Comput. Aided Geom. Des. 111 (2024) 102318

Contents lists available at ScienceDirect

Computer Aided Geometric Design

journal homepage: www.elsevier.com/locate/cagd

BrepMFR: Enhancing machining feature recognition in B-rep models through deep learning and domain adaptation

Shuming Zhang [a], Zhidong Guan [a], Hao Jiang [b], Xiaodong Wang [a,*], Pingan Tan [a]

[a] School of Aeronautic Science and Engineering, Beihang University, Beijing, China
[b] School of Mechanical Engineering and Automation, Beihang University, Beijing, China

A R T I C L E I N F O

Keywords:
Machining feature recognition
Boundary representation
Deep learning
Graph neural network
Domain adaptation

A B S T R A C T

Feature Recognition (FR) plays a crucial role in modern digital manufacturing, serving as a key technology for integrating Computer-Aided Design (CAD), Computer-Aided Process Planning (CAPP) and Computer-Aided Manufacturing (CAM) systems. The emergence of deep learning methods in recent years offers a new approach to address challenges in recognizing highly intersecting features with complex geometric shapes. However, due to the high cost of labeling real CAD models, neural networks are usually trained on computer-synthesized datasets, resulting in noticeable performance degradation when applied to real-world CAD models. Therefore, we propose a novel deep learning network, BrepMFR, designed for Machining Feature Recognition (MFR) from Boundary Representation (B-rep) models. We transform the original B-rep model into a graph representation as network-friendly input, incorporating local geometric shape and global topological relationships. Leveraging a graph neural network based on Transformer architecture and graph attention mechanism, we extract the feature representation of high-level semantic information to achieve machining feature recognition. Additionally, employing a two-step training strategy under a transfer learning framework, we enhance BrepMFR's generalization ability by adapting synthetic training data to real CAD data. Furthermore, we establish a large-scale synthetic CAD model dataset inclusive of 24 typical machining features, showcasing diversity in geometry that closely mirrors real-world mechanical engineering scenarios. Extensive experiments across various datasets demonstrate that BrepMFR achieves state-of-the-art machining feature recognition accuracy and performs effectively on CAD models of real-world mechanical parts.

1. Introduction

Since the early 1950s, the emerging of Computer Numerical Control (CNC) technology spurred the research and development of Computer-Aided Manufacturing (CAM) and Computer-Aided Design (CAD). These modern digital technologies have greatly improved the efficiency of product development. In digital design and manufacturing, the design department utilizes three-dimensional design software to create CAD models that incorporates both lower-level geometric information and higher-level design intent. Subsequently, the manufacturing department, leveraging Computer-Aided Process Planning (CAPP) systems, generates a series of

* Corresponding author.
E-mail addresses: by2005413@buaa.edu.cn (S. Zhang), 07343@buaa.edu.cn (Z. Guan), jianghao14789@buaa.edu.cn (H. Jiang), d5062010@163.com (X. Wang), tpa1999@buaa.edu.cn (P. Tan).

https://doi.org/10.1016/j.cagd.2024.102318

Available online 25 April 2024

machining instructions for the production of components. During the machining processes, specific shapes are machined on the workpiece by machine tools. These shapes, imbued with manufacturing semantics, are referred to as machining features.

However, due to the constraints of heterogeneous software systems, CAD models in the three main stages of digital production - CAD, CAPP, and CAM - often only convey low-level information such as geometric surfaces and curves. Crucial high-level semantic information, including design features and machining features, is frequently lost. These features are typically represented as collections of specific geometric and topological elements (such as surfaces, curves, points, B-rep faces and edges), in Boundary Representation (B-rep). To enhance process planning and manufacturing efficiency, CAPP systems employ automatic Machining Feature Recognition (MFR) to reanalyze CAD models from a manufacturing perspective. This involves grouping geometric and topological elements to identify machining features, such as holes, slots, steps, and chamfers. Subsequently, based on specific parameters of these machining features, such as radius, depth, and axial dimensions, suitable manufacturing processes for CAM can be intelligently generated. This includes defining machining steps, selecting machining tools, and planning CNC paths. Therefore, machining feature recognition is considered a crucial step in manufacturing automation, effectively bridging the gap between design and manufacturing in the integration of CAD, CAPP, and CAM (Brousseau et al., 2008).

Over the past few decades, feature recognition has emerged as a dynamic research area within the CAD/CAM field. Scholars have proposed a variety of geometric reasoning-based feature recognition methods: Semantics-based approach (Vandenbrande and Requicha, 1993; Zhang et al., 2017), Rule-based approach (Zehtaban and Roller, 2016), Graphs-based approach (Joshi and Chang, 1988; Huang and Yip-Hoi, 2002; Fu et al., 2003), Volume decomposition approach (Woo and Sakurai, 2002; Geng et al., 2016), and Hint-based approach (Vandenbrande and Requicha, 1993; Han et al., 1997). These traditional methods leverage geometric and topological information for the identification of machining features. In recent years, the advent of deep learning methods has introduced a novel approach to tackle this challenge. Depending on the representation of the three-dimensional shape, learning-based machining feature recognition methods can be classified into point clouds (Lei et al., 2022), voxels (Ning et al., 2023; Peddireddy et al., 2021; Lee et al., 2021), polygon meshes (Shi et al., 2020; Jia et al., 2023), multiple views (Shi et al., 2022), and B-rep (Cao et al., 2020; Colligan et al., 2022; Wu et al., 2024). These data-driven methods showcase robust recognition and classification capabilities, possessing considerable potential for substantial advancement in the field of CAD model feature recognition.

Existing machining feature recognition methods have undergone extensive research and application, yet they still encounter challenges in several aspects. Firstly, algorithms tailored for specific predefined features lack generality, as the diverse nature of features. Secondly, the time cost of feature recognition is relatively high, particularly when handling multiple types of features in complex, intersecting, or large structural components. Thirdly, learning-based methods often require the conversion of the original CAD models into representations such as point clouds, voxels, or images, leading to a decrease in the accuracy of geometric shape representation. Furthermore, neural networks trained on artificially synthesized datasets demonstrate significant performance degradation when applied to real-world CAD models. This is attributed to the inconsistency in the distribution of training and testing data.

To address aforementioned challenges, we propose a novel deep learning network named BrepMFR. This network is specifically designed to directly execute machining feature recognition on B-rep models within the CAD/CAM domain. The original B-rep model is converted into a graph representation for network-friendly input, where graph nodes and edges respectively correspond to B-rep faces and edges. Subsequently, we leverage a graph neural network based on the Transformer architecture and graph attention mechanism to encode both local geometric shape and global topological relationships, achieving high-level semantic extraction and prediction of machining feature categories. Furthermore, to enhance the performance of neural networks on real-world CAD models, we adopt a two-step training strategy within a novel transfer learning framework. This achieved cross-domains adaptation, thereby further strengthening the generalization capabilities of BrepMFR. For network training, we establish a large-scale synthetic CAD dataset, CADSynth, comprising 24 typical machining features. This dataset exhibits greater geometric diversity and is more representative of mechanical engineering than the existing synthetic datasets MFCAD (Cao et al., 2020) and MFCAD++ (Colligan et al., 2022). Testing results across multiple datasets demonstrate that BrepMFR achieves state-of-the-art machining feature recognition accuracy, showcasing excellent performance on CAD models of mechanical components from the real world. Codes and datasets related to this study could be found in https://github.com/zhangshuming0668/BrepMFR.

2. Related work

2.1. Conventional feature recognition

Traditional feature recognition algorithms rely on the geometric and topological information of models. Henderson (1984), Donaldson and Corney (1993), Vosniakos and Davies (1993) and Chan and Case (1994) proposed a series of rule-based methods based on defined boundary patterns and expert systems. Although this approach is straightforward and easy to implement, the non-uniqueness and incompleteness of rule definitions make it less flexible and pose challenges in dealing with intersecting features.

The topological structure of B-rep models shares inherent similarity with graphs, allowing the issue of feature recognition to be viewed as a sub-graph recognition problem within the B-rep face-edge graph. Joshi and Chang (1988) was the first to utilize an attributed adjacency graph (AAG) for feature recognition. Building on this, Gao and Shah (1998) and Yuen and Venuvinod (1999) attempted two aspects of improvement: completing extensive attribute information to enhance the expressiveness of graphs; and employing various matching strategies.

Vandenbrande and Requicha (1993) proposed the hint-based method to address the issue of recognizing intersecting features. This method extracts all feature hints from a B-rep model and infers possible true features through geometric reasoning. Subsequently, it utilizes solid modeling operations to construct the complete features corresponding to the extracted feature hints. Further improvements have been made by Huang and Yip-Hoi (2002) and Fu et al. (2003) in terms of reasoning methods, feature classification, and feature probability ordering.

Woo (1982) proposed the Alternating Sum of Volumes (ASV) Decomposition method, which represents the solid as the difference and union of convex bodies in a tree structure. Feature recognition can be achieved by examining leaf nodes or their combinations. Subsequently, Tang and Woo (1991), Kim and Wilde (1992) and Dong and Vijayan (1997) addressed the convergence of the decomposition and expanded the applicability of the method.

2.2. Deep learning-based feature recognition

Prabhakar and Henderson (1992) first demonstrated the application of artificial neural networks in 3D solid models feature recognition. In recent years, more and more scholars have introduced deep learning methods into feature recognition research, proposing various deep neural network models for different 3D shape representations. Point clouds represent 3D geometric shapes as a set of points. MFPointNet (Lei et al., 2022) is a machining feature recognition method based on down sampling point cloud neural networks, enhancing recognition efficiency while reducing the complexity of the neural network. 3D voxels present a structured and regular form of data, naturally suitable for encoding with convolutional neural networks (CNNs). Zhang et al. (2018) introduced FeatureNet, a voxel-based 3D CNN for processing feature recognition, and established a dataset with 24 typical machining features. Ning et al. (2023) combined voxel-based 3D CNN with graph-based methods to achieve improved processing feature recognition. For 3D shapes represented as polygon meshes, Takaishi et al. (2020) employed the k-means method and KNN algorithm to cluster mesh vertexes, achieving the recognition of geometric shape features. Shi et al. (2020) proposed a method based on heat kernel features. According to the heat kernel eigenvalues of mesh nodes, the surface mesh is divided into a finite number of regions, and an adjacency graph is generated. Subsequently, 2D CNN is employed to classify these graph embeddings for feature recognition.

B-rep is the most commonly used data format in the CAD field. Cao et al. (2020) first converted B-rep model into graph representation, and then use graph neural networks (GNNs) to conduct representation learning. Subsequently, Colligan et al. (2022) introduced a hierarchical structure graph convolutional model as an improvement. Jayaraman et al. (2021) presented the UV-Net model, utilizing CNNs to encode the surfaces and curves in B-rep models. They then employed graph convolutional networks (GCN) with information propagation to learn graph feature representations. To comprehensively capture the geometric and topological information in B-rep representation, Lambourne et al. (2021) introduced a novel graph convolution method, BRepNet, tailored to the B-rep data structure. This method performs convolution centered around coedges, enabling the learning of both global and local topological relationships in B-rep models. Lee et al. (2023) proposed BRepGAT to segment machining feature in B-rep models based on Graph Attention Networks (GAT). Wu et al. (2024) proposed a multi-task network, AAGNet, capable of simultaneously performing semantic segmentation, instance segmentation, and bottom face segmentation.

2.3. Domain adaptation

Traditional deep learning methods assume that training and testing data adhere to the independent and identically distribution. However, real-world environments are complex and dynamic, limiting the application of deep learning methods in practical scenarios. To address this issue, Transfer Learning has been proposed to handle differences in data domains, tasks, or distributions between the training and testing phases (Pan and Yang, 2009; Zhuang et al., 2020). Domain Adaptation, a branch of transfer learning (Wang and Deng, 2018), focuses on knowledge transfer between source and target domains with inconsistent data distributions, while the tasks in both domains remain the same.

