TOPOLOGY MATTERS: EVALUATING TOPOLOGICAL FIDELITY IN CAD GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Geometric similarity and execution validity do not guarantee structural correctness in Computer-Aided Design (CAD): small geometric perturbations can change a solid's connected components, tunnels, or cavities. We find that such changes can arise during representation preprocessing alone: among 7,590 DeepCAD references reconstructed from both original CAD programs and quantized vectors, 762 (10.0%) have different global topology. Motivated by this gap, we study topology fidelity across five neural CAD generators, using the Betti signature of each final solid as a diagnostic. Under standard generation, exact signature fidelity ranges from 78.2% to 88.9%, and error directions differ across generation paradigms: text-conditioned models tend to omit existing structures, while the evaluated reconstruction-based models show more balanced or over-realized errors. Fidelity is also substantially lower on construction-novel patterns, with a gap persisting after standardization over measured topological complexity and construction length. Finally, even when a model reproduces the baseline topology, it rarely realizes explicit topology-changing instructions, with strict success rates of 0–1.7% across the evaluated text-conditioned models. These findings show that aggregate topology fidelity alone is insufficient to characterize CAD generators: models can differ substantially in their failure modes, sensitivity to construction patterns, and ability to modify existing topology.[1]

1 INTRODUCTION

Computer-aided design (CAD) is widely used in industrial product development, providing precise and editable geometric solids for design and manufacturing (Camba et al., 2016). Recent neural CAD generators model CAD construction as sketches and operations (Wu et al., 2021; Xu et al., 2022), learn structured latent representations (Xu et al., 2024; Li et al., 2025a), or generate executable CAD programs from language or images (Khan et al., 2024; Xu et al., 2025; Wang et al., 2026b). As these models advance, reliably evaluating the quality of generated CAD becomes increasingly important.

Evaluating generated CAD models remains non-trivial because an effective protocol must account for geometric fidelity, basic CAD validity, and the intended structure of the design (Ma et al., 2026). Existing protocols primarily rely on geometric similarity metrics, such as Chamfer Distance (CD) (Wu et al., 2021), F-score at a threshold τ (Khan et al., 2024), and intersection-over-union (IoU) (Guan et al., 2025). For program-based generators, validity is commonly assessed through execution success and solid validity (Alrashedy et al., 2025; Chen et al., 2025), and recent benchmarks extend these checks to executable requirements and engineering constraints (Doris et al., 2026; Wang et al., 2026a; Dong et al., 2026).

However, neither geometric similarity nor validity guarantees structural correctness in CAD models (Xu et al., 2024; Wu et al., 2025; Li et al., 2025b). Beyond geometric shape, a CAD solid has a topology that describes its fundamental structural properties, such as the connectivity of its components and the presence of holes and enclosed cavities. These properties distinguish solids that may be geometrically similar but structurally different. In particular, **Small geometric deviations**

[1] All experimental data and code are available at: `https://anonymous.4open.science/r/CADTopo-evaluation-F874`.

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.

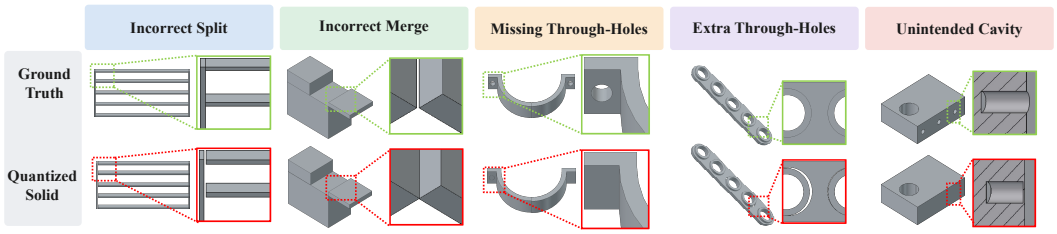


Figure 1: Illustrative examples showing how small geometric deviations introduced by DeepCAD's quantized representation can induce discrete changes in global topology.

can induce discrete changes in the topology of a solid, such as sealing an intended through-hole, disconnecting components that should remain connected, or creating unintended cavities. Figure 1 presents five illustrative examples of such changes arising from DeepCAD's quantized representation (Wu et al., 2021). For example, in the incorrect-split case, a bar that is intended to connect to a vertical support is shifted slightly during quantization, leaving a small gap between the two parts. Although the discrepancy is local and small in geometric magnitude, the reconstructed solid thus has a different global topology.

Such changes can be introduced by the CAD representation before generation even begins. We examine the vectorization pipeline used by DeepCAD (Wu et al., 2021), which independently normalizes sketch profiles and maps continuous geometric parameters onto an 8-bit discrete grid. This process can break exact relations that topology depends on, including profile alignment, equality of coaxial radii, and complete penetration of a cut. Using the official CAD builder, we reconstruct the same test references from both their original CAD programs and their quantized vector representations, and compare the global topology of the resulting solids. Among 7,590 references for which both reconstructions admit reliable topology extraction, 762 (10.0%) have a different global topology after vectorization. **Thus, perturbations that are nearly negligible under geometric metrics can make the reconstructed solid topologically inequivalent to the original.** More details of this representation-level analysis are provided in Appendix F.

These representation-level changes are concerning because they can remain inconspicuous under the geometric and validity-oriented criteria commonly used to evaluate CAD generation (Wu et al., 2021). More broadly, if a solid can undergo a discrete topological change despite only a small geometric perturbation, then strong performance under standard evaluation does not by itself establish structural correctness. This motivates us to ask how reliably current CAD generators preserve the intended global topology, and under which conditions this fidelity breaks down.

Existing evaluations primarily emphasize geometric similarity, execution success, mesh validity, or task-specific criteria, but these measures do not directly test whether the generated solid preserves its intended global topology. Recent topology-aware signals (Xu et al., 2025; Preintner et al., 2025) provide useful additional information, yet they do not fully characterize how topological fidelity varies across generation conditions. In particular, it remains unclear how fidelity changes with structural complexity, whether it persists for unfamiliar construction patterns, and whether generators can realize explicit topology modifications.

To answer these questions, we use global topology as a diagnostic probe for structural fidelity in CAD generation. We represent each solid by its Betti signature $(\beta_0, \beta_1, \beta_2)$, where β_0, β_1, and β_2 count connected components, tunnels, and enclosed cavities, respectively (Hatcher, 2002; Rote & Vegter, 2006). This representation is intrinsic to the final solid and independent of geometric scale or Boundary Representation (B-Rep) decomposition (De Floriani et al., 2010). We further stress-test whether topological fidelity remains robust under changing construction patterns and topology-changing instructions. Together, these evaluations distinguish topology preservation under standard generation from robustness to unfamiliar constructions and the ability to transform topology.

We find three systematic patterns: topology errors exhibit directional biases across generation paradigms, topology fidelity degrades substantially on construction-novel patterns, and models struggle to realize explicitly requested topology changes. These findings reveal that high topology fidelity under standard generation does not necessarily imply robust performance beyond familiar construction patterns or under explicit topology-changing instructions.

Our study makes three contributions.

- We formulate topology preservation as a core capability of neural CAD generation and use global topology as a representation-independent probe of structural correctness.
- We develop a three-part diagnostic evaluation protocol covering topology reproduction, generalization to construction-novel patterns, and topology-directed control.
- We reveal systematic failure modes that are obscured by aggregate geometric or validity-oriented evaluation, including paradigm-specific directional error patterns, a persistent construction-novelty gap, and a strong asymmetry between topology preservation and transformation.

2 PRELIMINARIES

Structural Levels in CAD Solids CAD solids contain structural information at different levels. Geometric properties, such as curvature and edge length, describe metric shape, while representation-level properties, such as B-Rep face counts and adjacency relations, depend on how the solid is encoded (Xu et al., 2024). In contrast, global properties such as connected components, through-holes, and enclosed cavities characterize the organization of the solid itself (De Floriani et al., 2010).

We focus on these representation-independent properties because they allow generated and target solids to be compared across different construction procedures and B-Rep decompositions. Global topology naturally captures this level of structure by describing the connectivity and global organization of the final solid, independent of geometric scale or representation.

Global Topology and Betti Numbers Global topology characterizes a solid through structural properties of the object as a whole: its connected components, tunnels, and enclosed cavities (Hatcher, 2002). Unlike geometric measurements, global topology is invariant to changes in scale, position, or other geometric deformations that preserve the same topological state. Unlike representation-level structure, it does not depend on how the solid is encoded. B-Rep decompositions offer a concrete illustration. For example, the lateral surface of a cylinder may be represented by a single face or decomposed into multiple faces. The B-Rep entities differ, but both representations describe the same solid and share the same global topology. Comparing global topology helps distinguish structural differences in the generated object from differences that arise from geometry or representation.

We quantify global topology using Betti numbers, a standard family of topological invariants (Rote & Vegter, 2006; Dey & Guha, 1998). For a solid M:

$$\boldsymbol{\beta}(M) = (\beta_0(M),\ \beta_1(M),\ \beta_2(M)) \tag{1}$$

where β_0 counts connected components, β_1 counts tunnels (e.g., through-holes), and β_2 counts enclosed cavities (e.g., internal voids).

These numbers provide a concise description of global structural properties, making global topology a concrete probe for evaluating whether a generator reliably preserves the intended structure of the solid.

3 METHODOLOGY

We study topological fidelity as the central structural property of CAD generation. Our methodology first establishes a representation-independent topology descriptor for generated solids, then measures fidelity under standard generation, and finally stress-tests whether this fidelity remains reliable under more challenging conditions. Concretely, we evaluate topological fidelity at three levels. First, we measure whether the generated solid exactly preserves the topology of its target. Second, we examine how fidelity changes with topological complexity and characterize the direction of topology errors. Third, we test the robustness of fidelity under two challenging settings: construction novelty, where the target follows a construction pattern not represented in training, and topology-directed intervention, where the desired topology is explicitly modified.

