturbation	identical region graph	p50	p99	worst
diag-split (re-partition)	99.9%	0	4×10^{-7}	0.002
NURBS re-expression	100%	0.0007	0.009	0.022
random rigid motion	100%	0	0.002	0.007

at machine precision) the region graph $G(P)$ is isomorphic (with identical node geometry) to the graph of these patches. G is therefore a function of the solid, not of P.

The next result shows that totals computed from the pieces of the same surface equals the total over the entire surface. This makes features computable from any file and, applied to the whole boundary, makes the canonical frame itself partition-invariant.

Proposition 2 (Additivity over subdivisions)**.** *Surface and edge integrals are additive over subdivisions. Therefore, subdividing any region into valid subregions does not change the resulting features, so they are invariant to the choice of valid partition P of ∂S.*

Next, we show that a partition-invariant model can use only the information from the shape.

Proposition 3 (Lower bound on partition-invariance)**.** *A predictor is partition-invariant if and only if its prediction depends only on the underlying solid geometry. Consequently, for any label Y (e.g., a modeling operation), the risk of a partition-invariant predictor is bounded below by the irreducible disagreement of Y among B-reps representing the same geometry. Thus, any predictor achieving lower risk must exploit information in the B-rep partition that is not determined by the geometry.*

Some features (e.g. the integrated normal, the curvature totals, the dihedral histograms, the occupancy grid) cannot be computed in closed form and they are evaluated by *quadrature* on a triangle mesh of the surface; that is the integral approximated as a finite sum, the integrand evaluated at one sample point per triangle in the mesh and weighted by that triangle's area. The next result guarantees that such features remain invariant up to tolerance proportional to the mesh error.

Lemma 1 (Stability of the mesh-computed features)**.** *The following holds for any mesh-computed feature that is Lipschitz continuous in the quadrature sample positions and area weights. A mesh perturbation of size ε (i.e. a re-triangulation of the same surface that changes each sample position and area weight by at most ε) changes every such feature by at most $O(\varepsilon)$.*

The final result shows that under the canonical frame, the features are invariant under rigid motion.

Proposition 4 (Rigid-motion invariance)**.** *The centroid and principal-axis frame transform equivariantly under rigid motions. Consequently, features expressed in this canonical frame (after the sign blending and degenerate-subspace averaging, which are functions of invariants alone) are exactly invariant to translation and rotation, and remain partition-invariant.*

Verification in practice. Table 1 measures invariance at the representation level. We take the 3000 sample test set of MFInstSeg (Wu et al., 2024) and make three different perturbations: repartitioning, NURBS re-expresson, and rigid motions. We compare the perturbed version with the original by first comparing the canonical region graph. We match the counts and then the largest feature disagreement after optimally matching regions. We show residuals in units of σ, the channel's standard deviation over the training set. We see that the invariance properties hold in practice. Appendix F provides an in-depth analysis of the invariance.

5 Experiments

We compare our method with popular baselines UV-Net (Jayaraman et al., 2021), AAGNet (Wu et al., 2024) and BrepNET (Lambourne et al., 2021). We also include a point cloud method

Table 2: Segmentation under automatic perturbations. We show that our method is robust to different perturbations of the datapoints across a range of benchmark segmentation tasks, while the performance of popular baselines degrades significantly.

dataset	model	clean	axis-split	diag-split	rotation	NURBS	composed
MFInstSeg	AAGNet	0.9851	0.6927	0.4663	0.4811	0.0604	0.0055
MFInstSeg	UV-Net	0.9725	0.5486	0.3001	0.3392	0.0035	0.0026
MFInstSeg	BRepNet	0.9828	0.7176	0.4626	0.7746	0.9564	0.4428
MFInstSeg	DGCNN	0.6622	0.6622	0.6622	0.1231	0.6622	0.1231
MFInstSeg	**Ours**	**0.9862**	**0.9870**	**0.9873**	**0.9867**	**0.9868**	**0.9862**
MFCAD++	AAGNet	0.9851	0.6737	0.4357	0.6027	0.0275	0.0056
MFCAD++	UV-Net	0.9770	0.5964	0.3569	0.4115	0.0079	0.0040
MFCAD++	BRepNet	0.9842	0.7593	0.5209	0.7491	0.9596	0.4661
MFCAD++	DGCNN	0.6571	0.6571	0.6571	0.1095	0.6571	0.1095
MFCAD++	**Ours**	**0.9870**	**0.9887**	**0.9884**	**0.9851**	**0.9856**	**0.9846**
CADSynth	AAGNet	0.9904	0.4698	0.4499	0.9463	0.0360	0.0219
CADSynth	UV-Net	0.9859	0.4257	0.3975	0.9141	0.0250	0.0089
CADSynth	BRepNet	0.9904	0.6487	0.6212	0.9766	0.9121	0.7249
CADSynth	DGCNN	0.6142	0.6142	0.6142	0.2673	0.6142	0.2673
CADSynth	**Ours**	**0.9922**	**0.9926**	**0.9930**	**0.9906**	**0.9909**	**0.9816**

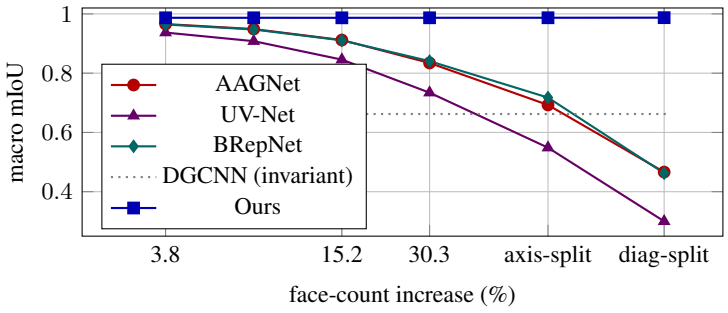


Figure 3: Re-paritition intensity effect on MFInstSeg (log scale in the x-axis). Every B-rep baseline degrades monotonically in the amount of face splitting while ours is flat at parity accuracy. DGCNN's point-cloud input is partition-invariant by construction, but at a massive accuracy cost.

DGCNN (Wang et al., 2019), which does not depend on the B-rep (but less accurate because the point clouds give less information than the B-rep). We use each baseline's published weights where they exist and retrain at a shared budget where they do not. Full details in Appendix B.

