BREPPREDIFF: DIFFUSION-BASED SELF-SUPERVISED REPRESENTATION LEARNING FOR BOUNDARY REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models have recently demonstrated strong potential for generating structured boundary representation (B-Rep) models by modeling complex geometric distributions and dependencies. However, their use for transferable B-Rep representation learning remains largely unexplored. Meanwhile, large-scale Computer-Aided Design (CAD) datasets are predominantly unlabeled, and models stored in intermediate exchange formats typically lack their original modeling histories, making scalable semantic representation learning from CAD data particularly challenging. To address this problem, we introduce **BRepPreDiff**, a diffusion-based self-supervised pre-training framework for learning transferable B-Rep representations across downstream tasks, including face-level semantic segmentation and shape classification. BRepPreDiff consists of a *topology-aware B-Rep encoder* and a *diffusion-based pre-training* strategy. The encoder employs edge-update attention to explicitly model geometric and topological interactions between adjacent B-Rep faces and their relations. During pre-training, we preserve the B-Rep topology while corrupting geometric features at sampled noise levels and train the encoder to predict the injected noise and reconstruct clean geometric features. This multi-level denoising objective enables the model to learn transferable geometric and structural features from unlabeled B-Rep data. For downstream adaptation, the pretrained encoder is jointly fine-tuned with lightweight MLP heads for face-level semantic segmentation and whole-model classification. BRepPreDiff achieves 86.64% mIoU on Fusion360Seg and 86.38% classification accuracy on TMCAD, exceeding the strongest compared baselines by 5.05 and 2.28 percentage points, respectively. We further introduce **BLEND**, a CAD dataset comprising 18K B-Rep models with face-level blend annotations, providing a large-scale benchmark for blend-face recognition and semantic supervision for design-intent recovery and feature-aware CAD analysis.

1 INTRODUCTION

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Nichol & Dhariwal, 2021; Song et al., 2021; Rombach et al., 2022) have recently demonstrated strong capabilities in generating structured boundary representations (B-Reps), showing that iterative denoising can effectively model the complex geometric distributions and dependencies underlying Computer-Aided Design (CAD) models (Xu et al., 2024; Guo et al., 2025; Li et al., 2025; Lee et al., 2025; Zhang et al., 2025; Hou et al., 2026; Wang et al., 2026a; 2025; 2026b; 2024). Beyond generation, this denoising process suggests a natural mechanism for representation learning: recovering clean geometry from progressively corrupted observations requires the model to capture regularities not only within individual geometric entities, but also across their topological relationships. This motivates us to investigate diffusion models not merely as generative models for B-Reps, but as a self-supervised objective for learning transferable geometric and topological representations.

Self-supervised representation learning is particularly well suited to CAD data: large-scale CAD repositories contain abundant B-Rep models, yet the vast majority of them are unlabeled, whereas datasets with annotations for downstream tasks such as face-level semantic segmentation or shape classification are typically much smaller. This imbalance between readily available unlabeled data

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.

and scarce task-specific annotations naturally favors a pre-training and fine-tuning paradigm, in which a representation encoder is first trained on large-scale unlabeled B-Reps and subsequently adapted to downstream CAD understanding tasks using limited labeled data.

Existing self-supervised representation learning methods for B-Reps, however, predominantly follow the masked autoencoding paradigm, where a subset of geometric or structural information is removed and reconstructed from the remaining context (Lou et al., 2023; Yao et al., 2026; Li et al., 2026). While effective, masked reconstruction typically defines a single corruption pattern and requires the model to recover missing information in one step. In contrast, diffusion-based pre-training exposes the model to a continuum of corruption levels and trains it to progressively recover clean geometric information from noisy observations. This multi-level denoising objective provides supervision across varying degrees of information degradation, potentially encouraging the encoder to learn more robust representations of the underlying geometric distribution and structural dependencies of B-Rep models. Motivated by the imbalance between abundant unlabeled CAD data and scarce task-specific annotations, together with the variable topology of B-Rep models and the limitations of existing mask-based pre-training paradigms, we introduce **BRepPreDiff**, a diffusion-based self-supervised representation learning framework designed specifically for B-Rep models.

In addition, blend-face recognition highlights both the practical importance of CAD understanding and the need for task-specific annotations. Identifying blend faces is essential for recovering design intent, feature-aware editing, and geometry simplification for simulation. Yet this task requires more than recognizing surface types, as the same cylindrical, toroidal, or spherical geometry can occur in both blend and non-blend faces, making geometric and topological context crucial. Existing public CAD datasets are synthetic and limited in scale, with few explicitly distinguishing edge-from vertex-blend faces, which limits the recognition performance and generalizability of deep models. To address this gap, we introduce **BLEND**, a dataset of approximately 18K real-world B-Rep models with face-level annotations for non-blend, edge-blend, and vertex-blend faces, providing a benchmark for evaluating blend-related tasks.

Our contributions are as follows:

- We develop a topology-aware B-Rep encoder with edge-update attention that explicitly captures geometric and topological interactions between adjacent faces.
- We enable transferable B-Rep feature learning through topology-preserving multi-level denoising, formulating diffusion as a self-supervised representation learning objective.
- We introduce **BLEND**, a dataset containing 18K real-world B-Rep models with fine-grained face-level blend annotations that explicitly distinguish edge-blend faces (EBFs) from vertex-blend faces (VBFs).

2 Related Work

B-Rep encoding. B-Rep encoders combine parametric geometry with explicit topology. BRepNet (Lambourne et al., 2021) aggregates local topological neighborhoods, while UV-Net (Jayaraman et al., 2021) combines sampled curve and surface features with face-adjacency message passing. Hierarchical CADNet (Colligan et al., 2022) and AAGNet (Wu et al., 2024) extend graph-based encoding to machining feature recognition. BRepMFR (Zhang et al., 2024) combines Transformer-based graph encoding with domain adaptation to improve transfer from synthetic to real CAD models. Attention-based methods, including BRepFormer (Dai et al., 2025) and Bringing Attention to CAD (Zou & Zhu, 2025), model broader geometric and topological dependencies. Our encoder builds on this direction with edge-update attention that jointly updates face and edge representations.

Self-supervised B-Rep learning. Existing methods learn transferable features from unlabeled CAD geometry through several objectives. SSL4CAD (Jones et al., 2023) learns hybrid implicit–explicit surface representations, and BRep-BERT (Lou et al., 2023) combines masked token prediction with subgraph contrastive learning. BRepMAE (Yao et al., 2026) and the hierarchical masked autoencoder of Li et al. (2026) reconstruct masked geometry and attributes, whereas Brep2Shape (Sun et al., 2026) predicts sampled geometry from analytical B-Rep descriptions. We instead study diffusion denoising across multiple noise levels as a pre-training objective, preserving

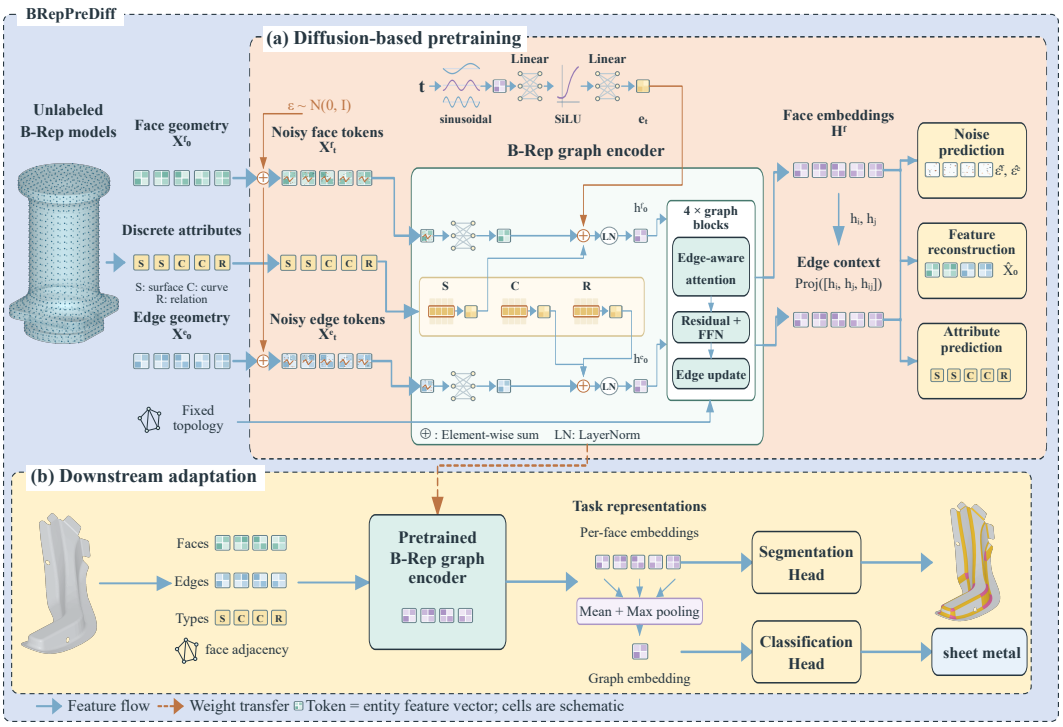

Figure 1: **Overview of BRepPreDiff.** (a) **Diffusion-based pretraining** learns a B-Rep encoder by predicting noise and reconstructing geometry. (b) **Downstream adaptation** transfers the pretrained encoder to face segmentation and whole-model classification.

the input topology while recovering corrupted geometric attributes for downstream segmentation and classification.

Diffusion models for CAD generation. BrepGen (Xu et al., 2024) generates B-Reps through hierarchical latent diffusion, while BrepGiff (Guo et al., 2025) uses graph-attention diffusion and DTGBrepGen (Li et al., 2025) separates topology and geometry generation. BrepDiff (Lee et al., 2025) adopts a single-stage formulation using masked UV grids, and HiFi-BRep (Hou et al., 2026) performs diffusion in a topology-aware latent space. These methods target B-Rep synthesis; our focus is learning a reusable encoder from existing B-Reps through denoising supervision.

3 Method

BRepPreDiff learns transferable B-Rep representations by recovering geometric attributes from noisy observations while retaining the original face-adjacency structure. Fig. 1 summarizes two stages. During pre-training, continuous face and edge features are corrupted, combined with unchanged categorical attributes, and processed by a time-conditioned graph encoder. Face embeddings and contextual edge embeddings support joint noise prediction, clean-feature reconstruction, categorical prediction, and relation prediction. During downstream adaptation, the encoder weights are transferred to a model that processes clean B-Reps for face segmentation or whole-model classification. We describe this data flow below and retain its principal equations in the main text; Appendix A provides supplementary feature definitions and implementation details.