Existing domain adaptation methods encompass those utilizing statistical metrics and those employing adversarial learning. Statistical methods minimize the distribution difference between source and target domains, enabling neural networks to generate features with similar distributions under the chosen statistical metric. Zhuang et al. (2015) proposed a feature encoding method capable of generating domain-shared features by minimizing the KL divergence. Long et al. (2017) introduced Joint Adaptation Networks, applying Maximum Mean Discrepancy (MMD) to achieve joint distribution matching. Lee et al. (2019) used Wasserstein distance to achieve distribution matching between domains from a geometric perspective. Inspired by Generative Adversarial Networks (GAN), adversarial learning-based domain adaptation methods employ adversarial training of feature extractors to extract domain-shared features. Ganin and Lempitsky (2015) proposed Domain Adversarial Neural Networks (DANN), implementing adversarial training through a gradient reversal layer inserted into the neural network. Tzeng et al. (2017) proposed a multi-adversarial domain adaptation method, designing a separate domain discriminator for each category. Adversarial training of the feature extractor and multiple domain discriminators achieved class-level conditional probability distribution matching. Zhang et al. (2020) applied an attention mechanism for feature weighting considering varying transferability of different features, with both the classifier and domain discriminator taking the weighted features as input.

3. Methodology

3.1. B-rep model graph representation

Due to neural networks operate on numerical inputs, the first concern in learning-based feature recognition is how to transform the geometric and topological information of B-rep models into a format suitable for neural network input.

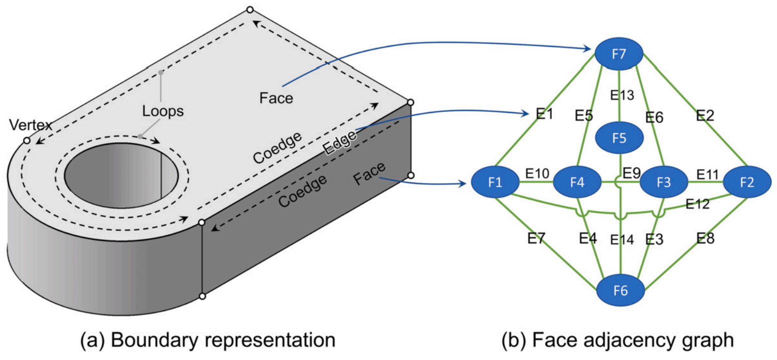


Fig. 1. (a) Boundary representation and (b) face adjacency graph for a 3D solid model.

3.1.1. *Boundary representation*

Boundary representation is a method employed for representing and editing solid models. Constructed from bottom to top, it utilizes topological entities such as Vertex, Edge, Face, Coedge, Loop, Shell, and Body, forming the topological data structure of a 3D solid model. This structure meticulously records the topological relationships of elements at various hierarchical levels, constituting the "skeleton" of the model.

To represent the geometric shape of solid models, B-rep relies on parameterized curves and surfaces. It forms a closed, single connected space by combining a set of consistently oriented trimming surfaces, creating the boundaries of geometric shapes. Faces in B-rep are bounded by sets of edges, which are portions of curves located on the surface. Therefore, our work focuses on the two most crucial B-rep elements, faces, and edges, transforming the B-rep model into the form of a face adjacency graph as input to neural networks. As shown in Fig. 1, a CAD model contains a set of B-rep faces $F = \{f_1, f_2, \ldots, f_{|F|}\}$ and a set of B-rep edges $E = \{e_1, e_2, \ldots, e_{|E|}\}$. The topological relationships between B-rep faces and edges can be represented in the form of a graph $G = (F, E)$, where graph nodes F represent B-rep faces, and graph edges E represent B-rep edges.

3.1.2. *Network-friendly representation of geometric*

B-rep models typically employ parametric or elementary analytic curves/surfaces to represent geometric shapes. Parametric curves/surfaces (such as Bézier surfaces, NURBS surfaces, B-spline curves, NURBS curves) are defined by varying numbers of control points and knot vectors. Analytic curves/surfaces (such as spheres, cylinders, tori, straight lines, arcs) are defined by mathematical expressions and parameters. To ensure compatibility and generality, we adopt a unified representation for these diverse curve and surface geometric shapes, following UV-Net (Jayaraman et al., 2021), as illustrated in Fig. 2. Initially, we transform the base surface of B-rep faces into NURBS surfaces. Subsequently, we evenly sample points on their two-dimensional parameter domains along both u and v directions. Each sample point is represented by a 7D vector $[X, Y, Z, N_x, N_y, N_z, T]$, recording three-dimensional coordinates (X, Y, Z), surface normals (N_x, N_y, N_z), and the relationship between points and trimming surfaces (T). This process results in a discrete representation $f_{\text{geom}} \in \mathbb{R}^{n \times n \times 7}$, where n represents the number of discrete points in the u or v direction. In our work, n is set to 5. Similarly, we evenly sample points on the parameter domain of curves, with each sample point represented by a 6D vector $[X, Y, Z, T_x, T_y, T_z]$ recording three-dimensional coordinates (X, Y, Z) and curve tangents (T_x, T_y, T_z). This approach yields a discrete representation of the curve $e_{\text{geom}} \in \mathbb{R}^{n \times 6}$.

Furthermore, the description of B-rep faces and edges includes other fundamental attributes. For B-rep faces, these attributes are surface type (f_{type}), area (f_{area}), the number of boundary loops (f_{loop}) and the number of adjacent faces (f_{adj}). As for edges, the fundamental attributes encompass curve type (e_{type}), length (e_{len}), convexity (e_{conv}), and dihedral angle (e_{ang}). The convexity of edges is determined through a connectivity test adapted and modified from (Joshi and Chang, 1988; Liu et al., 1996). This test involves assessing whether the faces sharing the edge form a concave or convex angle, or tangential continuity (G1 continuity).

3.1.3. *Network-friendly representation of proximity*

In addition to the local geometric shapes of B-rep faces and edges mentioned in the previous section, the global topological relationships between these geometric elements are crucial for the representation learning of B-rep models. We express the proximity between geometric elements in the following three aspects:

Extended adjacency relations. Traditional graph-based methods (Joshi and Chang, 1988) use an adjacency matrix to represent the adjacency relationship between any two B-rep faces. If two faces are directly adjacent, the corresponding element value in the adjacency matrix is 1; otherwise, it is 0. However, this approach has limitations as it cannot represent the proximity between faces that are not directly adjacent. Therefore, we employ the Floyd-Warshall algorithm (Roy et al., 1959; Warshall, 1962) to compute the shortest paths between any pair of B-rep faces on the face adjacency graph presented in Sec. 3.1.1, and take the number of edges on this path as the shortest distance. For instance, the shortest distance between directly adjacent faces $F1$ and $F7$ in Fig. 1 is 1; while the shortest distance between non-adjacent faces $F1$ and $F3$ is 2. Consequently, we use the shortest distances between face pairs as the element values in the adjacency matrix, obtaining an extended adjacency matrix $D_{\text{g}} \in \mathbb{Z}^{|F| \times |F|}$.

Spatial positional relations. The relative spatial positions of B-rep faces in three-dimensional space also reflect important geometric and topological information. We use statistical-based geometric descriptors D2 distance and A3 distance (Rea et al., 2005)

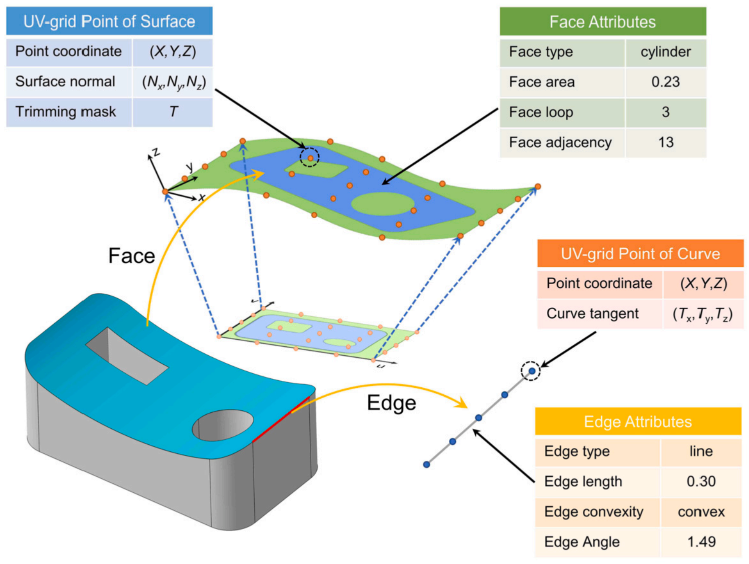


Fig. 2. Geometric and attributes of B-rep faces and edges.

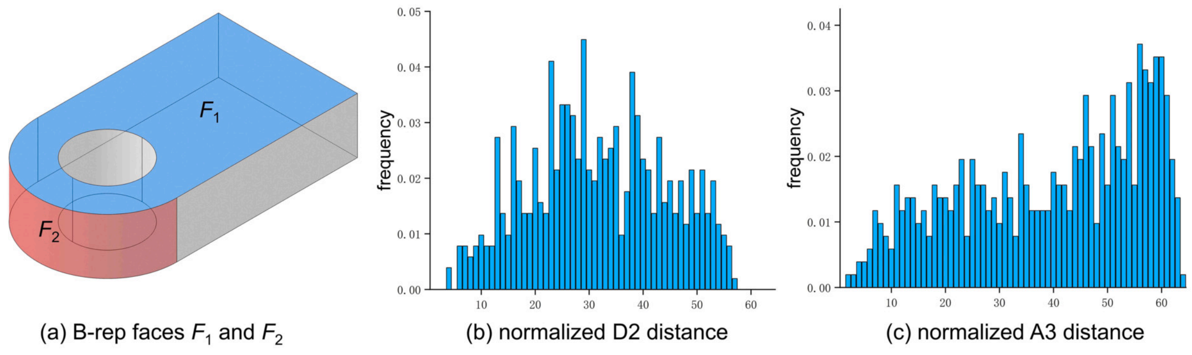


Fig. 3. (a) 3D spatial position of face F1 and F2 in a B-rep model; (b) (c) frequency distribution histograms of normalized D2 and A3 distance between F1 and F2.

to express these relations. Specifically, for two pairwise faces, 512 point pairs are randomly selected, and the ratio of the Euclidean distance of each point pair to the diagonal length of the whole solid model bounding box is calculated. Then we take the marginal distributions of the above ratio as the D2 distance between two faces, approximated by dividing the [0,1] interval into 64 sub-intervals and calculating the frequency of each distance ratio among them, $d^{\mathrm{D2}}(f_i, f_j) \in \mathbb{Z}^{64}$. Likewise, we obtain the approximate marginal distribution $d^{\mathrm{A3}}(f_i, f_j) \in {}^{64}$ for the A3 distance. Fig. 3 shows the three-dimensional spatial position of face F_1 and F_2 in a B-rep model, along with the frequency distribution histograms corresponding to D2 and A3 distances. By calculating D2 and A3 distances between every pair of faces in the B-rep model, we can obtain the spatial positional relations matrices $D_{\mathrm{d2}} \in \mathbb{Z}^{|F|\times|F|\times 64}$ and $D_{\mathrm{a3}} \in \mathbb{Z}^{|F|\times|F|\times 64}$.

Face-edge relations. Faces in the B-rep model are represented as trimming surfaces, with loops composed of multiple edges defining the boundary. Moreover, any two adjacent faces are connected by common edges, namely, coedges. Therefore, edges play a pivotal role in the topological construction. In the B-rep graph described in Sec. 3.1.1, for any two ordered faces pairs (f_i, f_j), we can find the shortest path between them and record the edges chain on this shortest path $\{e_{ij,1}, e_{ij,2}, \ldots, e_{ij,N}\}$.