There are several ways to automatically generate different valid B-reps for a solid. We explore:

- **axis-split.** Re-partitioning each face by sectioning two axis-aligned planes
- **diag-split.** Re-partitioning each face is sectioned by four planes including diagonals
- **split-k.** Re-partitioning where exactly k faces are split.
- **rotation.** Applies one random rigid motion per part, shared by every model.
- **NURBS re-expression.** Rewrite each analytic surface as its exact B-spline form.
- **composed.** Applies diag-split, rotation and re-expression at once.

Segmentation Benchmark: We use standard benchmark datasets and compare the accuracy on the original task vs accuracy on a task where the datapoints are perturbed automatically. Table 2 shows the results. All accuracy numbers are macro mIoU (per-class intersection-over-union of predicted vs. true face labels), averaged over classes with 1.0 being the perfect score. For each perturbation, the geometry of the solid remains unchanged (up to a very small tolerance). We see that our method has robust performance under these perturbations, while the popular baselines' performances decay significantly. It's worth noting that DGCNN consumes point clouds sampled from the geometry,

Table 3: Retrieval self-identification on 3000 real Fusion 360 parts. We report the rank-1 accuracy of retrieving a part's own clean B-rep (with retrieval database consisting of all the clean B-reps) from a query that is the same solid under one perturbation. DGCNN's queries are re-sampled point clouds so there is some variance even in the columns it's theoretically invariant to.

model	axis-split	diag-split	rotation	NURBS	composed
AAGNet	22.6%	6.3%	25.5%	0.7%	0.2%
UV-Net	5.5%	1.0%	7.3%	0.3%	0.1%
BRepNet	10.6%	7.8%	16.5%	**98.1%**	3.6%
DGCNN	85.9%	85.6%	7.7%	86.3%	7.8%
Ours	**96.1%**	**96.3%**	**88.3%**	80.5%	**76.1%**

Table 4: Predictive churn on 3,000 MFInstSeg solids. Percentage of individual face predictions that change with perturbation, and the fraction of solids with at least one change.

	axis-split	diag-split	rotation	NURBS	composed
faces flipped					
AAGNet	11.90%	27.29%	35.83%	74.68%	96.81%
UV-Net	21.66%	45.69%	50.29%	97.80%	98.19%
BRepNet	10.75%	24.45%	8.98%	1.20%	29.90%
Ours	**0.00%**	**0.00%**	**0.00%**	**0.00%**	**0.02%**
solids affected					
AAGNet	85.63%	100.00%	96.77%	99.93%	100.00%
UV-Net	96.00%	100.00%	99.83%	100.00%	100.00%
BRepNet	86.00%	99.30%	64.10%	20.87%	99.10%
Ours	**0.00%**	**0.00%**	**0.00%**	**0.03%**	**0.33%**

so re-partitioning and re-expression remains unchanged but is affected by rotation. Figure 3 shows how performance monotonically decays for the B-rep baselines as we increase the amount of re-partitioning, while our method remains stable.

Retrieval Benchmark: We build a label-free retrieval benchmark from 3000 real Fusion 360 Gallery parts. The retrieval database consists of the clean B-reps, and each query is the *same* solid under one perturbation. No retrieval training is performed. Table 3 shows the results. BRepNet is best with NURBS perturbation as re-expression changes nothing its features read, so its embeddings barely move but performs poorly under other perturbations. Ours is the only model that consistently performs across perturbations.

Predictive Churn. It is quite common for models to have very different individual predictions, even with similar accuracy. This is known as predictive churn (Jiang et al., 2021); which can be seen as a more direct measure of stability than for e.g. accuracy on downstream tasks. We see in Table 4 that our method has very low churn while the baselines all exhibit a high amount, sometimes near 100% in terms of number of face predictions changed, a clear sign of representation collapse.

Why augmentation is not the answer. A natural approach is to train with the perturbations applied via data augmentation. Table 5 shows the results with each baseline trained with each of the augmentations, evaluated on every perturbation. Augmentation genuinely helps and in all cases, substantially improves the results for its corresponding perturbation; however, the transfer is not as strong across different perturbations. Furthermore in all cases, our method without any data augmentation surpasses the performance on each individual perturbation, suggesting that our method of fixing the problem at the architecture level is superior to augmenting the data.

Kernel-induced re-partitioning. In our automatic perturbations based on re-partitioning, we use `SplitShape` command in OpenCASCADE on the B-rep. We show that our results also hold when we don't use a splitting operator and instead rewrite the CAD program itself in OpenCASCADE in the following way: whenever a sketch is extruded by extent E, we make the same sketch extruded E/2, plus a second copy on a plane offset by E/2, extruded E/2 so in the end it is still the same

Table 5: Data augmentation matrix: every baseline and every augmentation, evaluated on every perturbation. We report macro mIoU on MFInstSeg test set.

model	clean	axis-split	diag-split	NURBS	composed
AAGNet + axis aug.	0.9850	0.9759	0.8558	0.0450	0.0054
+ diag aug.	0.9828	0.9774	0.9703	0.0469	0.0043
+ NURBS aug.	0.9851	0.6778	0.4711	0.9853	0.2358
+ composed aug.	0.9802	0.9421	0.5725	0.9674	0.9502
UV-Net + axis aug.	0.9876	0.9630	0.7622	0.0072	0.0066
+ diag aug.	0.9827	0.9561	0.9421	0.0104	0.0053
+ NURBS aug.	0.9744	0.5420	0.3108	0.9744	0.1893
+ composed aug.	0.9717	0.8152	0.5083	0.9456	0.9326
BRepNet + axis aug.	0.9849	0.9830	0.9192	0.9587	0.3037
+ diag aug.	0.9828	0.9793	0.9780	0.9565	0.6115
+ NURBS aug.	0.9785	0.7238	0.4752	0.9588	0.5832
+ composed aug.	0.9816	0.9740	0.8125	0.9586	0.9639
Ours (no aug.)	0.9862	**0.9870**	**0.9873**	**0.9868**	**0.9862**

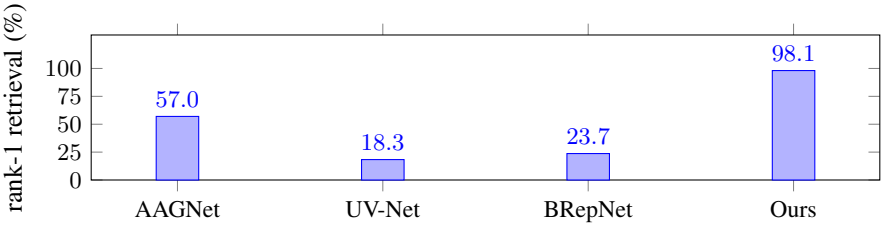


Figure 4: Kernel-induced re-partitioning in OpenCASCADE: rank-1 retrieval accuracy.