3.1 B-Rep Graph and Structure-Preserving Corruption

We represent a solid as a directed attributed face-adjacency graph

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X}_0^f, \mathbf{X}_0^e, \mathbf{s}, \mathbf{c}, \mathbf{r}), \tag{1}$$

where the N faces form $\mathcal{V}$ and each shared topological curve induces two directed edges in $\mathcal{E}$. The matrices $\mathbf{X}_0^f \in \mathbb{R}^{N\times d_f}$ and $\mathbf{X}_0^e \in \mathbb{R}^{M\times d_e}$ contain clean, preprocessed face and edge features, with $M = |\mathcal{E}|$. Face features describe area, centroid, oriented normal, curvature, and a sampled UV grid of points, normals, and trimming indicators. Edge features contain curve length, relative face orientation, and sampled curve points and tangents. The categorical attributes $\mathbf{s}$, $\mathbf{c}$, and $\mathbf{r}$ encode surface type, curve type, and smooth/sharp adjacency relations. Geometric channels are standardized within each graph; trimming indicators remain binary and edge tangents are normalized before corruption. Appendix A.1 specifies the feature layout, preprocessing, and corruption schedule.

As shown in Fig. 1(a), we preserve $\mathcal{V}$, $\mathcal{E}$, and all categorical attributes and corrupt only the face and edge feature tensors. For each graph, we sample one timestep $t \sim \mathrm{Uniform}\{0,\dots,T-1\}$ and form

$$\mathbf{X}_t^q = \sqrt{\bar{\alpha}_t}\,\mathbf{X}_0^q + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}^q, \qquad \boldsymbol{\epsilon}^q \sim \mathcal{N}(\mathbf{0},\mathbf{I}), \quad q \in \{f,e\}, \tag{2}$$

where $\bar{\alpha}_t = \prod_{\tau=0}^{t}(1-\beta_\tau)$. Face and edge noise are independent, while their shared timestep gives a common corruption level within the solid. Preserving connectivity maintains entity correspondence and provides neighboring context for denoising. Importantly, the noisy tensors represent observations in feature space and therefore need not correspond to valid CAD geometry.

3.2 Topology-aware B-Rep encoder with edge-update attention

The central encoder in Fig. 1(a) first projects face and edge features into a common latent space. A sinusoidal timestep embedding passes through two linear layers separated by SiLU to obtain $\boldsymbol{\gamma}(t)$. The initial states are

$$\mathbf{h}_i^{(0)} = \mathrm{LN}\Big(\mathbf{W}_f \mathbf{x}_{t,i}^f + \mathbf{E}_s(s_i) + \boldsymbol{\gamma}(t)\Big), \tag{3}$$

$$\mathbf{e}_{ji}^{(0)} = \mathrm{LN}\big(\mathbf{W}_e \mathbf{x}_{t,ji}^e + \mathbf{E}_c(c_{ji}) + \mathbf{E}_r(r_{ji})\big), \tag{4}$$

where $\mathbf{E}_s$, $\mathbf{E}_c$, and $\mathbf{E}_r$ are learned categorical embeddings and LN is layer normalization. Time conditioning enters the face stream and propagates to edge states through subsequent graph blocks.

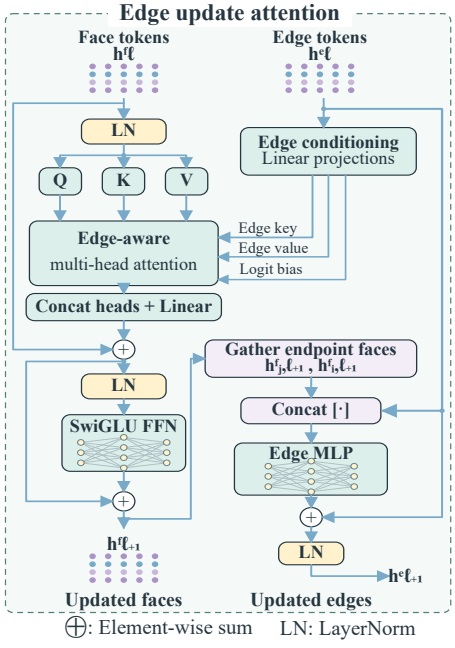

Figure 2: Edge-update attention uses edge tokens to guide face attention and updated face tokens to refine edge representations.

Edge-update attention. Fig. 2 illustrates a graph block that couples face and edge updates: edge tokens first condition message passing between adjacent faces, and the updated face tokens then refine their shared edge tokens. At layer l, let $\mathbf{h}_i^{(l)}$ denote the token of face i and $\mathbf{e}_{ji}^{(l)}$ the token of the directed edge from face j to face i. The face stream first applies layer normalization, $\bar{\mathbf{h}}_i^{(l)} = \mathrm{LN}(\mathbf{h}_i^{(l)})$, and projects the normalized tokens into queries, keys, and values. In parallel, three learned projections of each edge token provide an edge key, an edge value, and a scalar logit bias.

The resulting attention weights and messages are

$$\alpha_{ji}^{(l)} = \mathrm{softmax}_{j\in\mathcal{N}(i)}\left[\frac{(\mathbf{W}_Q\bar{\mathbf{h}}_i^{(l)})^\top(\mathbf{W}_K\bar{\mathbf{h}}_j^{(l)} + \mathbf{W}_{K,e}\mathbf{e}_{ji}^{(l)})}{\sqrt{d_h}} + \mathbf{w}_b^\top \mathbf{e}_{ji}^{(l)}\right], \tag{5}$$

$$\mathbf{m}_i^{(l)} = \sum_{j\in\mathcal{N}(i)} \alpha_{ji}^{(l)}\left(\mathbf{W}_V\bar{\mathbf{h}}_j^{(l)} + \mathbf{W}_{V,e}\mathbf{e}_{ji}^{(l)}\right). \tag{6}$$

where $\mathcal{N}(i) = \{j : (j,i) \in \mathcal{E}\}$ is the set of incoming neighbors and d_h is the dimension of each head. The edge key and logit bias modulate the relative importance of adjacent faces, while the edge

value injects shared-curve information into the aggregated message. Attention is normalized over the incoming neighbors of each destination face, preserving the face-adjacency structure.

The head outputs are concatenated, linearly projected, and added to the input face token through a residual connection. A pre-normalized SwiGLU feed-forward network with a second residual connection then produces the updated face token:

$$\widetilde{\mathbf{h}}_i^{(l)} = \mathbf{h}_i^{(l)} + \mathrm{Drop}\Big(\mathbf{W}_O \Big\|_{a=1}^{H} \mathbf{m}_i^{(l,a)}\Big), \tag{7}$$

$$\mathbf{h}_i^{(l+1)} = \widetilde{\mathbf{h}}_i^{(l)} + \mathrm{Drop}\Big(\mathrm{SwiGLU}\Big(\mathrm{LN}(\widetilde{\mathbf{h}}_i^{(l)})\Big)\Big), \tag{8}$$

where H is the number of heads, a indexes the heads, and $\|$ denotes concatenation.

The edge stream gathers the two updated endpoint face tokens and concatenates them with the input edge token. An edge MLP predicts a residual correction, which is added to the input edge token and normalized:

$$\mathbf{e}_{ji}^{(l+1)} = \mathrm{LN}\Big(\mathbf{e}_{ji}^{(l)} + \mathrm{Drop}\Big[\mathrm{MLP}_e^{(l)}([\mathbf{h}_j^{(l+1)} \| \mathbf{h}_i^{(l+1)} \| \mathbf{e}_{ji}^{(l)}])\Big]\Big). \tag{9}$$

This face–edge coupling allows each shared-curve representation to incorporate information from both incident surfaces and condition face attention in the next block. Appendix A.2 specifies the timestep embedding and attention-head parameterization.

The distinctive role of edge-update attention is to make shared-edge representations adaptive mediators of face interactions: edge geometry guides face aggregation, and the resulting face context refines edge representations for subsequent layers. This reciprocal update explicitly couples surface and curve information throughout the encoder. Attention is computed only along existing adjacency edges between incident faces, embedding the local B-Rep topology into feature propagation and keeping the number of attention scores proportional to the number of graph edges.

3.3 Joint Geometric and Attribute Prediction

The noise prediction, feature reconstruction, and attribute prediction branches in Fig. 1(a) jointly supervise the encoder through four losses: noise, reconstruction, categorical, and relation prediction. Face decoders use $\mathbf{h}_i^{(L)}$ directly, while edge decoders use a contextual projection of the incident face embeddings and final edge state:

$$\mathbf{z}_i^f = \mathbf{h}_i^{(L)}, \qquad \mathbf{z}_{ji}^e = \mathrm{MLP}_{\mathrm{ctx}}([\mathbf{h}_j^{(L)} \| \mathbf{h}_i^{(L)} \| \mathbf{e}_{ji}^{(L)}]). \tag{10}$$

For each type of entity $q \in \{f, e\}$, independent MLP heads predict injected noise $\widehat{\boldsymbol{\epsilon}}^q = D_\epsilon^q(\mathbf{Z}^q)$ and clean features $\widehat{\mathbf{X}}_0^q = D_x^q(\mathbf{Z}^q)$, where $\mathbf{Z}^q$ stack the corresponding entity states. The clean-feature prediction is a separate decoder output. The two continuous losses are

$$\mathcal{L}_{\mathrm{noise}} = \sum_{q\in\{f,e\}} \mathrm{MSE}(\widehat{\boldsymbol{\epsilon}}^q, \boldsymbol{\epsilon}^q), \tag{11}$$

$$\mathcal{L}_{\mathrm{recon}} = \sum_{q\in\{f,e\}} \mathrm{MAE}(\widehat{\mathbf{X}}_0^q, \mathbf{X}_0^q). \tag{12}$$

The attribute prediction branch uses separate MLP heads to produce surface-type logits $\widehat{\mathbf{O}}^s = D_s(\mathbf{Z}^f)$, curve-type logits $\widehat{\mathbf{O}}^c = D_c(\mathbf{Z}^e)$, and adjacency-relation logits $\widehat{\mathbf{O}}^r = D_r(\mathbf{Z}^e)$. These define the categorical and relational losses:

$$\mathcal{L}_{\mathrm{cat}} = \mathrm{CE}(\widehat{\mathbf{O}}^s, \mathbf{s}) + \mathrm{CE}(\widehat{\mathbf{O}}^c, \mathbf{c}), \tag{13}$$

$$\mathcal{L}_{\mathrm{rel}} = \mathrm{CE}(\widehat{\mathbf{O}}^r, \mathbf{r}). \tag{14}$$

Here MSE and MAE average over the entries of each tensor, and CE averages cross-entropy over the corresponding entities. All edge-specific terms are zero when there are no adjacency edges. The categorical attributes remain unchanged during corruption and serve as both encoder inputs and prediction targets.

