Journal of Computational Design and Engineering, 2023, 10, 2384–2400
DOI: 10.1093/jcde/qwad106
Advance access publication date: 28 November 2023
Research Article

BRepGAT: Graph neural network to segment machining feature faces in a B-rep model

Jinwon Lee[1], Changmo Yeo[2], Sang-Uk Cheon[2], Jun Hwan Park[2] and Duhwan Mun[2,*]

[1]Department of Industrial & Management Engineering, Gangneung-Wonju National University, 150 Namwon-ro, Heungeop-myeon, Wonju, Gangwon-do 26403, Republic of Korea
[2]School of Mechanical Engineering, Korea University, 145 Anam-ro, Seongbuk-gu, Seoul 02841, Republic of Korea
*Correspondence: dhmun@korea.ac.kr

Abstract

In recent years, there have been many studies using artificial intelligence to recognize machining features in three-dimensional models in the computer-aided design (CAD)/computer-aided manufacturing field. Most of these studies converted the original CAD data into images, point clouds, or voxels for recognition. This led to information loss during the conversion process, resulting in decreased recognition accuracy. In this paper, we propose a novel deep learning model called the boundary representation graph attention network (BRepGAT) to segment faces in an original boundary representation (B-rep) model containing machining features. We define descriptors that represent information about the faces and edges of the B-rep model from the perspective of feature recognition. These descriptors are extracted from the B-rep model and transformed into homogeneous graph data, which are then passed to graph networks. BRepGAT recognizes machining features from the graph data input. Our experimental results using the MFCAD18++ dataset showed that BRepGAT achieved state-of-the-art recognition accuracy (99.1%). Furthermore, BRepGAT showed relatively robust performance on other datasets besides MFCAD18++.

Keywords: graph neural networks, deep learning, boundary representation, machining feature, face segmentation

1. Introduction

Product lifecycle management (PLM) refers to the activities managing all stages of a product's lifecycle, from opportunity identification, design, production, distribution, service, to disposal (Son *et al.*, 2022). Each stage of PLM is organically interconnected, necessitating smooth data exchange between the stages. During the product design process, engineers use mechanical computer-aided design (CAD) systems to design a product's three-dimensional (3D) geometry. The data structure of mechanical CAD systems combines a hybrid structure, consisting of boundary representation (B-rep) model, which represents 3D geometry, and procedural model, which represents the modeling history.

In the process of exchanging CAD models generated by mechanical CAD systems between various enterprises, intellectual property security concerns and stability issues during the data exchange process frequently necessitate the exclusion of procedural models. As a result, only the B-rep model data are typically transferred, stored in neutral formats such as ISO 10303, commonly referred to as STEP (Hwang *et al.*, 2009; Kim, Mun, *et al.*, 2011). Although B-rep models are advantageous for visualizing 3D product shape, they have the drawback of lacking machining feature information within the model, which is necessary for computer-aided manufacturing (CAM) systems used in product production (Lee *et al.*, 2022b). Consequently, a computer-aided process planning (CAPP) stage exists between CAD and CAM systems. CAPP involves identifying machining features from the CAD model and determining the sequence and methods for processing these features. Machining feature recognition is a critical component of the CAPP process.

Machining feature recognition has been an active area of research for several decades (Lee *et al.*, 2022a). In recent years, the CAD/CAM domain has seen an increasing number of studies focused on utilizing artificial intelligence (AI) for recognizing feature patterns in CAD models as the advancements in AI (Kim, Kim, *et al.*, 2022). However, the majority of these studies have conducted recognition tasks after converting the original CAD models into alternative representations such as images, point clouds, or voxels (Liang *et al.*, 2022; Zhang, He, *et al.*, 2020). Consequently, the conversion process has led to information loss, resulting in decreased recognition accuracy.

In some studies, researchers attempted to segment machining feature faces from CAD models by extracting meaningful feature data that affect recognition and processing it through a deep neural network (DNN). However, since DNNs perform operations on all nodes, an information overload occurred during the process of identifying specific face types, utilizing not only adjacent faces but also unrelated ones (Yeo, Kim, *et al.*, 2021). This negatively impacts the recognition of machining feature patterns, acting as a factor that degrades performance.

In this paper, we propose a method for machining feature segmentation from an original CAD model (B-rep model) using BRepGAT, a graph-based DNN. We extract descriptors that effectively represent machining features for faces and edges of the original CAD model and convert them into homogeneous graph data. Sub-

Received: May 26, 2023. **Revised:** November 18, 2023. **Accepted:** November 19, 2023

sequently, we input these graph data into our proposed BRepGAT to segment machining features on a face-by-face basis.