Table 6: Region ablation: we keep features the same but test merging (into regions) vs not merging (keeping the existing faces) for the input graph. We use MFInstSeg test split under axis/diag re-partitioning. "Solids stable" is the fraction of test solids whose predictions are bit-identical between the clean file and its re-partitioned version.

	mIoU		solids stable	
	axis	diag	axis	diag
face nodes (merging off)	0.5653	0.3502	4.80%	0.07%
region nodes (merging on)	**0.9828**	**0.9840**	**99.73%**	**99.70%**

solid but now each extrusion is partitioned into two. The dataset we use is 1,049 pairs, generated procedurally based on CAD-Recode (Rukhovich et al., 2025). Figure 4 shows the results and our method outperforms significantly.

Region Ablation In Table 6, we show the importance of face merging in our region-based approach. We compare the robustness under our automatic re-partitioning perturbation using an input graph with the region-based nodes (with merging) vs the input graph without merging (using the original faces as nodes).

Kernel round-trips. Here we explore the effect of a real-world perturbation: a plain import/export round-trip in Rhino, a popular commercial CAD software which uses the openNURBS kernel. Table 7 shows the results: AAGNet and UV-Net collapse catastrophically, while BRepNet holds better: its features do not read surface parameterizations. Our method is still the most stable.

Human study. The human study runs in FreeCAD where we have two experts who labeled 25 parts. One labeler made 3 different FreeCAD scripts for each part and a second labeler made one script for each part. We compare both variation in scripts made by the same expert and variation

Table 7: Robustness to an ordinary Rhino import/export of the entire MFInstSeg test split. We show macro mIoU as in Table 2 and additionally the predictive churn at both face-level and solid-level (if at least one face's prediction in the solid was changed).

	clean	round-trip	faces flipped	solids affected
AAGNet (publ.)	0.9851	0.2838	47.04%	99.50%
UV-Net	0.9725	0.0433	89.98%	100.00%
BRepNet	0.9828	0.9674	0.37%	5.13%
Ours	0.9862	**0.9870**	**0.00%**	**0.04%**

Table 8: The human study: segmentation predictive churn (percentage) between two independently authored FreeCAD scripts of the same geometry.

	AAGNet	UV-Net	BRepNet	Ours
scripts from same expert	11.8	30.1	12.6	**0.5**
scripts from different experts	13.1	15.1	12.5	**0.2**

Table 9: Fusion 360 under the automatic perturbations, same conventions as Table 2

model	clean	axis-split	diag-split	rotation	NURBS	composed
AAGNet	**0.7412**	0.4842	0.3151	0.5512	0.2473	0.0697
UV-Net	0.6977	0.3772	0.2218	0.4576	0.2895	0.0950
BRepNet	0.7162	0.3541	0.2419	0.5167	0.5707	0.2415
DGCNN (points)	0.4025	0.4025	0.4025	0.1436	0.4025	0.1436
Ours	0.6148	**0.6048**	**0.5725**	**0.6153**	**0.6150**	**0.6142**

made by different experts. The results are in Table 8, and full details are in Appendix C. We compare the methods on the segmentation predictive churn on each methods' corresponding MFInstSeg-trained checkpoint for both within-expert and cross-expert B-rep pairs. We see that our method is far more consistent than the baselines.

The cost of invariance (a limitation). Our method has a limitation in situations the information in the B-rep itself can be useful and not using it would hurt performance. One such situation is the Fusion 360 Gallery segmentation, where it labels each face by the modeling operation that created it. As such, a partition-invariant model cannot always recover such labels, while a model free to read the boundary decomposition can exploit residual traces of the history. Table 9 shows the results. We do see that our method performs worse than the B-rep baselines on the clean dataset; however, when automatic perturbations are applied, our robustness overcomes the decay in the baselines when faced with the perturbations.

6 Conclusion

We showed that existing B-rep encoders are not robust to variation in the B-rep used to represent the solid. We proposed the *canonical region graph*, an input representation whose nodes, features and coordinate frame are derived from the solid itself with theoretical invariance guarantees. It matches the strongest baseline on standard benchmarks and is stable under variation of B-rep.

Our findings may have implications for B-rep generators that count re-descriptions of the same solid as distinct or novel samples (Xu et al., 2024; Lee et al., 2025b; Xu et al., 2025; Li et al., 2025a;b) and those that use element-matching reconstruction scores (Guo et al., 2022), which can penalize correct solids decomposed differently from the one reference B-rep.

Future work includes extending the method to use feature trees and construction histories but with invariance constraints in order to leverage potentially useful information contained in the B-rep. Another direction is extending the method to work for assemblies, which are files with multiple parts, instead of just a single part, as well as 2D sketches and drawings.

AI use statement. Anthropic's Claude Fable 5 implemented the method, baselines, and evaluation code, generated the perturbed benchmark variants, assisted in developing the conceptual framework and formulating mathematical claims and proofs, helped design the experimental methodology, and assisted in drafting the manuscript, tables, and figures.

OpenAI's GPT-6 Astra was used as a separate auditing tool for the implementation and mathematical claims. In multiple rounds, it generated and executed adversarial counterexamples and test cases against the implementation and claims. This process led to code and claim revisions, with corrected implementations re-tested against the resulting fixes. The authors reviewed the results and manually assessed the resulting claims and experimental conclusions.

The CAD models in the human-study dataset were authored entirely by human experts. The final manuscript was substantially reviewed and edited by the authors. We take full responsibility for the final content of this work.

Reproducibility statement. Our benchmark experiments use MFInstSeg, MFCAD++, CADSynth, and Fusion 360 Gallery segmentation. Appendix B describes the perturbation procedures, training settings, and evaluation protocol. Appendix E specifies the canonical region graph construction, including feature definitions and numerical tolerances. Appendix C describes the human-study, geometric verification, and evaluation criteria.

REFERENCES

Industrial automation systems and integration—product data representation and exchange—part 42: Integrated generic resource: Geometric and topological representation, 2025. URL `https://www.iso.org/standard/91386.html`.

Matteo Ballegeer and Dries F Benoit. Fov-net: Rotation-invariant cad b-rep learning via field-of-view ray casting. *arXiv preprint arXiv:2602.24084*, 2026.