3.2. Network architecture

Existing methods for recognizing machining features in B-rep models based on graph neural networks predominantly rely on Message Passing Neural Networks (MPNNs). While effective in specific tasks, these approaches face inherent limitations such as restricted receptive fields and network depth. Inspired by recent advancements in graph transformers (Ying et al., 2021; Zhao et al., 2021), we incorporate the transformer framework (Vaswani et al., 2017) into the representation learning of B-rep models, aiming to enhance the neural network's feature extraction capabilities for complex CAD models. The architecture of our proposed BrepMFR deep learning network is illustrated in Fig. 4.

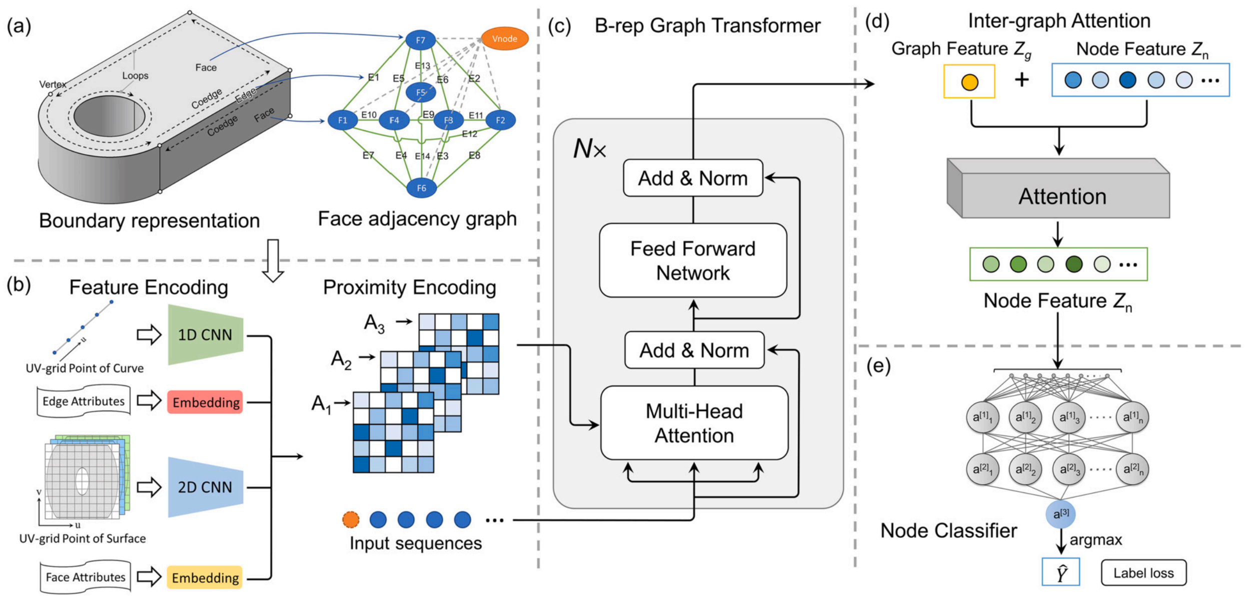


Fig. 4. The network architecture of BrepMFR.

Table 1
The detail of encoding layers for face attributes and edge attributes.

Face Attributes			Edge Attributes		
Inputs	Encoding layers	Outputs	Inputs	Encoding layers	Outputs
$f_{\text{type,i}} \in \mathbb{Z}^1$	Embedding(8, 32)	$h_{\text{type,i}} \in \mathbb{R}^{32}$	$e_{\text{type,i}} \in \mathbb{Z}^1$	Embedding(8, 32)	$l_{\text{type,i}} \in \mathbb{R}^{32}$
$f_{\text{area,i}} \in \mathbb{R}^1$	Linear(1, 32)	$h_{\text{area,i}} \in \mathbb{R}^{32}$	$e_{\text{len,i}} \in \mathbb{R}^1$	Linear(1, 32)	$l_{\text{len,i}} \in \mathbb{R}^{32}$
$f_{\text{loop,i}} \in \mathbb{Z}^1$	Embedding(256, 32)	$h_{\text{loop,i}} \in \mathbb{R}^{32}$	$e_{\text{conv,i}} \in \mathbb{Z}^1$	Embedding(3, 32)	$l_{\text{conv,i}} \in \mathbb{R}^{32}$
$f_{\text{adj,i}} \in \mathbb{Z}^1$	Embedding(256, 32)	$h_{\text{adj,i}} \in \mathbb{R}^{32}$	$e_{\text{ang,i}} \in \mathbb{R}^1$	Linear (1, 32)	$l_{\text{ang,i}} \in \mathbb{R}^{32}$

3.2.1. Input feature encoder

The network-friendly representation of geometry and proximity introduced in Sec. 3.1.1 serves as the input to our neural network. Initially, following UV-Net, a 2D CNN with seven channels (three for XYZ coordinates, three for normals, and one for trimming mask) processes the discrete representation of surfaces, $f_{\text{gome,i}} \in \mathbb{R}^{n\times n\times 7}$, to yield the feature representation, $h_{\text{geom,i}} \in \mathbb{R}^{128}$. The configuration of 2D CNN consists of sequential layers: Conv(7, 64, 3)→Conv(64, 128, 3)→Conv(128, 256, 3)→Pool(1, 1)→FC(256, 128), where Conv(i, o, k) is a 2D convolutional layer with i input channels, o output channels, and kernel size k, Pool(n, n) is an adaptive average pooling layer that outputs an $n \times n$ feature map, and FC(i, o) is a fully connected layer that maps an i-dimensional vector to an o-dimensional vector. Similarly, a 1D CNN with six channels (three for XYZ coordinates and three for tangents) processes the discrete representation of curves, $e_{\text{geom,i}} \in \mathbb{R}^{n\times 6}$, to generate the edge geometry's feature representation, $l_{\text{geom,i}} \in \mathbb{R}^{128}$. The 1D CNN employs convolutional and pooling layers: Conv-1d(6, 64, 3)→Conv-1d(64, 128, 3) →Conv-1d(128, 256, 3)→Pool(1, 1)→FC(256, 128).

Next, we encode the fundamental attributes of faces and edges through learnable linear layers or embedding layers. These attributes include 1D vectors: $f_{\text{type,i}} \in \mathbb{Z}^1$ for surface types (plane, cylinder, cone, sphere, torus, Bézier, B-Spline, Nurbs), $e_{\text{type,i}} \in \mathbb{Z}^1$ for curve types (line, circle, ellipse, parabola, hyperbola, Bézier, B-Spline, Nurbs), $e_{\text{conv,i}} \in \mathbb{Z}^1$ for edge convexity (concave, convex, smooth), and $f_{\text{loop,i}} \in \mathbb{Z}^1$, $f_{\text{adj,i}} \in \mathbb{Z}^1$ for the number of loops and adjacent faces for faces, with values ranging from 1 to 256. These vectors are encoded into 32D feature vectors via learnable embedding layers. For the remaining attributes, $f_{\text{area,i}}$, $e_{\text{len,i}}$, $e_{\text{ang,i}}$, we apply linear layers without biases for mapping to 32D feature vectors. The configurations of these layers within the neural network are detailed in Table 1. Finally, we concatenate all feature vectors of each geometric element to form a 256D feature vector, representing a B-rep face or edge:

$$h_i = Concat(h_{\text{geom,i}}, h_{\text{type,i}}, h_{\text{area,i}}, h_{\text{loop,i}}, h_{\text{adj,i}}) \in \mathbb{R}^{256} \tag{1}$$

$$l_i = Concat(l_{\text{geom,i}}, l_{\text{type,i}}, l_{\text{len,i}}, l_{\text{conv,i}}, l_{\text{ang,i}}) \in \mathbb{R}^{256} \tag{2}$$

For the extended adjacency matrix $D_g \in \mathbb{Z}^{|F|\times|F|}$, each element $d_{g(i,j)}$ denotes the minimum distance between B-rep faces f_1 and f_2 in the adjacency graph, constrained to a value range of 1 to 256. We obtain a feature representation for this minimum distance between B-rep faces using a learnable embedding layer (256,64):

$$A_1 \in \mathbb{R}^{|F|\times|F|\times 64} \tag{3}$$

For the matrices $D_{\mathrm{d2}} \in \mathbb{Z}^{|F|\times|F|\times 64}$ and $D_{\mathrm{a3}} \in \mathbb{Z}^{|F|\times|F|\times 64}$, which represent the D2 and A3 distances between faces, we combine them together to obtain the feature representation of spatial position relationships:

$$A_2 \in \mathbb{R}^{|F|\times|F|\times 64},\ A_{2(i,j)} = d^{\mathrm{D2}}(f_i, f_j) + d^{\mathrm{A3}}(f_i, f_j) \tag{4}$$

For the shortest path of edge chains $\{e_{ij,1}, e_{ij,2}, \ldots, e_{ij,N}\}$ between any ordered pair of faces (f_i, f_j), we summarize the feature encoding of these B-rep edges:

$$A_3 \in {}^{|F|\times|F|\times 64},\ A_{3(i,j)} = \frac{1}{N}\sum_{n=1}^{N}(l_{ij,n} \odot w_i) \tag{5}$$

where $w_i \in \mathbb{R}^{64}$ is a learnable weight coefficient that depends on the positional order. For example, it can be learned to decrease as the position advances along the path. Consequently, in the path from B-Rep face F_i to F_j, the edges closer to the starting face F_i are considered more important.

3.2.2. Graph transformer

We developed a B-rep graph transformer to encode the B-rep models' graph representation by adopting the standard Transformer architecture and its adaptation for graphs, known as Graphormer (Ying et al., 2021). The graph transformer consists of six layers of Transformer blocks. Each block includes sixteen attention heads and pre-layer normalization. Firstly, the feature representations $H^{(0)} = [h_1^{\mathrm{T}}, h_2^{\mathrm{T}}, \ldots, h_{|F|}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{|F|\times 256}$ of all graph nodes are sequentially fed into the network. From the hidden state $H^{(i-1)}$ of layer $i-1$, the hidden state $H^{(i)}$ of layer i can be obtained through a multi-head self-attention module and a fully connected feedforward network (FFN), with subsequent layer normalization.

$$\begin{aligned} H'^{(i)} &= \mathrm{MultiHead}(norm(\mathrm{H}^{(i-1)})) + \mathrm{H}^{(i-1)} \\ H^{(i)} &= \mathrm{FFN}(norm(\mathrm{H}'^{(i)})) + \mathrm{H}'^{(i)} \end{aligned} \tag{6}$$

Each layer consists of M self-attention modules:

$$\begin{aligned} &\mathrm{MultiHead(H) = Concat(head_1, \ldots, head_M)W^O} \\ &\mathrm{head}_m = \mathrm{self-att}\left(H, A_1, A_2, A_3\right) \\ &\quad = \mathrm{softmax}\left(\frac{Q_m K_m^{\mathrm{T}}}{\sqrt{d_k}} + A_1\left(W_{a1}^m\right)^{\mathrm{T}} + A_2\left(W_{a2}^m\right)^{\mathrm{T}} + A_3\left(W_{a3}^m\right)^{\mathrm{T}}\right) V_m, \\ &\forall m \in \{1, \ldots, M\},\ Q_m = HW_q^m,\ K_m = HW_k^m,\ V_m = HW_v^m \end{aligned} \tag{7}$$

Here, $W^O \in \mathbb{R}^{Md_v\times 256}$ is the output projection matrix, $W_q^m \in \mathbb{R}^{256\times d_k}$, $W_k^m \in \mathbb{R}^{256\times d_k}$ and $W_v^m \in \mathbb{R}^{256\times d_v}$ are projection matrices, while $W_{a1}^m \in \mathbb{R}^{64}$, $W_{a2}^m \in \mathbb{R}^{64}$ and $W_{a3}^m \in \mathbb{R}^{64}$ are weight embedding matrices.

To obtain the graph feature of the entire B-Rep graph, our network utilizes the same graph pooling method as Graphformer. A virtual node, denoted as Vnode, is introduced, connecting to all nodes, thereby extending the sequence lengths of input and output to $(|F|+1)$. The output of the final layer encompasses the feature representations of all graph nodes $Z_n = [z_1^{\mathrm{T}}, z_2^{\mathrm{T}}, \cdots z_{|F|}^{\mathrm{T}}] \in \mathbb{R}^{|F|\times 256}$, and the feature representation $Z_g \in \mathbb{R}^{256}$ corresponding to the entire B-Rep model.