The experiments jointly optimize all four losses using

$$\mathcal{L}_{\mathrm{pre}} = \lambda_{\mathrm{noise}}\mathcal{L}_{\mathrm{noise}} + \lambda_{\mathrm{recon}}\mathcal{L}_{\mathrm{recon}} + \lambda_{\mathrm{cat}}\mathcal{L}_{\mathrm{cat}} + \lambda_{\mathrm{rel}}\mathcal{L}_{\mathrm{rel}}, \tag{15}$$

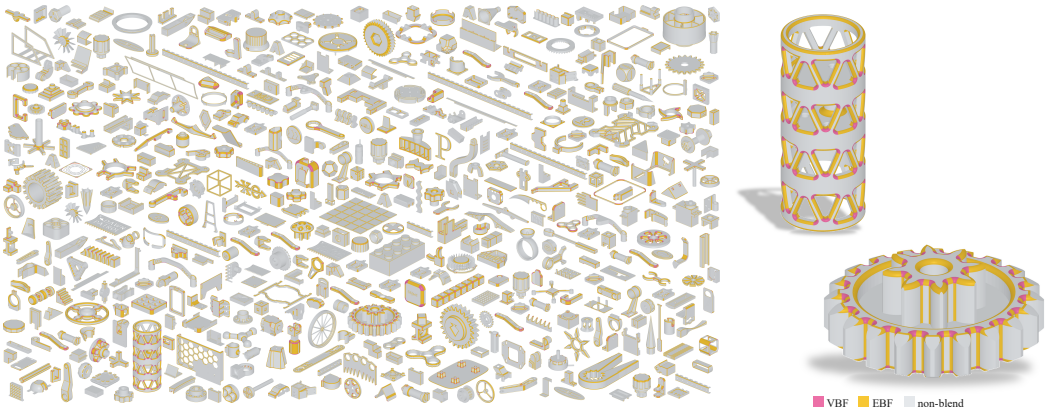

Figure 3: **BLEND dataset.** Examples of B-Rep models for face-level blend recognition.

Noise prediction supplies supervision across corruption levels, direct L1 reconstruction anchors the learned states to clean geometry, and categorical and relation prediction supervise surface and curve types, and smooth/sharp adjacency attributes, respectively. Together, these objectives train the shared encoder to represent continuous geometry and discrete B-Rep attributes.

3.4 Downstream Adaptation

As illustrated by the weight-transfer arrow in Fig. 1(b), downstream models retain the pretrained graph encoder and discard the pre-training prediction heads. The encoder receives clean preprocessed features and categorical attributes, with the timestep term omitted from equation 4.

For segmentation, a head predicts each face label from $\mathbf{h}_i^{(L)}$. For whole-model classification, the Mean+Max pooling module combines complementary statistics of all face embeddings. Its residual implementation is

$$\boldsymbol{\mu}_g = \frac{1}{N_g} \sum_{i \in \mathcal{V}_g} \mathbf{h}_i^{(L)}, \qquad \mathbf{m}_g = \max_{i \in \mathcal{V}_g} \mathbf{h}_i^{(L)}, \qquad \mathbf{g}_g = \boldsymbol{\mu}_g + R([\boldsymbol{\mu}_g \| \mathbf{m}_g]), \tag{16}$$

where the maximum is element-wise, and R is an MLP with a zero-initialized final projection. The graph embedding therefore starts as mean pooling and learns a correction from the concatenated mean and max statistics. Task-specific MLP heads map face or graph embeddings to class logits, using cross-entropy plus soft Dice for segmentation and cross-entropy for classification.

4 BLEND Dataset

Identifying blend faces is essential for recovering design intent, editing CAD models, and simplifying geometry for simulation. Edge-blend faces (EBFs) smooth sharp edges between adjacent faces, while vertex-blend faces (VBFs) connect neighboring blends where multiple blended edges meet.

The public dataset released with FilletRec (Gao et al., 2025) contains only 4,486 synthetic CAD models and provides binary fillet/non-fillet labels without distinguishing between EBFs and VBFs. However, EBFs and VBFs arise from different local blending operations and play distinct topological roles. Treating them as a single category therefore obscures important design intent and limits fine-grained blend-feature recognition and subsequent feature-aware CAD processing.

To address these limitations, we introduce **BLEND**, comprising 17,869 real-world B-Rep models randomly sampled from a cleaned subset of the ABC dataset (Koch et al., 2019). Engineers with CAD modeling experience manually annotate the faces and cross-check the labels, yielding 1,030,152 annotations across three classes: non-blend, EBF, and VBF. Fig. 3 shows representative models.

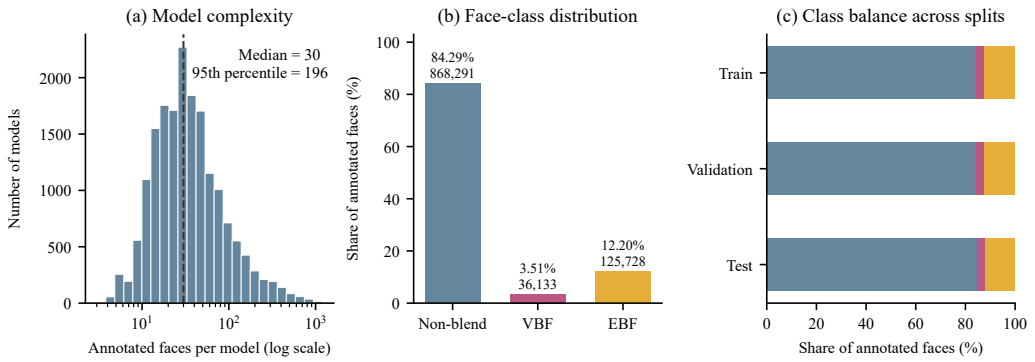

Figure 4: **BLEND dataset statistics.** (a) Faces per model on a logarithmic horizontal axis; the dashed line marks the median. (b) Face counts and proportions by class. (c) Face-class proportions within each split.

BLEND spans models with 3–1,097 faces, enabling evaluation across a wide range of B-Rep complexity (Fig. 4). EBFs and VBFs account for only 12.20% and 3.51% of all faces, respectively, making recognition of sparse blend faces a key challenge. Further statistics and split details are provided in Appendix B.

5 Experiments

5.1 Settings

Pre-training corpus. We pre-train on 512,605 B-Rep models collected from ten data sources: a processed subset of ABC (Koch et al., 2019), obtained by filtering invalid models, removing assemblies to retain only single-part models, and deduplicating the remaining data; the Fusion 360 reconstruction (Willis et al., 2021a) and assembly (Willis et al., 2021b) datasets; and the training splits of seven downstream benchmarks: BLEND, Fusion 360 Segmentation (Lambourne et al., 2021), MFCAD++ (Colligan et al., 2022), TMCAD (Zou & Zhu, 2025), SolidLetters (Jayaraman et al., 2021), CADSynth (Zhang et al., 2024), and MFInstSeg (Wu et al., 2024). All task-specific annotations are removed before pre-training (Table 5 in Appendix B).

Training configuration. The encoder uses four Edge Update Attention layers with hidden width 128, four attention heads, and dropout of 0.1. The forward diffusion process uses $T = 1000$ steps with a linear variance schedule from 10^{-4} to 2×10^{-2}. We pre-train with AdamW for 100 epochs, a constant learning rate of 10^{-4}, weight decay of 10^{-4}, and a batch size of 128. The full objective uses noise, clean-reconstruction, categorical, and relation weights of 1.0, 0.5, 0.5, and 0.3, respectively.

The entire encoder is fine-tuned jointly with the task-specific MLP head using AdamW for 200 epochs, a learning rate of 3×10^{-4}, a weight decay of 10^{-4}, and a batch size of 256.

Evaluation Protocol. We fix downstream splits before pre-training and exclude all validation and test geometries from the pre-training corpus. Fine-tuning uses only training labels, with validation reserved for model selection. Appendix B describes the downstream benchmarks and lists their class counts and split sizes. We evaluate segmentation at the face level and classification at the model level; Appendix C defines accuracy, F1, and mIoU and their aggregation conventions.

5.2 Main Results

For fair comparisons, all self-supervised baselines are pre-trained on the same corpus as BRepPreDiff, except Brep2Shape. Because Brep2Shape does not publicly release its B-Rep model preprocessing code, we use its officially released checkpoint for downstream fine-tuning. All methods use the same downstream training, validation, and test splits as BRepPreDiff, with hyperparameters taken from their official code or original papers.

Table 1: **Test performance on synthetic and real-world B-Rep datasets.** Methods are grouped into B-Rep encoding and self-supervised learning. We report segmentation accuracy (Acc) and mean intersection over union (mIoU), and classification accuracy for SolidLetters and TMCAD. All results are percentages (higher is better). Bold values indicate the best result in each column; "–" denotes unavailable results.

Method	Synthetic datasets							Real-world datasets				
	MFCAD++		CADSynth		MFInstSeg		SolidLetters	Fusion360Seg		BLEND		TMCAD
	Acc	mIoU	Acc	mIoU	Acc	mIoU	Acc	Acc	mIoU	Acc	mIoU	Acc
B-Rep Encoding												
BRepNet	96.57	93.37	99.02	97.01	96.70	93.37	–	93.38	78.68	98.47	91.47	–
UV-Net	98.92	96.47	99.46	97.62	98.93	96.34	97.38	88.77	67.75	98.35	91.89	83.99
AAGNet	99.24	98.63	99.63	**99.33**	99.14	98.40	–	80.90	73.75	76.43	53.03	–
BRepMFR	**99.70**	**99.05**	97.91	92.87	**99.67**	**98.82**	–	68.27	28.86	97.26	86.38	–
BRT	98.52	96.17	99.51	98.21	98.51	95.83	96.89	96.37	81.59	98.27	93.12	84.10
Self-Supervised Learning												
Brep2Shape	99.39	98.09	**99.87**	99.30	99.32	97.64	95.94	95.99	80.86	83.73	47.48	81.39
SSL4CAD	93.34	77.98	95.55	81.44	92.26	75.45	95.01	80.65	46.06	93.75	73.97	74.52
BRep-BERT	98.29	96.59	99.47	98.34	97.05	94.71	96.15	90.46	69.96	98.74	93.88	83.90
BRepMAE	99.53	98.67	99.83	99.24	99.55	98.51	96.80	92.78	79.08	98.86	94.10	83.86
BRepPreDiff	99.62	98.87	99.65	98.97	99.45	98.29	**97.46**	**96.93**	**86.64**	**98.99**	**94.92**	**86.38**

Quantitative Results. As shown in Table 1, BRepPreDiff consistently outperforms all baselines across the three real-world benchmarks, surpassing the strongest competing methods by 5.05 and 0.82 mIoU points on Fusion360Seg and BLEND, respectively, and by 2.28 accuracy points on TMCAD. It also achieves the highest accuracy on SolidLetters (97.46%) and remains competitive on MFCAD++, CADSynth, and MFInstSeg, where the top-performing methods already achieve similarly high scores. These results demonstrate the strong transferability of BRepPreDiff across diverse CAD datasets and downstream tasks.

Table 2: Training cost and cached-input inference latency on Fusion360Seg. Batch sizes are 16 for training (Appendix D) and 1 for inference (Fig. 6).