Matteo Ballegeer, Toon Van Camp, Willem Jaspers, Alp Bayar, Aung Nyein Soe, Martin Roelfs, Dries F Benoit, Bieke Decraemer, and Joost R Duflou. Cad-feature enhanced machine learning for manufacturing effort estimation on sheet metal bending parts. *arXiv preprint arXiv:2605.12266*, 2026.

Yongkang Dai, Xiaoshui Huang, Yunpeng Bai, Hao Guo, Hongping Gan, Ling Yang, and Yilei Shi. Brepformer: Transformer-based b-rep geometric feature recognition. In *Proceedings of the 2025 International Conference on Multimedia Retrieval*, pp. 155–163, 2025.

Mikell Groover and EWJR Zimmers. *CAD/CAM: computer-aided design and manufacturing*. Pearson Education, 1983.

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.

Omar M Hafez and Mark M Rashid. A robust workflow for b-rep generation from image masks. *Graphical Models*, 128:101174, 2023.

Negar Heidari and Alexandros Iosifidis. Geometric deep learning for computer-aided design: A survey. *IEEE Access*, 13:119305–119334, 2025.

Pradeep Kumar Jayaraman, Aditya Sanghi, Joseph G Lambourne, Karl DD Willis, Thomas Davies, Hooman Shayani, and Nigel Morris. Uv-net: Learning from boundary representations. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11698–11707. IEEE, 2021.

Pradeep Kumar Jayaraman, Joseph G Lambourne, Nishkrit Desai, Karl DD Willis, Aditya Sanghi, and Nigel JW Morris. Solidgen: An autoregressive model for direct b-rep synthesis. *arXiv preprint arXiv:2203.13944*, 2022.

Heinrich Jiang, Harikrishna Narasimhan, Dara Bahri, Andrew Cotter, and Afshin Rostamizadeh. Churn reduction via distillation. *arXiv preprint arXiv:2106.02654*, 2021.

Benjamin Jones, James Noeckel, Milin Kodnongbua, Ilya Baran, and Adriana Schulz. B-rep matching for collaborating across cad systems. *arXiv preprint arXiv:2306.03169*, 2023.

Sanjay Joshi and Tien-Chien Chang. Graph-based heuristics for recognition of machined features from a 3d solid model. *Computer-aided design*, 20(2):58–66, 1988.

Mingi Kim, Yongjun Kim, Jungwoo Kang, and Hyungki Kim. Brepcoder: A unified multimodal large language model for multi-task b-rep reasoning. *arXiv preprint arXiv:2602.22284*, 2026.

Joseph G Lambourne, Karl DD Willis, Pradeep Kumar Jayaraman, Aditya Sanghi, Peter Meltzer, and Hooman Shayani. Brepnet: A topological message passing system for solid models. *arXiv preprint arXiv:2104.00706*, 2021.

Jihui Lee, Changmo Yeo, Kyung Cheol Bae, and Duhwan Mun. Replacing nurbs surfaces with analytic surfaces based on isocurve characteristics in b-rep models. *Journal of Computational Design and Engineering*, 12(8):78–106, 2025a.

Jinwon Lee, Changmo Yeo, Sang-Uk Cheon, Jun Hwan Park, and Duhwan Mun. Brepgat: Graph neural network to segment machining feature faces in a b-rep model. *Journal of Computational Design and Engineering*, 10(6):2384–2400, 2023.

Mingi Lee, Dongsu Zhang, Clément Jambon, and Young Min Kim. Brepdiff: Single-stage b-rep diffusion model. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–11, 2025b.

Jing Li, Yihang Fu, and Falai Chen. Dtgbrepgen: A novel b-rep generative model through decoupling topology and geometry. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21438–21447. IEEE, 2025a.

Pu Li, Wenhao Zhang, Weize Quan, Biao Zhang, Peter Wonka, and Dongming Yan. Brepgpt: Autoregressive b-rep generation with voronoi half-patch. *ACM Transactions on Graphics (TOG)*, 44(6):1–18, 2025b.

Yifei Li, Kang Wu, Wenming Wu, and Xiao-Ming Fu. Masked brep autoencoder via hierarchical graph transformer. *arXiv preprint arXiv:2603.14927*, 2026.

JianFei Liang, FaZhi He, RuBin Fan, YiYang Chu, and Xiaohu Yan. Cadcl: Reconstruct parametric cad models from b-rep via contrastive learning. *Journal of Computational Design and Engineering*, 12(10):176–184, 2025.

Yilin Liu, Duoteng Xu, Xingyao Yu, Xiang Xu, Daniel Cohen-Or, Hao Zhang, and Hui Huang. Hola: B-rep generation using a holistic latent representation. *ACM Transactions on Graphics (TOG)*, 44(4):1–25, 2025.

Yilin Liu, Pradeep Jayaraman, Chinthala Reddy, Xiang Xu, and Hooman Shayani. Dualbrep: A dual-field continuous representation for b-rep modelling. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–11, 2026.

Yunzhong Lou, Xueyang Li, Haotian Chen, and Xiangdong Zhou. Brep-bert: Pre-training boundary representation bert with sub-graph node contrastive learning. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 1657–1666, 2023.

Michael A Park, Robert Haimes, Nicholas J Wyman, Patrick A Baker, and Adrien Loseille. Boundary representation tolerance impacts on mesh generation and adaptation. In *AIAA aviation 2021 forum*, pp. 2992, 2021.

Dacheng Qi, Chenyu Wang, Jingwei Xu, Tianzhe Chu, Zibo Zhao, Wen Liu, Wenrui Ding, Yi Ma, and Shenghua Gao. Pointer-cad: Unifying b-rep and command sequences via pointer-based edges & faces selection. *arXiv preprint arXiv:2603.04337*, 2026.

Dafei Qin, Rui Xu, Zeyu Shen, Kaichun Qiao, Hongyang Lin, Qixuan Zhang, Huaijin Pi, Lan Xu, Jingyi Yu, Wenping Wang, et al. Autoregressive b-rep shape generation with parametric surfaces. *arXiv preprint arXiv:2607.17093*, 2026a.

Feiwei Qin, Chenqi Luo, Junhao Hou, Meie Fang, and Ligang Liu. Brep-gd: a graph diffusion model for cad boundary representation generation. *IEEE Transactions on Visualization and Computer Graphics*, 2026b.

Danila Rukhovich, Elona Dupont, Dimitrios Mallis, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. Cad-recode: Reverse engineering cad code from point clouds. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9801–9811. IEEE, 2025.