3.2.3. Inter-graph attention

The feature representations Z_n of B-Rep faces and the feature representation Z_g of the entire B-Rep model capture local and global information respectively. Simply aggregating Z_n and Z_g as the input for machining feature label classifier may not be the optimal strategy, as the potentially unequal importance of Z_n and Z_g. Therefore, we implement an inter-graph attention mechanism to effectively integrate Z_g into Z_n, resulting in a refined feature representation for each B-Rep face. Specifically, we compute two attention weights, att_{n} and att_{g}, through a linear transformation layer serving as the attention function f_{att}.

$$\left[att_{\mathrm{n}}, att_{\mathrm{g}}\right] = f_{att}(\left[Z_{\mathrm{n}}, Z_{\mathrm{g}}\right]) \tag{8}$$

Next, we normalize the attention weights using a softmax layer:

$$\begin{aligned} att_{\mathrm{n}} &= \frac{\exp(att_{\mathrm{n}})}{\exp(att_{\mathrm{n}} + att_{\mathrm{g}})} \\ att_{\mathrm{g}} &= \frac{\exp(att_{\mathrm{g}})}{\exp(att_{\mathrm{n}} + att_{\mathrm{g}})} \end{aligned} \tag{9}$$

Finally, based on the attention weights, we aggregate the local feature Z_n and the global feature Z_g to obtain the final representation of B-rep faces:

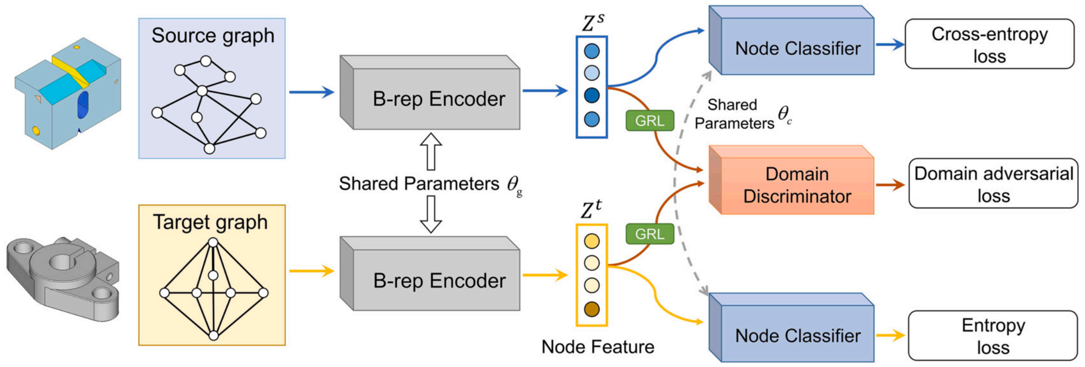


Fig. 5. The framework of adversarial domain adaptation.

$$Z = att_{\mathrm{n}} Z_n + att_{\mathrm{g}} Z_g \tag{10}$$

3.2.4. Node classifier

Machining feature recognition of B-rep models can be considered as a multi-class node classification problem. We employ a multi-layer perceptron (MLP) as the graph node classifier $f_c(z;\theta_c)$ to predict the specific machining feature category for each B-rep face. The node classifier consists of three fully connected layers: FC(256, 1024)→FC(1024, 256)→FC(256, K), where K is the number of feature categories in the dataset. Both fully connected layers do not have biases and include batch normalization and the LeakyReLU activation function. The input to the node classifier is denoted as $Z = [z_1^{\mathrm{T}}, z_2^{\mathrm{T}}, \cdots z_{|F|}^{\mathrm{T}}]$, and the output is $P = [p_1^{\mathrm{T}}, p_2^{\mathrm{T}}, \cdots p_{|F|}^{\mathrm{T}}]$, where $p_i \in \mathbb{R}^K$ represents the likelihood of B-rep face f_i belonging to each machining feature category. Finally, the argmax function is applied to calculate the predicted machining feature category $\hat{Y} = [\hat{y}_1, \hat{y}_2, \cdots \hat{y}_{|F|}]$ for B-rep faces.

The loss function of the graph node classifier is the cross-entropy between the predicted values and the ground truth labels:

$$\mathcal{L}_{label} = -\frac{1}{|F|}\sum_{i=1}^{|F|} y_i \log(\hat{y}_i) \tag{11}$$

3.3. Adversarial domain adaptation

Domain adaptation theory suggests that reducing the feature representation distribution gap between source and target domains can enhance the transferability of knowledge learned by neural networks from the source to the target domain (Ben-David et al., 2006, 2010). In our work, we define the B-rep graph of labeled CAD models as the source domain graph, $\mathcal{G}^s = (F^s, E^s)$, which includes a set of B-rep faces F^s, an edge set E^s, and the corresponding label matrix $Y^s \in \mathbb{R}^{N^s \times K}$ with N^s representing the number of B-rep faces in the graph $\mathcal{G}^s$, and K denoting the number of machining feature categories. Likewise, the B-rep graph of unlabeled CAD models is regarded as the target domain graph, $\mathcal{G}^t = (F^t, E^t)$. Both the source ($\mathcal{G}^s$) and target ($\mathcal{G}^t$) domain graph are input to the B-rep encoder $f_g(\mathcal{G};\theta_{\mathrm{g}})$, which is composed of the input feature encoder, graph transformer, and inter-graph attention module as described in Sec. 3.2. This process results in the feature representations of graph nodes $Z^s \in \mathbb{R}^{N^s \times d_E}$ and $Z^t \in {}^{N^t \times d_E}$.

We employ an adversarial training strategy, as illustrated in Fig. 5, to perform domain adaptation and achieve alignment of Z^s and Z^t in the feature space. This involves training a Domain Discriminator $f_d(z;\theta_d)$ to distinguish whether a B-rep face feature z comes from the source or target domain. It implies that the B-rep encoder f_g has successfully learned cross-domain invariant feature representations when $f_d(z;\theta_d)$ is unable to differentiate between the feature distributions of Z^s and Z^t. Such common features across domains can enhance the neural network's performance in node classification task within the target domain. The adversarial domain discrimination loss is defined as follows:

$$\mathcal{L}_{adv} = \sum_{i=0}^{N^s}\left(\log[f_d(z_i^s)]\right) + \sum_{i=0}^{N^t}\left(\log[1 - f_d(z_i^t)]\right) \tag{12}$$

where $f_d(z_i^s) \in \{0,1\}$ and $f_d(z_i^t) \in \{0,1\}$ are the domain labels predicted by the domain discriminator, indicating whether the example comes from the source domain or the target domain.

As the adversarial training process is a min-max game, the objective of domain discriminator f_d is to minimize the domain discrimination loss $\mathcal{L}_{adv}$, while the objective of B-rep encoder f_g is to maximize the same loss $\mathcal{L}_{adv}$. To facilitate this min-max game, we introduce a Gradient Reversal Layer (GRL) acting on Z^s and Z^t. The gradient reversal layer is defined as $Q(z) = z$, with the reversed gradient being $\frac{\partial Q(z)}{\partial z} = -\mathrm{I}$, where I represents the identity matrix. Thus, the adversarial domain discrimination loss can be rewritten as:

$$\mathcal{L}_{adv} = \sum_{i=0}^{N^s}\left(\log[f_d(Q(z_i^s))]\right) + \sum_{i=0}^{N^t}\left(\log[1 - f_d(Q(z_i^t))]\right) \tag{13}$$

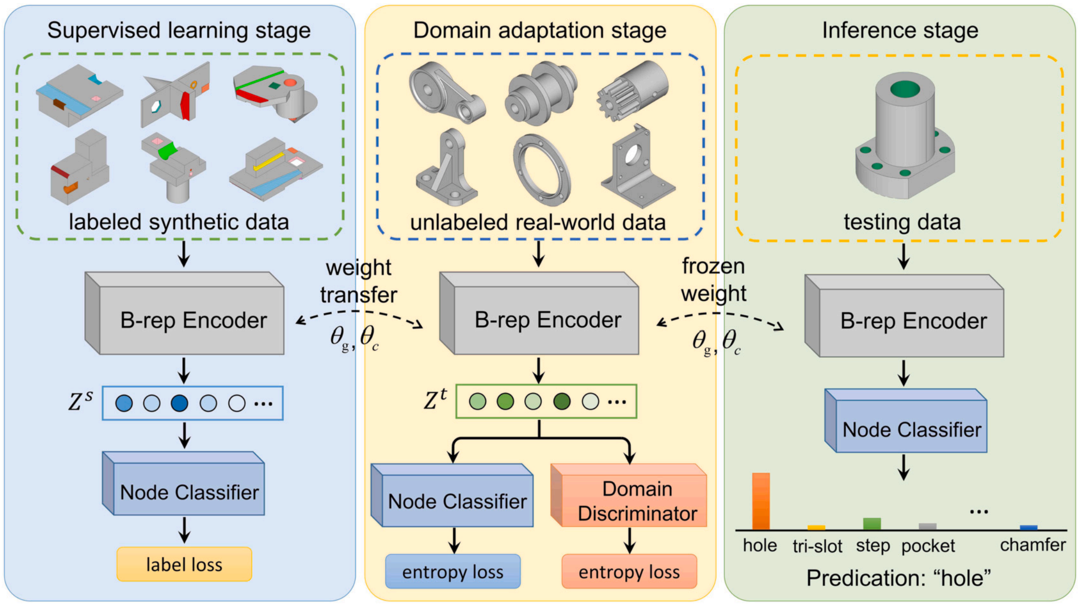


Fig. 6. The pipeline of the two-step training strategy under a transfer learning framework.

The overall optimization objective is to:

$$\min_{\theta_g,\theta_d} \mathcal{L}_{adv} \tag{14}$$

According to the prediction results of the Node Classifier $f_c(z;\theta_c)$, the Cross-entropy loss $\mathcal{L}_{label}$ for source domain data can be computed according to Equation (11). Since the target domain data lacks real labels, we employ entropy loss:

$$\mathcal{L}_{entropy} = -\frac{1}{N^t}\sum_{i=1}^{N^t} \hat{y}_i \log(\hat{y}_i) \tag{15}$$

where $\hat{y}_i$ is the predicted probability distribution of the i-th graph node in the target domain graph.

3.4. Overall algorithm

We implement and utilize the neural network BrepMFR following the pipeline illustrated in Fig. 6. The training process for BrepMFR is outlined in Algorithm 1, adopting a two-step training strategy under a transfer learning framework. In the supervised learning stage, a large amount of labeled synthetic CAD model data is used to train the B-rep Encoder $f_g(\mathcal{G};\theta_g)$ and Node Classifier $f_c(z;\theta_c)$. The optimization goal is defined as follows:

$$\min_{\theta_g,\theta_c} \mathcal{L}_{label} \tag{16}$$

In the domain adaptation stage, we incorporate the domain discriminator $f_d(z;\theta_d)$ into the pre-trained B-rep Encoder $f_g(\mathcal{G};\theta_g)$ and Node Classifier $f_c(z;\theta_c)$. The network is jointly trained using labeled synthetic CAD models as the source data and unlabeled real-world CAD models as the target data. The optimization objective is as follows:

$$\min_{\theta_g,\theta_c,\theta_d} \mathcal{L}_{label} + \alpha\mathcal{L}_{entropy} + \beta\mathcal{L}_{adv} \tag{17}$$

where α and β are trade-off parameters used to balance the two loss functions. In our work, $\alpha = 0.1$ and $\beta = 0.3$.

After two stages of training, in the inference stage, the parameters of the entire network are frozen in preparation for deployment. By inputting CAD models from the test set into the network, we obtain the predicted probability distribution across machining feature categories. This allows for identifying the machining feature category to which each face in the B-rep model belongs.