Evaluated Models. We evaluate five CAD generation models spanning different paradigms. DeepCAD (Wu et al., 2021) and SkexGen (Xu et al., 2022) generate CAD construction sequences using learned latent representations and disentangled codebooks, respectively. Text2CAD (Khan et al., 2024), CAD-Coder (Guan et al., 2025), and Cadrille-SFT (Kolodiazhnyi et al., 2026) represent text-conditioned and language-model-based CAD generation approaches. Together, they enable comparisons of topological fidelity across generation paradigms. All models are evaluated using the same topology-based protocol, with implementation details provided in the appendix G.

3.1 Topology Extraction

To compare models with different internal representations and construction procedures, we extract topology directly from the final CAD solid rather than from its construction history or B-Rep decomposition. Given a CAD solid in STEP format, we tessellate its boundary into a triangular mesh, weld duplicate vertices along shared face boundaries, and compute simplicial homology using GUDHI (Maria et al., 2014). Since the resulting mesh represents the boundary ∂M rather than the solid M itself, we recover the Betti numbers of the enclosed solid as

$$\beta_0(M) = N, \qquad \beta_1(M) = \frac{1}{2}\beta_1(\partial M), \qquad \beta_2(M) = \beta_2(\partial M) - N,$$

where N is the number of connected solid components. The resulting Betti signature

$$\boldsymbol{\beta}(M) = (\beta_0, \beta_1, \beta_2)$$

captures the numbers of connected components, tunnels, and enclosed cavities, respectively. This provides a compact, representation-independent description of the solid's global topology. We assign a topological signature only to watertight welded meshes; invalid or non-watertight samples are excluded from topology-based comparisons. The extraction pipeline is validated using synthetic solids with known topology and an independent homology implementation; details are provided in Appendix A.

3.2 Measuring Topological Fidelity

For a generated solid M_{gen} and its reference M_{gt}, we define topology preservation by exact equality of their Betti signatures:

$$\text{Match}(M_{\text{gen}}, M_{\text{gt}}) = \mathbb{1}\left[\boldsymbol{\beta}_{\text{gen}} = \boldsymbol{\beta}_{\text{gt}}\right].$$

We report the resulting exact match rate as Signature Fidelity, together with per-component accuracy for β_0, β_1, and β_2. To examine how fidelity changes with topological complexity, we define a coarse topological complexity score

$$C_{\text{topo}} = (\beta_0 - 1) + \beta_1 + \beta_2,$$

and report Signature Fidelity across complexity strata. This score is used only to group samples by coarse complexity; topology correctness is always determined by the full Betti signature. We further analyze the direction of topology errors. Comparing generated and target signatures allows us to distinguish under-realization, where existing structures are lost, over-realization, where spurious structures are introduced, and mixed errors, where both occur simultaneously. This diagnostic separates models that fail in qualitatively different ways even when their aggregate fidelity is similar.

3.3 Stress-testing Topological Fidelity

High fidelity under standard generation does not necessarily imply that topology remains reliable under more challenging conditions. We therefore stress-test topological fidelity in two settings that probe different aspects of generation: construction-pattern novelty and topology-directed intervention.

Fidelity under Construction Novelty. We first test whether topological fidelity persists when the target construction pattern is not represented in the model's training corpus. For each reference CAD program, we derive a structural fingerprint from its sequence of construction commands and extrusion Boolean operations, while discarding continuous geometric parameters. The fingerprint therefore captures the construction pattern rather than dimensions or placement. A test sample is construction-covered if its fingerprint appears in the evaluated model's training corpus and construction-novel otherwise; coverage is computed separately for each model.

We first compare Signature Fidelity between construction-covered and construction-novel samples. We then examine the covered–novel difference within topological-complexity strata ($C_{\text{topo}} = 0$, 1–2, and ≥ 3). Finally, we directly standardize the covered–novel comparison to account for differences in the distributions of topological complexity and command-sequence length between the two groups. We estimate uncertainty using 1,000 sample-level bootstrap resamples. Full details of the fingerprint definition, model-specific coverage construction, and standardization procedure are provided in Appendix B. This analysis tests whether high topological fidelity is limited to familiar construction patterns or remains robust under construction novelty.

Fidelity under Topology-Directed Intervention. We next test whether topological fidelity extends to cases where the desired topology is explicitly changed. This setting differs from standard reproduction: the model must not preserve the original topology, but instead realize a newly specified topological state. We focus on through-holes and augment the original prompts with additional instructions that request topology changes, including adding new through-holes or removing existing ones. We further introduce topology-neutral distractors as controls to determine whether observed structural changes arise from topology-specific instructions rather than prompt modifications alone. The exact prompt templates are provided in Appendix C.

To isolate topology modification from ordinary generation failure, we evaluate only cases whose baseline generation already matches the reference topology exactly. We select 300 add-hole bases (reference $\beta = (1, 0, 0)$) and 300 remove-hole bases (reference $\beta = (1, 1, 0)$). For topology-changing instructions, we report strict hit, where the full requested topology is realized.

Together, these two stress tests examine whether topological fidelity is robust beyond standard generation: whether it persists under unfamiliar constructions and whether it supports deliberate transitions to explicitly specified topological states.

4 Experimental Results and Analysis

We evaluate topological fidelity following the framework in Section 3. We first measure fidelity under standard generation, then diagnose how it varies with topological complexity and failure direction. Finally, we stress-test fidelity under construction novelty and topology-directed intervention.

4.1 Topological Fidelity under Standard Generation

We first measure topological fidelity under standard generation settings. For each generated solid, we compare its Betti signature against that of the corresponding reference. Exact signature matching serves as the primary measure, while invalid generations are excluded from topology-based comparisons and separately reported through the invalid rate (F0).

Overall Topology Fidelity. Table 1 shows that all five generators achieve substantial topological fidelity under standard generation, with Signature Fidelity ranging from 78.20% to 88.85%. Accounting for invalid generations further reduces end-to-end fidelity, showing that execution failure contributes additional structural failures. Across all models, β_1 has the lowest per-component accuracy, indicating that errors in tunnel structure account for a larger share of the component-level mismatches than errors in connected components or enclosed cavities. We next diagnose how topological fidelity degrades and how the resulting errors differ across generation paradigms.

Fidelity by Topological Complexity. Figure 2 (a) shows a systematic decline in Signature Fidelity as topological complexity increases. Within each generation setting, fidelity remains above roughly 92% for $C_{\text{topo}} = 0$ but drops substantially at $C_{\text{topo}} \geq 3$. This consistent decline indicates that more topologically complex targets pose a greater challenge to faithful topology realization. Representative examples of topology errors are provided in Appendix D.1.

Table 1: Topological evaluation on the test set. For each model, we report the number of invalid rate (F_0), per-component Betti accuracy, and exact Betti-signature fidelity among valid generations. Sig+Inv Fidelity counts invalid generations as failures.

Model	invalid (F0)	β_0 acc	β_1 acc	β_2 acc	Sig Fidelity	Sig+Inv Fidelity
DeepCAD	10.15%	95.00%	92.59%	99.61%	**88.85%**	79.84%
SkexGen	1.71%	93.57%	88.91%	98.96%	**84.25%**	82.81%
Text2CAD	8.85%	94.09%	84.18%	99.66%	**80.37%**	73.26%
CAD-Coder	3.95%	94.83%	86.77%	99.87%	**84.07%**	80.75%
Cadrille-SFT	5.26%	93.26%	82.45%	99.73%	**78.20%**	74.08%

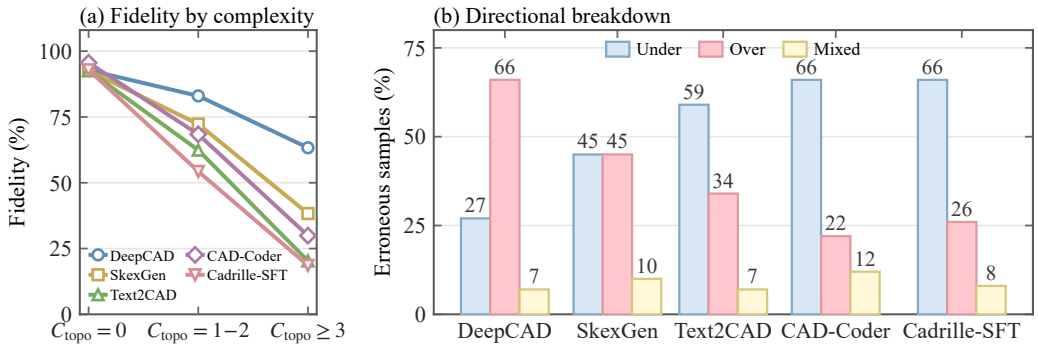


Figure 2: Topology fidelity and error direction across CAD generators. (a) Signature fidelity by topological complexity C_{topo}. (b) Error-direction distribution among valid samples with a signature mismatch. Percentages are computed over each model's mismatched samples; error directions are defined in Eq. 2.