Zeyu Shen, Mingyang Zhao, Dong-Ming Yan, and Wencheng Wang. Mesh2brep: B-rep reconstruction via robust primitive fitting and intersection-aware constraints. *IEEE Transactions on Visualization and Computer Graphics*, 31(10):6661–6676, 2025.

Ian Stroud. *Boundary representation modelling techniques*. Springer, 2006.

Rui Sun, Shuming Gao, and Wei Zhao. An approach to b-rep model simplification based on region suppression. *Computers & Graphics*, 34(5):556–564, 2010.

Christopher M Tierney, Liang Sun, Trevor T Robinson, and Cecil G Armstrong. Using virtual topology operations to generate analysis topology. *Computer-Aided Design*, 85:154–167, 2017.

Muhammad Usama, Didier Stricker, Mohammad Sadil Khan, and Muhammad Zeshan Afzal. Brepclip: Contrastive multimodal pretraining on brep primitives for cad understanding. *arXiv preprint arXiv:2606.05515*, 2026.

Tamás Várady, Ralph R Martin, and Jordan Cox. Reverse engineering of geometric models—an introduction. *Computer-aided design*, 29(4):255–268, 1997.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5): 1–12, 2019.

Kevin J Weiler. *Topological structures for geometric modeling (Boundary representation, manifold, radial edge structure)*. Rensselaer Polytechnic Institute, 1986.

Karl DD Willis, Pradeep Kumar Jayaraman, Hang Chu, Yunsheng Tian, Yifei Li, Daniele Grandi, Aditya Sanghi, Linh Tran, Joseph G Lambourne, Armando Solar-Lezama, et al. Joinable: Learning bottom-up assembly of parametric cad joints. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15828–15839. IEEE, 2022.

Hongjin Wu, Ruoshan Lei, Yibing Peng, and Liang Gao. Aagnet: A graph neural network towards multi-task machining feature recognition. *Robotics and Computer-Integrated Manufacturing*, 86: 102661, 2024.

Rundi Wu, Chang Xiao, and Changxi Zheng. Deepcad: A deep generative network for computer-aided design models. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6752–6762. IEEE, 2021.

Xiang Xu, Joseph Lambourne, Pradeep Jayaraman, Zhengqing Wang, Karl Willis, and Yasutaka Furukawa. Brepgen: A b-rep generative diffusion model with structured latent geometry. *ACM Transactions on Graphics (TOG)*, 43(4):1–14, 2024.

Xiang Xu, Pradeep Jayaraman, Joseph Lambourne, Yilin Liu, Durvesh Malpure, and Pete Meltzer. Autobrep: Autoregressive b-rep generation with unified topology and geometry. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pp. 1–12, 2025.

Can Yao, Kang Wu, Zuheng Zheng, Siyuan Xing, and Xiao-Ming Fu. Brepmae: Self-supervised masked brep autoencoders for machining feature recognition. *arXiv preprint arXiv:2602.22701*, 2026.

Shuming Zhang, Zhidong Guan, Hao Jiang, Tao Ning, Xiaodong Wang, and Pingan Tan. Brep2seq: a dataset and hierarchical deep learning network for reconstruction and generation of computer-aided design models. *Journal of Computational Design and Engineering*, 11(1):110–134, 2024.

Yingzhong Zhang, Yufei Fu, Jia Jia, and Xiaofang Luo. An approach to automatic boundary segmentation of solid models using virtual topology: Toward reconstruction of design features. *Journal of Computational Design and Engineering*, 7(3):367–385, 2020.

A PROOFS

Proof of Proposition 1. Analyticity gives uniqueness of the surface decomposition: two distinct analytic surfaces agree at most on a set with empty interior in either, so every point of ∂S off a measure-zero seam set lies in exactly one maximal surface, and the patches are the connected components $\Sigma_1, \ldots, \Sigma_m$ of the corresponding subsets of ∂S. Let P be a valid partition. Each face lies in exactly one Σ_j, and the faces contained in Σ_j cover it with disjoint interiors. It remains to show the edge-adjacency graph of the faces covering one connected Σ_j is connected: for x, y in the interiors of two such faces, take a path in Σ_j avoiding the (finitely many) face vertices; the path crosses face boundaries finitely often, each crossing at an interior point of a shared piecewise-regular curve, i.e. a positive-length shared edge. The sequence of faces along the path is a walk in the adjacency graph. Hence regions = patches, for every P; region adjacencies and shared boundary curves are likewise determined by the patches alone. □

Proof of Proposition 2. Throughout, φ denotes an arbitrary integrable function on ∂S (the integrand of any surface feature) and dA the surface area element. If region R is cut into faces $f_1, \ldots, f_k$ with disjoint interiors, then $\int_R \varphi \, dA = \sum_i \int_{f_i} \varphi \, dA$ because seams have measure zero. For boundary features — integrals $\int_\Gamma \psi \, d\ell$ of a function ψ along a boundary curve Γ, $d\ell$ the arc-length element: by Proposition 1, ∂R and each shared boundary $\partial R \cap \partial R'$ (R' ranging over the other regions) are partition-independent sets; internal seams introduced by splitting separate faces of the *same* region and are excluded by construction, so boundary integrals also decompose over faces. □

Proof of Proposition 3. Partition-invariance of a predictor f means $f(x) = f(x')$ whenever its inputs x, x' are B-reps of the same solid, i.e. f factors through the geometry map g — the map sending a B-rep to the solid it bounds (unrelated to the rigid motion g of Proposition 5). For any such f, a B-rep-valued random input X, and its label Y, $\Pr[f(X) \neq Y] = \mathbb{E}\big[\Pr[f \neq Y \mid g(X)]\big] \geq \mathbb{E}\big[\min_c \Pr[Y \neq c \mid g(X)]\big]$, the Bayes risk of predicting Y from geometry alone (the minimum ranging over label values c). If two histories with different labels produce the same solid with positive probability, this bound is strictly positive; conversely any predictor with lower risk cannot factor through g and therefore depends on the representation of ∂S — the partition. □

