Masked BRep Autoencoder via Hierarchical Graph Transformer

Yifei Li[1], Kang Wu[1], Wenming Wu[2], and Xiao-Ming Fu[1⋆]

[1] University of Science and Technology of China, China
[2] Hefei University of Technology, China

Abstract. We introduce a novel self-supervised learning framework that automatically learns representations from input computer-aided design (CAD) models for downstream tasks, including part classification, modeling segmentation, and machining feature recognition. To train our network, we construct a large-scale, unlabeled dataset of boundary representation (BRep) models. The success of our algorithm relies on two key components. The first is a masked graph autoencoder that reconstructs randomly masked geometries and attributes of BReps for representation learning to enhance the generalization. The second is a hierarchical graph Transformer architecture that elegantly fuses global and local learning by a cross-scale mutual attention block to model long-range geometric dependencies and a graph neural network block to aggregate local topological information. After training the autoencoder, we replace its decoder with a task-specific network trained on a small amount of labeled data for downstream tasks. We conduct experiments on various tasks and achieve high performance, even with a small amount of labeled data, demonstrating the practicality and generalizability of our model. Compared to other methods, our model performs significantly better on downstream tasks with the same amount of training data, particularly when the training data is very limited.

Keywords: CAD · Masked autoencoder · Hierarchical graph transformer

1 Introduction

As a core component of modern digital industrial manufacturing, computer-aided design (CAD) enhances process efficiency by using computers to create, modify, analyze, and optimize product models, thereby laying a crucial foundation across industries. In addition to creating models, CAD involves a range of critical tasks, such as machining feature recognition, face segmentation, and shape classification [4, 7, 10, 16, 20, 23, 29, 33, 39].

Machine learning for processing CAD models has emerged as a leading research focus, driven by their superior generalizability and efficiency compared to traditional handcrafted methods [8, 12]. In particular, since boundary representations (BReps) are the dominant representation for CAD models due to their precise geometric description, we focus on learning algorithms for BReps.

⋆ Corresponding author: Xiao-Ming Fu, fuxm@ustc.edu.cn

However, the practical value of learning-based methods is limited by the inherent characteristics of CAD data. A key characteristic is the scarcity of models, as they often contain sensitive commercial intellectual property that enterprises closely guard. Hence, constructing large-scale, high-quality labeled datasets is challenging, consequently limiting the performance of learning algorithms.

To address this limitation, a viable way is to perform representation learning on large-scale unlabeled datasets, enabling artificial intelligence (AI) models to automatically extract meaningful features for downstream tasks, thereby improving their performance with limited labeled data. However, it faces two main challenges. First, the combination of discrete topology with continuous geometry in BReps complicates the effective handling. Second, the inconsistent and hierarchical nature of BRep graph structures hinders understanding and analysis.

Many learning methods have been proposed for processing CAD models [3,8, 10,16,23,35,38,39]. Some require a large amount of annotated data [3,23,35,38, 39] and are limited to specific downstream tasks, such as feature recognition and surface segmentation. Other works [8,10,16,28] train on unlabeled data, but their graph neural network (GNN) encoders only aggregate local information, ignoring global dependencies. Recently, Transformer-based architectures have emerged to address this [35, 39]. Especially, BRepFormer [4] successfully applies generic graph Transformers to capture long-range global dependencies. However, their flat, single-level attention block is ineffective for extreme scale variations (e.g., large planes vs. small fillets) in CAD models and is easily overwhelmed by the severe data redundancy (e.g., planar regions require fewer samples) introduced by dense geometric sampling, leading to inefficient representation learning.

In this paper, we introduce a self-supervised learning framework for acquiring effective representations of BReps from a large-scale, unlabeled dataset. There are two essential technical components. First, the framework inherits from the elegant masked autoencoder [6]. By representing the BRep as a geometric attributed adjacency graph (gAAG) [23], it learns to reconstruct randomly masked face/edge geometries and their attributes from input CAD models, while keeping the graph topology invariant to enable effective representation learning. Second, we propose a tailored hierarchical graph Transformer, serving as its core encoder, with both global and local learning to tackle the heterogeneity and extreme scale variations of practical BRep data. Its architecture centers on a progressive, coarse-to-fine cross-attention model that processes CAD face geometries via dual-resolution UV grids to capture long-range geometric dependencies, followed by a message-passing neural network (MPNN) to aggregate local topological information. In summary, the progressive abstraction is coupled with the self-supervised masked reconstruction task that guides the encoder to distill essential CAD topologies and geometries rather than merely memorizing raw sampling coordinates.

Extensive experiments on a dataset of 283,018 models demonstrate that our pre-trained hierarchical encoder achieves superior generalization. It significantly outperforms existing representative methods across multiple downstream tasks, especially under highly limited supervision (e.g., 0.1% labeled data).

2 Related Work

Learning for BRep The diverse geometries and complex topologies of BReps pose significant challenges for neural network learning. Methods [5, 13, 29, 31, 33, 37] circumvent these challenges by converting BRep models into alternative 3D representations, like point clouds, voxels, or meshes, but fail to exploit the topological information inherent in BReps. A more faithful way treats BRep as a structured graph [1, 9, 12, 22], where each BRep face is represented as a node and each BRep edge as a graph edge. UVNet [8] samples geometric signals from both surfaces and curves to convert BRep models into graphs and applies GNN to aggregate information across the BRep topology. Owing to its effectiveness in handling BRep geometry and topology, UVNet has inspired a series of follow-up works [14, 15, 18, 20, 21, 23, 32, 35, 38]. Among them, Wu et al. [23] incorporate attributes to achieve improved performance. Zou et al. [39] leverage Transformer-based attention to capture long-range dependencies across the topology, but they do not use the graph-structured messaging mechanism to aggregate neighborhood information. Moreover, BRepMFR [35] and BRepFormer [4] introduce graph Transformers for global interactions. However, applying single-level attention uniformly remains inherently challenging for CAD models' extreme scale variations, as it struggles to decouple macroscopic topologies from dense microscopic details. To address this, we propose a hierarchical architecture featuring a cross-scale mutual attention block that establishes a dual-resolution information bottleneck, explicitly decoupling macro-anchors from micro-features. Finally, an MPNN aggregates local graph topology, seamlessly synergizing global hierarchical abstraction with local messaging for robust representation learning.