4. Synthetic dataset CADSynth

For deep neural network training, a large-scale datasets with appropriate labels is essential. To our knowledge, existing CAD model datasets such as DMU-Net (Dekhtiar et al., 2018), ABC (Koch et al., 2019), FabWave-3D (Angrish et al., 2019), and Fusion 360 Gallery (Willis et al., 2021) offer extensive collections of manually designed CAD models. However, these datasets lack labels

Algorithm 1: Training Algorithm of BrepMFR.

```
Input: Labeled samples batches $B^s = \{\mathcal{G}_i^s = (V_i^s, E_i^s), Y_i^s\}_{i=1}^{n_s}$ from source domain, unlabeled samples batches $B^t = \{\mathcal{G}_i^t = (V_i^t, E_i^t)\}_{i=1}^{n_t}$ from target domain.
Initialize parameters $\theta_g$ for B-rep Encoder $f_g$, $\theta_c$ for node classifier $f_c$, $\theta_d$ for domain discriminator $f_d$;
Stage-1
Pretrain $f_g$ and $f_c$ based on $\{\mathcal{G}_i^s = (V_i^s, E_i^s), Y_i^s\}_{i=1}^{n_s}$, update $\theta_g$ and $\theta_c$;
Stage-2
while not converge do
    for $i = 1$ to max iter do
        $Z_i^s \leftarrow f_g(\mathcal{G}_i^s, \theta_g)$, $Z_i^t \leftarrow f_g(\mathcal{G}_i^t, \theta_g)$;
        $\hat{Y}_i^s \leftarrow f_c(Z_i^s)$, $\hat{Y}_i^t \leftarrow f_c(Z_i^t)$;
        $\hat{D}_i^s \leftarrow f_d(Z_i^s)$, $\hat{D}_i^t \leftarrow f_d(Z_i^t)$;
        $\theta \leftarrow \{\theta_g, \theta_c, \theta_d\}$;
        $\theta \leftarrow \theta - lr \cdot \nabla \{\mathcal{L}_{label}(\hat{Y}_i^s, Y_i^s) + \alpha \cdot \mathcal{L}_{entropy}(\hat{Y}_i^t) + \beta \cdot \mathcal{L}_{adv}(\hat{D}_i^s, \hat{D}_i^t)\}$;
    end
end
```

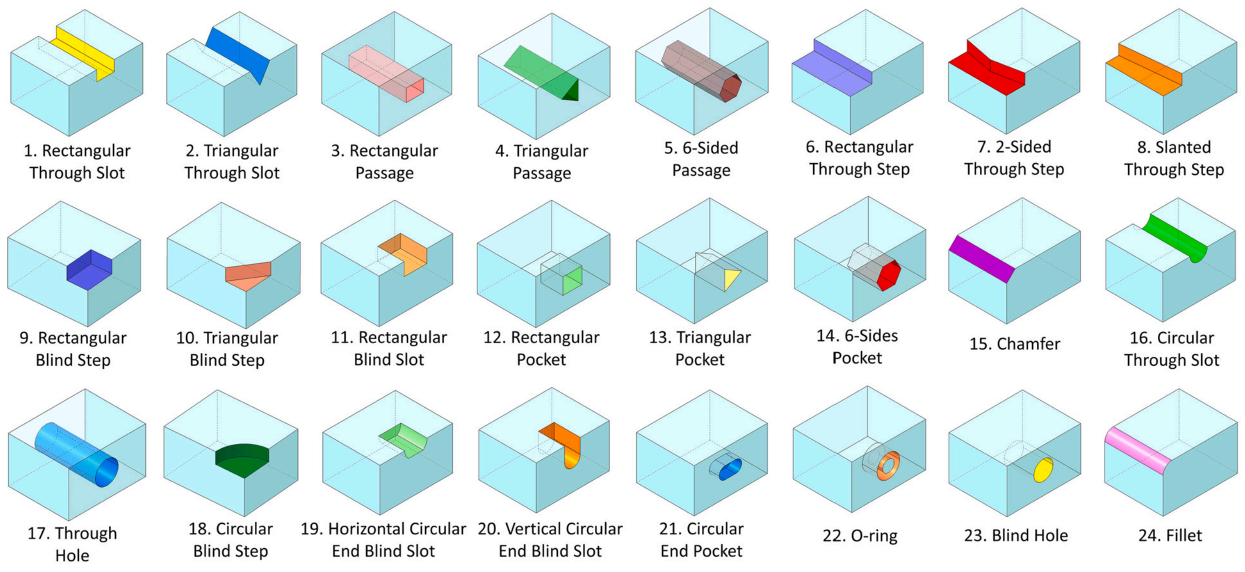


Fig. 7. Machining feature classes.

for machining feature recognition. Zhang et al. (2018) initially introduced the FeatureNet dataset, which incorporates 16 typical machining features. Subsequently, MFCAD (Cao et al., 2020) and MFCAD++ (Colligan et al., 2022) were established, which cover various machining features and cases with intersecting features. Nevertheless, it is noteworthy that the features in these datasets are attached to a single cubic entity, which does not accurately reflect the complexity of real-world mechanical parts. Building on this foundation, we establish a large-scale CAD dataset, CADSynth.

4.1. Machining feature type and label

The CADSynth dataset encompasses 24 machining features that are the same as those in MFCAD and MFAD++. The geometric shapes and indexes for each machining feature are illustrated in Fig. 7.

4.2. Dataset generation

CADSynth employs a random synthesis algorithm to combine various primitive elements (cuboid, prism, cylinder, cone, and sphere) to form the primary shape of a CAD model. This process is viewed as a state update, with the current geometric shape denoted by a global variable G, initially $G = \emptyset$. The first primitive is placed at the origin of the global coordinate system, with its size and orientation position randomly selected. Subsequently, a surface is selected from the current geometric shape based on certain rules (surface area as the selection weight) to serve as the reference plane. The placement and dimensions of subsequent primitives are guided by general spatial geometric constraints and rules, including coplanarity, concentricity, tangency, coplane, coedge and equal height. This approach imparts certain engineering characteristics and standardization to the final generated CAD model.

Regarding the incorporation of machining features into the primary shape, we adopted the methodology outlined by MFCAD++. This involves a three-step process: sketch plane selection, sketch outline definition, and extrusion depth determination. Following

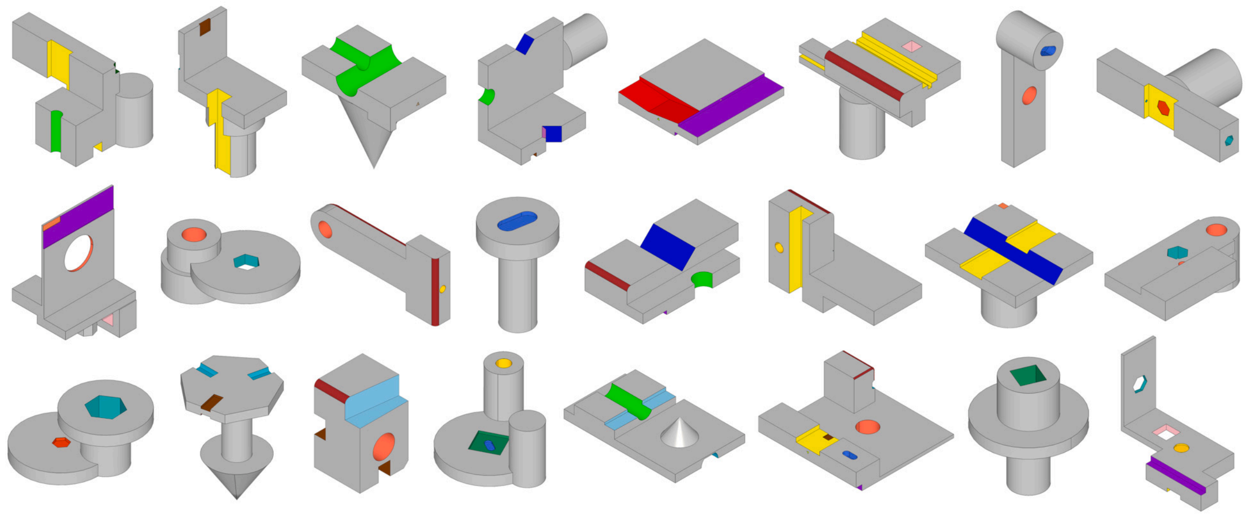


Fig. 8. General spatial geometric constraints.

these steps, a variety of machining features are methodically integrated into the CAD model's primary shape. To preserve the attributes of existing machining features, a specific order for adding features is essential. For example, a "through" feature should not cut a "blind" feature into a "non-blind" feature. We categorize the 24 machining features into the following five groups, adding features in the order: step→slot→hole→blind-slot→blind-hole→fillet.

In the end, we synthesized a dataset containing over 100,000 CAD designs. Fig. 8 displays samples of CAD models from our dataset.

5. Experimental results and discussion

In this section, we will experimentally evaluate the performance of our proposed BrepMFR in the following aspects:

1. Comparison with Existing Methods: How does the BrepMFR perform in comparison to existing methods?
2. Cross-Domain Performance and Domain Adaptation Impact: How does BrepMFR perform across different datasets, and how much performance improvement can domain adaptation bring?
3. Application to Real CAD Models: What is the performance of BrepMFR when applied to real-world CAD models?

5.1. Implementation detail

5.1.1. Network setting

The BrepMFR network is implemented in PyTorch and trained for max 200 epochs with a batch size of 64 on a NVIDIA RTX 3090 GPU. The AdamW optimizer with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and weight decay coefficient = 0.01 is employed during training. The initial learning rate is set to 0.001, and a warm-up stage of 50,00 steps is used. We incorporate the ReduceLROnPlateau scheduler to vary the learning rate when the loss function value no longer decreases.

5.1.2. Dataset

We conducted training and testing of BrepMFR on three different datasets: MFCAD, MFCAD++, and our CADSynth dataset. The CAD models in these datasets are represented in B-rep format, with machining feature labels for each face.

The MFCAD dataset consists of 15,488 CAD models, covering 16 machining feature types formed by planes. The dataset is split into training, validation, and testing sets by a 60/20/20% ratio.

MFCAD++ is an enhanced version of MFCAD, with an increased number of machining feature types to 24 and the introduction of more complex intersecting features. This dataset contains 59,655 CAD models split into training, validation, and testing sets by a 70/15/15% ratio.

The CADSynth dataset, generated using the automatic synthesis algorithm described in Sec. 4, includes 100,000 CAD designs across 24 machining feature categories. The dataset is randomly divided into training (80%), validation (10%), and test (10%) sets. To enhance training efficiency, the data is normalized, and the bounding box range of each CAD model along the x, y, and z axes is scaled to [-1,1].

Table 2
Machining feature recognition accuracy.