Proof of Proposition 4. Write $C(S)$ for the boundary's area centroid, $I_C(S)$ for its inertia tensor, $V(S)$ for the matrix of principal axes (eigenvectors of I_C), and $g = (\mathbf{R}, \mathbf{t})$ for a rigid motion — rotation $\mathbf{R}$, translation $\mathbf{t}$ — with gS the moved solid. Translation: the centroid is area-mean of a boundary integral, so $C(gS) = \mathbf{R}C(S) + \mathbf{t}$ is immediate; centred coordinates $x - C$ eliminate $\mathbf{t}$. Rotation: for the centred inertia $I_C = \int \big(\|x-C\|^2 E - (x-C)(x-C)^\top\big) dA$ — x ranging over ∂S and E the 3×3 identity matrix — substituting $x \mapsto \mathbf{R}x + \mathbf{t}$ gives $I_C(gS) = \mathbf{R} I_C(S) \mathbf{R}^\top$, whose eigenvectors are $\mathbf{R}V$ with the same eigenvalues. The residual ambiguity group is $\{\pm 1\}^3$ on signs times $SO(2)$ (rotations within a plane; $SO(3)$, all 3D rotations, for triple degeneracy) within each degenerate eigenspace; the sign choice is a function of the third moments — the integrals $\int \langle v_j, x - C\rangle^3 \, dA$ along the principal axes v_j, invariants after centring and rotation — and the degenerate-subspace average is by construction constant on orbits of the ambiguity group. Features that are tensors expressed in this frame are therefore fixed points of the whole nuisance group. Partition-invariance is inherited because every ingredient — C, I_C, the third moments — is an integral over ∂S (Proposition 2). An improper orthogonal transformation can reverse the handedness of the skew-oriented eigenframe. Independent sign fixing does not by itself preserve uniqueness under reflection, because coordinates expressed in the corresponding reflected frame can coincide. The frame is therefore constrained to be right-handed: when the sign fixes land left-handed, the determinant correction is distributed over the near-tied smallest-skew axes by continuously reweighting their sign blends (reducing to a single axis flip when the minimum is unique). Mirror pairs are then distinguished whenever all three skew statistics exceed the blend band, the smallest skew magnitude is untied, and the inertia spectrum is simple; at a tie or inside the band, discrimination degrades continuously to zero and is not guaranteed. □

Proof of Lemma 1. Soft binning assigns mass m at coordinate t to bins $\lfloor t \rfloor, \lfloor t \rfloor + 1$ with weights $1 - \{t\}, \{t\}$, writing $\lfloor t \rfloor$ for the integer part and $\{t\}$ for the fractional part of t; here w is the bin width, coordinates are measured in units of w, and in our histograms the mass m is the sample's

quadrature weight — its triangle's area (its segment's length for the boundary histograms). The map $t \mapsto$ (weight vector) is piecewise linear with slope $\pm 1/w$ into two coordinates, so a perturbation δ of one sample moves the histogram by at most $2m|\delta|/w$ in L_1; summing over samples gives the bound. The histogram is also linear in the masses, each mass spreading with total weight one, so perturbing an area weight by δ moves it by at most $|\delta|$ in L_1: Lipschitz constant 1 in the weights. The remaining mesh-computed features — the integrated normal and the curvature totals — are plain quadrature sums $\sum_T a_T\, \varphi(x_T)$ over triangles T with area weights a_T, sample positions x_T, and integrand φ (unit normals; curvatures): linear in the a_T with coefficients bounded by $\sup|\varphi|$, and Lipschitz in the x_T with constant $\mathrm{Lip}(\varphi) \sum_T a_T$, where $\mathrm{Lip}(\varphi)$ — finite because each integrand is smooth on the fixed analytic surface the samples lie on — is the integrand's Lipschitz constant there. Grid blending is Lipschitz in the blend weight for fixed binnings; this does not establish continuity across spatial bin boundaries. The histogram and quadrature bounds above establish the result for these features. Neither hard variant admits any bound: hard binning moves a sample's full mass m between bins when a perturbation $\delta \to 0$ crosses a bin edge (an L_1 jump of $2m$ regardless of δ), and a hard sign choice ($w \in \{0, 1\}$) likewise has no finite Lipschitz constant — this is why the rotation row of Table 1 concentrates its residual in histogram channels. □

B Experiment implementation details

Perturbation construction. Re-partitioning uses `BRepFeat_SplitShape` along section curves that lie on the faces they cut. A split face's children inherit its label. The acceptance tolerance quoted throughout as "a very small tolerance" is relative and unit-free: a perturbed solid is kept only if $|\Delta V|/V \leq 10^{-9}$ and $|\Delta A|/A \leq 10^{-9}$, i.e. volume (V) and surface area (A) each match the original to within one part in 10^9, independent of the model's units. Every other tolerance in the paper is relative in the same sense unless stated otherwise.

Rotation and re-expression. Rotation draws a uniform random axis and an angle uniform in $[0.2, \pi]$ and every model sees the identical rotated solid. Re-expression uses OpenCASCADE's `NurbsConvert`. Labels transfer to perturbed files by index where face order survives (rotation) and by multi-point interior on-surface matching with an unambiguity requirement; faces with no unambiguous parent are explicitly rejected and excluded from the metric identically for every model, with rejected-face counts and surface-area share reported per dataset and perturbation (with the clean solid first moved by the same rotation, so both solids sit in the same model-space coordinates). A small fraction of perturbed files crash some baseline's own feature extractor; such parts are dropped from that column for all models.

Training. For our models: we use AdamW, lr 10^{-3}, a OneCycleLR schedule, batch 64, label smoothing 0.05, EMA 0.999; d=512, 8 EdgeGAT layers, 8 heads (17.5M parameters). Synthetic benchmarks train 80 epochs; Fusion 360 trains 200 epochs with inverse-frequency class weights at exponent 0.3. Released checkpoints retain their original training protocols; retrained baselines are identified separately.

Evaluation. We evaluate on the official test split, and only for examples whose perturbations pass the validity test. Checkpoints were always selected by validation, but the choice among the Fusion 360 feature variants involved repeated test evaluations and validation-based selection would pick a variant within 0.3 mIoU of the top performing result. DGCNN on MFInstSeg and Fusion 360 uses a configuration of 4096 points and doubled epochs. We train and evaluate our method on three random initializations, trained once on that dataset's official training split, and for the baselines we use their released weights (or one retraining at the shared budget where none are released). Every number in the paper, on every perturbation and study, is an inference-mode evaluation of these fixed checkpoints. Epoch checkpoints were selected by validation mIoU. Feature-variant development on Fusion 360 involved repeated test evaluations, so its test results should not be interpreted as an untouched development holdout. After fixing the representation and training protocol, we trained three random initializations per dataset. The main tables report the run selected by validation mIoU. Baselines use released weights where available and otherwise a retraining under the stated protocol. Reported evaluations use fixed checkpoints in inference mode, with no further adaptation during evaluation. DGCNN on MFInstSeg and Fusion 360 uses 4,096 points and twice the stated epoch budget.