Method	Params. (M)	Time (s/epoch)	Peak mem. (MiB)	Inference (ms/model)
BRep-BERT†	25.040	140.08	1,393.13	10.45
BRepMAE	4.304	135.13	1,020.64	23.57
BRepPreDiff	**1.844**	**29.17**	**153.40**	**6.79**

† Our reimplementation.

Table 2 compares the efficiency of BRepPreDiff with BRep-BERT and BRepMAE on Fusion360Seg using a single NVIDIA A800 80 GB GPU in FP32. We measure training time per epoch with a batch size of 16 and cached-input inference latency with a batch size of 1, excluding offline feature extraction; detailed protocols are provided in Appendix D and Fig. 6. With only 1.844M parameters, BRepPreDiff uses 92.6% and 57.1% fewer parameters than BRep-BERT and BRepMAE, respectively, while achieving $4.8\times$ and $4.6\times$ training speedups. Its peak memory usage drops to 153.40 MiB, corresponding to reductions of 89.0% and 85.0%, while inference latency decreases from 10.45 ms and 23.57 ms per model to 6.79 ms. Despite these substantial efficiency gains, BRepPreDiff also achieves higher Fusion360Seg mIoU than both baselines (Table 1).

Qualitative Results. Appendix E presents qualitative comparisons on SolidLetters, Fusion360Seg, and BLEND (Figures 5 and 7–10).

Limited supervision. Table 3 demonstrates the label efficiency of BRepPreDiff across different supervision ratios. On TMCAD and Fusion360Seg, BRepPreDiff consistently outperforms BRepMAE in both accuracy and mIoU, with the only exception (mIoU at 0.1% labels on Fusion360Seg). The advantage is particularly pronounced under extremely limited supervision: on BLEND, BRepPreDiff improves mIoU by 17.9 percentage points with only 0.1% labels. As more labels become available (e.g., 1% and 2%), the performance gap narrows, while BRepPreDiff regains the advantage

Table 3: Test accuracy and mIoU (%) at different labeled training-model ratios. Bold marks the higher value for each dataset, ratio, and metric.

Dataset	Method	0.1%		1%		2%		3%		100%	
		Acc.	mIoU	Acc.	mIoU	Acc.	mIoU	Acc.	mIoU	Acc.	mIoU
BLEND	BRepMAE	92.53	64.17	**97.76**	**87.78**	**98.05**	89.23	98.15	90.27	98.86	94.10
	BRepPreDiff	**96.28**	**82.07**	97.57	86.99	97.97	**89.53**	**98.26**	**91.22**	**98.99**	**94.92**
Fusion360Seg	BRepMAE	67.65	**37.36**	82.30	53.16	84.70	61.47	86.00	63.15	92.78	79.08
	BRepPreDiff	**73.42**	36.79	**83.16**	**54.00**	**87.09**	**62.38**	**89.59**	**67.93**	**96.93**	**86.64**
TMCAD	BRepMAE	14.67	5.02	45.94	27.94	53.72	35.84	61.96	44.42	83.86	71.09
	BRepPreDiff	**21.25**	**10.97**	**51.24**	**35.01**	**59.43**	**43.76**	**65.13**	**49.80**	**86.38**	**75.94**

from 3% supervision onward. Overall, these results demonstrate the transferability of BRepPreDiff under limited supervision.

5.3 Ablation Study

Table 4 examines encoder architecture, geometric inputs, adaptation, and pre-training on Fusion360Seg.

Table 4: Ablations on Fusion360Seg (in %).

Variant	Fusion360Seg		
	Acc.	macro-F1	mIoU
Full model	**96.93**	**92.23**	**86.64**
Encoder architecture			
E1: Fixed edges	96.74	90.91	84.90
E2: Edge-only updates	96.66	91.82	85.97
MP: MLP message passing	96.48	90.36	84.01
Input information			
w/o face geometry	96.76	91.21	85.27
w/o edge geometry	94.12	87.45	79.35
w/o face and edge grids	84.01	76.97	64.06
Adaptation and pre-training			
Frozen-encoder linear probing	71.95	48.77	36.25
Training from scratch	96.89	91.81	85.93
Cross-dataset generalization			
Pre-training w/o Fusion360Seg	97.02	91.65	85.98

Encoder architecture. Keeping edge embeddings fixed across layers (E1) or excluding face context from edge updates (E2) lowers performance, supporting iterative edge updates guided by neighboring faces. The further decline with MLP message passing highlights the efficiency of our proposed edge-update attention.

Input information. Removing face and edge grids causes the largest degradation, highlighting the importance of sampled surface and curve geometry. Masking edge geometry is also more harmful than masking basic face attributes and surface types while retaining face grids, underscoring the contribution of edge information.

Effects of fine-tuning and pre-training. Joint fine-tuning with an MLP head substantially outperforms frozen-encoder linear probing, supporting task-specific adaptation. Pre-training provides a modest additional gain over training from scratch under full supervision.

Cross-dataset generalization. When Fusion360Seg is excluded from pre-training, BRepPreDiff retains comparable performance after fine-tuning on its training split. This result demonstrates that the learned representations transfer effectively to datasets unseen during pre-training.

6 Conclusion

We introduced **BRepPreDiff**, a framework that combines topology-preserving diffusion pre-training with face-conditioned edge-update attention. Compared with MAE-based methods, BRepPreDiff enables more efficient self-supervised learning, with a smaller model size and only one-quarter of the fine-tuning time. In addition, we introduced **BLEND**, an 18K-model benchmark with fine-grained annotations for blend-face recognition. Extensive experiments across multiple benchmarks demonstrate the effectiveness and superior performance of BRepPreDiff.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript writing and language polishing, including improvements to grammar, clarity, and readability. These uses were limited to manuscript preparation and presentation. The authors take full responsibility for the accuracy, originality, and integrity of the final manuscript.

REFERENCES

Andrew R. Colligan, Trevor T. Robinson, Declan C. Nolan, Yang Hua, and Weijuan Cao. Hierarchical CADNet: Learning from B-Reps for machining feature recognition. *Computer-Aided Design*, 147:103226, 2022.

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.

Jiali Gao, Taoran Liu, Hongfei Ye, and Jianjun Chen. FilletRec: A lightweight graph neural network with intrinsic features for automated fillet recognition. *arXiv preprint arXiv:2511.05561*, 2025.

Hao Guo, Xiaoshui Huang, Hao jiacheng, Yunpeng Bai, Hongping Gan, and Yilei Shi. BrepGiff: Lightweight generation of complex B-rep with 3D GAT diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26587–26596, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Junhao Hou, Chenqi Luo, Pufan Wang, Jiaying Lu, Yusheng Liu, Feiwei Qin, Meie Fang, and Kun Zhou. HiFi-BRep: High-Fidelity latent representation for robust B-Rep generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27199–27208, 2026.

Pradeep Kumar Jayaraman, Aditya Sanghi, Joseph G. Lambourne, Karl D. D. Willis, Thomas Davies, Hooman Shayani, and Nigel Morris. UV-net: Learning from boundary representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11703–11712, 2021.

Benjamin T. Jones, Michael Hu, Milin Kodnongbua, Vladimir G. Kim, and Adriana Schulz. Self-supervised representation learning for CAD. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21327–21336, 2023.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9601–9611, 2019.

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

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*, SIGGRAPH Conference Papers '25, pp. 1–11, 2025.

Jing Li, Yihang Fu, and Falai Chen. DTGBrepGen: A novel B-rep generative model through decoupling topology and geometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21438–21447, 2025.

Yifei Li, Kang Wu, Wenming Wu, and Xiao-Ming Fu. Masked BRep autoencoder via hierarchical graph transformer. *arXiv preprint arXiv:2603.14927*, 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*, CIKM '23, pp. 1657–1666, 2023.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Yuanxu Sun, Yuezhou Ma, Haixu Wu, Guanyang Zeng, Muye Chen, Jianmin Wang, and Mingsheng Long. Brep2Shape: Boundary and shape representation alignment via self-supervised transformers. In *Forty-third International Conference on Machine Learning*, 2026.

Hanxiao Wang, Mingyang Zhao, Yiqun Wang, Weize Quan, and Dong-Ming Yan. VQ-CAD: Computer-aided design model generation with vector quantized diffusion. *Computer Aided Geometric Design*, 111:102327, 2024.

Hanxiao Wang, Biao Zhang, Weize Quan, Dong-Ming Yan, and Peter Wonka. iFlame: Interleaving full and linear attention for efficient mesh generation. *arXiv preprint arXiv:2503.16653*, 2025.

Hanxiao Wang, Yuan-Chen Guo, Ying-Tian Liu, Zi-Xin Zou, Biao Zhang, Weize Quan, Ding Liang, Yan-Pei Cao, and Dong-Ming Yan. FACE: A face-based autoregressive representation for high-fidelity and efficient mesh generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12719–12729, 2026a.

Hanxiao Wang, Ying-Tian Liu, Yuan-Chen Guo, Qi-Yuan Feng, Zi-Xin Zou, Ding Liang, Biao Zhang, and Yan-Pei Cao. Nexus: Native mesh generation with diffusion. *ACM Transactions on Graphics*, 45(4), 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 Transactions on Graphics (TOG)*, 40(4), 2021a.

Karl DD Willis, Pradeep Kumar Jayaraman, Hang Chu, Yunsheng Tian, Yifei Li, Daniele Grandi, Aditya Sanghi, Linh Tran, Joseph G Lambourne, Armando Solar-Lezama, and Wojciech Matusik. JoinABLe: Learning bottom-up assembly of parametric CAD joints. *arXiv preprint arXiv:2111.12772*, 2021b.

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 (C):102661, 2024.

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

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.

Aijia Zhang, Weiqiang Jia, Qiang Zou, Yixiong Feng, Xiaoxiang Wei, and Ye Zhang. Diffusion-CAD: Controllable diffusion model for generating computer-aided design models. *IEEE Transactions on Visualization and Computer Graphics*, 31(12):10188–10199, 2025.

Shuming Zhang, Zhidong Guan, Hao Jiang, Xiaodong Wang, and Pingan Tan. BrepMFR: Enhancing machining feature recognition in B-rep models through deep learning and domain adaptation. *Computer Aided Geometric Design*, 111:102318, 2024.

Qiang Zou and Lizhen Zhu. Bringing attention to CAD: Boundary representation learning via transformer. *Computer-Aided Design*, 189:103940, 2025.

A Detailed Formulation and Architecture

This appendix presents the complete formulation in the same subsection order as the method: graph features and corruption, the topology-aware encoder, joint prediction, and downstream adaptation. All model equations are stated explicitly for self-contained reading. The tensors $\widetilde{\mathbf{X}}^f$ and $\widetilde{\mathbf{X}}^e$ denote clean preprocessed face and edge features.