Dataset	Network	Accuracy	Per-class accuracy	mIoU
MFCAD	BrepMFR	99.99	**99.99**	99.99
	UV-Net	99.99	99.98	99.97
	BRepNet	99.99	99.98	99.98
	AGGNet	99.99	99.98	99.99
	Hierarchical CADNet	99.98	99.94	99.89
MFCAD++	BrepMFR	**99.76**	**99.64**	**99.30**
	UV-Net	99.49	99.10	98.35
	BRepNet	99.43	98.99	98.31
	AGGNet	99.57	99.21	99.08
	Hierarchical CADNet	98.78	97.55	95.56
CADSynth	BrepMFR	**99.96**	**99.92**	**99.83**
	UV-Net	99.74	99.54	99.02
	BRepNet	99.67	99.27	98.77
	AGGNet	99.80	99.67	99.25
	Hierarchical CADNet	99.53	98.72	97.96

5.1.3. Evaluation metrics

To evaluate the performance of the BrepMFR network, we refer to related work and adopt the following three evaluation metrics. The primary metric evaluates the accuracy of machining feature classification on individual faces in B-rep models. This accuracy is calculated as the proportion of correctly classified B-rep faces out of the total B-rep faces within the CAD models:

$$Accuracy = \frac{|Correct\ Faces|}{|Total\ Faces|} \tag{18}$$

The second metric is per-class accuracy, which represents the average accuracy across all machining feature classes:

$$Per-class\ accuracy = \frac{1}{K}\sum_{i=1}^{K}\frac{|y : y = i \cap \hat{y} : \hat{y} = i|}{|y : y = i|} \tag{19}$$

The third metric is the mean Intersection Over Union (mIoU), commonly used to evaluate the similarity and diversity between two sample sets. In our machining feature recognition task, mIoU is determined as the average proportion of correctly classified B-rep faces over the union of B-rep faces that have either the actual or predicted labels within the same category:

$$mIoU = \frac{1}{K}\sum_{i=1}^{K}\frac{|y : y = i \cap \hat{y} : \hat{y} = i|}{|y : y = i \cup \hat{y} : \hat{y} = i|} \tag{20}$$

5.2. Comparative results

We evaluated the BrepMFR network on MFCAD, MFCAD++, and CADSynth datasets, comparing its performance with existing deep learning methods for machining feature recognition, including UV-Net (Jayaraman et al., 2021), BRepNet (Lambourne et al., 2021), AAGNet (Wu et al., 2024) and Hierarchical CADNet (Colligan et al., 2022). The experimental results are presented in Table 2, showcasing that our proposed BrepMFR network demonstrates superior or comparable performance across all three datasets. On the MFCAD dataset, BrepMFR achieves near-perfect performance with Accuracy, Per-class accuracy, and mIoU all reaching 99.99%. On the MFCAD++ and CADSynth datasets, where the geometric complexity of CAD models increases, the performance slightly decreases but maintains above 99% across the evaluation metrics. Fig. 9 presents the accuracy per feature class, where the x-axis denotes the predicted feature type index, the y-axis denotes the ground truth feature type index. The diagonal of the confusion matrix represents accurate predictions, and it can be seen that more than 99% accuracy is achieved for most categories. Fig. 10 displays randomly selected examples from the test set, highlighting the predicted machining features through different colors of faces.

5.3. Domain adaptation results

To quantitatively validate the effectiveness of domain adaptation methods in enhancing the cross-datasets generalization capabilities for machining feature recognition tasks, we trained the BrepMFR network on a source dataset and tested it on different target domain datasets. Specifically, we conducted machining feature recognition across three domains through four transfer learning tasks, including MFCAD→CADSynth, MFCAD→MFCAD++, MFCAD++ →CADSynth, and CADSynth→MFCAD++, with results presented in Table 3. The results reveal a notable performance decline when the pre-trained network is directly applied to the target domain datasets without any domain adaptation. Notably, networks trained on the MFCAD dataset exhibited lower cross-datasets feature recognition accuracy, achieving only 22.91% for CADSynth and 54.80% for MFCAD++. Similarly, networks trained on the MF-CAD++ dataset and directly applied to the CADSynth dataset saw a decrease in feature recognition accuracy to 61.82%. However,

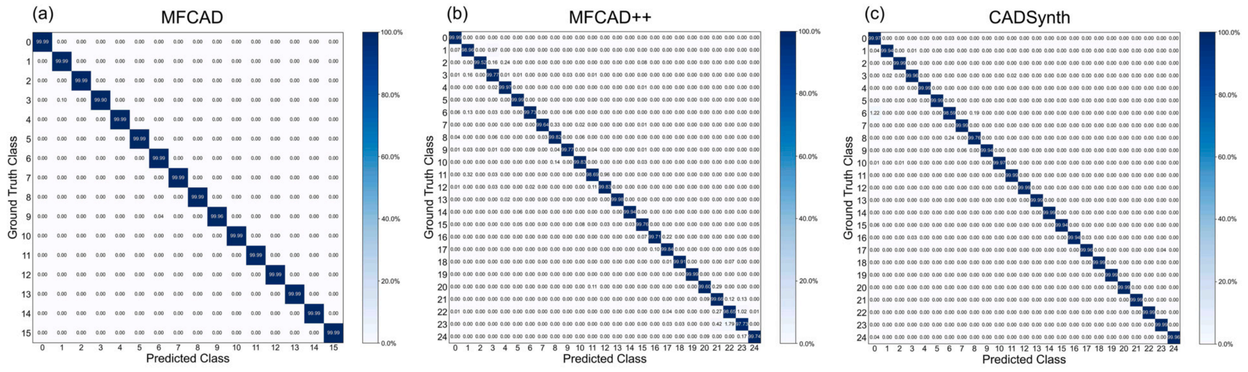


Fig. 9. Confusion matrix for MFCAD, MFCAD++ and CADSynth datasets.

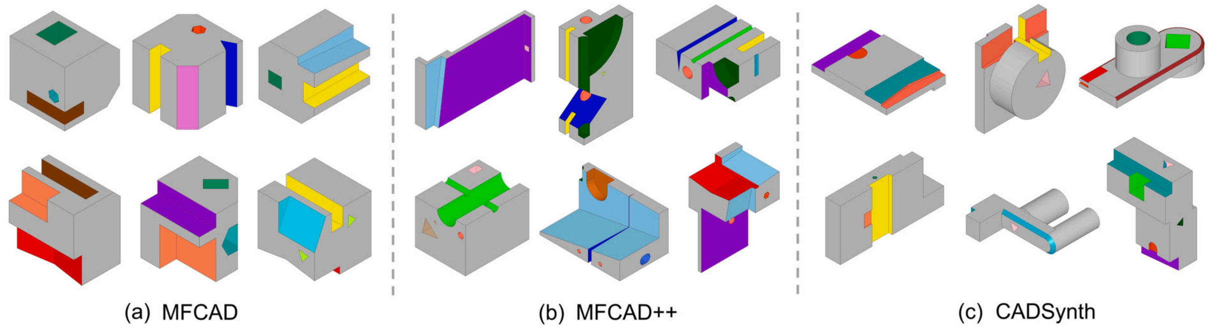


Fig. 10. Examples for machining feature recognition results on test set.

Table 3
Machining feature recognition accuracy of BrepMFR on target domains. The first row corresponds to the accuracy when trained on the source domain dataset, and the last row corresponds to the accuracy when trained on the target domain dataset with known class labels. The middle row indicates the accuracy following domain adaptation, with the extent of the gap covered between the lower and upper bounds presented in brackets.

Method	Source / Target	MFCAD / CADSynth	MFCAD / MFCAD++	MFCAD++ / CADSynth	CADSynth / MFCAD++
Source Only (baseline)		22.91	54.80	61.82	81.50
Domain Adaptation		85.71 (81.5%)	87.08 (71.8%)	92.74 (81.1%)	90.32 (48.3%)
Train on Target		99.96	99.76	99.96	99.76

networks trained on our large-scale synthesized dataset CADSynth, and tested on the MFCAD++ dataset, while experiencing a performance drop, still maintained a high accuracy rate of 81.50%. This suggests that the CADSynth dataset, reflecting a richer variety of geometric shapes and complex combinations of machining features, enhances the network's generalization capabilities.

Next, we employed adversarial domain adaptation methods to align features between the source domain dataset and target domain dataset. It can be observed that after domain adaptation, the performance of the BrepMFR on target domain datasets has significantly improved. This indicates that our proposed domain adaptation approach enhances the network's generalization ability, supporting machining feature recognition in CAD models across different domain datasets.

5.4. Ablation study

Ablation studies explore the impact of removing or replacing a specific component in the neural network on the overall system performance. In this section, we designed the following ablation experiments to evaluate the impact of different factors on the machining feature recognition and domain adaptation performance. These ablation studies were conducted on the CADSynth dataset.

5.4.1. Ablation study on input features of geometric

The geometric features inputted into BrepMFR network consist of the fundamental attributes, as well as UV-grid points parameters of B-rep faces and edges, which respectively describe the overall geometric properties and precise geometric shapes of the B-rep model. In this ablation study, we established the model with full input features as the baseline and then removed each of these three types of input features to create ablated models. Results presented in Table 4 indicate that the removal of any input features leads to a decrease in feature recognition accuracy. Among them, the exclusion of face attributes results in the greatest performance decline, implying its greater importance compared to other input features. It is worth noting that, compared to the baseline, all ablated models

Table 4
Ablation study on input features of geometric. The bold numbers are the best results for each evaluation metrics, and the values in brackets are the changes relative to baseline.

Input	Accuracy	Per-class accuracy	mIoU
Full (baseline)	**99.96**	**99.92**	**99.83**
No Face Attr.	99.85 (-0.11)	99.65 (-0.27)	99.08 (-0.75)
No Edge Attr.	99.91 (-0.05)	99.82 (-0.10)	99.62 (-0.21)
No UV-grid	99.89 (-0.07)	99.75 (-0.17)	99.55 (-0.28)

Table 5
Ablation study on input features of proximity. The bold numbers are the best results for each evaluation metrics, and the values in brackets are the changes relative to baseline.

Input	Accuracy	Per-class accuracy	mIoU
No A1/2/3 (baseline)	97.33	95.67	91.50
A1	99.24 (+1.91)	98.07 (+2.40)	96.24 (+4.74)
A1 + A2	99.60 (+2.27)	99.39 (+3.72)	98.39 (+6.89)
A1 + A2 + A3	**99.96** (+2.63)	**99.92** (+4.25)	**99.83** (+8.33)

Table 6
Ablation study on domain adaptation. The bold numbers are the best results for each evaluation metrics, and the values in brackets are the changes relative to baseline.

Method	Accuracy	Per-class accuracy	mIoU
$\mathcal{L}_{label}$ (baseline)	61.82	85.53	61.64
$\mathcal{L}_{label} + \mathcal{L}_{entropy}$	64.69 (+2.87)	90.68 (+5.15)	75.44 (+13.80)
$\mathcal{L}_{label} + \mathcal{L}_{adv}$	91.75 (+29.93)	95.16 (+9.63)	81.23 (+19.59)
$\mathcal{L}_{label} + \mathcal{L}_{adv} + \mathcal{L}_{entropy}$	**92.74** (+30.92)	**95.70** (+10.17)	**83.53** (+21.89)

show relatively minor performance decreases (approximately 0.1% decrease in Accuracy), indicating the insensitivity of the feature recognition task to input geometric information. We believe that this comprehensive, even "redundant" input geometric information is beneficial for enhancing the robustness and generality of neural network, thereby enabling its application in more complex tasks such as model classification, model retrieval, and 3D reconstruction.