The contribution of the paper is as follows: (i) proposed a graph-based artificial neural network for segmenting machining features; (ii) established a process for handling B-rep CAD models within artificial neural networks; (iii) demonstrated state-of-the-art (SOTA) accuracy in machining feature recognition; (iv) showed relatively robust performance on datasets not used in training; and (v) reconstructed the labels of the MFCAD18++ dataset from a CAM perspective.

This paper is organized as follows: Section 2 introduces related research on graph-based artificial neural networks and machining feature recognition. Section 3 explains the machining feature dataset used in this paper and the newly defined labeling. Section 4 describes the feature data used in the graph and the network of our proposed BRepGAT. Section 5 presents the results and analysis of machining feature segmentation. In Section 6, we conclude with a summary of this paper and provide the potential for future works.

2. Literature Review

2.1. Graph neural networks

Graph neural network (GNN) is a deep learning algorithm that processes graph data (Scarselli *et al.*, 2008). It handles structured data and learns features considering the nodes, edges, and global-level attributes of the graph. GNNs are used to solve various graph-related problems such as classification, clustering, and link prediction. Graph convolutional network (GCN) is a type of GNN that uses convolutional layers to process graph data (Kipf & Welling, 2016; Li *et al.*, 2021). This approach aims to generate node embeddings by integrating information from adjacent nodes. GCNs are used to encode the local structure of graphs to improve the performance of applications. Message passing neural network (MPNN) is a deep learning model for processing graph-based data and is related to GCNs (Fu *et al.*, 2023; Gilmer *et al.*, 2017). MPNNs learn the overall representation of the graph by repeatedly passing and aggregating information using the nodes and edges of the graph. Graph attention network (GAT) introduced the attention mechanism to model relationships between nodes in the graph (Velickovic *et al.*, 2017). As a result, weights are dynamically assigned, allowing for the effective aggregation of information from neighboring nodes. Furthermore, it explicitly encodes the neighboring structure within the graph.

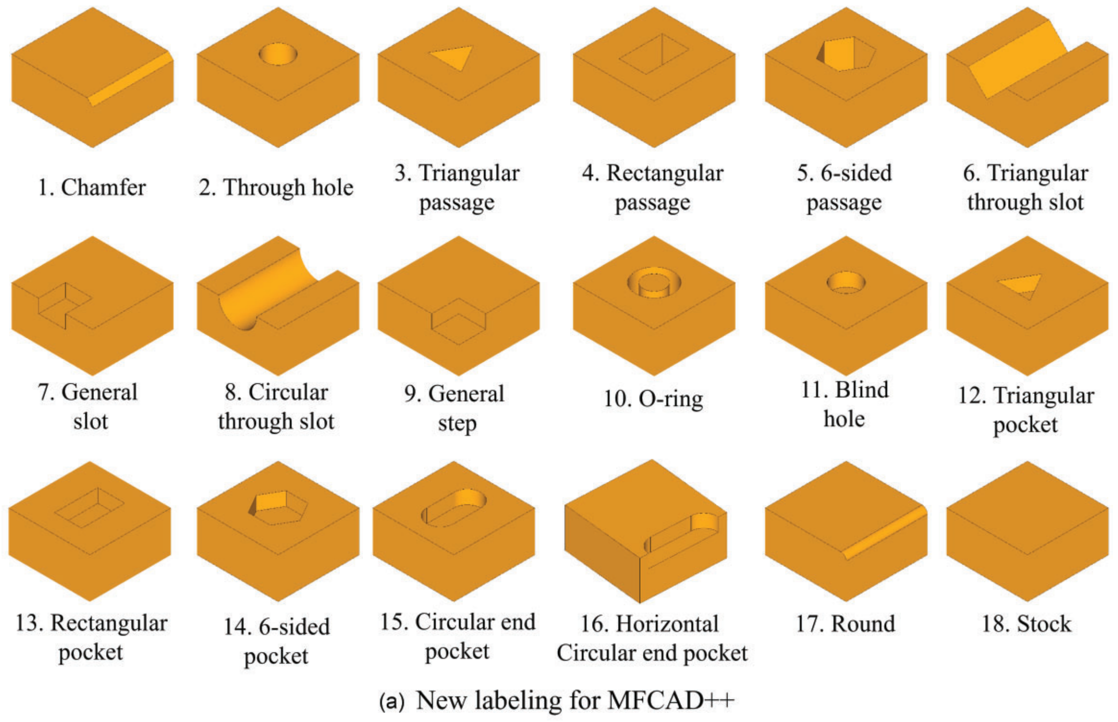
Graph Sample and Aggregative Learning (GraphSAGE) samples information from neighboring nodes and uses aggregation functions to generate embeddings for the central node (Hamilton *et al.*, 2017). This enables predictions for new nodes or dynamic graphs in large graphs. GraphSAGE reduced computational complexity by selecting a fixed-size neighborhood through sampling strategies and can be used for both unsupervised and supervised learning. Mixture of neighborhoods aggregates neighboring nodes in feature space using various Gaussian kernels (Monti *et al.*, 2017). This explicitly encodes the relative positions between the node and its neighbors, capturing various structures.