Self-supervised learning for CAD models Self-supervised learning is especially valuable for CAD models, where publicly available, well-annotated datasets are scarce. Jones et al. [10] propose a reconstruction-based self-supervised learning method that, while effective in few-shot settings, is limited to CAD models composed of basic geometric primitives. BRep-BERT [16] converts BRep entities into discrete tokens via a GNN tokenizer and trains a Transformer with a masked entity modeling task. However, this discretization step introduces a significant bottleneck by compressing geometries and topologies into a limited number of token IDs. BRepGen [26] uses a hierarchical tree representation to encode BRep models into a latent space, which cannot be effectively used for downstream tasks. Zhang et al. [31] operate on point clouds, which inherently lack topological and detailed geometric information. Our method builds upon the masked autoencoder (MAE) [6] framework. Closely related to our work, Yao et al. [28] successfully apply MAE to BRep models by masking latent features and employing an MPNN. In contrast, we enforce a stricter information bottleneck by directly masking raw geometric inputs and use a hierarchical GT to capture long-range global dependencies prior to local topology aggregation.

Graph Transformer Unlike traditional GNNs, which rely solely on local message passing, GTs also use attention mechanisms to capture long-range depen-

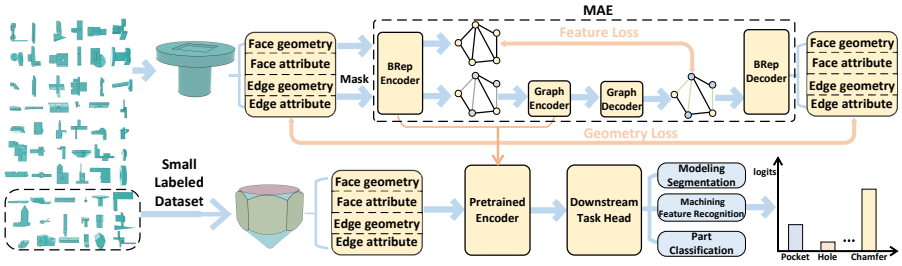


Fig. 1: Given a CAD model, our method first applies a BRep encoder to construct a gAAG from the extracted BRep information. A Graph encoder is then implemented to update the graph features with both global and local information. During pre-training, we train an MAE by reconstructing the randomly masked faces and edges for BRep representation learning (the upper part of the figure). In the fine-tuning stage, we train a new network, formed by connecting a task-specific head behind the encoder, using a small amount of labeled data with different loss functions (lower part of the figure).

dencies, thereby becoming powerful architectures for graph-structured data to achieve more expressive and flexible representations. Most prior work is centered on molecular property prediction, where input graphs represent chemical compounds, such as Graph-BERT [34], GROVER [19], GraphiT [17], Graph-Trans [25], SAN [11], Graphormer [30], and Sgformer [24]. While GTs have shown immense potential, their direct application to BRep data requires careful architectural design. Although the pioneers [4, 35] have introduced GTs to BReps, they predominantly inherit flat attention structures from general-purpose GTs, which limits their effectiveness on complex CAD topologies. To bridge this gap, we propose a hierarchical GT encoder tailored to BReps. By seamlessly integrating a dual-resolution information bottleneck with local message passing, our architecture overcomes the limitations of flat attention, enabling highly expressive and transferable representation learning across diverse downstream tasks.

3 Method

Our proposed self-supervised BRep representation learning method consists of two parts: BRep information extraction (Sec. 3.1) and pre-training (Sec. 3.2). For pre-training, we employ the MAE framework to reconstruct the masked geometries in the input models and propose a tailored hierarchical graph Transformer encoder with a cross-scale mutual attention block. After pre-training, we concatenate a specific-task network behind our encoder to form a new network, and then train it with a small amount of labeled data for various downstream tasks (Sec. 3.3). The pipeline of our method is shown in Fig. 1.

3.1 BRep information extraction

To address extreme scale variations (e.g., large planes vs. small fillets) and reduce surface data redundancy (e.g., planar regions that require fewer sample points)

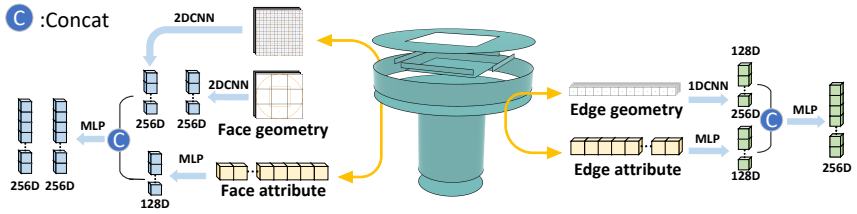


Fig. 2: Our BRep encoder extracts and fuses geometric and attribute features into a unified gAAG representation via parallel CNN and MLP branches.

in CAD models, we propose a dual-resolution geometry extraction strategy that explicitly separates global shape from local detail.

Extracting face information For each face, to capture both its local and global geometric information, we perform UV parameterization [8] and uniformly sample two different resolution grids (3×3 for a global semantic guidance and 13×13 for fine-grained local details). We extract the coordinates (3D), normal (3D), and trimming indicator (1D) of each grid point to form a 7D feature as the face geometric information. The face attribute for each grid point is a 16D vector that contains face type (6D), area (1D), centroid (3D), and bounding box (6D).

Extracting edge information Similar to the node feature construction, we uniformly sample 13 points on each BRep edge to extract the information of the edges. The edge geometric information is a 12D vector, containing coordinate (3D), tangent (3D), and the normals of the two incident faces (6D), while the attribute is a 9D vector with edge type (5D), length (1D), and convexity (3D).

3.2 Pre-training

Encoder Our encoder has two parts: (1) BRep encoder and (2) Graph encoder.

BRep encoder Raw BRep models consist of highly heterogeneous data, intricately intertwining continuous geometric with discrete topological information. To systematically process such heterogeneous data while strictly preserving its structural integrity, we project the BRep into a unified latent graph space via a gAAG [23]. In this formulation, the gAAG nodes represent the extracted face features, while the graph edges represent the BRep topological edge features.