Table 10: The human study's full results for predictive churn: the percentage of faces whose predicted label changes between two independently authored constructions of the same geometry. Upper block: within-designer pairs. Lower block: the between-designer.

	AAGNet	UV-Net	BRepNet	Ours
geometry-exact pairs (87 of them)	11.8	30.1	12.6	**0.5**
all 113 pairs (bodies pooled)	11.7	27.5	11.1	**0.6**
raw file vs. its canonical twin	11.4	12.1	9.0	**0.2**
raw cross-version pairs	13.5	35.9	13.1	**0.3**
cross-designer geometry-exact pairs (24 of them)	13.1	15.1	12.5	**0.2**
2nd designer's solids, axis-split	18.9	64.1	30.6	**0.2**
2nd designer's solids, diag-split	27.4	75.4	39.8	**0.0**
unchanged faces across designers	15.7	18.9	15.0	**0.7**

Retrieval embeddings. We use mean+max pools of each model's own node features, taken from the released checkpoints without any retrieval training; we cosine similarity on the embeddings to find the top-1 in the database.

C THE HUMAN STUDY IN DETAIL

Calibration. Before the study we checked how a designer's habits reach the exported file. They do indeed affect the exported file: e.g. draw a circular profile as two half-arcs and the exported cylindrical face arrives split in two; model half a part and mirror it, and the mirror plane leaves extra seam faces. One pass of the standard cleanup (`refine`, OCC's `UnifySameDomain`) removed many artifacts, and the kernel normalizes some route choices before they even reach the file (building a cylinder by lofting between two circles stores the same analytic cylinder as extruding one). Because much of the software stack applies such cleanup automatically, habit-level variation rarely reaches a consumer; the study therefore cleans every file first, and the deployment-scale measurements target the paths where no cleanup intervenes i.e. kernels that store everything as NURBS (Rhino's openNURBS) and translation between kernels.

Multi-body intake. Some parts are built as several separate pieces rather than one solid, and two files of the same part need not list those pieces in the same order. Pieces are therefore matched by size and position (volume and centroid). Evaluating piece by piece and pooling the per-face results recovers every program the one-solid intake rejects (27 of the 100 are genuinely multi-body), so all 100 human programs are usable.

Staging and audit. All files from both experts are produced by executing their 100 FreeCAD programs through one deterministic runner. Files come from people re-typing dimensions off drawings, and a single mistyped value makes the two solids genuinely different shape. Sameness is therefore tested per pair. Pairs are gated by a spatial identity check: maximum bidirectional deviation of interior on-surface sample points within 1e-6 relative (agreement at the stated sampling resolution, not an exact certificate) — with volume/area agreement at 1e-6 as a confirmatory scalar check. Scalars alone are insufficient: a feature moved inside a part can leave every scalar unchanged. Among 122 pairs satisfying the scalar gate, checks at three sampling densities confirm 116 within the stated spatial tolerance; three are tolerance-boundary cases and three are geometrically different. The confirmed set contains 92 within-designer pairs and 24 cross-designer pairs. The primary comparison further requires successful predictions from every compared model, leaving 87 within-designer pairs and all 24 cross-designer pairs. Boundary cases, different geometries, and model-specific extraction failures are reported separately.

Scoring support. Consistency for every model is computed on the same face support: faces whose cross-construction correspondence is accepted (ambiguous or off-surface transfers are excluded identically for all models), restricted to pairs on which every compared model produces a prediction. Five within-expert pairs (two designs) are unparseable by the baselines' own feature extractors; they are excluded from the paired comparison but reported, and per-model coverage accompanies each table.

Between-designer geometric identity. Free modeling from a drawing does not produce geometrically identical solids. The experts were instructed to match the output dimensions after making a first pass independently. Of the 35 staged cross-designer pairs, there were 26 pairs whose volume and surface area agree within the specified tolerance. Then checking additional points on both models' surfaces, we classified 24 as confirmed within tolerance, one as tolerance-boundary, and one as geometrically different.

Derived measurements. Re-partitioning the second user's own solids exercises human-authored geometry under our synthetic families. The results are in Table 10.

The unchanged-face cell. For pairs whose geometry does not match exactly, consistency is scored only on the faces both users built identically (matched at 10^{-6} in surface type, area and centroid). One caveat applies to every model equally: all of these are graph networks, so a prediction on an unchanged face may legitimately move because the *rest* of the part differs. This is therefore a softer test.

D CANONICAL FRAME DETAILS

Some of the model's features describe placement i.e. where a region sits and which way it is oriented. Coordinates require a reference frame, and the file's own frame is arbitrary: rotate the part and every such feature changes, which is exactly the rotation sensitivity the baselines exhibit. We compute the canonical frame from the solid itself: the eigenvectors of its boundary inertia tensor, which is built from integrals and is therefore partition-invariant too.

An inertia frame, however, is not automatically well-defined. Eigenvectors come with two ambiguities: each axis is defined only up to sign, and when two eigenvalues are (nearly) equal the axes are defined only up to a rotation within that subspace. We resolve both. Axis signs are fixed by the solid's third-order moments (its skew along each axis), exactly with a closed-form cubic over the mesh triangles. When a skew is too close to the threshold, the two sign choices are blended continuously instead of hard-switched; when eigenvalues are nearly equal, frame-dependent features are blended toward their exact averages over rotations of the degenerate subspace: with per-gap degeneracy weights a and b, a feature X is replaced by the convex combination

$$(1-a)\cdot(1-b)\cdot\ X\ +\ a\cdot(1-b)\cdot\ P_{01}(X)\ +\ (1-a)\cdot b\cdot\ P_{12}(X)\ +\ a\cdot b\cdot\ P_3(X),$$

where P_{01} and P_{12} are the closed-form averages over rotations within each eigenvalue pair's plane and P_3 is the full rotational average ($\mathrm{tr}(X)/3\cdot I$ for tensors, 0 for vectors). The projections are applied independently to the original value (composing them sequentially is order-dependent when both gaps are small) so the result is exactly basis-independent at any full degeneracy, pairwise or triple, and continuous in the eigenvalue gaps. A genuinely symmetric solid has no preferred frame and correctly receives none.