2.2. Machining feature recognition

Conventional methods for recognizing features in 3D CAD models can be divided into graph-based, volume decomposition, hint-based, and similarity-based approaches (Yeo, Cheon, *et al.*, 2021). Graph-based methods represent the relationships between faces and edges in a graph structure, analyzing patterns where the graphs of two models match. Elinson *et al.* represented the relationships of machining features as graphs and compared their similarities for similarity assessment of solid models (Elinson *et al.*, 1997). However, there were limitations in recognizing complex shapes, as they compared similarities only for simple forms such as milling and drilling. Volume decomposition methods involve recognizing shapes by decomposing complex models into simple volumes of various forms. Woo proposed a fast volume decomposition as an improvement over the traditional cell-based decomposition method, which requires numerous calculations (Woo, 2003). They generated volumes by performing cellular decomposition through localized face extension and cell collection using seed cells. Hint-based methods involve defining hints for the object to be recognized, then identifying the shape that best matches the hint information. An object-oriented feature finder was introduced to find hints from faces of slots, holes, and pockets (Vandenbrande & Requicha, 1993). Similarity-based methods involve finding features or local shapes by comparing the similarities between two models. Jeon, Lee, and Yang generated probability distribution histograms using feature vectors that represent the geometric properties of shape models. They then compared the similarities between existing and new models based on these histograms (Jeon *et al.*, 2016).

Zhang *et al.* proposed FeatureNet, which utilizes 3D-CNN to find machining features (Zhang, Jaiswal, *et al.*, 2018). They designed a network with the aim of recognizing a dataset composed of 24 machining features. Peddireddy *et al.* trained feature recognition using 3D-CNN to identify machining processes in CAD models and then further trained the network through transfer learning to improve performance (Peddireddy *et al.*, 2020). However, overfitting occurred in the training dataset, leading to a decrease in classification accuracy for the validation dataset. Some studies captured images at various angles by rotating the 3D model as input data for feature recognition (Kim, Yeo, *et al.*, 2020; Shi *et al.*, 2020). Based on the captured images, they trained a 2D image neural network and combined the information obtained from various angles. Lee *et al.* classified multi-instance machining features using a voxel-based network. In addition, they predicted the locations of the classified machining features using class activation mapping (Hilbig *et al.*, 2023; Lee *et al.*, 2022b).

Cao *et al.* are pioneers in utilizing GNN for machining feature recognition in 3D CAD models. They proposed an innovative method for generating large datasets containing machining features and presented an informative graph representation to convey 3D CAD models to a GNN (Cao *et al.*, 2020). BRepNet (Lambourne *et al.*, 2021) is designed to leverage the topological relationships between faces and edges in B-rep models, without the need to convert them into approximations such as meshes or point clouds. The architecture defines convolutional kernels with respect to the oriented co-edges in the B-rep data structure, allowing for the extraction of feature vectors from neighboring entities of each co-edge. By concatenating these vectors in a known order, convolution is performed as a matrix/vector multiplication. Specific entities are mapped to learnable parameters in the convolutional kernels, thus enabling the recognition of patterns in the input data. Approach of BRepNet demonstrates an effective means of recognizing design elements, directly on B-rep models, without the need for intermediate data representations. The representation employed by UV-Net (Jayaraman *et al.*, 2021) unifies the U and V parameter domains of curves and surfaces to model geometry. UV-Net used an efficient network architecture that couple's image and GCNs in a manner that is both compute and memory-efficient. In the B-rep representation, each face and edge contain a