Specifically, our BRep encoder employs five parallel branches to independently embed the geometries and attributes: two 2D CNNs encode the face geometries, and an MLP encodes the attributes (Fig. 2). The outputs of the two face geometry embeddings are both 256D vectors, while the attribute embedding is a 128D vector. We then concatenate each dual-resolution face geometric vector with its corresponding attribute vector, feeding them into another MLP to produce two decoupled 256D embeddings, denoted as f^i_{low} and f^i_{high}, which serve as the node features for the gAAG.

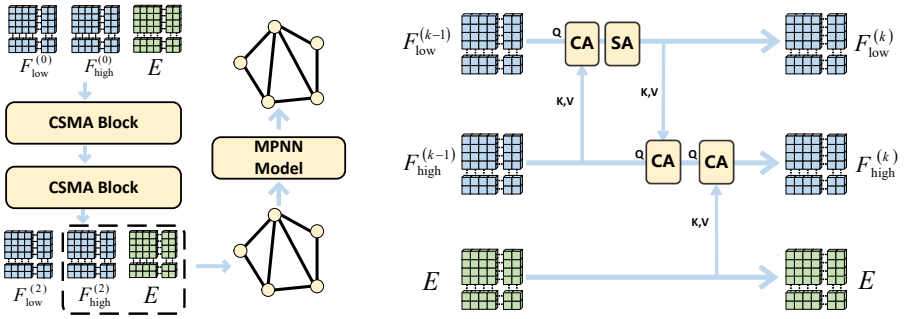


Fig. 3: Left: The overall architecture of the Graph encoder, processing multi-resolution face features F_{low}, F_{high}, and edge features E. Right: The internal structure of the k-th CSMA block. CA and SA denote cross-attention and self-attention layers, respectively, while Q, K, and V represent queries, keys, and values.

Similarly, the geometries and attributes of BRep edges are extracted using a 1D CNN and an MLP. They are transferred to 256D and 128D vectors and then concatenated to generate a unified 256D embedding e^i as the graph edge feature. Details are provided in the supplementary.

Graph encoder Standard GNNs suffer from over-smoothing [36], failing to capture long-range geometric dependencies. Simultaneously, directly applying standard Transformers to dense BRep samplings is inefficient, as the attention mechanism can be easily overwhelmed by redundant local details.

To address this, our hierarchical graph Transformer encoder combines a cross-scale mutual attention (CSMA) module to capture global geometric structures and an MPNN [23] to enforce local topological connections (Fig. 3 - Left).

– **Cross-Scale Mutual Attention.** To efficiently process features, we group node features by grid resolution into low-resolution features F_{low} and high-resolution features F_{high} [2], along with edge features E. To make the network focus on key geometric changes rather than being distracted by large, flat surfaces, our CSMA block uses sparse, low-resolution features to query and extract essential details from dense, high-resolution features (Fig. 3 - Right):

$$F_{\text{low}}^{(k)} = \text{SA}\Big(\text{CA}\big(F_{\text{low}}^{(k-1)},\, F_{\text{high}}^{(k-1)}\big)\Big), \tag{1}$$

$$F_{\text{high}}^{(k)} = \text{CA}\Big(\text{CA}\big(F_{\text{high}}^{(k-1)},\, F_{\text{low}}^{(k)}\big),\, E\Big), \tag{2}$$

where CA is multi-head cross-attention (queries from the first argument, keys/values from the second), and SA is self-attention. With $F_{\text{low}}^{(0)} = F_{\text{low}}$ and $F_{\text{high}}^{(0)} = F_{\text{high}}$, the updated $F_{\text{low}}^{(k)}$ aggregates a global context to guide the refinement of $F_{\text{high}}^{(k)}$ alongside explicit topological edge features E. We stack two such CSMA blocks to ensure the overall structure and local details are fully integrated. Crucially, this asymmetric bottleneck prevents the network

from being overwhelmed by redundant flat surfaces and avoids the over-smoothing typical of GNNs. It ensures that crucial local details are accurately retained and integrated into the global context.

– **Local Topological Message Passing.** Although the CSMA blocks capture the overall structure, BRep faces are geometrically connected by local edges. To explicitly model these precise neighborhood relationships, we pass the updated face features $F_{\text{high}}^{(2)}$ and edge features E into an MPNN:

$$\left(F_{\text{latent}},\, E_{\text{latent}}\right) = \text{MPNN}\Big(F_{\text{high}}^{(2)},\, E;\, \mathcal{G}_{\text{adj}}\Big)\,, \tag{3}$$

where $\mathcal{G}_{\text{adj}}$ is the explicit adjacency graph. By placing this local MPNN after the global Transformer, our encoder effectively combines the overall 3D geometric structure with precise face-to-face connections. Detailed configurations are in the supplementary.

Two-Stage Decoder Reconstructing heavily masked BRep models in a single step is highly challenging. Instead of directly predicting raw 3D geometries and attributes from the compressed latent space, we design a two-stage decoding strategy: a graph decoder followed by a BRep decoder.

1. The graph decoder first reconstructs the intermediate node and edge features within the graph space.
2. Subsequently, the BRep Decoder maps these recovered features back to explicit geometries and attributes.

This two-stage breakdown not only eases the reconstruction but also allows the intermediate graph features to serve as additional supervision signals, thereby stabilizing the pre-training process.

Graph decoder To first recover the intermediate node and edge features from the masked context, the graph decoder employs an MPNN with an architecture symmetric to the graph encoder:

$$\left(F_{\text{recon}},\, E_{\text{recon}}\right) = \text{MPNN}(F_{\text{latent}},\, E_{\text{latent}};\, \mathcal{G}_{\text{adj}})\,, \tag{4}$$

where F_{recon} and E_{recon} represent the reconstructed graph node and edge features, respectively. During training, these features are directly supervised by the features extracted from the unmasked BRep using the BRep encoder.

BRep decoder Subsequently, the BRep decoder utilizes four parallel branches to map these recovered graph features back to the explicit geometric and attribute details. For the continuous geometric structures, we employ FoldingNet [27] branches, which are naturally suited for deforming latent representations into 3D surfaces (7D face geometry) and curves (12D edge geometry). Simultaneously, two standard MLPs are used to regress the 16D face discrete attributes and 9D edge discrete attributes. Detailed layer-wise configurations are provided in the supplementary material.