5.4.2. *Ablation study on input features of proximity*

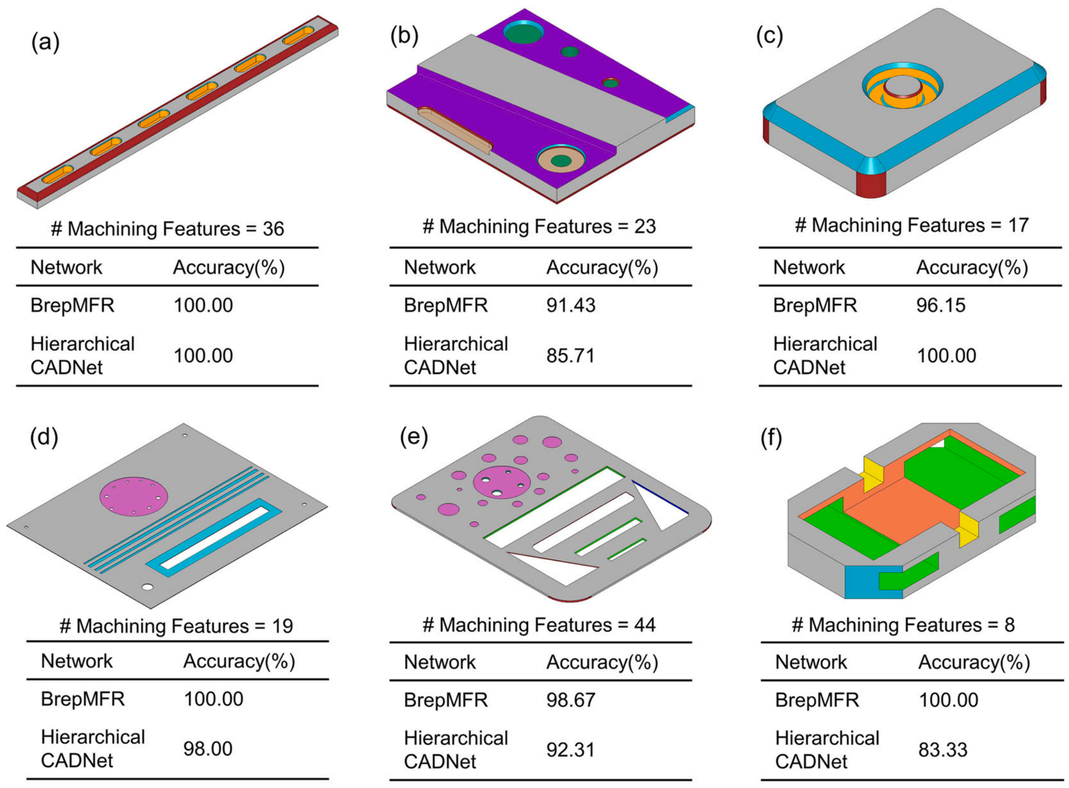
In the BrepMFR network, we integrated three matrices, A1, A2, and A3, to characterize the adjacency relationships between B-rep faces, their spatial positions, and the faces-edges connections. To evaluate the effectiveness of this design, we utilized a model without the proximity matrices A1, A2, and A3 as the baseline (consistent with the standard Transformer architecture), and observe substantial performance gaps compared to the state-of-the-art (SOTA), as indicated in Table 5. The gradual incorporation of the proximity matrices A1, A2, and A3 led to a stepwise enhancement in the network's performance, with increases of 2.63%, 4.25%, and 8.33% in the evaluation metrics. This demonstrates that matrices A1, A2, and A3 are crucial in the BrepMFR, facilitating the neural network model's accurate understanding of complex 3D solid models.

5.4.3. *Ablation study on domain adaptation*

To implement adversarial domain adaptation, we integrated two terms into the joint training loss function: the entropy loss $\mathcal{L}_{entropy}$, and the adversarial domain discrimination loss $\mathcal{L}_{adv}$. Ablation experiments on the transfer learning task MFCAD++ →CADSynth were conducted to evaluated the contributions of these losses. We used the model trained only with $\mathcal{L}_{label}$ as the baseline, then incrementally added $\mathcal{L}_{entropy}$ and $\mathcal{L}_{adv}$ for domain adaptation, with results shown in Table 6. We observed that solely incorporating the $\mathcal{L}_{entropy}$ to the loss function results in a mere 2.87% accuracy improvement on the target domain data over the baseline. Incorporating $\mathcal{L}_{adv}$ significantly improved BrepMFR's performance on the target domain dataset. Finally, combining $\mathcal{L}_{entropy}$ and $\mathcal{L}_{adv}$ yielded optimal cross-domain machining feature recognition, achieving increases of 30.92%, 10.17%, and 21.89% in accuracy, per-class accuracy, and mIoU.

5.5. *Case study and discussion*

Existing methods such as Hierarchical CADNet (Colligan et al., 2022) can also achieve feature recognition. Therefore, we compare BrepMFR with them through several test cases to demonstrate the advantages of our proposed approach. The comparative case study with Hierarchical CADNet is shown in Fig. 11, utilizing test cases from the public dataset associated with the Hierarchical CADNet research. In test cases (a)-(c), with a larger number of fillet and chamfer features, BrepMFR failed to detect several chamfer features in the latter two cases. For test cases (d) and (e), with large aspect ratios and thin sheet shapes, and test case (f), which exhibits



(a) # Machining Features = 36

Network	Accuracy(%)
BrepMFR	100.00
Hierarchical CADNet	100.00

(b) # Machining Features = 23

Network	Accuracy(%)
BrepMFR	91.43
Hierarchical CADNet	85.71

(c) # Machining Features = 17

Network	Accuracy(%)
BrepMFR	96.15
Hierarchical CADNet	100.00

(d) # Machining Features = 19

Network	Accuracy(%)
BrepMFR	100.00
Hierarchical CADNet	98.00

(e) # Machining Features = 44

Network	Accuracy(%)
BrepMFR	98.67
Hierarchical CADNet	92.31

(f) # Machining Features = 8

Network	Accuracy(%)
BrepMFR	100.00
Hierarchical CADNet	83.33

Fig. 11. The comparative case study with Hierarchical CADNet.

highly intersecting features, BrepMFR outperforms Hierarchical CADNet. These results indicate that the BrepMFR network trained on the CADSynth dataset demonstrates superior generalization to new CAD models not seen in the training set.

To better integrate with downstream applications such as CAPP and CAM, post-processing of BrepMFR recognition results is necessary. The objective is to extract individual machining features' components (B-rep faces and edges) along with their design and manufacturing parameters (e.g., radius, axis, length, width). We design a rule-based post-processing algorithm following the boundary patterns for machining features proposed by Sunil and Pande (2009), along with B-rep topology query interfaces provided by Open CASCADE Technology (OCCT). This algorithm groups faces belonging to the same machining feature instance into sets; in cases of intersection, sets are further split or merged based on predefined rules.

To evaluate the performance of BrepMFR on real-world CAD models, we chose several CAD models from the ABC dataset for machining feature recognition. The predicted results of BrepMFR and the separated machining features through post-processing method are illustrated in Fig. 12. For example, in case (b), BrepMFR accurately identified the faces belonging to rectangular through slots (highlighted in yellow). In post-processing, we grouped these 27 faces into 6 machining feature instances: typically, each through slot comprises 3 connected faces, including a bottom face and two parallel side faces; specifically, there is one through slot segmented into 4 parts by three other slots. By testing the parallelism and coplanarity between faces, we classified these 4 parts into the same through slot instance. Similarly, in case (a), the pocket is defined as a set of faces, where a base face (fully accessible for machining) is concavely attached to a number of side faces. These results demonstrate that BrepMFR accurately identifies all isolated and intersecting features, indicating its capability to handle typical machining features in engineering parts.

6. Conclusion

In this paper, we introduced BrepMFR, a novel deep neural network designed for machining feature recognition in B-rep models. BrepMFR leverages the structural characteristics of B-rep data, incorporating improvements of both Transformer and Graphormer to effectively utilize local geometric shape information and topological relationships in 3D B-rep models. In summary, the novelty and contribution of this paper are as follows:

1. We proposed a novel deep learning method, BrepMFR, specifically designed for B-rep models, capable of performing machining feature recognition tasks. BrepMFR outperforms existing learning-based methods.
2. We established a synthetic dataset, CADSynth, comprising 100,000 CAD designs. This dataset is more representative of engineering realities compared to existing synthetic CAD datasets.

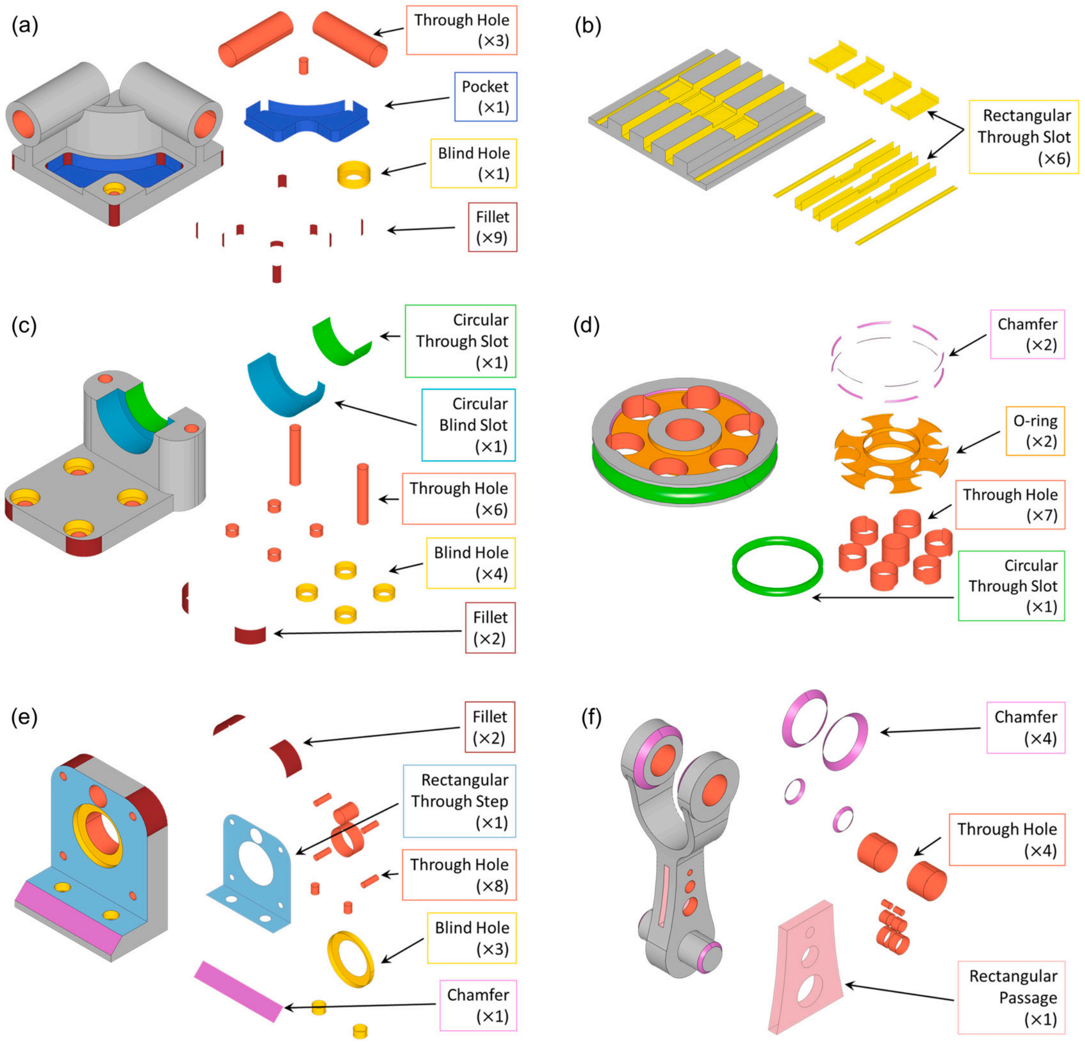


Fig. 12. Machining feature recognition results for real-world CAD models from ABC dataset. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

3. We implemented a domain adaptation framework capable of effectively enhancing the cross-domain adaptability of deep neural networks. This framework enhances the application capabilities of BrepMFR in engineering scenarios.

However, there are still limitations and challenges that need to be addressed in future work. Our current work is limited to 24 types of basic machining features. Subsequent efforts should focus on diversifying the range of features and handling complex, intersection scenarios. Additionally, constructing a large-scale open-source real CAD dataset comprising labeled machining features is an important future endeavor. This will enhance the generalization capabilities of our approach and drive advancements in associated research domains.

CRediT authorship contribution statement

Shuming Zhang: Writing – original draft, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Zhidong Guan:** Writing – review & editing, Project administration, Funding acquisition, Conceptualization. **Hao Jiang:** Writing – review & editing, Writing – original draft, Software, Investigation, Data curation. **Xiaodong Wang:** Writing – review & editing, Validation, Supervision, Project administration. **Pingan Tan:** Writing – review & editing, Validation, Investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was funded by National Key Project of China (Grant No. GJXM92579).

References

Angrish, A., Craver, B., Starly, B., 2019. "fabsearch": a 3d cad model-based search engine for sourcing manufacturing services. J. Comput. Inf. Sci. Eng. 19, 041006. https://doi.org/10.1115/1.4043211.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W., 2010. A theory of learning from different domains. Mach. Learn. 79, 151–175. https://doi.org/10.1007/s10994-009-5152-4.