A.1 B-Rep Graph and Structure-Preserving Corruption

We represent a B-Rep solid as a directed attributed graph

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X}^f, \mathbf{X}^e, \mathbf{s}, \mathbf{c}, \mathbf{r}), \tag{17}$$

where $\mathcal{V} = \{v_i\}_{i=1}^N$ contains the N B-Rep faces. For every pair of faces that share a topological edge, we insert two directed graph edges, so $(j, i) \in \mathcal{E}$ denotes a message from face v_j to face v_i. The graph contains $M = |\mathcal{E}|$ directed edges. The matrices $\mathbf{X}^f \in \mathbb{R}^{N \times d_f}$ and $\mathbf{X}^e \in \mathbb{R}^{M \times d_e}$ store continuous face and edge attributes, respectively. The vectors $\mathbf{s}$, $\mathbf{c}$, and $\mathbf{r}$ contain the surface type, curve type, and adjacency-relation category. For face v_i, the continuous feature vector is

$$\mathbf{x}_i^f = \left[\log(1 + A_i), \mathbf{p}_i^{\text{cen}}, \mathbf{n}_i^{\text{cen}}, \boldsymbol{\kappa}_i, \{\mathbf{p}_{iuv}, \mathbf{n}_{iuv}, m_{iuv}\}_{u,v=1}^U\right], \qquad d_f = 11 + 7U^2. \tag{18}$$

Here, A_i is the face area, $\mathbf{p}_i^{\text{cen}} \in \mathbb{R}^3$ is its centroid, and $\mathbf{n}_i^{\text{cen}} \in \mathbb{R}^3$ is the oriented unit normal at the center of the parametric domain. The curvature vector $\boldsymbol{\kappa}_i \in \mathbb{R}^4$ contains the minimum, maximum, mean, and Gaussian curvatures. At each location of a uniform $U \times U$ UV grid, $\mathbf{p}_{iuv} \in \mathbb{R}^3$ and $\mathbf{n}_{iuv} \in \mathbb{R}^3$ record the surface point and oriented unit normal, while $m_{iuv} \in \{0, 1\}$ indicates whether the sample lies inside or on the boundary of the trimmed face. The discrete variable $s_i \in \{0, \ldots, S-1\}$ stores the Open CASCADE surface category. For a directed edge $e = (j, i)$ induced by a shared B-Rep curve, we use

$$\mathbf{x}_{ji}^e = \left[\log(1 + \ell_{ji}), \frac{\theta_{ji}}{\pi}, (\mathbf{n}_j^{\text{cen}})^\top \mathbf{n}_i^{\text{cen}}, \left\{\mathbf{p}_{jik}^e, \boldsymbol{\tau}_{jik}\right\}_{k=1}^K\right], \qquad d_e = 3 + 6K, \tag{19}$$

where ℓ_{ji} is the curve length, θ_{ji} is the angle between the two center normals, and $\mathbf{p}_{jik}^e, \boldsymbol{\tau}_{jik} \in \mathbb{R}^3$ are a sampled curve point and its unit tangent. The categorical attribute $c_{ji} \in \{0, \ldots, C_e - 1\}$ specifies the curve type. The relation category r_{ji} distinguishes a smooth connection from a sharp connection using the configured threshold $\theta_{ji} \leq 5^\circ$. The default representation uses $U = K = 10$, giving $d_f = 711$ face channels and $d_e = 63$ edge channels, with vocabularies $S = C_e = 32$ and $C_r = 4$. The default input preprocessing standardizes each continuous channel within a graph. For a feature channel q, this operation is

$$\widetilde{x}_{a,q} = \frac{x_{a,q} - \mu_q(\mathcal{G})}{\max(\sigma_q(\mathcal{G}), \varepsilon_{\text{norm}})}, \qquad \varepsilon_{\text{norm}} = 10^{-6}, \tag{20}$$

where a indexes either faces or edges. The trim indicator m_{iuv} remains binary, and sampled edge tangents are normalized to unit length after preprocessing. All diffusion targets below refer to these preprocessed continuous features.

Perturbing B-Rep connectivity can destroy the validity of the solid and changes the entity correspondence required by face-level learning. We therefore keep $\mathcal{V}$, $\mathcal{E}$, $\mathbf{s}$, $\mathbf{c}$, and $\mathbf{r}$ fixed, and apply Gaussian corruption only to $\widetilde{\mathbf{X}}^f$ and $\widetilde{\mathbf{X}}^e$. For each graph $\mathcal{G}_g$ in a mini-batch, we sample

$$t_g \sim \text{Uniform}\{0, \ldots, T-1\}. \tag{21}$$

The same t_g is assigned to every face and edge of that graph, while the Gaussian noise is sampled independently for the face and edge tensors. With

$$\beta_t = \beta_{\min} + \frac{t}{T-1}(\beta_{\max} - \beta_{\min}), \qquad \alpha_t = 1 - \beta_t, \qquad \bar{\alpha}_t = \prod_{\tau=0}^{t} \alpha_\tau, \tag{22}$$

the corrupted continuous attributes are

$$\mathbf{X}_t^f = \sqrt{\bar{\alpha}_t}\,\widetilde{\mathbf{X}}^f + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}^f, \qquad \boldsymbol{\epsilon}^f \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{23}$$

$$\mathbf{X}_t^e = \sqrt{\bar{\alpha}_t}\,\widetilde{\mathbf{X}}^e + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}^e, \qquad \boldsymbol{\epsilon}^e \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{24}$$

The implementation uses $T = 1000$, $\beta_{\min} = 10^{-4}$, and $\beta_{\max} = 2 \times 10^{-2}$.

A.2 Topology-aware B-Rep encoder with edge-update attention

The encoder maps the heterogeneous B-Rep attributes to a common latent space of dimension d. Let $\mathbf{u}_i^f$ and $\mathbf{u}_{ji}^e$ denote its continuous face and edge inputs. They are the noised features during pre-training and the clean preprocessed features during downstream fine-tuning. During pre-training, every graph is assigned one diffusion timestep t, and its sinusoidal embedding is transformed by an MLP,

$$\boldsymbol{\gamma}(t) = \mathrm{MLP}_t\left([\sin(t\omega_0), \ldots, \sin(t\omega_{d_t/2-1}), \cos(t\omega_0), \ldots, \cos(t\omega_{d_t/2-1})]\right), \tag{25}$$

where $\omega_k = 10000^{-k/(d_t/2-1)}$. The initial face and edge states are

$$\mathbf{h}_i^{(0)} = \mathrm{LN}\left(\mathbf{W}_f \mathbf{u}_i^f + \mathbf{E}_s(s_i) + \boldsymbol{\gamma}(t)\right), \tag{26}$$

$$\mathbf{e}_{ji}^{(0)} = \mathrm{LN}\left(\mathbf{W}_e \mathbf{u}_{ji}^e + \mathbf{E}_c(c_{ji}) + \mathbf{E}_r(r_{ji})\right). \tag{27}$$

The time term is omitted during downstream encoding. The categorical B-Rep attributes are not noised. They condition the encoder through learned embeddings and also serve as categorical and relation prediction targets during pre-training. Each encoder layer performs sparse multi-head attention over the incoming directed edges. For attention head $a \in \{1, \ldots, H\}$ and edge (j, i), let $\overline{\mathbf{h}}_i^{(l)} = \mathrm{LN}(\mathbf{h}_i^{(l)})$. The query, edge-conditioned key, and edge-conditioned value are

$$\mathbf{q}_i^{(l,a)} = \mathbf{W}_Q^{(l,a)} \overline{\mathbf{h}}_i^{(l)}, \tag{28}$$

$$\mathbf{k}_{ji}^{(l,a)} = \mathbf{W}_K^{(l,a)} \overline{\mathbf{h}}_j^{(l)} + \mathbf{W}_{K,e}^{(l,a)} \mathbf{e}_{ji}^{(l)}, \tag{29}$$

$$\mathbf{v}_{ji}^{(l,a)} = \mathbf{W}_V^{(l,a)} \overline{\mathbf{h}}_j^{(l)} + \mathbf{W}_{V,e}^{(l,a)} \mathbf{e}_{ji}^{(l)}. \tag{30}$$

The attention score contains an additional learned edge bias,

$$z_{ji}^{(l,a)} = \frac{(\mathbf{q}_i^{(l,a)})^\top \mathbf{k}_{ji}^{(l,a)}}{\sqrt{d/H}} + (\mathbf{w}_b^{(l,a)})^\top \mathbf{e}_{ji}^{(l)}, \tag{31}$$

$$\alpha_{ji}^{(l,a)} = \frac{\exp(z_{ji}^{(l,a)})}{\sum_{k:(k,i)\in\mathcal{E}} \exp(z_{ki}^{(l,a)})}. \tag{32}$$

Thus, the attention normalization is performed independently over all messages arriving at the same destination face. The node update combines all heads through residual attention and a pre- normalized SwiGLU feed-forward block,

$$\widetilde{\mathbf{h}}_i^{(l)} = \mathbf{h}_i^{(l)} + \mathrm{Drop}\left(\mathbf{W}_O^{(l)} \Big\Vert_{a=1}^{H} \sum_{j:(j,i)\in\mathcal{E}} \alpha_{ji}^{(l,a)} \mathbf{v}_{ji}^{(l,a)}\right), \tag{33}$$

$$\mathbf{h}_i^{(l+1)} = \widetilde{\mathbf{h}}_i^{(l)} + \mathrm{Drop}\left(\mathrm{SwiGLU}(\mathrm{LN}(\widetilde{\mathbf{h}}_i^{(l)}))\right). \tag{34}$$

After the face update, the edge representation is also updated rather than being kept as a static attention bias,

$$\mathbf{e}_{ji}^{(l+1)} = \mathrm{LN}\left(\mathbf{e}_{ji}^{(l)} + \mathrm{Drop}\left[\mathrm{MLP}_e^{(l)}\left([\mathbf{h}_j^{(l+1)} \| \mathbf{h}_i^{(l+1)} \| \mathbf{e}_{ji}^{(l)}]\right)\right]\right). \tag{35}$$

This update allows subsequent layers to refine the shared-curve representation using both incident faces. The default encoder uses $L = 4$ layers, $H = 4$ heads, hidden dimension $d = 128$, and dropout probability 0.1.