Masked Pre-training To learn robust BRep representations, we formulate a self-supervised masked autoencoding task. Crucially, instead of masking latent features [28], we randomly mask the raw geometries and attributes of input faces and edges at a high ratio (e.g., 70%). This strict input-level corruption forces the network to deduce missing structures from the surviving global context, preventing trivial local shortcuts.

Specifically, we reconstruct the edge features and only the high-resolution face features, along with their original geometric information. Since higher-resolution grids encapsulate richer geometric information, reconstructing them enables the model to accurately capture small geometric structures.

Our objective includes five loss terms to provide comprehensive supervision across both latent features and explicit geometries. It includes a latent feature alignment loss $\mathcal{L}_{\text{feat}}$ to penalize the divergence between reconstructed graph features and the target features extracted from the original unmasked BRep. Crucially, a stop-gradient operation is applied to this target branch to prevent representation collapse and ensure the network learns meaningful semantics, alongside four explicit reconstruction losses ($\mathcal{L}_{\text{geom}}^{\text{face}}$, $\mathcal{L}_{\text{attr}}^{\text{face}}$, $\mathcal{L}_{\text{geom}}^{\text{edge}}$ and $\mathcal{L}_{\text{attr}}^{\text{edge}}$) for face/edge geometries and attributes. Detailed formulations are provided in the supplementary material. We optimize the loss functions over all faces and edges to ensure our network accurately recovers the masked information.

3.3 Fine-tuning

After pre-training, we attach a task-specific head behind the pre-trained encoder for downstream tasks. The entire network is then fine-tuned by different loss functions with a small amount of labeled data from the target dataset. Details for each downstream task are provided in the supplementary materials.

4 Experiments

All experiments are implemented in PyTorch and executed on a server equipped with a single NVIDIA A800 GPU. We first perform self-supervised pre-training, followed by task-specific evaluation across three downstream benchmarks: (1) machining feature recognition, (2) modeling segmentation, and (3) part classification. Unless otherwise specified, all results are obtained under identical hardware configurations, and the best results are in bold for statistics comparisons.

4.1 Datasets and Data Isolation Protocol

To rigorously evaluate our method and prevent potential data leakage (i.e., data contamination), we enforce a strict data-isolation protocol across all experiments. Specifically, the CAD models are randomly split into training, validation, and testing subsets according to the task: a 70%/15%/15% split is used for pre-training, model segmentation, and classification, while an 80%/10%/10% split is adopted for machining feature recognition. Crucially, the testing sets are strictly

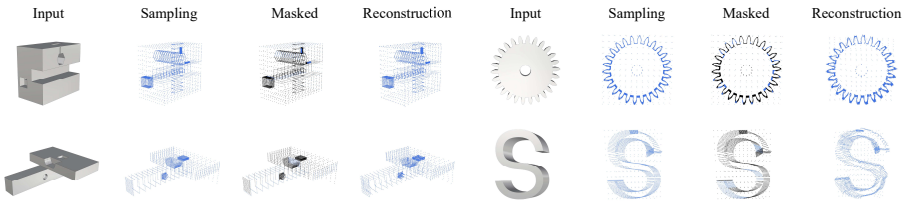


Fig. 4: Four reconstruction examples. For each example, the sequence from left to right shows: (1) the input BRep model, (2) the full surface point cloud, (3) the masked input where black denotes masked faces and blue denotes unmasked faces, and (4) the reconstructed point cloud.

held out and remain completely unseen during both the self-supervised pre-training and downstream fine-tuning phases, ensuring a fair evaluation.

Utilized benchmarks Our experiments utilize the following benchmarks:

- MFInstSeg [23] (62,495 models): per-face machining-feature labels; 25 classes.
- MFCAD++ [3] (59,665 models): per-face machining-feature labels; 25 classes.
- CADSynth [35] (100,000 models): per-face machining-feature labels; 25 classes.
- Fusion 360 Gallery Segmentation [12] (35,858 models): per-face modeling-operation labels; 8 classes.
- SolidLetter [8] (96,861 models): alphabetic CAD models with shape-level labels; 26 classes.

Pre-training corpus. Following our strict isolation protocol, the pre-training dataset is constructed by merging only the training splits (70%) of MFInstSeg, MFCAD++, CADSynth, Fusion 360 Gallery, and a 25,000-sample subset of SolidLetter. While the total collection of all datasets amounts to 283,018 models, this sanitized pre-training corpus comprises exactly 198,113 unique, unlabeled CAD models, ensuring absolute isolation from any downstream testing data.

Downstream adaptation. For downstream tasks, we utilize the training and validation splits of the respective target datasets for fine-tuning, and report all final metrics exclusively on the strictly isolated testing splits.

4.2 Pre-training

In the pre-training stage, we use the AdamW optimizer with a batch size of 128 and an initial learning rate of 10^{-4}. The learning rate follows a cosine decay schedule. Pre-training is conducted for 100 epochs from scratch, with 70% of faces and edges randomly masked in each sample. We show four examples of face-coordinate reconstruction in Fig. 4. The results show that our algorithm can correctly reconstruct the geometry of the masked BRep faces. Therefore, our algorithm does learn an effective BRep representation. See supplementary for more reconstruction results.

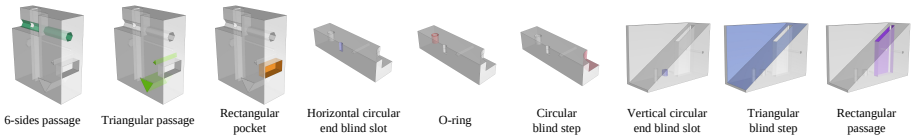


Fig. 5: Machining feature recognition under the full supervised setting. We accurately predict various machining features, such as passages, pockets, slots, and steps.

4.3 Comparisons

We use face-level accuracy (Acc) and mean intersection over union (mIoU) to evaluate performance on machining feature recognition and modeling segmentation. For part classification, we report shape-level accuracy only. The formulas of Acc and mIoU are shown in the supplementary.