Direction of Topology Errors To characterize these failure patterns, we compare the generated and target Betti signatures through $\Delta\boldsymbol{\beta} = \boldsymbol{\beta}^{gen} - \boldsymbol{\beta}^{gt} \in \mathbb{Z}^3$.

$$\text{direction}(\Delta\boldsymbol{\beta}) = \begin{cases} \textit{under-realization}, & \forall i:\ \Delta\beta_i \leq 0\ \wedge\ \exists i:\ \Delta\beta_i < 0, \\ \textit{over-realization}, & \forall i:\ \Delta\beta_i \geq 0\ \wedge\ \exists i:\ \Delta\beta_i > 0, \\ \textit{mixed}, & \exists\, i \neq j:\ \Delta\beta_i < 0\ \wedge\ \Delta\beta_j > 0. \end{cases} \tag{2}$$

Figure 2(b) reveals clear differences in error direction across generation models. Representative examples of under-, over-, and mixed-realization errors are provided in Appendix D.2.

DeepCAD shows an apparent over-realization tendency (66% vs. 27%), while SkexGen is nearly balanced. In contrast, Text2CAD, CAD-Coder, and Cadrille-SFT predominantly under-realize topology (59%, 66%, and 66%, respectively). A closer analysis of DeepCAD suggests that part of its apparent over-realization arises from representation-induced artifacts during vectorization, which can perturb geometric relations and consequently alter topology (Appendix F).

Across the evaluated text-conditional generators, under-realization is consistently more prevalent than over-realization, in contrast to the reconstruction-based models.

☞ **Finding 1:** *Topology errors exhibit distinct directional patterns across generation paradigms: reconstruction-based models show predominantly balanced or over-realization errors, whereas text-conditional generators exhibit a pronounced tendency to under-realize existing topological structures.*

4.2 Construction-Pattern Novelty

We next stress-test topological fidelity under construction-pattern novelty. We divide test samples into construction-covered cases, whose construction patterns are represented in the training corpus,

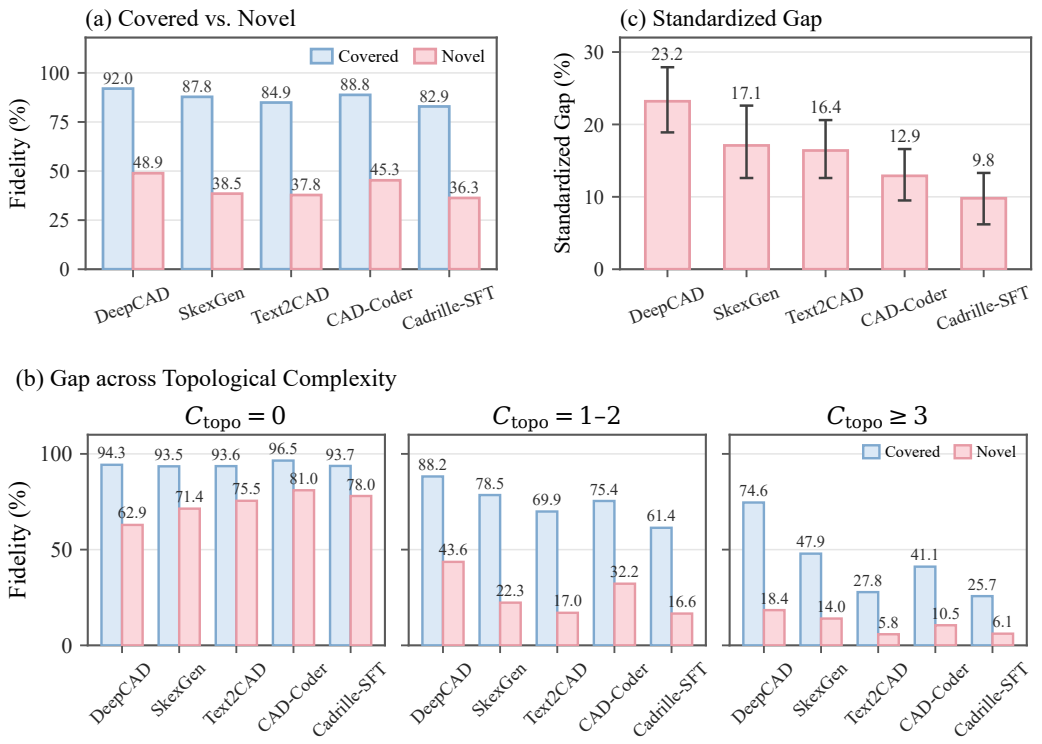


Figure 3: Topology fidelity under construction-pattern novelty. (a) Signature fidelity for construction-covered and construction-novel patterns. (b) Signature fidelity within each topological complexity stratum. (c) Covered-minus-novel fidelity gap, after direct standardization over joint bins of target topological complexity and command-sequence length. Error bars show 95% sample-level bootstrap confidence intervals.

and construction-novel cases, whose patterns are not. We then compare Signature Fidelity between the two groups.

Covered–Novel Gap Figure 3(a) shows a substantial covered–novel fidelity gap for all five generators. Signature Fidelity is 92.0% versus 48.9% for DeepCAD, 87.8% versus 38.5% for SkexGen, 84.9% versus 37.8% for Text2CAD, 88.8% versus 45.3% for CAD-Coder, and 82.9% versus 36.3% for Cadrille-SFT. The corresponding raw gaps range from 43.2 to 49.3 percentage points.

Gap across Topological Complexity We next examine whether construction-pattern novelty remains associated with lower topology fidelity after controlling for target topological complexity. We stratify test cases by the target C_{topo} and compare fidelity between covered and novel patterns within each stratum. Figure 3(b) shows that novel patterns consistently have lower fidelity across all five generators, with fidelity dropping to 5.8–18.4% at $C_{\text{topo}} \geq 3$. Thus, although higher topological complexity decreases fidelity, it does not fully explain the construction-pattern effect: even within the same complexity level, construction-novel patterns remain substantially harder to realize correctly.

Standardized Covered–Novel Gap Finally, Figure 3(c) reports the covered–novel gap after jointly standardizing over target topological complexity C_{topo} and command-sequence length L.

The standardized gap remains positive for every generator, ranging from 9.8 to 23.2 percentage points, with all 95% bootstrap confidence intervals excluding zero. Thus, the difference between covered and novel patterns cannot be fully explained by these measured complexity factors alone. Representative examples comparing construction-covered and construction-novel cases are provided in Appendix D.3.

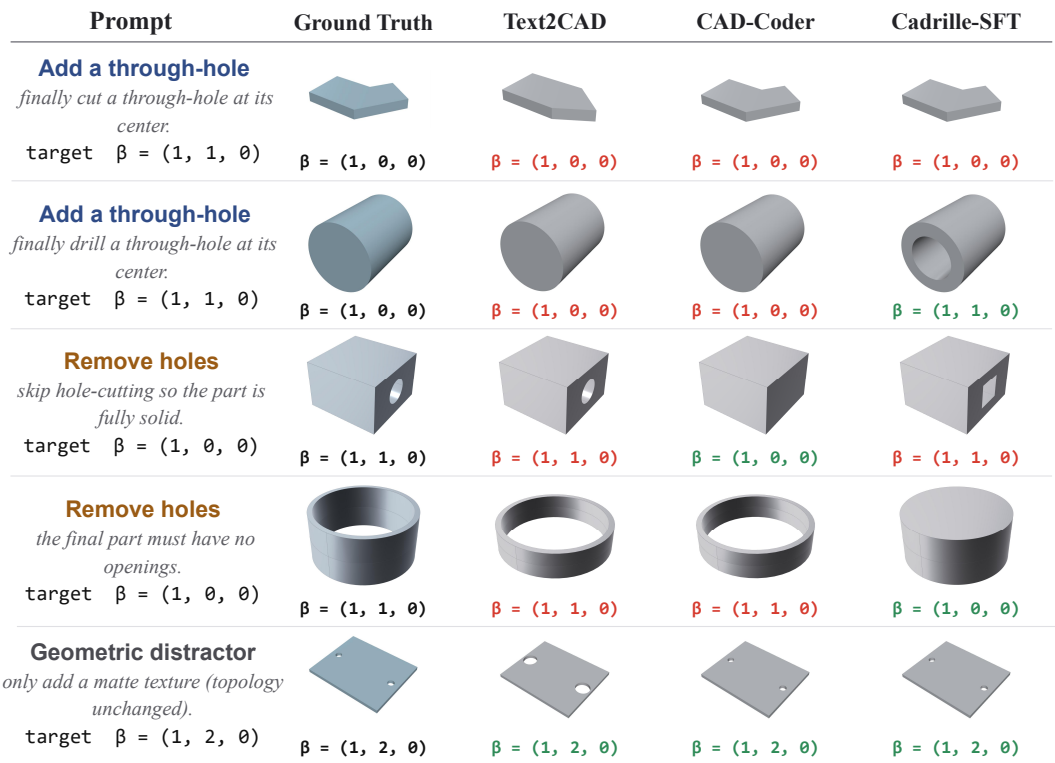


Figure 4: Representative examples of topology-directed intervention under exact baseline fidelity. For each case, the *target* signature denotes the desired topological state after intervention. Green and red signatures indicate successful and unsuccessful realization of the requested topological state.

These results indicate that topology realization is sensitive to construction-pattern novelty: patterns absent from the training distribution are substantially harder to realize correctly.

☞ **Finding 2:** *Topology realization is sensitive to construction-pattern novelty. The gap remains after accounting for measured topological and construction complexity.*

4.3 Topology-Directed Intervention

We finally stress-test topological fidelity under explicit topology modifications, conditioning on baseline generations that exactly match the reference topology. We evaluate instructions to add a through-hole, remove holes, or apply a topology-neutral geometric distractor.

Figure 4 shows that representative examples of topology-directed intervention under exact baseline fidelity. The strict-hit rates for adding a through-hole are 0.3%, 0.0%, and 1.7% for Text2CAD, CAD-Coder, and Cadrille-SFT, respectively, while the corresponding rates for removing all holes are 0.3%, 0.7%, and 1.7%. Topology-neutral distractor instructions serve as a control for prompt augmentation itself: they preserve the baseline topology in 99.5%, 99.7%, and 98.1% of cases across the three generators, indicating that adding an instruction alone does not alter the generated topology.