A.3 Joint Geometric and Attribute Prediction

The corrupted graph is processed by the shared encoder. From the final face state $\mathbf{h}_i^{(L)}$, three independent MLP heads predict the face noise, clean face feature, and surface- type logits,

$$(\widehat{\boldsymbol{\epsilon}}_i^f, \widehat{\mathbf{x}}_{0,i}^f, \widehat{\mathbf{o}}_i^s) = (D_\epsilon^f(\mathbf{h}_i^{(L)}), D_x^f(\mathbf{h}_i^{(L)}), D_s(\mathbf{h}_i^{(L)})). \tag{36}$$

For edge (j, i), the decoder first forms a contextual edge state

$$\mathbf{z}_{ji} = \mathrm{MLP}_{\mathrm{ctx}}([\mathbf{h}_j^{(L)} \| \mathbf{h}_i^{(L)} \| \mathbf{e}_{ji}^{(L)}]), \tag{37}$$

and uses four heads to predict edge noise, the clean edge feature, curve-type logits, and relation logits,

$$(\widehat{\boldsymbol{\epsilon}}_{ji}^e, \widehat{\mathbf{x}}_{0,ji}^e, \widehat{\mathbf{o}}_{ji}^c, \widehat{\mathbf{o}}_{ji}^r) = (D_\epsilon^e(\mathbf{z}_{ji}), D_x^e(\mathbf{z}_{ji}), D_c(\mathbf{z}_{ji}), D_r(\mathbf{z}_{ji})). \tag{38}$$

The implementation jointly predicts injected noise, clean continuous geometry, surface and curve categories, and adjacency relations through these heads. Let MSE and MAE denote elementwise means over the corresponding tensor. The continuous denoising and reconstruction losses are

$$\mathcal{L}_{\mathrm{noise}} = \mathrm{MSE}(\widehat{\boldsymbol{\epsilon}}^f, \boldsymbol{\epsilon}^f) + \mathrm{MSE}(\widehat{\boldsymbol{\epsilon}}^e, \boldsymbol{\epsilon}^e), \tag{39}$$

$$\mathcal{L}_{\mathrm{recon}} = \mathrm{MAE}(\widehat{\mathbf{X}}_0^f, \widetilde{\mathbf{X}}^f) + \mathrm{MAE}(\widehat{\mathbf{X}}_0^e, \widetilde{\mathbf{X}}^e). \tag{40}$$

Cross-entropy averages over the corresponding entities. The categorical objectives are

$$\mathcal{L}_{\mathrm{cat}} = \mathrm{CE}(\widehat{\mathbf{O}}^s, \mathbf{s}) + \mathrm{CE}(\widehat{\mathbf{O}}^c, \mathbf{c}), \tag{41}$$

$$\mathcal{L}_{\mathrm{rel}} = \mathrm{CE}(\widehat{\mathbf{O}}^r, \mathbf{r}). \tag{42}$$

For a graph without adjacency edges, all edge-specific terms are set to zero. The complete pre-training objective is

$$\mathcal{L}_{\mathrm{pre}} = \lambda_{\mathrm{noise}}\mathcal{L}_{\mathrm{noise}} + \lambda_{\mathrm{recon}}\mathcal{L}_{\mathrm{recon}} + \lambda_{\mathrm{cat}}\mathcal{L}_{\mathrm{cat}} + \lambda_{\mathrm{rel}}\mathcal{L}_{\mathrm{rel}}, \tag{43}$$

with $(\lambda_{\mathrm{noise}}, \lambda_{\mathrm{recon}}, \lambda_{\mathrm{cat}}, \lambda_{\mathrm{rel}}) = (1.0, 0.5, 0.5, 0.3)$ in the reported experiments. Noise prediction trains the encoder to identify the applied corruption, direct L1 reconstruction anchors the representation to clean geometric attributes, categorical prediction supervises surface and curve types, and relation prediction supervises smooth/sharp adjacency attributes. All four losses jointly train the shared encoder.

A.4 Downstream adaptation

After pre-training, only the parameters of the B-Rep graph encoder are loaded into a downstream model. Unless an explicit freezing ablation is performed, the encoder and the newly initialized task head are optimized jointly. For face segmentation with an MLP head, each final face embedding produces class logits $\mathbf{z}_i = \mathrm{MLP}_{\mathrm{seg}}(\mathbf{h}_i^{(L)})$. The discriminative segmentation loss combines cross-entropy with a soft Dice term,

$$\mathcal{L}_{\mathrm{seg}}^{\mathrm{MLP}} = \mathrm{CE}(\mathbf{Z}, \mathbf{y}) + \lambda_{\mathrm{Dice}}\mathcal{L}_{\mathrm{Dice}}, \tag{44}$$

$$\mathcal{L}_{\mathrm{Dice}} = 1 - \frac{1}{C_y}\sum_{c=1}^{C_y} \frac{2\sum_i p_{ic}y_{ic} + \delta}{\sum_i p_{ic} + \sum_i y_{ic} + \delta}, \tag{45}$$

where $p_{ic} = \mathrm{softmax}(\mathbf{z}_i)_c$, $\delta = 10^{-6}$, and $\lambda_{\mathrm{Dice}} = 0.3$. Whole-model classification first aggregates the face embeddings. The default Mean+Max pooling computes

$$\boldsymbol{\mu}_g = \frac{1}{N_g}\sum_{i\in\mathcal{V}_g} \mathbf{h}_i^{(L)}, \qquad \mathbf{m}_g = \max_{i\in\mathcal{V}_g} \mathbf{h}_i^{(L)}, \tag{46}$$

$$\mathbf{g}_g = \boldsymbol{\mu}_g + R([\boldsymbol{\mu}_g \| \mathbf{m}_g]), \tag{47}$$

where R is a residual MLP whose final projection is initialized to zero. This initialization starts optimization from exact mean pooling while allowing the model to learn a max-statistic correction. An MLP maps $\mathbf{g}_g$ to model-level logits and is trained with cross-entropy. At inference, the encoder processes clean B-Rep features and the task head produces class logits in one forward pass. The predicted face or model label is the index of the largest logit; softmax converts logits to class probabilities.

Table 5: Composition of the BRepPreDiff pre-training and downstream task corpus.

Source	pre-training/ train	valuation	test
BRepPreDiff pre-training	172,287	-	-
Fusion 360 Reconstruction	27,958	-	-
Fusion 360 Assembly	23,029	-	-
BLEND	14,295	1,787	1,787
Fusion 360 Segmentation	24,964	5,350	5,366
MFCAD++	41,766	8,950	8,949
TMCAD	8,709	1,090	1,087
SolidLetters	69,660	7,744	19,392
CADSynth	79,941	9,994	9,993
MFInstSeg	49,996	6,249	6,250
Total	512,605		

B Dataset Details

Table 5 summarizes the pre-training sources and downstream dataset splits.

BLEND. BLEND contains 17,869 STEP files paired by model identifier with 17,869 `.seg` files, each storing one integer label per face, for a total of 1,030,152 face annotations. The training, validation, and test sets contain 826,018, 101,154, and 102,980 faces, respectively. Across the benchmark, there are 868,291 non-blend faces, 125,728 EBFs, and 36,133 VBFs. EBFs occur in 9,401 models and VBFs in 3,826 models, with overlap between these groups. The 95th percentile of model size is 196 faces. We use a fixed, project-specific split of 14,295/1,787/1,787 training/validation/test models (approximately $80\%/10\%/10\%$), with pairwise disjoint model identifiers.

Fusion 360 Segmentation. Fusion 360 Segmentation (Lambourne et al., 2021) is a face-level semantic segmentation benchmark derived from parametric CAD modeling operations. Faces are assigned one of eight operation-related classes, including extrude, cut, fillet, chamfer, and revolve-related surfaces. We retain the official test set of 5,366 models from release s2.0.0/s2.0.1 and partition its official training pool of 30,314 models into 24,964 training and 5,350 validation models. Thus, validation uses approximately 17.65% of the official training pool, rather than the default 20% in UV-Net or 18% in BRepNet. The union of our training and validation sets exactly matches the official training set.

MFCAD++. MFCAD++ (Colligan et al., 2022) is a synthetic machining-feature dataset designed for recognizing manufacturing semantics directly from CAD geometry. It contains a 25-class face segmentation task. We follow the official training/validation/test split.

TMCAD. TMCAD (Zou & Zhu, 2025) is used for whole-model mechanical-part classification. The ten categories in our processing pipeline are bearing, bolt, bracket, coupling, flange, gear, nut, pulley, screw, and shaft. We construct a stratified $80\%/10\%/10\%$ training/validation/test split and remove 11 unparseable models, yielding 8,709/1,090/1,087 models.

SolidLetters. SolidLetters (Jayaraman et al., 2021) is a 26-class whole-model classification benchmark of solid letter shapes. We preserve the official training/test boundary and reserve approximately 10% of the cleaned official training pool for validation, yielding 69,660/7,744/19,392 training/validation/test models.

CADSynth. CADSynth (Zhang et al., 2024) provides synthetic CAD models for machining-feature recognition. We use its face-level semantic labels for a 25-class task comprising 24 machining features and stock/background. We follow the official training/validation/test split.

MFInstSeg. MFInstSeg (Wu et al., 2024) provides machining-feature annotations for B-Rep models. We evaluate its 25-class face-level semantic segmentation task using a project-specific

Table 6: Downstream datasets used by BRepPreDiff. Split sizes count CAD models. "Unit" denotes the unit on which accuracy is accumulated.

Dataset	Task	Classes	Train	Val.	Test	Unit
BLEND	face segmentation	3	14,295	1,787	1,787	face
Fusion360Seg	face segmentation	8	24,964	5,350	5,366	face
MFCAD++	face segmentation	25	41,766	8,950	8,949	face
CADSynth	face segmentation	25	79,941	9,994	9,993	face
MFInstSeg	face segmentation	25	49,996	6,249	6,250	face
TMCAD	shape classification	10	8,709	1,090	1,087	model
SolidLetters	shape classification	26	69,660	7,744	19,392	model

$80\%/10\%/10\%$ split of 49,996/6,249/6,250 training/validation/test models. The present experiments evaluate semantic labels; instance-segmentation performance is not reported.

Data augmentation. Both self-supervised pre-training and supervised fine-tuning use a mixture of canonical and randomly rotated inputs. Each training sample retains its canonical orientation with probability 0.5; otherwise, a random rotation $R \in \mathrm{SO}(3)$ is sampled from a normalized four-dimensional Gaussian quaternion. The same rotation is applied to all orientation-dependent geometric quantities, while rotation-invariant attributes, topology, and task labels remain unchanged. Rotation augmentation is applied before per-graph normalization and, during pre-training, before diffusion corruption, making the augmented clean geometry the denoising target. Validation and test samples remain in their canonical orientations without test-time augmentation.

C Evaluation Metrics

For C classes, let TP_c, FP_c, and FN_c denote the entries induced by the test-set confusion matrix for class c, and let N be the total number of evaluated faces or models. We report accuracy, macro-F1, weighted-F1, and mean intersection over union (mIoU):

$$\mathrm{Acc} = \frac{1}{N}\sum_{c=1}^{C}\mathrm{TP}_c, \tag{48}$$

$$\mathrm{F1}_c = \frac{2\mathrm{TP}_c}{2\mathrm{TP}_c + \mathrm{FP}_c + \mathrm{FN}_c}, \qquad \mathrm{MacroF1} = \frac{1}{C}\sum_{c=1}^{C}\mathrm{F1}_c, \tag{49}$$

$$\mathrm{IoU}_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FP}_c + \mathrm{FN}_c}, \qquad \mathrm{mIoU} = \frac{1}{C}\sum_{c=1}^{C}\mathrm{IoU}_c. \tag{50}$$

Weighted-F1 is $\sum_{c=1}^{C}(n_c/N)\mathrm{F1}_c$, where $n_c = \mathrm{TP}_c + \mathrm{FN}_c$ is the support of class c. For classification, mIoU is computed from the model-level confusion matrix; it does not measure spatial overlap. For segmentation, the confusion matrix is accumulated over all test faces before computing the metrics, rather than averaging a per-model score. Macro-F1 and mIoU therefore expose errors on rare classes that may be obscured by overall accuracy.