Machining feature recognition We evaluate our method on per-face machining-feature recognition (25 classes) using MFInstSeg, MFCAD++, and CADSynth. We compare against fully supervised baselines, AAGNet [23] and the recent state-of-the-art BRepFormer [4], adopting their official implementations, alongside a recent self-supervised pre-training method, BRepMAE [28]. All datasets follow our strict 80/10/10 isolation protocol.

To assess few-shot performance, we fine-tune the network under varying supervision ratios (0.1%–100%). We use the AdamW optimizer for 200 epochs, setting learning rates to 10^{-4} for the task-specific head and 10^{-5} for the pre-trained encoder.

As shown in Table 1, while the fully supervised BRepFormer demonstrates strong capacity under full supervision (100%), training Transformers from scratch with extremely limited data (e.g., 0.1%) is inherently challenging, leading to a drastic performance drop (e.g., falling to 33.43% Acc and 7.47% mIoU on MFInstSeg). In contrast, pre-training methods effectively alleviate this issue. Specifically, our method maintains a robust 88.75% Acc and 66.38% mIoU on MFInstSeg at the 0.1% setting. Under the same pre-training data scale, our method demonstrates superior data efficiency, consistently outperforming BRepMAE across most low-data regimes (e.g., achieving 96.19% vs. 94.86% Acc on MFInstSeg at 0.5%, and 89.19% vs. 86.38% Acc on MFCAD++ at 0.1%). This confirms that our hierarchical MAE pre-training extracts more transferable and robust representations. Furthermore, under full supervision, our method remains highly competitive with the fully supervised upper bound (e.g., 99.53% vs. BRepFormer's 99.62% Acc on MFInstSeg), proving that we achieve strong few-shot generalization without sacrificing the model's overall representational capacity. To further illustrate the effectiveness of our method, Figure 5 shows representative results from the machining feature recognition task.

Modeling segmentation We evaluate our pre-trained encoder on the Fusion 360 Gallery Segmentation dataset (8 classes). We compare against BRepBERT [16], BRT [39], and BRepFormer [4] under 10-shot, 20-shot, and full-data

Table 1: Comparison across MFInstSeg, MFCAD++, and CADSynth under varying supervision ratios. Accuracy (Acc, %) and mIoU (%) are reported.

		Method	0.1%	0.5%	1%	1.5%	2%	3%	100%
MFInstSeg	Acc	AAGNet	46.49	76.62	85.19	90.30	92.31	92.37	99.13
		BRepFormer	33.43	86.14	91.23	95.77	96.51	97.64	**99.62**
		BRepMAE	87.93	94.86	96.91	97.32	97.61	98.23	99.57
		Ours	**88.75**	**96.19**	**97.31**	**97.67**	**97.92**	**98.26**	99.53
	mIoU	AAGNet	34.01	67.51	77.50	84.64	86.76	87.32	98.36
		BRepFormer	7.47	62.66	73.89	86.17	88.25	91.85	**98.75**
		BRepMAE	**66.53**	83.38	90.39	91.33	91.93	93.59	98.60
		Ours	66.38	**87.21**	**90.81**	**91.74**	**92.60**	**93.86**	98.47
MFCAD++	Acc	AAGNet	51.22	81.47	87.38	90.64	91.64	94.42	99.27
		BRepFormer	32.95	83.92	92.49	95.82	96.93	97.70	**99.71**
		BRepMAE	86.38	94.21	96.51	97.03	97.45	97.86	99.60
		Ours	**89.19**	**95.94**	**97.34**	**97.60**	**97.97**	**98.36**	99.61
	mIoU	AAGNet	38.23	71.88	80.66	84.62	86.28	90.91	98.65
		BRepFormer	6.36	58.02	77.80	86.80	90.00	92.56	**99.27**
		BRepMAE	65.75	83.35	89.78	91.41	91.87	93.63	98.82
		Ours	**67.29**	**86.26**	**90.97**	**92.12**	**93.25**	**94.57**	99.03
CADSynth	Acc	AAGNet	58.53	87.36	95.58	96.08	97.01	97.14	99.66
		BRepFormer	59.34	65.60	97.49	98.33	98.65	98.73	**99.88**
		BRepMAE	93.55	97.22	98.27	98.75	99.12	99.18	99.77
		Ours	**93.82**	**98.54**	**98.91**	**99.25**	**99.29**	**99.42**	99.86
	mIoU	AAGNet	47.17	81.04	92.40	93.42	94.92	94.61	99.37
		BRepFormer	10.97	82.61	89.84	93.04	94.38	94.49	**99.51**
		BRepMAE	78.40	88.91	92.91	94.70	96.26	96.50	99.06
		Ours	**79.29**	**93.72**	**95.53**	**96.85**	**97.03**	**97.55**	99.43

settings, adopting their official implementations or reported results. The dataset is split into 70% training, 15% validation, and 15% testing.

As shown in Table 2, our method achieves the highest performance across all settings. Under limited supervision (10-shot and 20-shot), fully supervised approaches like BRT and BRepFormer naturally experience performance drops since they are trained from scratch. Benefiting from the pre-trained representations, our model yields 72.33% Acc and 34.63% mIoU in the 10-shot scenario, demonstrating strong few-shot generalization and substantially exceeding the self-supervised baseline BRep-BERT (59.26% Acc). Furthermore, while the fully supervised BRepFormer excels in machining feature recognition, its flat, single-level attention mechanism faces inherent challenges when handling the extreme scale variations of CAD surfaces. In contrast, our hierarchical architecture with the CSMA bottleneck and local message passing seamlessly integrates global geometric contexts with crucial local details. Consequently, our method not only excels in few-shot scenarios but also maintains the highest accuracy (97.02%) and mIoU (86.75%) under full supervision (See Fig.6).

Part classification We evaluate our pre-trained encoder on the part classification task using the SolidLetter dataset, which consists of CAD models labeled across 26 alphabetic classes. We compare our method with strong baselines:

Table 2: Performance comparison on Fusion 360 Gallery Segmentation under different levels of supervision. The k-shot setting denotes using k labeled training samples in total. Accuracy (Acc, %) and mIoU (%) are reported.