These results reveal a distinct limitation beyond baseline topology realization: even when models initially produce the correct topology, they rarely modify the topological structure according to explicit intervention instructions.

☞ **Finding 3:** *Even when baseline topology is correctly reproduced, models rarely achieve exact realization of the requested topological state under the evaluated topology-changing instructions.*

5 Discussion

Our findings show that topology errors in CAD generation are systematic rather than uniform. Different generation paradigms exhibit distinct directional biases, with reconstruction-based models tending to over-realize or balance errors, while text-conditional generators more often under-realize existing topological structures. Topological fidelity also depends on how structures are constructed: construction-novel patterns remain substantially harder to realize even at comparable structural difficulty, highlighting the importance of construction-pattern diversity in training data. Finally, topology preservation and transformation are strongly asymmetric: models generally retain existing topology but rarely execute explicit topology-changing instructions, suggesting that topology transformation constitutes a distinct modeling challenge rather than simply a harder form of preservation.

6 Related Work

CAD Generation Methods Deep learning for CAD generation has advanced rapidly. DeepCAD (Wu et al., 2021) first formulated CAD models as sketch-extrude sequences and trained a Transformer autoencoder on a large dataset. Later works followed this paradigm: SkexGen (Xu et al., 2022) disentangled topology and geometry into separate codebooks, enabling controlled generation; HNC (Xu et al., 2023) used hierarchical code trees for control. Text2CAD (Khan et al., 2024) introduced a task-specific text-to-CAD Transformer. More recently, LLM-backed approaches have become increasingly prominent. CAD-MLLM (Xu et al., 2025) fine-tunes Vicuna with LoRA for multimodal alignment; Cadrille (Kolodiazhnyi et al., 2026) and CAD-Coder (Guan et al., 2025) generate executable CadQuery scripts via vision or code LLMs, with the latter further incorporating chain-of-thought and geometric reward. An alternative line of work models the boundary representation directly: BrepGen (Xu et al., 2024) adopts diffusion for B-Rep generation. Existing research has increasingly recognized the importance of structural information and explored topology-aware generation strategies. However, whether these representations enable CAD generation models to truly capture the underlying structural principles of CAD solids remains an open question.

Evaluation of CAD Generation Models Evaluation protocols for CAD generation have primarily focused on geometric similarity, such as Chamfer Distance, F-score, and IoU, together with compilation validity (Wu et al., 2021; Khan et al., 2024; Koch et al., 2019; Willis et al., 2021). Recent benchmarks extend evaluation toward broader aspects: multi-modal generation (CADBench, Doris et al. (2026); Text2CAD-Bench, Wang et al. (2026a)), executable functional testing (CADTest-Bench, Mallis et al. (2026)), and functionality-, manufacturability-, and assemblability-oriented criteria (MUSE, Dong et al. (2026)). However, these metrics target observable quality or task-specific properties, not structural organization. They do not directly examine whether CAD generation models can capture and respect the underlying structural organization of generated solids. Among these properties, topology provides a representation-independent description of a solid's connectivity and global organization. Recent work has begun to incorporate topology-related signals: CAD-MLLM reports mesh-level validity, while EvoCAD considers Euler characteristic (Xu et al., 2025; Preintner et al., 2025). These signals are useful quality criteria but do not directly test whether generated solids preserve their intended global topology. We therefore use global topology as a diagnostic lens for structural capability, with additional comparisons provided in Appendix E.

7 Conclusion and future work

We presented an empirical study of topological fidelity in CAD generation using a representation-independent Betti signature of the final solid. Across standard generation, varying topological complexity, construction-pattern novelty, and topology-directed intervention, we found systematic directional biases across generation paradigms, a persistent covered–novel gap beyond measured complexity, and a strong asymmetry between topology preservation and transformation.

Several limitations remain. Betti numbers capture global topological invariants but not local structural details or construction semantics; our construction-pattern fingerprint only coarsely characterizes construction novelty, and the study is primarily based on DeepCAD. The topology-directed intervention is limited to through-hole modifications. Future work can extend to richer structural descriptors, broader CAD datasets, and more diverse topology transformations.

AI USE STATEMENT

In this work, we used generative AI tools to assist in designing and providing feedback on the research methodology and experiments, implementing methods, and cleaning and reformatting datasets. We have not used generative AI tools for translation or result interpretation, and the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to edit software code and improve the readability of the manuscript. We have reviewed all AI-assisted work. AI-assisted code was executed and tested by the authors, and the topology-extraction and evaluation pipelines were validated using the procedures described in the paper and appendix. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work evaluates existing CAD generation models and generated CAD solids using research datasets and does not involve human-subject experiments. The study focuses on diagnostic evaluation of structural and topological fidelity. The reported topology metrics characterize generated CAD models in the experimental setting and are not intended to replace application-specific engineering or safety validation.

REPRODUCIBILITY STATEMENT

All experimental data and code are provided through the anonymous repository linked in the paper. The paper and appendix document the topology-extraction procedure, evaluation metrics, construction-pattern analysis, topology-directed interventions, dataset processing, and experimental protocols for the evaluated CAD generators. The topology-extraction pipeline is additionally validated using synthetic solids with known topology and an independent homology implementation, supporting reproduction and verification of the reported results.

REFERENCES

Kamel Alrashedy, Pradyumna Tambwekar, Zulfiqar Haider Zaidi, Megan Langwasser, Wei Xu, and Matthew Gombolay. Generating cad code with vision-language models for 3d designs. In *International Conference on Learning Representations*, volume 2025, pp. 52236–52262, 2025.

Jorge D Camba, Manuel Contero, and Pedro Company. Parametric cad modeling: An analysis of strategies for design reusability. *Computer-aided design*, 74:18–31, 2016.

Jiali Chen, Xusen Hei, HongFei Liu, Yuancheng Wei, Zikun Deng, Jiayuan Xie, Yi Cai, and Qing Li. Cadreview: Automatically reviewing cad programs with error detection and correction. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9909–9927, 2025.

Leila De Floriani, Laura Papaleo, and Annie Hui. Topmesh - a tool for extracting topological information from non-manifold objects. In *Proceedings of the International Conference on Computer Graphics Theory and Applications (VISIGRAPP 2010) - GRAPP*, pp. 21–29. INSTICC, SciTePress, 2010. ISBN 978-989-674-026-9. doi: 10.5220/0002819400210029.

Tamal K. Dey and Sumanta Guha. Computing homology groups of simplicial complexes in r3. *J. ACM*, 45(2):266–287, March 1998. ISSN 0004-5411. doi: 10.1145/274787.274810.

Xiaoyu Dong, Zhi Li, and Xiao-Ming Wu. Muse: Benchmarking manufacturable, functional, and assemblable text-to-cad generation, 2026.

Anna C. Doris, Jacob Thomas Sony, Ghadi Nehme, Era Syla, Amin Heyrani Nobari, and Faez Ahmed. Cadbench: A multimodal benchmark for ai-assisted cad program generation, 2026.

Yandong Guan, Xilin Wang, XiMing Xing, Jing Zhang, Dong Xu, and Qian Yu. Cad-coder: Text-to-cad generation with chain-of-thought and geometric reward. In *Advances in Neural Information*

Processing Systems, volume 38, Main Conference, pp. 59765–59789. Curran Associates, Inc., 2025. doi: 10.52202/085713-1999.

Allen Hatcher. *Algebraic topology*, volume 1. Cambridge university press Cambridge, 2002.

Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin Sheikh, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts. In *Advances in Neural Information Processing Systems*, volume 37, pp. 7552–7579. Curran Associates, Inc., 2024. doi: 10.52202/079017-0242.

Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9593–9603, 2019. doi: 10.1109/CVPR.2019.00983.

Maksim Kolodiazhnyi, Denis Tarasov, Dmitrii Zhemchuzhnikov, Alexander Nikulin, Ilya Zisman, Anna Vorontsova, Anton Konushin, Vladislav Kurenkov, and Danila Rukhovich. cadrille: Multi-modal cad reconstruction with reinforcement learning, 2026.

Jing Li, Yihang Fu, and Falai Chen. Dtgbrepgen: A novel b-rep generative model through decoupling topology and geometry, June 2025a.

Yuan Li, Cheng Lin, Yuan Liu, Xiaoxiao Long, Chenxu Zhang, Ningna Wang, Xin Li, Wenping Wang, and Xiaohu Guo. Caddreamer: Cad object generation from single-view images. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21448–21457. IEEE, 2025b.

Weijian Ma, Shizhao Sun, Ruiyu Wang, and Jiang Bian. Cadmorph: Geometry-driven parametric cad editing via a plan–generate–verify loop. *Advances in Neural Information Processing Systems*, 38:100279–100297, 2026.

Dimitrios Mallis, Marco Wang, Ahmet Serdar Karadeniz, Elisa Ricci, Anis Kacem, and Djamila Aouada. Text-to-cad evaluation with cadtests, 2026.

Clément Maria, Jean-Daniel Boissonnat, Marc Glisse, and Mariette Yvinec. The gudhi library: Simplicial complexes and persistent homology. In Hoon Hong and Chee Yap (eds.), *Mathematical Software – ICMS 2014*, pp. 167–174, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. ISBN 978-3-662-44199-2.

Tobias Preintner, Weixuan Yuan, Adrian König, Thomas Bäck, Elena Raponi, and Niki Van Stein. Evocad: Evolutionary cad code generation with vision language models. In *2025 IEEE 37th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 504–511, 2025. doi: 10.1109/ICTAI66417.2025.00075.