One further ambiguity remains: the sign fixes alone can land on either a right- or left-handed set of axes, and a mirrored solid would then receive the mirrored frame — and identical features. We therefore constrain the frame to be right-handed. When the sign fixes land left-handed, the correction is applied on the axis whose skew is least trusted, and when several axes are nearly tied for that role the correction is blended across them continuously, the same way the sign choices themselves are blended. Mirror pairs are consequently distinguished when all three skews are confident, the least-trusted one is untied, and the inertia spectrum is non-degenerate; otherwise discrimination degrades continuously and is not guaranteed (near-ties occur on 4.3% of MFInstSeg parts).

E THE PROCEDURE IN FULL

Type selection on ambiguous patches. A patch's samples do not always determine its surface type uniquely: more than one analytic fit can clear the acceptance tolerance. The simplest admissible type wins i.e. the one with the fewest defining parameters (plane, then sphere, cylinder, cone, torus). Fits meeting the machine-precision threshold take priority; multiple qualifying fits are resolved by the deterministic type-selection rule.

stage	what it does
rescale	the solid is uniformly scaled to unit boundary area (factor $1/\sqrt{A}$), so all later tolerances are size- and unit-independent; uniform scaling is rotation-invariant, unlike the bounding-box normalization the baselines use
surface keys	every face is labelled by the surface it lies on. For the five analytic types the label is the handful of numbers that define the surface (a cylinder's axis and radius, a plane's normal and offset), written in a unique standard form and rounded to 10^{-6} so the same surface always produces the same label. Kernels often re-save a simple surface in the generic freeform (NURBS) format — the way $1/2$ can be written 0.5000 — so every freeform face is first tested against the five types by fitting sampled points and normals: a fit is accepted below $10^{-6}\cdot\mathrm{scale}$ residual (genuine matches measure $\sim 10^{-9}$, non-matches $O(1)$), and the face is then labelled as its analytic self; truly freeform surfaces are labelled by their defining control points
regions	faces are grouped transitively into *regions*: two faces join the same region when they share an edge and carry the same surface label. Regions are the graph's nodes; a face's prediction is its region's prediction
canonical frame	positions and directions are expressed in the solid's own natural axes (the principal axes of its boundary inertia, as in mechanics) rather than the file's arbitrary frame. Each axis's sign is fixed by the solid's asymmetry (third-order moments); when an axis's asymmetry is too small to trust (skew within 0.01–0.04) the two choices are blended, and when two axes are interchangeable (near-equal inertia) features are blended toward the exact rotational average over the ambiguous subspace; mirror images are deliberately kept distinct
node features (115)	per region: which surface type it is and its defining numbers; exact integrals over the region (area, centroid, average normal direction, spread of positions and normals, total curvatures); how many boundary loops it has and how long they are; how much of its border is sharp, concave, convex; a smoothed 8-bin histogram of the bend angle along its border; and a coarse 4^3 map of where its area sits within its own bounding box
edge features (10)	per pair of touching regions: how long the shared border is, the average and distribution (6 soft bins) of the bend angle across it, and how much of it is sharp or concave
quadrature	all sampled quantities are computed on the kernel's triangle mesh, refined to at most 2M triangles per solid; histogram bins overlap smoothly so a value near a bin edge changes features continuously (Lemma 1)
network	a standard graph network over the region graph: 8 layers of edge-conditioned attention, width 512, 8 heads, dropout 0.1; a linear head scores each region and every face inherits its region's score

F REPRESENTATION-LEVEL VERIFICATION IN FULL

This section verifies the *representation* with no model involved. For each solid and its perturbed twin, both canonical region graphs are built and compared directly. We test whether a single bijection between the two graphs' nodes simultaneously preserves adjacency and matches features: node counts first, then a feature-matched assignment checked as a permuted edge set with per-edge feature residuals. Where symmetric, feature-identical nodes make that assignment fail adjacency spuriously, a colored graph-isomorphism search decides whether *any* structure-preserving bijection exists, and every reported residual (node, edge, and constant-channel drift) is then recomputed under that one common bijection. "Identical structure" below means this full test passed, not a node-count match. If the construction is invariant as claimed, structures match and the feature differences are numerical noise. Differences are reported in units of σ (the feature's standard deviation across the training set). This is the empirical backing for the propositions, measured at benchmark scale and independent of any training.

Feature classes. Features split into two classes by how their invariance is established. *Exact* features (areas, moments, types, parameters) are accumulated by the kernel's exact integrator, so their invariance is provable and their measured drift is machine noise. *Mesh-tolerance* features (anything computed on the triangle mesh: normal and curvature integrals, histograms, the grid) carry *no invariance guarantee*: they drift whenever a perturbation makes the kernel re-mesh the solid or the canonical frame is recomputed. For them, the claim is weaker and three-part: Lemma 1 bounds how fast they *can* drift, the table below measures how far they actually do, and the experiment sec-

tions price what that drift costs at prediction level. Only the exact class is invariant by proof. The measured drifts below are raw relative errors on the split stress set (not σ units).

family	class	measured residual
area, centroid, moments, type, axes/radii, loops, boundary length	exact (GProp)	$\leq 10^{-15}$
normal integrals	mesh-tolerance	$\leq 1.8 \times 10^{-7}$
curvature integrals	mesh-tolerance	0
dihedral histograms (node, edge)	mesh-tolerance	$\leq 3 \times 10^{-8}$
inertia-frame grid	mesh-tolerance	$\leq 2.3 \times 10^{-5}$

Table 11: Feature drift between each solid and its perturbed twin, by perturbation: N pairs; "ident." = the fraction whose region graphs have identical structure; then percentiles over pairs of each pair's *largest* channel residual, in σ units, separately for the exact and mesh-sampled feature classes sampled features drift where a perturbation forces a re-mesh and, for frame-expressed channels, where the canonical frame is itself recomputed.

			exact features (σ)				sampled features (σ)			
perturbation	N	ident.	p50	p90	p99	worst	p50	p90	p99	worst
constr. fixtures	24	$24/24$	0	0	0	0	2e-15	0.003	0.003	0.003
curved-split	3	$3/3$	0	0	0	0	0.001	0.003	0.003	0.003
diag-split	3000	99.9%	0	0	4e-7	0.002	0.015	0.040	0.40	13.5
NURBS re-expr.	3000	100%	0.0007	0.003	0.009	0.022	0.041	0.29	3.2	94
random rigid motion	3000	100%	0	0	0.002	0.007	4.8	9.7	15	38
all combined	3000	99.3%	0.0006	0.003	0.007	3.2	4.8	9.7	15	80
human cross-ver.	42	$38/42$	0	10^{-6}	$<10^{-4}$	$<10^{-4}$	0.010	1.9	3.6	3.6