Model	10-shot		20-shot		Full	
	Acc	mIoU	Acc	mIoU	Acc	mIoU
BRep-BERT	59.26	19.02	62.30	24.20	95.14	82.88
BRT	54.98	13.54	56.69	14.99	94.48	79.23
BRepFormer	50.24	12.31	51.75	13.38	94.02	78.73
Ours	**72.33**	**34.63**	**74.31**	**38.22**	**97.02**	**86.75**

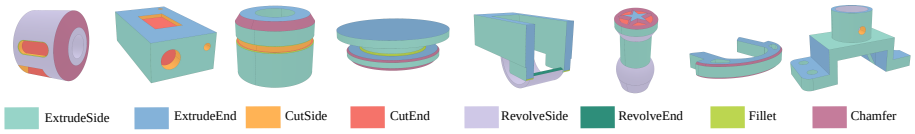


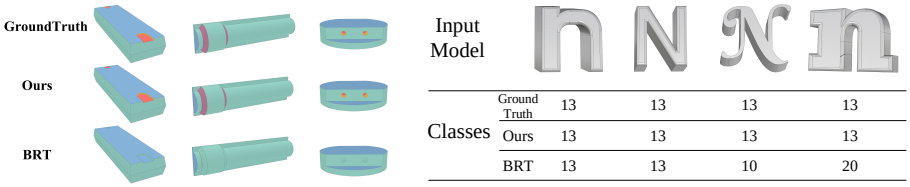
Fig. 6: Examples of modeling segmentation results on the Fusion 360 Gallery segmentation dataset under the full supervised setting. Our model accurately predicts per-face labels across 8 operation types, with distinct colors for each.

BRep-BERT [16], BRT [39], and BRepFormer [4]. For BRep-BERT, we report results from the original paper, as the publicly available code is currently unavailable. For BRT and BRepFormer, we adopt their official implementations and follow the recommended hyperparameter settings.

To assess performance under different levels of supervision, we conduct experiments in three settings: 10-shot, 20-shot, and full-data. In the few-shot settings, 10 or 20 labeled examples are sampled per class for training. The dataset is split in a class-balanced manner: for each alphabet class, 70% of the models are used for training, 15% for validation, and 15% for testing. All models are trained using the same experimental setup as described previously.

Our method achieves state-of-the-art performance on the SolidLetter benchmark across all supervision levels. Under 10-shot and 20-shot settings, our model achieves 82.51% and 89.58% accuracy, respectively, outperforming the self supervised baseline BRep-BERT (68.71% and 75.92%) as well as fully supervised approaches like BRepFormer (76.53% and 84.25%) and BRT (54.84% and 62.69%) by substantial margins (See Fig 7 Right). Notably, even with very limited labeled data, our method maintains high classification accuracy, demonstrating strong generalization and data efficiency. Under full supervision, our method continues to yield the best performance, achieving 97.91% accuracy, slightly surpassing BRep-BERT (97.76%), BRepFormer (97.59%), and BRT (97.35%).

We further evaluate the cross-dataset generalization of our model by removing SolidLetter samples from the pre-training corpus (denoted as Ours*). As shown in Table 3, even without exposure to the target dataset during pre-training, our method achieves 81.83% and 88.47% accuracy in the 10-shot and 20-shot settings, respectively, substantially outperforming all baselines, including the strong BRepFormer. Under full supervision, Ours* attains 97.78%, closely



Classes	Ground Truth	13	13	13	13
	Ours	13	13	13	13
	BRT	13	13	10	20

Fig. 7: Performance comparison with BRT under the 10-shot setting. Left: The segmentation task, where our method predicts face labels more accurately. Right: The part classification task, where our method correctly identifies all shapes as **n** (class 13), while BRT misclassifies the last two.

Table 3: Comparison on SolidLetter classification under different levels of supervision. The k-shot setting denotes using k labeled training samples for each class. "Ours*" denotes our model pre-trained without SolidLetter data. Accuracy (Acc, %) is reported.

Model	10-shot Acc	20-shot Acc	Full Acc
BRep-BERT	68.71	75.92	97.76
BRT	54.84	62.69	97.35
BRepFormer	76.53	84.25	97.59
Ours*	81.83	88.47	97.78
Ours	**82.51**	**89.58**	**97.91**

matching the best results. These findings confirm the robustness and transferability of our pre-trained encoder across different geometric domains.

4.4 Ablation study

To isolate the contributions of our core designs, we conduct ablation studies on MFInstSeg (Table 4). Additional ablations are detailed in the supplementary material.

1. Dual-Resolution vs. Network Width. We compare our dual-resolution (3×3 and 13×13) CSMA block against *Single Res.* (13×13 only) and *Triple Res.* (adding 7×7). *Single Res.* consistently lags behind our method. To further validate this, we scale the latent embedding dimension of the single-resolution variant to 384D and 512D. Despite the increased parameters, both wider variants still underperform our default setting (e.g., 87.17% vs. 88.75% at 0.1% supervision), demonstrating that simply widening a flat network cannot effectively capture multi-scale geometric dependencies. Furthermore, *Triple Res.* slightly underperforms our approach (e.g., 99.42% vs. 99.53% at 100%), demonstrating that an intermediate resolution introduces mild attention dilution. Thus, our extreme coarse-to-fine bottleneck provides the optimal balance.

2. Masking Strategies and Ratios. High-ratio masking is essential for robust representation learning. Pre-training with simple autoencoding (0% mask ratio, *w/o Masking*) causes a drastic performance collapse, dropping to 66.41% at 0.1% supervision. We evaluate the sensitivity of our framework to different masking ratios (50%, 60%, and 80%). While our architecture remains robust across these regimes, our default configuration (70% masking) yields the optimal

Table 4: Ablation study on MFInstSeg under various supervision ratios. The core architectural and pre-training designs are evaluated.