Günter Rote and Gert Vegter. *Computational Topology: An Introduction*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-33259-6. doi: 10.1007/978-3-540-33259-6_7.

Liang Wang, Heng Meng, Zekai Xiang, Jin Liu, Pingyi Zhou, Litao Chen, and Yongqiang Tang. Text2cad-bench: A benchmark for llm-based text-to-parametric cad generation, 2026a.

Ruiyu Wang, Shizhao Sun, Weijian Ma, and Jiang Bian. Cad-tokenizer: Towards text-based cad prototyping via modality-specific tokenization. In *International Conference on Learning Representations*, volume 2026, pp. 147972–147992, 2026b.

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), July 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459818.

Jianyu Wu, Yizhou Wang, Xiangyu Yue, Xinzhu Ma, Jinyang Guo, Dongzhan Zhou, Wanli Ouyang, and Shixiang Tang. Cmt: A cascade mar with topology predictor for multimodal conditional cad generation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7014–7024. IEEE, 2025.

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, 2021. doi: 10.1109/ICCV48922.2021.00670.

Jingwei Xu, Chenyu Wang, Zibo Zhao, Wen Liu, Yi Ma, and Shenghua Gao. Cad-mllm: Unifying multimodality-conditioned cad generation with mllm, 2025.

Xiang Xu, Karl D.D. Willis, Joseph G Lambourne, Chin-Yi Cheng, Pradeep Kumar Jayaraman, and Yasutaka Furukawa. SkexGen: Autoregressive generation of CAD construction sequences with disentangled codebooks. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 24698–24724. PMLR, 17–23 Jul 2022.

Xiang Xu, Pradeep Kumar Jayaraman, Joseph George Lambourne, Karl D.D. Willis, and Yasutaka Furukawa. Hierarchical neural coding for controllable CAD model generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38443–38461. PMLR, 23–29 Jul 2023.

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 Trans. Graph.*, 43(4), July 2024. ISSN 0730-0301. doi: 10.1145/3658129.

Part I

Appendix

Table of Contents

A topology-validation **14**
- A.1 Synthetic solids with known topology 14
- A.2 Resolution and Tolerance Sensitivity 15
- A.3 Independent Homology Cross-Check 15

B Construction-Pattern Fingerprint and Coverage **15**
- B.1 Model-Specific Coverage Construction 16

C Topology-Directed Intervention Prompts **17**

D Sample Analysis **18**
- D.1 Qualitative Examples of Topology Errors 18
- D.2 Representative Directional Error Samples 18
- D.3 Representative Construction-Novelty Samples 18

E Why Full Betti Signatures? A Comparison with Existing Topology Metrics **19**

F Representation-Level Effects in DeepCAD **21**

G Model Implementation Details **22**

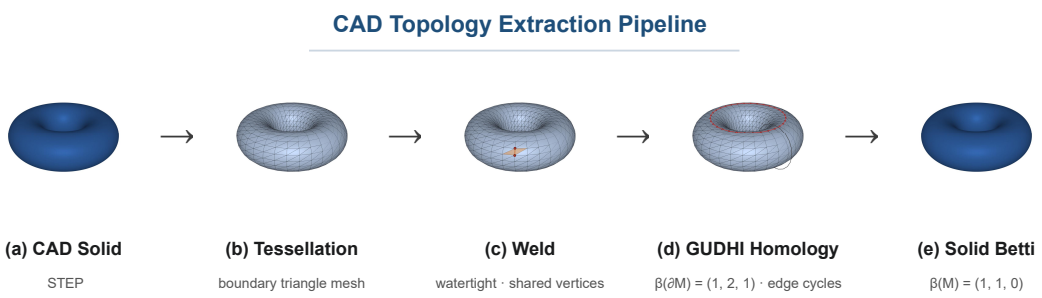


Figure 5: Topology extraction pipeline. Generated and reference CAD solids are converted from STEP to welded boundary meshes, from which homology is computed and converted into the solid's Betti signature.

Table 2: Synthetic validation of the frozen topology judge. Expected signatures are known from construction, and all tested cases are recovered correctly.

Case	Expected $(\beta_0, \beta_1, \beta_2)$	Recovered	Watertight	Result
Cube	$(1, 0, 0)$	$(1, 0, 0)$	✓	PASS
Washer (1 through-hole)	$(1, 1, 0)$	$(1, 1, 0)$	✓	PASS
Plate, 2 holes	$(1, 2, 0)$	$(1, 2, 0)$	✓	PASS
Plate, 4 holes	$(1, 4, 0)$	$(1, 4, 0)$	✓	PASS
Plate, 8 holes	$(1, 8, 0)$	$(1, 8, 0)$	✓	PASS
Blind pocket	$(1, 0, 0)$	$(1, 0, 0)$	✓	PASS
Counterbore, through	$(1, 1, 0)$	$(1, 1, 0)$	✓	PASS
Counterbore, blind	$(1, 0, 0)$	$(1, 0, 0)$	✓	PASS
Hollow box (1 cavity)	$(1, 0, 1)$	$(1, 0, 1)$	✓	PASS
Two cavities	$(1, 0, 2)$	$(1, 0, 2)$	✓	PASS
Two disjoint bodies	$(2, 0, 0)$	$(2, 0, 0)$	✓	PASS
Torus	$(1, 1, 0)$	$(1, 1, 0)$	✓	PASS
Two tori	$(2, 2, 0)$	$(2, 2, 0)$	✓	PASS
Through-hole + cavity	$(1, 1, 1)$	$(1, 1, 1)$	✓	PASS
14/14 tested cases correctly labeled.				

A TOPOLOGY-VALIDATION

This appendix validates the topology extraction pipeline used throughout the paper. The pipeline converts each CAD solid from STEP into a tessellated and welded boundary mesh, computes simplicial homology, and maps the resulting homology to the solid Betti signature $(\beta_0, \beta_1, \beta_2)$, as described in Sec. 3.1. Topological labels are emitted only for watertight meshes; samples that fail the watertightness or zero-volume checks are treated as invalid and excluded from topology comparisons.

Labeling and failure policy. The three Betti components are not obtained in the same way. β_0 is taken directly from the reconstructed CAD solid as the number of connected solids, whereas β_1 and β_2 are derived from the homology of the welded boundary mesh. Thus, β_0 does not depend on mesh connectivity. A topology label is emitted only when the welded mesh is watertight; if the default tessellation tolerance fails this check, the pipeline retries once at a finer tolerance. Zero-volume solids or builds that cannot produce a valid solid, as well as meshes that remain non-watertight after retry, are marked invalid and excluded from topology comparisons rather than assigned a Betti signature.

A.1 SYNTHETIC SOLIDS WITH KNOWN TOPOLOGY

We first validate the judge on procedurally constructed solids with known topology. The suite covers the three Betti components and distinguishes global topological features from local surface features that should not change the global signature.

Table 3: Resolution sweeps under the production tessellation policy (10^{-3}, with one retry at 10^{-4}). *Correct* denotes an emitted signature matching the known topology; *excluded* denotes rejection by the watertightness gate. No tested case produced an incorrect label.

Sweep	Range tested	Outcome	Limiting behaviour
Through-hole radius, $\beta_1 = 1$	$0.25 \to 5{\times}10^{-4}$	9/9 correct	none observed
Enclosed box cavity, $\beta_2 = 1$	$0.6 \to 2{\times}10^{-3}$	9/9 correct	none observed
Two-body gap, $\beta_0 = 2$	$10^{-1} \to 10^{-8}$	8/8 correct	welding does not merge the tested bodies

The suite confirms that the judge distinguishes global topology from local surface geometry. In particular, blind pockets and blind counterbores retain the signature $(1, 0, 0)$, whereas their through-hole counterparts yield $\beta_1 = 1$. The judge therefore responds to global homology rather than to surface indentations alone.

A.2 Resolution and Tolerance Sensitivity

We next examine whether the mesh-based judge remains stable for small topological features. Feature dimensions are varied over several orders of magnitude on unit-bounding-box solids, and each case is classified as correctly labeled or conservatively excluded.

The resolution sweeps show that the pipeline correctly resolves the tested topological features down to $5{\times}10^{-4}$ of the bounding-box scale without producing an incorrect label. For the tested two-body configuration, the welding procedure does not merge the separate bodies even at a 10^{-8} gap. On 20 adversarially selected real samples covering multi-body, high-genus, and cavity-bearing cases, the emitted signatures were invariant across tessellation tolerances from 10^{-2} to 10^{-5}, with no label changes or watertightness flips. Sweeping the vertex-welding precision from 10^{-4} to 10^{-10} likewise produced no label changes. At the deliberately destructive 10^{-3} welding setting, 15/20 meshes became non-watertight and were excluded by the same validity gate.

A.3 Independent Homology Cross-Check

We independently re-implemented the homology computation without GUDHI, using union-find for connected components and GF(2) boundary-matrix rank calculations for the remaining Betti components. On 646 stratified real solids, the two implementations agreed bit-identically on all 646 cases.

B Construction-Pattern Fingerprint and Coverage

To characterize whether a construction pattern (a representation-specific aspect of CAD structure) is represented in the training set, we derive a structural fingerprint for each reference solid:

$$\phi(M) = \big(L, c_1, c_2, \ldots, c_L\big), \tag{3}$$

where L is the number of construction tokens and c_i denotes the command token at position i, including the boolean operation attribute for extrusion commands.

We discard numerical parameters, so the fingerprint captures only the construction-level organization, not dimensions or placement. A test case is construction-covered if its fingerprint appears in the evaluated model's training corpus, and construction-novel otherwise; coverage is defined separately for each model.