D FULL-EPOCH TRAINING COST AT A FIXED BATCH SIZE

Table 7 reports a complete training epoch on Fusion360Seg for each available implementation. All measurements use one NVIDIA A800-SXM4-80GB GPU, a batch size of 16 CAD models, one optimizer update per batch, FP32 with TF32 disabled, two CPU threads, and four DataLoader workers. Each run loads its trained downstream checkpoint, performs three warm-up batches, restores the model state, and rebuilds the optimizer before timing.

Table 7: **Full-epoch training cost on Fusion360Seg at batch size 16.** Parameters include the downstream task head. Peak memory is the maximum allocated CUDA memory during the measured epoch.

Method	Params. (M)	Train CADs	Time (s/epoch)	Time (s/1k CADs)	Peak mem. (MiB)
BRepNet	0.181	24,964	20.09	0.80	53.88
UV-Net	1.365	24,964	36.02	1.44	1,562.77
AAGNet	0.376	24,955	75.36	3.02	86.88
BRepMFR	5.390	24,964	261.15	10.46	17,192.43
BRT	2.292	23,321	214.66	9.20	10,390.11
SSL4CAD†	0.047	24,964	9.69	0.39	28.39
BRep-BERT‡	25.040	24,964	140.08	5.61	1,393.13
BRepMAE	4.304	24,918	135.13	5.42	1,020.64
BRepPreDiff	1.844	24,964	29.17	1.17	153.40

† SSL4CAD denotes the local simplified implementation.
‡ BRep-BERT denotes our reimplementation based on the original paper (Lou et al., 2023); official code is not publicly available.

E Qualitative Comparisons and Inference Latency

Selected examples. We visualize ten test models from each of Fusion360Seg, BLEND, and SolidLetters. Aggregate test performance is reported in Table 1. In all figures, SSL4CAD* denotes the local simplified implementation.

Figures 7–10 compare face segmentation results. Colors encode face labels, column headings give face counts, and percentages indicate per-model face accuracy.

Input model										
	(01)	(02)	(03)	(04)	(05)	(06)	(07)	(08)	(09)	(10)
Ground truth	c (2)	j (9)	d (3)	f (5)	j (9)	j (9)	j (9)	l (11)	u (20)	x (23)
BRepPreDiff (Ours)	**c (2)**	**j (9)**	**d (3)**	**f (5)**	**j (9)**	**j (9)**	**j (9)**	**l (11)**	**u (20)**	**x (23)**
UV-Net	i (8)	g (6)	d (3)	f (5)	j (9)	g (6)	d (3)	i (8)	v (21)	x (23)
BRT	i (8)	g (6)	g (6)	i (8)	g (6)	j (9)	i (8)	i (8)	v (21)	v (21)
SSL4CAD*	i (8)	g (6)	p (15)	i (8)	g (6)	g (6)	j (9)	i (8)	v (21)	r (17)
BRep-BERT	t (19)	z (25)	s (18)	i (8)	f (5)	i (8)	z (25)	l (11)	v (21)	z (25)
BRepMAE	l (11)	g (6)	s (18)	i (8)	q (16)	g (6)	b (1)	i (8)	u (20)	z (25)

Figure 5: **Classification on SolidLetters test models.** Entries show letters and class IDs ($a = 0, \ldots, z = 25$); red entries indicate incorrect predictions. Uppercase and lowercase letters share the same class.

Inference timing. Figure 6 reports measurements on an NVIDIA A800 80 GB GPU in FP32 with batch size one. We use ten models per dataset, with ten warm-up runs and 30 timed repetitions per model, and measure synchronized wall-clock time. Methods are measured sequentially on the same GPU. Offline STEP feature extraction is excluded.

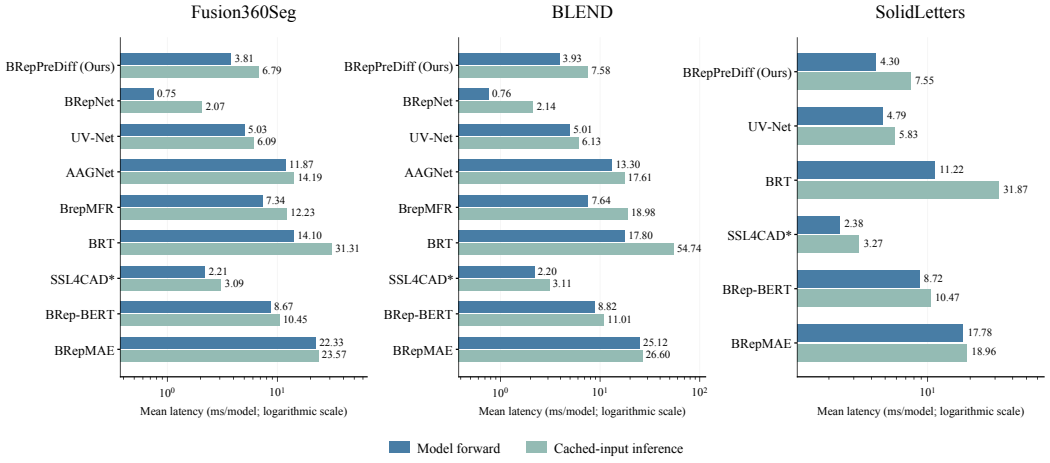

Figure 6: **Mean inference latency on the selected CAD models.** Bars distinguish model-forward time from inference with cached inputs. Latency is measured in milliseconds per model on logarithmic axes; lower is better.

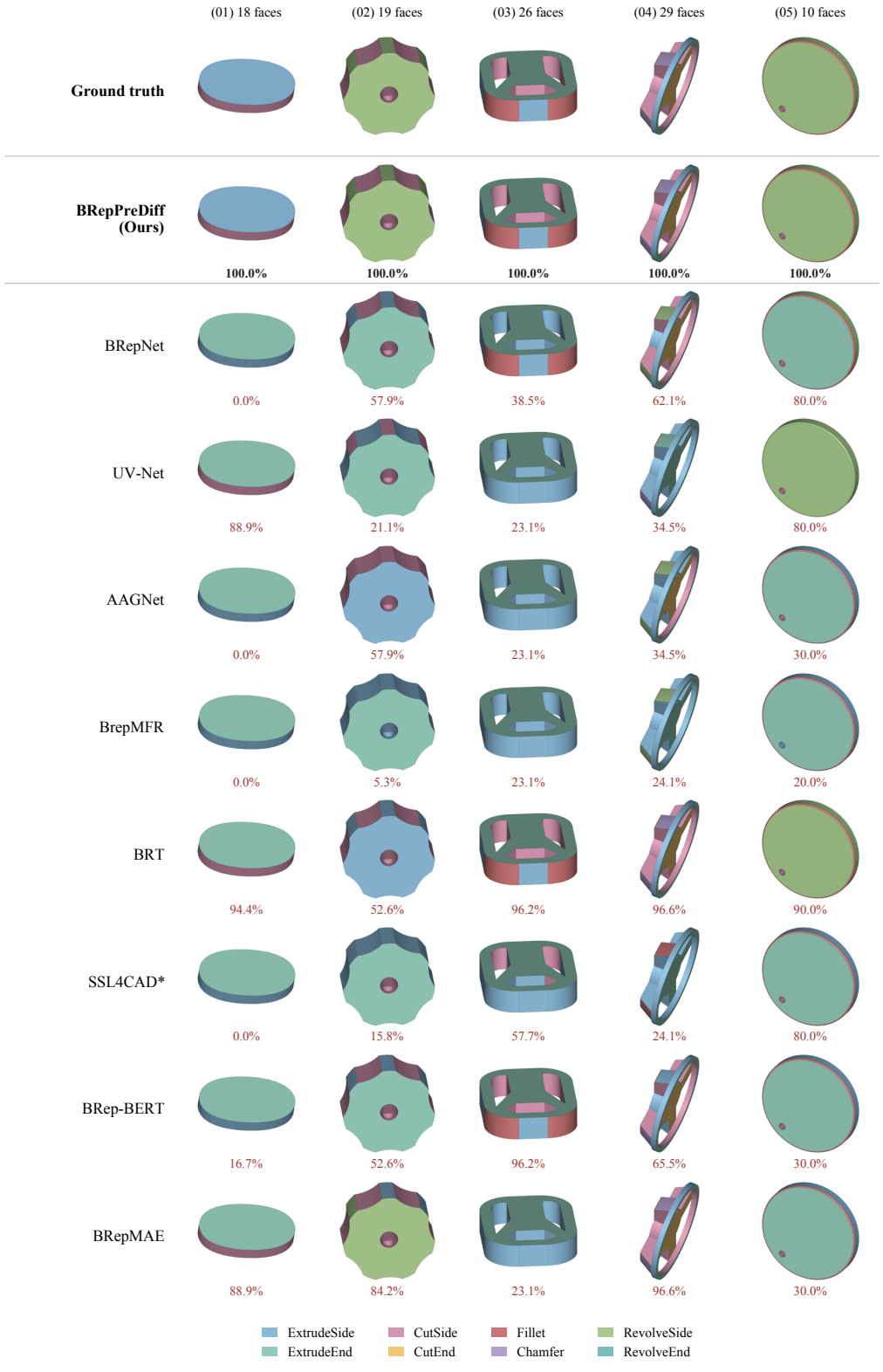

Figure 7: **Face segmentation on Fusion360Seg, examples 01–05.** Ground-truth labels and predictions are shown with a shared class-color legend. Percentages indicate face accuracy for each model.