Configuration	0.1%	0.5%	1%	1.5%	2%	3%	100%
Single Res. (13×13 only)	85.83	93.14	95.19	95.88	97.16	97.42	99.27
Single Res. (384D)	86.11	94.55	96.41	97.02	97.33	97.66	99.35
Single Res. (512D)	87.17	95.46	96.95	97.53	97.75	98.03	99.41
Triple Res. (w/ 7×7)	86.41	94.46	96.40	97.04	97.38	97.78	99.42
MPNN first	76.39	92.10	94.95	95.62	96.12	96.41	99.06
w/o Masking (0% Mask)	66.41	84.19	92.99	94.20	95.33	95.96	99.43
Mask ratio (50%)	86.99	96.02	97.18	97.43	97.60	98.02	99.49
Mask ratio (60%)	87.41	96.12	97.19	97.46	97.73	98.05	99.51
Mask ratio (80%)	87.01	95.66	96.86	97.37	97.62	97.96	99.46
Mask gAAG	78.63	92.98	94.36	95.11	95.75	96.48	99.16
Frozen encoder	82.01	92.71	94.70	95.74	96.15	96.56	98.64
Ours (Default: 70% Mask)	**88.75**	**96.19**	**97.31**	**97.67**	**97.92**	**98.26**	**99.53**

balance, consistently achieving the highest accuracy. Furthermore, instead of masking input geometries, randomly masking the gAAG features after the BRep encoder (*Mask gAAG*) underperforms (e.g., 78.63% at 0.1%), confirming that input-level geometry masking establishes a stronger information bottleneck.

3. Architectural Routing (Global to Local). Reversing our architecture to apply the MPNN before the Transformer (*MPNN first*) significantly degrades performance (e.g., 76.39% vs. 88.75% at 0.1%). This proves that establishing a global, coarse-to-fine geometric context before local topological message passing serves as a superior inductive bias for BRep data.

4. Representation Transferability. When strictly freezing the pre-trained encoder and only updating the downstream MLP head (*Frozen encoder*), the model still achieves a remarkable 82.01% accuracy at 0.1% supervision. This strong performance with frozen representations demonstrates that our pre-training framework extracts high-quality, task-agnostic geometric features.

4.5 Computational Cost Analysis

To provide details on the computational cost of the proposed method, we report the model parameters, floating-point operations (FLOPs), inference latency, and throughput. The profiling is conducted using CAD models with an average of 30 faces on a single NVIDIA A800 GPU, consistent with the hardware setup described in the main paper.

As shown in Table 5, we report the metrics separately for the self-supervised pre-training stage (comprising the full masked autoencoder architecture) and the downstream task stage (comprising the pre-trained encoder and the task-specific network).

Pre-training Cost: During the pre-training stage, the full architecture contains 8.15M parameters and requires 20.98G FLOPs per forward pass. The average latency is 25.99 ms per model, corresponding to a throughput of 38.47 models per second, which constitutes a one-time computational cost.

Downstream Cost: For downstream tasks, the pre-training decoder is replaced by specific task networks tailored for the respective applications. The

Table 5: Computational cost profiling for the pre-training and downstream stages. Profiling is based on CAD models with an average of 30 faces.

Stage	Parameters (M)	FLOPs (G)	Latency (ms)	Throughput (models/s)
Pre-training	8.15	20.98	25.99	38.47
Downstream	6.09	5.26	10.42	96.01

adapted downstream network comprises 6.09M parameters and requires 5.26G FLOPs. The latency in this stage is 10.42 ms per model, with a throughput of 96.01 models per second.

5 Conclusion

We propose the Masked BRep Autoencoder, a novel self-supervised framework for CAD representation learning. Our method effectively captures both the geometric and topological structures of BRep models by directly reconstructing the raw geometries and attributes of masked faces and edges, establishing a rigorous information bottleneck. By featuring a hierarchical architecture that seamlessly integrates global geometric contexts via the cross-scale mutual attention (CSMA) bottleneck and preserves crucial local details through explicit local message passing, our encoder learns rich, transferable representations.

Extensive experiments on machining feature recognition, modeling segmentation, and part classification benchmarks demonstrate the effectiveness and versatility of our approach. Notably, our method achieves strong performance with very limited supervision and remains competitive under full-data settings. These results highlight the practicality and generalizability of our framework for a wide range of downstream tasks. In the future, as larger datasets become available, our framework has the potential to scale to broader applications, such as BRep generative models.

Limitations Although our exhaustive dual-resolution sampling is crucial for capturing fine-grained details, it inherently increases the computational and storage overhead during data preprocessing. Consequently, scaling this face-level representation to massive industrial assemblies introduces significant memory bottlenecks, restricting the maximum sequence length during training.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive suggestions and comments. This work is supported by the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China and the National Natural Science Foundation of China (92570201, 62272429).

References

1. Bian, S., Grandi, D., Hassani, K., Sadler, E., Borijin, B., Fernandes, A., Wang, A., Lu, T., Otis, R., Ho, N., et al.: Material prediction for design automation using graph representation learning. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. vol. 86229, p. V03AT03A001. American Society of Mechanical Engineers (2022)
2. Cho, I., Yoo, Y., Jeon, S., Kim, S.J.: Representing 3d shapes with 64 latent vectors for 3d diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 28556–28566 (October 2025)
3. Colligan, A.R., Robinson, T.T., Nolan, D.C., Hua, Y., Cao, W.: Hierarchical cadnet: Learning from b-reps for machining feature recognition. Computer-Aided Design **147**, 103226 (2022)
4. Dai, Y., Huang, X., Bai, Y., Guo, H., Gan, H., Yang, L., Shi, Y.: Brepformer: Transformer-based b-rep geometric feature recognition. In: Proceedings of the 2025 International Conference on Multimedia Retrieval. pp. 155–163 (2025)
5. Guo, H.X., Liu, Y., Pan, H., Guo, B.: Implicit conversion of manifold b-rep solids by neural halfspace representation. ACM Transactions on Graphics (TOG) **41**(6), 1–15 (2022)
6. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16000–16009 (2022)
7. Hou, J., Luo, C., Qin, F., Shao, Y., Chen, X.: Fus-gcn: Efficient b-rep based graph convolutional networks for 3d-cad model classification and retrieval. Advanced Engineering Informatics **56**, 102008 (2023)
8. Jayaraman, P.K., Sanghi, A., Lambourne, J.G., Willis, K.D., Davies, T., Shayani, H., Morris, N.: Uv-net: Learning from boundary representations. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11703–11712 (2021)
9. Jones, B., Hildreth, D., Chen, D., Baran, I., Kim, V.G., Schulz, A.: Automate: A dataset and learning approach for automatic mating of cad assemblies. ACM Transactions on Graphics (TOG) **40**(6), 1–18 (2021)
10. Jones, B.T., Hu, M., Kodnongbua, M., Kim, V.G., Schulz, A.: Self-supervised representation learning for cad. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 21327–21336 (2023)
11. Kreuzer, D., Beaini, D., Hamilton, W.L., Létourneau, V., Tossou, P.: Rethinking graph transformers with spectral attention. In: Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems (2021), `https://openreview.net/forum?id=huAdB-Tj4yG`
12. Lambourne, J.G., Willis, K.D., Jayaraman, P.K., Sanghi, A., Meltzer, P., Shayani, H.: 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)
13. Lei, R., Wu, H., Peng, Y.: Mfpointnet: A point cloud-based neural network using selective downsampling layer for machining feature recognition. Machines **10**(12), 1165 (2022)
14. Li, Z., Tong, X., Shi, P., Cai, M., Han, F.: Classification of non-standard mechanical parts with graph convolutional networks. Engineering Applications of Artificial Intelligence **153**, 110879 (2025)