We compare full-signature topology fidelity between covered and novel cases. To separate construction novelty from generation difficulty, we first stratify by C_{topo} and then estimate a residual gap after jointly standardizing over $C_{\text{topo}} \in \{0,\ 1-2,\ \geq 3\}$ and command-sequence length $L \leq 6$, $7 \leq L \leq 14$, $L \geq 15$:

$$\Delta_{\text{std}} \ = \ \sum_{c} w_c \cdot \big(F_{\text{covered}}(c) - F_{\text{novel}}(c)\big), \qquad w_c \ = \ \frac{n_{\text{novel}}(c)}{N_{\text{novel}}}, \tag{4}$$

where $F_g(c)$ is the signature fidelity of group $g \in \{\text{covered}, \text{novel}\}$ within cell c. $n_{\text{novel}}(c)$ is the number of novel cases in cell c, and N_{novel} is the total number of novel cases across the retained

cells. The weights reproduce the novel group's difficulty composition, so Δ_{std} compares the two groups as if they faced the same mix of measured difficulties.

We exclude cells with fewer than five samples in either group and report the fraction of novel samples retained after filtering. Uncertainty is estimated using 1,000 sample-level bootstrap resamples.

Complete command vocabulary The command-type sequence in Eq. 3 is drawn from the *entire* command vocabulary of the DeepCAD representation (`cadlib/macro.py`), which consists of five structural tokens and one sequence delimiter:

Index	Command	Role
0	LINE	sketch primitive (straight segment)
1	ARC	sketch primitive (circular arc)
2	CIRCLE	sketch primitive (full circle)
3	EOS	end-of-sequence marker (trailing padding; *stripped* from the fingerprint)
4	SOL	start-of-loop marker (delimits loops within a sketch)
5	EXT	sketch extrusion

Extrusion boolean operation Each EXT row additionally carries its boolean operation type, read from the operation field (column 15) of the extrusion vector: NEWBODY, JOIN, CUT, and INTERSECT (`EXTRUDE_OPERATIONS` in `cadlib/macro.py`). The boolean operation is a discrete structural attribute—it determines how material is added or removed and is therefore topology-relevant, whereas all continuous parameters (coordinates, radii, angles, extrusion distances) are discarded.

Fingerprint definition A structural fingerprint captures the *skeleton* of a construction sequence: which commands are used, in what order, and with which boolean operation—but not their numerical parameters.

To compute it, we first strip the trailing EOS padding; in the released DeepCAD vectors, EOS appears only as the sequence terminator and padding, while loop boundaries within a sketch are marked by SOL tokens, which are retained. Each remaining token is then encoded as a pair: (command type, boolean operation if the command is an extrusion; otherwise $\perp$). The fingerprint is defined as the combination of (i) the length of this encoded sequence and (ii) a compact hash of the entire sequence of pairs.

Two sequences are considered structurally identical if and only if they have the same length and exactly the same (command, boolean-op) pair at every position. The hash is merely a deterministic identifier and introduces no additional abstraction; it is used solely for efficient dictionary lookups when determining whether a fingerprint appears in the training set.

Coverage definition For each model m, let $D^{(m)}_{\text{train}}$ denote its training corpus and let $\Phi^{(m)}_{\text{train}}$ be the set of fingerprints extracted from that corpus. A test case is considered *covered* for model m if its fingerprint appears in $\Phi^{(m)}_{\text{train}}$, and *novel* otherwise.

B.1 MODEL-SPECIFIC COVERAGE CONSTRUCTION

Table 4 summarizes the model-specific coverage construction. For each generator, the covered/novel partition is computed against its own training corpus.

DeepCAD: the official DeepCAD training split.

SkexGen: we reconstruct the model's training exposure by running its official preprocessing pipeline end-to-end on the raw DeepCAD JSON corpus (215,093 files, of which 178,238 belong to the official train/validation/test split; the remaining files are mostly over-length or invalid sequences that do not fit the fixed-size command representation [2]), including conversion, normalization, parsing, and deduplication. The released parsing script assigns any sample absent from the official split

[2]These extra sequences contribute only 107 additional fingerprints, and no test sample's coverage label depends on them: restricting the corpus to the official train split leaves the covered/novel partition unchanged.

Table 4: Model-specific coverage construction. For each model, we report the training corpus used, the number of training sequences, the number of distinct structural fingerprints, and the resulting covered/novel partition among the 8,052 test samples with successfully extracted fingerprints.

Model	Training corpus	# train seq.	# fingerprints	covered	novel
DeepCAD	DeepCAD train	161,240	24,824	7,019	1,033
SkexGen	SkexGen-preprocessed	116,936	23,943	7,005	1,047
Text2CAD	Text2CAD train	159,049	24,461	7,015	1,037
CAD-Coder	Text2CAD derived	159,049	24,461	7,015	1,037
Cadrille-SFT	Text2CAD-compatible[†]	159,049	24,461	7,015	1,037

Condition	Instruction suffix (verbatim)
add one through-hole	"Finally, cut a through hole passing through the center of the part."
	"Finally, drill a through hole through the center of the part."
	"As a last step, create a cylindrical through hole in the center of the part."
remove through-holes	"Make sure the final part has no holes or openings of any kind."
	"Do not create any holes; keep the final part completely solid."
	"Skip any hole-cutting steps; the final part must be fully solid."
topology-neutral distractor	"Round the outer edges slightly for a better finish."
	"Finally, apply a smooth polished finish to all outer surfaces."
	"Give the outer surfaces a matte texture."

Table 5: Fixed instruction suffixes used for topology-directed intervention and topology-neutral control conditions.

to the training pool (`utils/dataset.py`); following the released code exactly, our reconstructed corpus therefore includes 22,728 such sequences. The two branches of the released training code apply different runtime filters (`dataset.py`): the sketch branch keeps deduplicated sequences with sketch pixel-sequence length (plus padding) at most 200, *excluding* invalid-marked sequences (73,686 sequences), while the extrusion branch keeps deduplicated sequences with at most five extrusions ($\texttt{num_se} \leq 5$) and applies *no* invalid filter (99,339 sequences). Since a sequence reaches the model whenever either branch includes it during training, we take the training corpus as the *union* of the two branches ($73{,}686 + 99{,}339 - 56{,}089 = 116{,}936$ sequences; 23,943 fingerprints). The intersection of the two branches (56,089 sequences) serves as a sensitivity control: under this stricter corpus the standardized covered–novel gap changes from 17.1 to 14.2 percentage points with the 95% confidence interval still excluding zero, leaving the covered–novel gap conclusion unchanged.

Text2CAD and CAD-Coder: the Text2CAD training split. CAD-Coder uses a corpus derived from this split.

Cadrille-SFT: the Text2CAD-compatible DeepCAD component used for supervised training. Cadrille also reports a 1M-sample CAD-Recode pretraining component expressed as free-form CadQuery code. Because this representation does not expose the command vocabulary used by our fingerprint, we restrict the model-specific coverage computation to the fingerprint-compatible training component.

[†]Coverage is defined with respect to the training data that are representable under the construction fingerprint used in this study.

C Topology-Directed Intervention Prompts

We focus on through-holes and augment each original description with a single additional instruction that either requests a topology change or introduces a topology-neutral distractor. Each augmented prompt retains the *verbatim* base description and appends one imperative suffix drawn from a fixed pool of three phrasings per condition (Table 5). Suffixes are assigned to base samples by random assignment uniformly over the three phrasings, so that each phrasing is used for a comparable share of cases. The three conditions are: adding one through-hole, removing existing through-holes, and applying a topology-neutral geometric change.

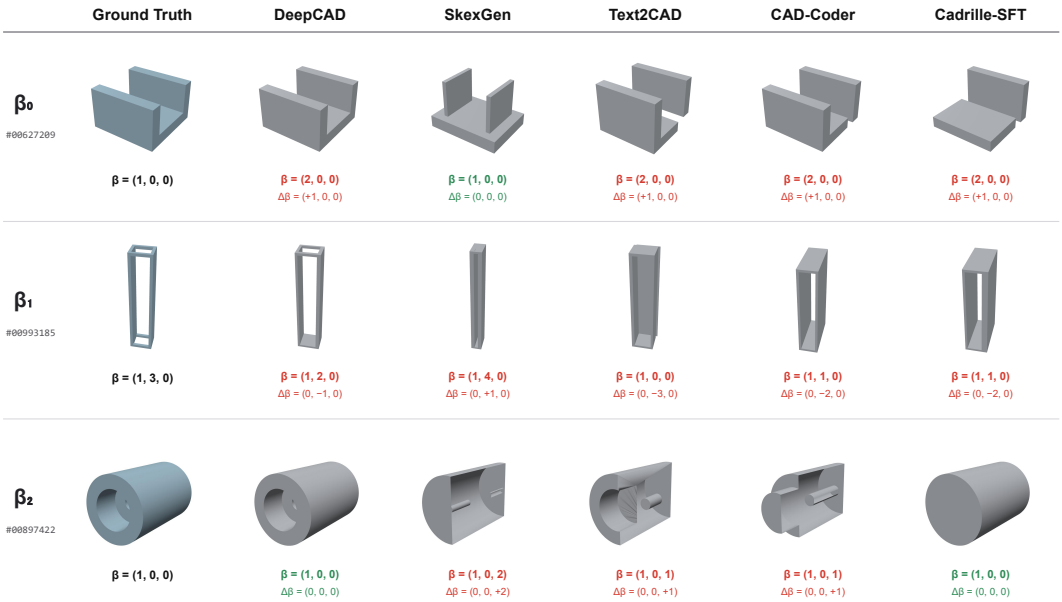


Figure 6: Representative topology errors across the three Betti components. The examples include incorrect connected components (β_0), missing or spurious tunnels (β_1), and incorrect enclosed cavities (β_2).