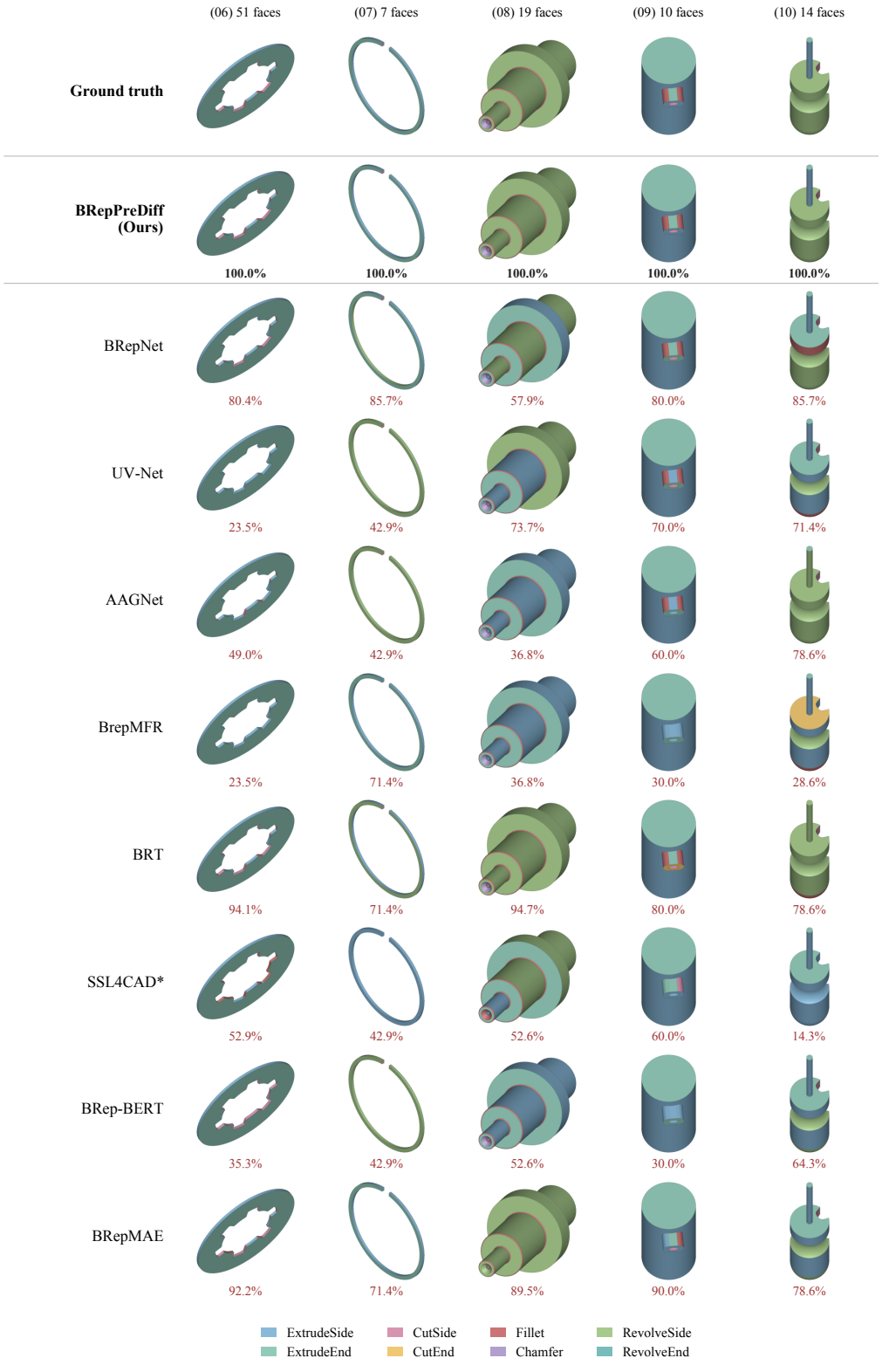

Figure 8: **Face segmentation on Fusion360Seg, examples 06–10.** The layout and color convention follow Figure 7. Percentages indicate face accuracy for each model.

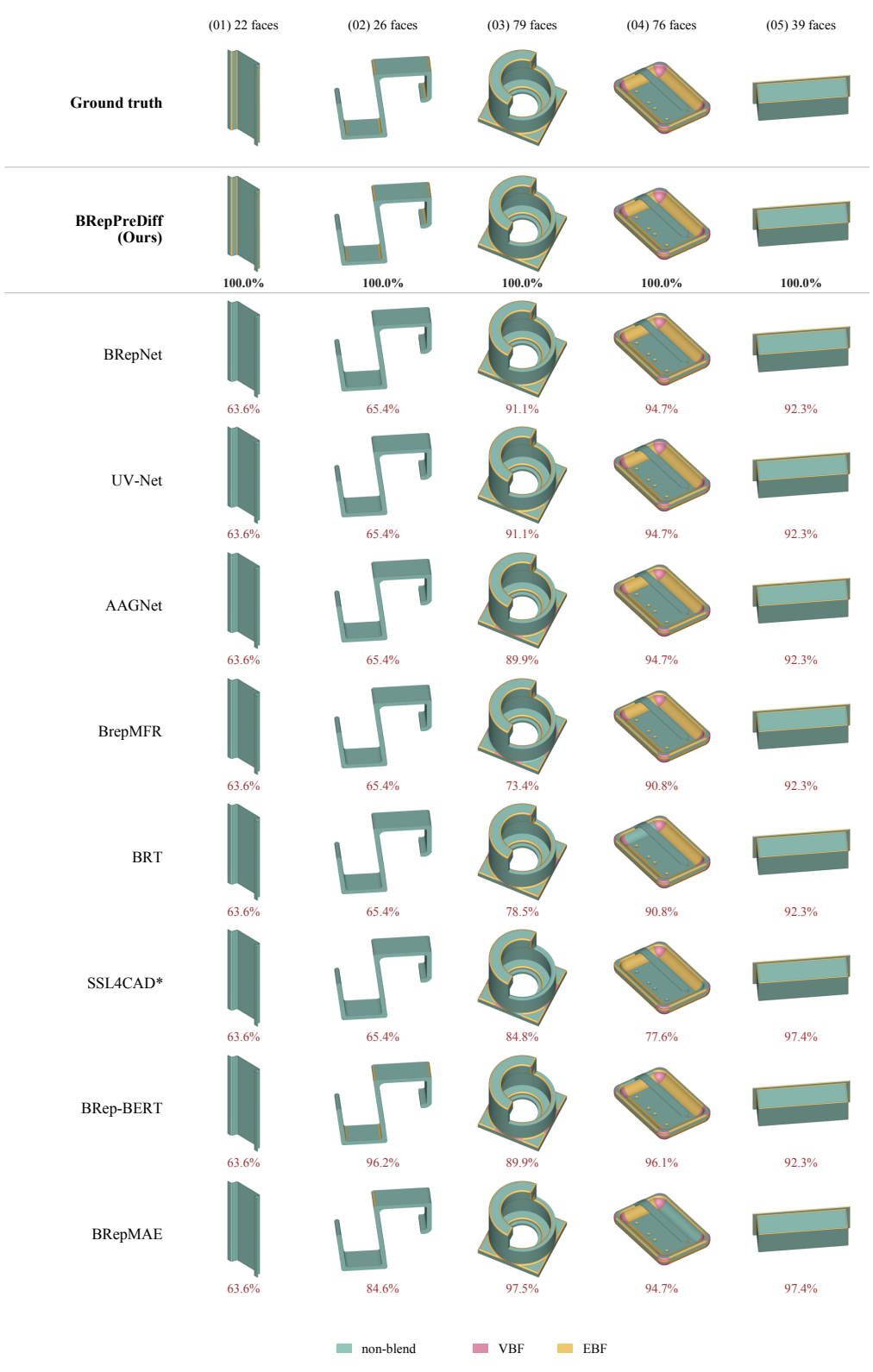

Figure 9: **Face segmentation on BLEND, examples 01–05.** Colors distinguish non-blend faces, VBFs, and EBFs. Percentages indicate face accuracy for each model.

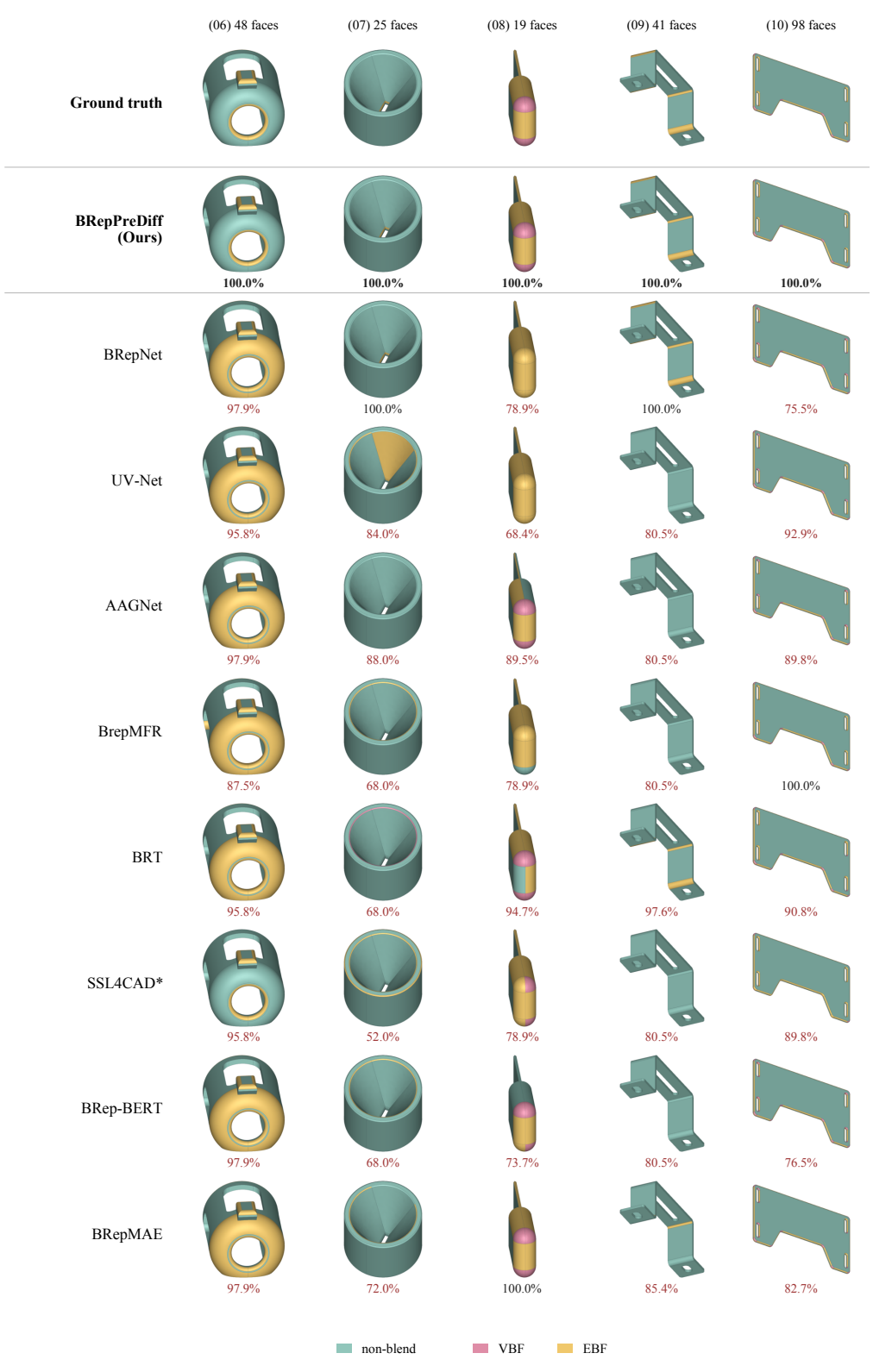

Figure 10: **Face segmentation on BLEND, examples 06–10.** The layout and color convention follow Figure 9. Percentages indicate face accuracy for each model.