15. Liu, X., Li, Y., Deng, T., Wang, P., Lu, K., Chen, J., Yang, D.: A supervised community detection method for automatic machining region construction in structural parts nc machining. Journal of Manufacturing Systems **62**, 367–376 (2022)
16. Lou, Y., Li, X., Chen, H., Zhou, X.: Brep-bert: Pre-training boundary representation bert with sub-graph node contrastive learning. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management. pp. 1657–1666 (2023)
17. Mialon, G., Chen, D., Selosse, M., Mairal, J.: Graphit: Encoding graph structure in transformers (2021)
18. Ning, F., Shi, Y., Cai, M., Xu, W.: Part machining feature recognition based on a deep learning method. Journal of Intelligent Manufacturing **34**(2), 809–821 (2023)
19. Rong, Y., Bian, Y., Xu, T., Xie, W., Wei, Y., Huang, W., Huang, J.: Self-supervised graph transformer on large-scale molecular data. Advances in Neural Information Processing Systems **33** (2020)
20. Shi, Y., Zhang, Y., Harik, R.: Manufacturing feature recognition with a 2d convolutional neural network. CIRP Journal of Manufacturing Science and Technology **30**, 36–57 (2020)
21. Wang, P., Yang, W.A., You, Y.: A hybrid learning framework for manufacturing feature recognition using graph neural networks. Journal of Manufacturing Processes **85**, 387–404 (2023)
22. Willis, K.D., Jayaraman, P.K., Chu, H., Tian, Y., Li, Y., Grandi, D., Sanghi, A., Tran, L., Lambourne, J.G., Solar-Lezama, A., et al.: Joinable: Learning bottom-up assembly of parametric cad joints. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 15849–15860 (2022)
23. Wu, H., Lei, R., Peng, Y., Gao, L.: Aagnet: A graph neural network towards multi-task machining feature recognition. Robotics and Computer-Integrated Manufacturing **86**, 102661 (2024)
24. Wu, Q., Zhao, W., Yang, C., Zhang, H., Nie, F., Jiang, H., Bian, Y., Yan, J.: Sgformer: Simplifying and empowering transformers for large-graph representations. Advances in Neural Information Processing Systems **36**, 64753–64773 (2023)
25. Wu, Z., Jain, P., Wright, M., Mirhoseini, A., Gonzalez, J.E., Stoica, I.: Representing long-range context for graph neural networks with global attention. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems. vol. 34, pp. 13266–13279. Curran Associates, Inc. (2021)
26. Xu, X., Lambourne, J., Jayaraman, P., Wang, Z., Willis, K., Furukawa, Y.: Brepgen: A b-rep generative diffusion model with structured latent geometry. ACM Transactions on Graphics (TOG) **43**(4), 1–14 (2024)
27. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 206–215 (2018)
28. Yao, C., Wu, K., Zheng, Z., Xing, S., Fu, X.M.: Brepmae: Self-supervised masked brep autoencoders for machining feature recognition. arXiv preprint arXiv:2602.22701 (2026)
29. Yao, X., Wang, D., Yu, T., Luan, C., Fu, J.: A machining feature recognition approach based on hierarchical neural network for multi-feature point cloud models. Journal of Intelligent Manufacturing **34**(6), 2599–2610 (2023)
30. Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., Liu, T.Y.: Do transformers really perform bad for graph representation? arXiv preprint arXiv:2106.05234 (2021)

31. Zhang, H., Wang, W., Zhang, S., Wang, Z., Zhang, Y., Zhou, J., Huang, B.: Point cloud self-supervised learning for machining feature recognition. Journal of Manufacturing Systems **77**, 78–95 (2024)
32. Zhang, H., Wang, W., Zhang, S., Zhang, Y., Zhou, J., Wang, Z., Huang, B., Huang, R.: A novel method based on deep reinforcement learning for machining process route planning. Robotics and Computer-Integrated Manufacturing **86**, 102688 (2024)
33. Zhang, H., Zhang, S., Zhang, Y., Liang, J., Wang, Z.: Machining feature recognition based on a novel multi-task deep learning network. Robotics and Computer-Integrated Manufacturing **77**, 102369 (2022)
34. Zhang, J., Zhang, H., Xia, C., Sun, L.: Graph-bert: Only attention is needed for learning graph representations. arXiv preprint arXiv:2001.05140 (2020)
35. Zhang, S., Guan, Z., Jiang, H., Wang, X., Tan, P.: Brepmfr: Enhancing machining feature recognition in b-rep models through deep learning and domain adaptation. Computer Aided Geometric Design **111**, 102318 (2024)
36. Zhang, W., Sheng, Z., Jiang, Y., Xia, Y., Gao, J., Yang, Z., Cui, B.: Evaluating deep graph neural networks. arXiv preprint arXiv:2108.00955 (2021)
37. Zhang, Z., Jaiswal, P., Rai, R.: Featurenet: Machining feature recognition based on 3d convolution neural network. Computer-Aided Design **101**, 12–22 (2018)
38. Zheng, X., Chen, H., He, F., Liu, X.: Sfrgnn-da: An enhanced graph neural network with domain adaptation for feature recognition in structural parts machining. Journal of Manufacturing Systems **81**, 16–33 (2025)
39. Zou, Q., Zhu, L.: Bringing attention to cad: Boundary representation learning via transformer. Computer-Aided Design p. 103940 (2025)