Topology-neutral distractor. The distractor condition changes the prompt while remaining topologically neutral: the added instruction concerns surface finishing and, by construction, does not request the creation or removal of holes or cavities. This condition serves as a control for prompt modification itself. If topology changes under a topology-changing instruction are driven by the intended structural instruction, the topology-neutral prompt modification should leave the realized topology unchanged. We additionally verify this empirically: relative to the exact baseline topology, the distractor condition preserves the Betti signature in 99.5%, 99.7%, and 98.1% of Text2CAD, CAD-Coder, and Cadrille-SFT cases, respectively.

D Sample Analysis

D.1 Qualitative Examples of Topology Errors

We provide representative examples of topology errors across different topology components. Each example compares the ground-truth solid with the generated solid and reports their corresponding Betti signatures. These cases illustrate the structural errors captured by our topology-aware evaluation (Figure 6).

D.2 Representative Directional Error Samples

We provide representative examples of under-realization, over-realization, and mixed topology errors (Figure 7).

D.3 Representative Construction-Novelty Samples

To provide an intuitive illustration of the construction-novelty effect, we present representative examples comparing construction-covered and construction-novel cases. Figure 8 compares cases where the construction fingerprint is observed during training with cases containing unseen fingerprints.

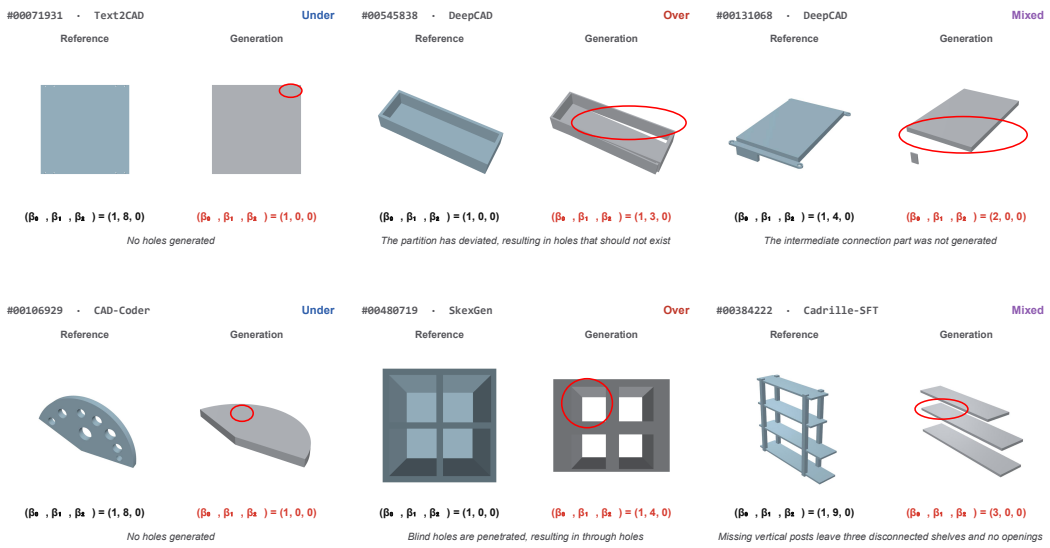


Figure 7: Representative examples of directional topology errors.

Table 6: Exact-match rates for the full Betti signature and Euler characteristic, computed on valid generations using the same topology labels as Table 1. χ-fidelity is the fraction of generations whose Euler characteristic matches the reference. A matching χ does not imply a matching Betti signature.

Model	Signature Fid. (%)	χ Fid. (%)
DeepCAD	88.85	89.45
SkexGen	84.25	85.20
Text2CAD	80.37	81.10
CAD-Coder	84.07	84.47
Cadrille-SFT	78.20	78.81

E Why Full Betti Signatures? A Comparison with Existing Topology Metrics

Existing CAD evaluation incorporates topology-related information in several more limited forms, including execution or mesh validity and scalar topological descriptors such as the Euler characteristic χ (Xu et al., 2025; Preintner et al., 2025). These criteria provide useful quality signals, but retain less information than the full Betti signature $(\beta_0, \beta_1, \beta_2)$. We compare these criteria on the same generations and under the same topology judge as Table 1.

Validity and mesh-defect measures are necessary but silent about topology Validity only certifies that a solid *exists*; it says nothing about whether the *right* solid was produced. Among generations that pass validity, 11.15%—21.8% still carry a wrong Betti signature (Table 1). Mesh-defect measures such as DangEL (Dangling Edge Length), SIR (Self Intersection Ratio), and FluxEE (Flux Enclosure Error) (Xu et al., 2025) inherit the same limitation: a model can emit a perfectly clean, watertight, non-self-intersecting mesh of the *wrong* solid. Such metrics are therefore insufficient as topology measures: they accept every topologically wrong but well-formed solid.

The Euler characteristic collapses distinct failure modes $\chi = \beta_0 - \beta_1 + \beta_2$ compresses a three-component vector into a scalar. Two consequences follow. First, χ *misses* errors entirely: 2.5–6.0% of signature errors keep χ intact, and these are almost exclusively cancellation errors — the most frequent collision pairs are $(1,0,0) \to (2,1,0)$ and $(1,1,0) \to (2,2,0)$, i.e., a spurious split body ($\Delta\beta_0 = +1$) exactly offset by an extra through-hole ($\Delta\beta_1 = +1$). Second, even when χ *does* change (12.4–17.8% of signature errors alter ≥ 2 Betti components, most with $\Delta\chi \neq 0$), the change

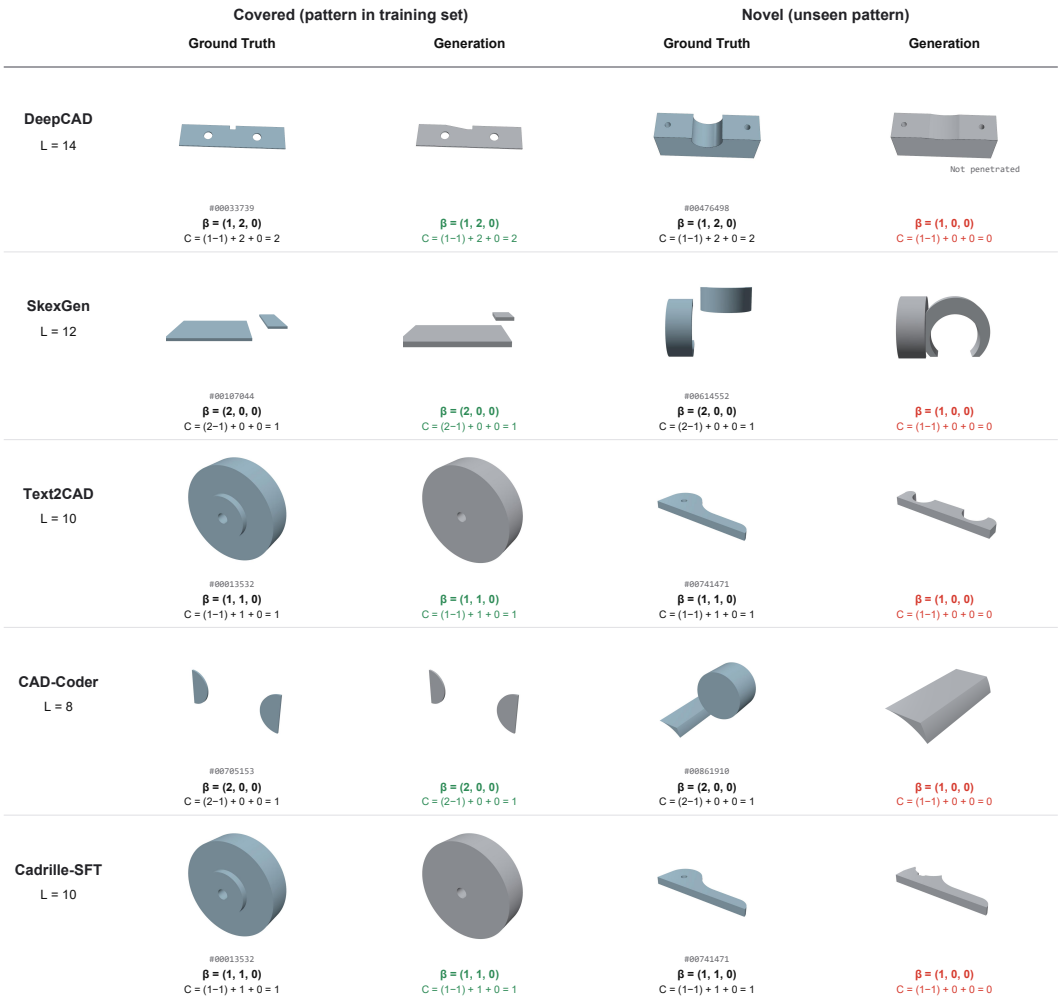


Figure 8: Representative samples illustrating topology realization on construction-covered and construction-novel patterns.

Table 7: Behavior of χ on signature-error samples. "χ-invisible" errors preserve the Euler characteristic; "multi-component" errors change at least two Betti components and cannot be uniquely attributed from χ alone.