Ben-David, S., Blitzer, J., Crammer, K., Pereira, F., 2006. Analysis of representations for domain adaptation. In: Advances in Neural Information Processing Systems.

Brousseau, E., Dimov, S., Setchi, R., 2008. Knowledge acquisition techniques for feature recognition in cad models. J. Intell. Manuf. 19, 21–32. https://doi.org/10.1007/s10845-007-0043-7.

Cao, W., Robinson, T., Hua, Y., Boussuge, F., Colligan, A.R., Pan, W., 2020. Graph representation of 3d cad models for machining feature recognition with deep learning. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers. p. V11AT11A003, https://doi.org/10.1115/DETC2020-22355.

Chan, A., Case, K., 1994. Process planning by recognizing and learning machining features. Int. J. Comput. Integr. Manuf. 7, 77–99. https://doi.org/10.1080/09511929408944597.

Colligan, A.R., Robinson, T.T., Nolan, D.C., Hua, Y., Cao, W., 2022. Hierarchical cadnet: learning from b-reps for machining feature recognition. Comput. Aided Des. 147, 103226. https://doi.org/10.1016/j.cad.2022.103226.

Dekhtiar, J., Durupt, A., Bricogne, M., Eynard, B., Rowson, H., Kiritsis, D., 2018. Deep learning for big data applications in cad and plm–research review, opportunities and case study. Comput. Ind. 100, 227–243. https://doi.org/10.1016/j.compind.2018.04.005.

Donaldson, I.A., Corney, J.R., 1993. Rule-based feature recognition for 2· 5d machined components. Int. J. Comput. Integr. Manuf. 6, 51–64. https://doi.org/10.1080/09511929308944555.

Dong, J., Vijayan, S., 1997. Features extraction with the consideration of manufacturing processes. Int. J. Prod. Res. 35, 2135–2156. https://doi.org/10.1080/002075497194778.

Fu, M., Ong, S.K., Lu, W.F., Lee, I., Nee, A.Y., 2003. An approach to identify design and manufacturing features from a data exchanged part model. Comput. Aided Des. 35, 979–993. https://doi.org/10.1016/S0010-4485(02)00160-4.

Ganin, Y., Lempitsky, V., 2015. Unsupervised domain adaptation by backpropagation. In: Proceedings of the 32nd International Conference on Machine Learning.

Gao, S., Shah, J.J., 1998. Automatic recognition of interacting machining features based on minimal condition subgraph. Comput. Aided Des. 30, 727–739. https://doi.org/10.1016/S0010-4485(98)00033-5.

Geng, W., Chen, Z., He, K., Wu, Y., 2016. Feature recognition and volume generation of uncut regions for electrical discharge machining. Adv. Eng. Softw. 91, 51–62. https://doi.org/10.1016/j.advengsoft.2015.10.005.

Han, J., Regli, W.C., Brooks, S., 1997. Hint-based reasoning for feature recognition: status report. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers. p. V005T32A020, https://doi.org/10.1115/DETC97/CIE-4485.

Henderson, M.R., 1984. Extraction of Feature Information from Three-Dimensional CAD Data. Purdue University.

Huang, Z., Yip-Hoi, D., 2002. High-level feature recognition using feature relationship graphs. Comput. Aided Des. 34, 561–582. https://doi.org/10.1016/S0010-4485(01)00128-2.

Jayaraman, P.K., Sanghi, A., Lambourne, J.G., Willis, K.D., Davies, T., Shayani, H., Morris, N., 2021. Uv-net: learning from boundary representations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11703–11712.

Jia, J.L., Zhang, S.W., Cao, Y.R., Qi, X.L., WeZhu, 2023. Machining feature recognition method based on improved mesh neural network. Iran. J. Sci. Technol. Trans. Mech. Eng. 47, 2045–2058. https://doi.org/10.1007/s40997-023-00610-8.

Joshi, S., Chang, T.C., 1988. Graph-based heuristics for recognition of machined features from a 3d solid model. Comput. Aided Des. 20, 58–66. https://doi.org/10.1016/0010-4485(88)90050-4.

Kim, Y.S., Wilde, D., 1992. A convergent convex decomposition of polyhedral objects. J. Mech. Des. 114, 468–476. https://doi.org/10.1115/1.2926575.

Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., Panozzo, D., 2019. 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.

Lambourne, J.G., Willis, K.D., Jayaraman, P.K., Sanghi, A., Meltzer, P., Shayani, H., 2021. 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.

Lee, C.Y., Batra, T., Baig, M.H., Ulbricht, D., 2019. Sliced Wasserstein discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10285–10295.

Lee, H., Lee, J., Kim, H., Mun, D., 2021. Dataset and method for deep learning-based reconstruction of 3d cad models containing machining features for mechanical parts. J. Comput. Des. Eng. 9, 114–127. https://doi.org/10.1093/jcde/qwab072.

Lee, J., Yeo, C., Cheon, S.U., Park, J.H., Mun, D., 2023. Brepgat: graph neural network to segment machining feature faces in a b-rep model. J. Comput. Des. Eng. 10, 2384–2400. https://doi.org/10.1093/jcde/qwad106.

Lei, R., Wu, H., Peng, Y., 2022. Mfpointnet: a point cloud-based neural network using selective downsampling layer for machining feature recognition. Dianji Yu Kongzhi Xuebao 10, 1165. https://doi.org/10.3390/machines10121165.

Liu, S.C., Gonzalez, M., Chen, J.G., 1996. Development of an automatic part feature extraction and classification system taking cad data as input. Comput. Ind. 29, 137–150. https://doi.org/10.1016/0166-3615(95)00081-X.

Long, M., Zhu, H., Wang, J., Jordan, M.I., 2017. Deep transfer learning with joint adaptation networks. In: Proceedings of the 34th International Conference on Machine Learning. PMLR, pp. 2208–2217.

Ning, F., Shi, Y., Cai, M., Xu, W., 2023. Part machining feature recognition based on a deep learning method. J. Intell. Manuf. 34, 809–821. https://doi.org/10.1007/s10845-021-01827-7.

Pan, S.J., Yang, Q., 2009. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22, 1345–1359. https://doi.org/10.1109/TKDE.2009.191.

Peddireddy, D., Fu, X., Shankar, A., Wang, H., Joung, B.G., Aggarwal, V., Sutherland, J.W., Jun, M.B.G., 2021. Identifying manufacturability and machining processes using deep 3d convolutional networks. J. Manuf. Process. 64, 1336–1348. https://doi.org/10.1016/j.jmapro.2021.02.034.

Prabhakar, S., Henderson, M.R., 1992. Automatic form-feature recognition using neural-network-based techniques on boundary representations of solid models. Comput. Aided Des. 24, 381–393. https://doi.org/10.1016/0010-4485(92)90064-H.

Rea, H., Sung, R., Corney, J., Clark, D., Taylor, N., 2005. Interpreting three-dimensional shape distributions. Proc. Inst. Mech. Eng., Part C, J. Mech. Eng. Sci. 219, 553–566. https://doi.org/10.1243/095440605X31427.

Roy, B., et al., 1959. Transitivité et connexité. C. R. Acad. Sci. Paris 249, 182.

Shi, P., Qi, Q., Qin, Y., Scott, P.J., Jiang, X., 2022. Highly interacting machining feature recognition via small sample learning. Robot. Comput.-Integr. Manuf. 73, 102260. https://doi.org/10.1016/j.rcim.2021.102260.

Shi, Y., Zhang, Y., Harik, R., 2020. Manufacturing feature recognition with a 2d convolutional neural network. CIRP J. Manuf. Sci. Technol. 30, 36–57. https://doi.org/10.1016/j.cirpj.2020.04.001.

Sunil, V., Pande, S., 2009. Automatic recognition of machining features using artificial neural networks. Int. J. Adv. Manuf. Technol. 41, 932–947. https://doi.org/10.1007/s00170-008-1536-z.

Takaishi, I., Kanai, S., Date, H., Takashima, H., 2020. Free-form feature classification for finite element meshing based on shape descriptors and machine learning. Comput-Aided Des. Appl. 17, 1049–1066.

Tang, K., Woo, T., 1991. Algorithmic aspects of alternating sum of volumes. part 1: Data structure and difference operation. Comput. Aided Des. 23, 357–366. https://doi.org/10.1016/0010-4485(91)90029-V.

Tzeng, E., Hoffman, J., Saenko, K., Darrell, T., 2017. Adversarial discriminative domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7167–7176.

Vandenbrande, J.H., Requicha, A.A., 1993. Spatial reasoning for the automatic recognition of machinable features in solid models. IEEE Trans. Pattern Anal. Mach. Intell. 15, 1269–1285. https://doi.org/10.1109/34.250845.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 30.

Vosniakos, G., Davies, B., 1993. A shape feature recognition framework and its application to holes in prismatic parts. Int. J. Adv. Manuf. Technol. 8, 345–351. https://doi.org/10.1007/BF01751095.

Wang, M., Deng, W., 2018. Deep visual domain adaptation: a survey. Neurocomputing 312, 135–153. https://doi.org/10.1016/j.neucom.2018.05.083.

Warshall, S., 1962. A theorem on Boolean matrices. J. ACM 9, 11–12.

Willis, K.D., Pu, Y., Luo, J., Chu, H., Du, T., Lambourne, J.G., Solar-Lezama, A., Matusik, W., 2021. Fusion 360 gallery: a dataset and environment for programmatic cad construction from human design sequences. ACM Trans. Graph. 40. https://doi.org/10.1145/3450626.3459818.

Woo, Y., 1982. Feature extraction by volume decomposition. In: Proceedings of Conference on CAD/CAM in Mechanical Engineering, pp. 76–94.

Woo, Y., Sakurai, H., 2002. Recognition of maximal features by volume decomposition. Comput. Aided Des. 34, 195–207. https://doi.org/10.1016/S0010-4485(01)00080-X.

Wu, H., Lei, R., Peng, Y., Gao, L., 2024. Aagnet: a graph neural network towards multi-task machining feature recognition. Robot. Comput.-Integr. Manuf. 86, 102661. https://doi.org/10.1016/j.rcim.2023.102661.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., Liu, T.Y., 2021. Do transformers really perform badly for graph representation? Adv. Neural Inf. Process. Syst. 34, 28877–28888.

Yuen, C., Venuvinod, P., 1999. Geometric feature recognition: coping with the complexity and infinite variety of features. Int. J. Comput. Integr. Manuf. 12, 439–452. https://doi.org/10.1080/095119299130173.

Zehtaban, L., Roller, D., 2016. Automated rule-based system for opitz feature recognition and code generation from step. Comput-Aided Des. Appl. 13, 309–319. https://doi.org/10.1080/16864360.2015.1114388.

Zhang, C., Zhao, Q., Wang, Y., 2020. Transferable attention networks for adversarial domain adaptation. Inf. Sci. 539, 422–433. https://doi.org/10.1016/j.ins.2020.06.016.

Zhang, Y., Luo, X., Zhang, B., Zhang, S., 2017. Semantic approach to the automatic recognition of machining features. Int. J. Adv. Manuf. Technol. 89, 417–437. https://doi.org/10.1007/s00170-016-9056-8.

Zhang, Z., Jaiswal, P., Rai, R., 2018. Featurenet: machining feature recognition based on 3d convolution neural network. Comput. Aided Des. 101, 12–22. https://doi.org/10.1016/j.cad.2018.03.006.

Zhao, J., Li, C., Wen, Q., Wang, Y., Liu, Y., Sun, H., Xie, X., Ye, Y., 2021. Gophormer: ego-graph transformer for node classification. https://doi.org/10.48550/arXiv.2110.13094.

Zhuang, F., Cheng, X., Luo, P., Pan, S.J., He, Q., 2015. Supervised representation learning: transfer learning with deep autoencoders. In: Proceedings of the 24th International Conference on Artificial Intelligence, pp. 4119–4125.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., He, Q., 2020. A comprehensive survey on transfer learning. Proc. IEEE 109, 43–76. https://doi.org/10.1109/JPROC.2020.3004555.