Model	#sig. errors	χ-invisible (%)	multi-comp. (%)	of which $\Delta\chi \neq 0$
DeepCAD	794	43 (5.4)	118 (14.9)	75
SkexGen	1014	61 (6.0)	180 (17.8)	119
Text2CAD	1316	49 (3.7)	163 (12.4)	114
CAD-Coder	1186	30 (2.5)	194 (16.4)	164
Cadrille-SFT	1546	43 (2.8)	194 (12.5)	151

is uninterpretable: $\Delta\chi = +1$ is equally consistent with a lost hole ($\Delta\beta_1 = -1$) and a split body ($\Delta\beta_0 = +1$). The scalar detects that *something* changed but cannot say *what* (Figure 9).

Full signatures enable component-level diagnosis Because the signature keeps all three components separate, topology errors can be localized to the affected structural components—connected components (β_0), tunnels (β_1), or enclosed cavities (β_2)—and further classified by direction (under- vs. over-realization). This component-level information is essential for the analyses in this paper: it enables the directional breakdown of topology errors, the complexity-stratified fidelity analysis, and the distinction between successful and unsuccessful topology transformations.

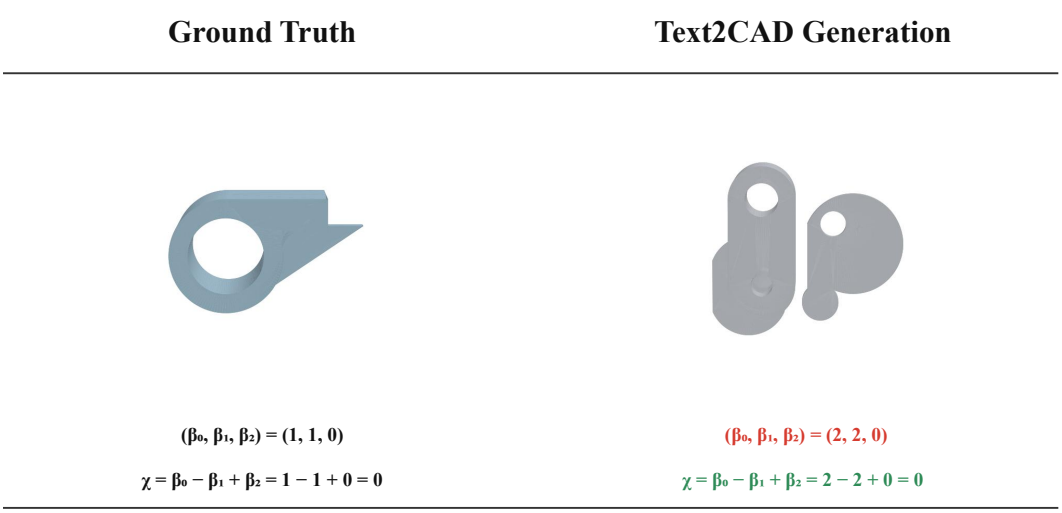


Figure 9: An Euler-characteristic-invisible topology error. The Text2CAD generation has a different Betti signature from the ground truth, $(1, 1, 0)$ versus $(2, 2, 0)$, but both have Euler characteristic $\chi = 0$.

F Representation-Level Effects in DeepCAD

The main text shows that DeepCAD's quantized representation can itself alter the topology of reconstructed CAD models. We investigate this effect in detail here and examine its implications for interpreting the directional topology errors observed for DeepCAD.

DeepCAD represents each CAD construction as a sequence of at most 60 commands with 17 fields each, EOS-padded into a fixed-size 60×17 integer matrix (`MAX_TOTAL_LEN=60`, `cadlib/macro.py`). Each 2D sketch profile is normalized into a unit square independently (the whole model is separately scaled into a 2×2×2 cube, with a per-profile scale factor s stored for recovery). This normalization itself is invertible; however, the profile coordinates, the recovery scale s, and the sketch-plane poses are subsequently *quantized independently* onto the same 8-bit (256-level) grid, so each profile recovers its placement with an independent rounding error (`Extrude.numericalize`, `cadlib/extrude.py`). Cross-profile coinciden that were exact in the original CAD JSON are therefore no longer guaranteed after vectorization, even though the representation preserves the information needed for geometric reconstruction and does not explicitly encode global topological relations across profiles.

Topology changes introduced by quantization. To quantify the resulting effect, we reconstruct the same reference models from both the original unquantized CAD JSON and the quantized vector using the same official CAD builder, and compare their full Betti signatures. Among 7,590 test references successfully labeled in both representations, 762 (10.0%) have different signatures.

The difference is directionally biased toward additional structure in the quantized reconstruction: under the direction definition of Eq. 2 applied to $\Delta\beta = \beta^{\text{vec}} - \beta^{\text{JSON}}$, 621 of the 762 disagreeing cases are net additions of structure, 115 are net removals, and only 26 are mixed.

Why small geometric errors can change topology. The representative cases share the same underlying mechanism: after per-profile normalization to the unit square, DeepCAD quantizes continuous parameters independently onto an 8-bit grid. Geometrically, each scalar deviation is bounded by half a quantization bin, but topology can depend critically on relations among parameters—such as alignment between contacting profiles, equality of coaxial radii, and whether a cut extends beyond the body thickness. Independent rounding can therefore break relations that are exact in the original model, allowing sub-bin geometric perturbations to propagate into changed connectivity, additional or missing handles, and disconnected solids.

Table 8: Direction of topology errors on the 6,828 references whose original and quantized DeepCAD representations have identical Betti signatures. For each model, n is the number of signature-error generations in this subset; directional percentages are computed over those errors. Under:Over is the ratio of under-realization to over-realization errors.

Model	n_{err}	Under	Over	Mixed	Under:Over
DeepCAD	217	100 (46%)	99 (46%)	18 (8%)	1.01×
SkexGen	740	342 (46%)	335 (45%)	63 (9%)	1.02×
Text2CAD	920	554 (60%)	310 (34%)	56 (6%)	1.79×
CAD-Coder	741	545 (74%)	122 (16%)	74 (10%)	4.47×
Cadrille-SFT	1081	720 (67%)	283 (26%)	78 (7%)	2.54×

Implications for interpreting directional errors The representation-level effect is relevant when interpreting DeepCAD's directional topology errors in Section 4.1. If topology is evaluated after reconstruction from the quantized representation, some topology changes are introduced before model prediction errors are considered. We therefore use the original unquantized CAD JSON to define the reference topology. As a sensitivity analysis, we remove the 762 references whose unquantized and quantized signatures disagree and recompute the directional analysis on the remaining 6,828 representation-consistent references (see Table 8).

The effect is substantial in the directional analysis. Among the 6,828 references for which the unquantized and quantized representations agree, DeepCAD's under:over error ratio changes from 0.4× to 1.01×.

These results suggest that a substantial part of DeepCAD's apparent over-realization is attributable to topology changes introduced by its quantized representation, rather than to the generator alone. The persistence of under-realization for the text-conditional generators on the same representation-consistent subset indicates that their directional tendency cannot be explained by this artifact.

This case illustrates why topology should be considered explicitly when designing and evaluating CAD representations. A representation designed primarily for geometric reconstruction can alter global topology as a side effect, while such changes may remain invisible to geometry-oriented evaluation. The DeepCAD case therefore provides a concrete example of how representation-level effects can confound the interpretation of topology errors in generated CAD.

G Model Implementation Details

All five models are evaluated under an identical topology-based protocol: each model produces one solid per test case, the solid is exported to STEP (or decoded to a B-Rep solid directly), and the reference extractor assigns the watertight Betti signature $\beta = (\beta_0, \beta_1, \beta_2)$. Exact signature matching against the reference defines topology fidelity; solids that cannot be built or are not watertight are counted as invalid (failure class F0). Table 9 summarizes the evaluation interface of each model; the per-model details follow.

Table 9: Evaluation interface of the five CAD generators. "Seq." denotes a CAD construction sequence in the DeepCAD command vocabulary; "Code" denotes an executable CadQuery program.

Model	**Paradigm**	**Input at test time**	**Output form**
DeepCAD	latent autoregressive	latent code (reconstruction)	Seq. → B-Rep
SkexGen	codebook disentangled	shape codes (reconstruction)	Seq. → STEP
Text2CAD	text-conditional seq.	expert-level description	Seq. → STEP
CAD-Coder	code-generating LM	expert-level description	Code → STEP
Cadrille-SFT	code-generating VLM	expert-level description	Code → STEP

DeepCAD. We use the officially released pretrained DeepCAD model. The generation side is decoded from the released 8-bit quantized command vectors with the official `vec2CADsolid` implementation, so that the evaluated geometry passes through the same quantization stage as the model's own reconstruction chain. The reference geometry is decoded independently from the unquantized CAD JSON sequences with the official `create_CAD` builder; the comparison of the two references is the subject of Appendix F. Both decoders are used unmodified.

SkexGen. We use the officially released SkexGen model and its reconstruction pipeline. Reconstructions of the test split are exported with the official converter and evaluated as STEP solids. No text input is involved; SkexGen serves as the reconstruction-based, non-text-conditioned baseline.

Text2CAD. We use the officially released Text2CAD checkpoint with greedy decoding. Each test case is conditioned on the expert-level textual description from the official test split (7,924 prompts covering the DeepCAD test cases with a valid vectorization). The generated command sequence is converted to a solid with the official CadSeqProc post-processing pipeline.

CAD-Coder. CAD-Coder generates CadQuery programs rather than command sequences. Each test description is fed to the model, and the returned program is executed in a sandboxed CadQuery 2.x environment to obtain the final solid.

Cadrille-SFT. Cadrille-SFT is the supervised fine-tune of the Cadrille model (Qwen2-VL backbone) on the DeepCAD-compatible training component. At test time the model is prompted with the expert-level description only (text-only interface) and generates a CadQuery program, which is executed as for CAD-Coder.

Decoding and reproducibility. All text-conditioned models decode greedily (temperature 0).