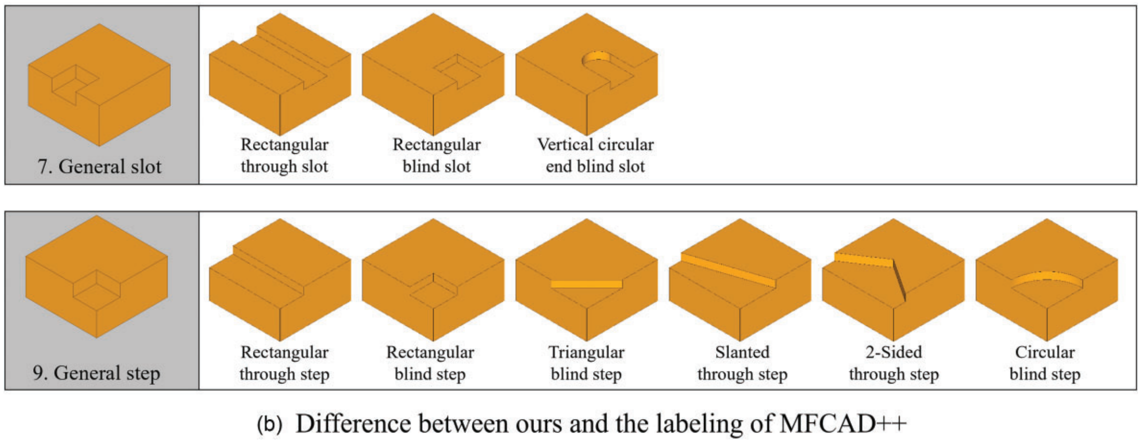


Figure 1: Labeling for machining features.

parametric surface and curve, respectively. These are represented by UV-grids, which UV-Net stores as node and edge attributes in the graph. Hierarchical CADNet (Colligan *et al.*, 2022) proposes a multi-stage neural network model to automatically recognize machining features from both B-rep and facet representations of 3D CAD models. Specifically, the model uses a structural analysis network to identify the overall structure and characteristic terrain of a part and then employs other sub-networks to accurately identify the location, orientation, and scale of each feature. These networks are the use of a multi-stage neural network structure that considers both facet and B-rep representations for feature recognition. Typically, general networks use a single network for feature recognition and do not consider both representations. CAD_GAT conducted a study in which they converted CAD data into Face Attribute Adjacency Graph and Face Attributed Adjacency Graph with Face Structure Fingerprint. They then utilized a GAT to identify the type of assembly interface (Yigang Wang, 2023).

The proposed Hierarchical CADNet achieved higher recognition accuracy compared with other existing models, showcasing the effectiveness of the multi-stage neural network approach.

3. Dataset and Labeling

3.1. MFCAD++ dataset

To train a deep learning network, a large-scale dataset containing machining features is required. There were various datasets containing machining features such as FeatureNet (Zhang, Jaiswal, *et al.*, 2018), Multi-machining Feature (Lee *et al.*, 2022b), and mechanical components benchmark (MCB) (Kim, Chi, *et al.*, 2020).Some of these datasets were served in the format of voxel or image. However, these datasets only have labels for classification of machining feature and do not have labels for segmentation of machining feature. Manual labeling for segmentation could be required time-consuming and may include personal subjective evaluations for labeling depending on the user. MFCAD (MF-

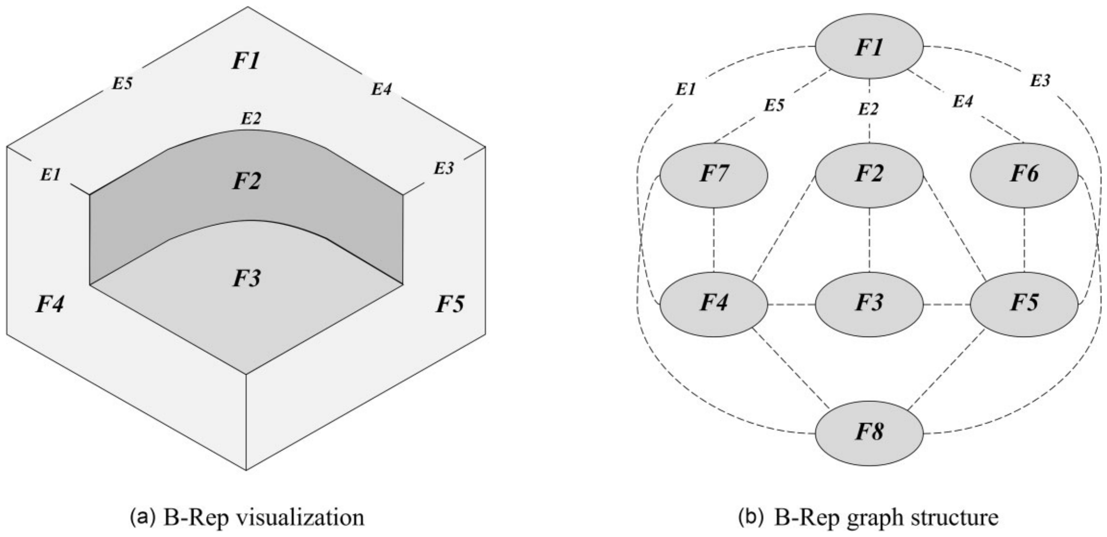


Figure 2: B-rep model and its corresponding graph model.

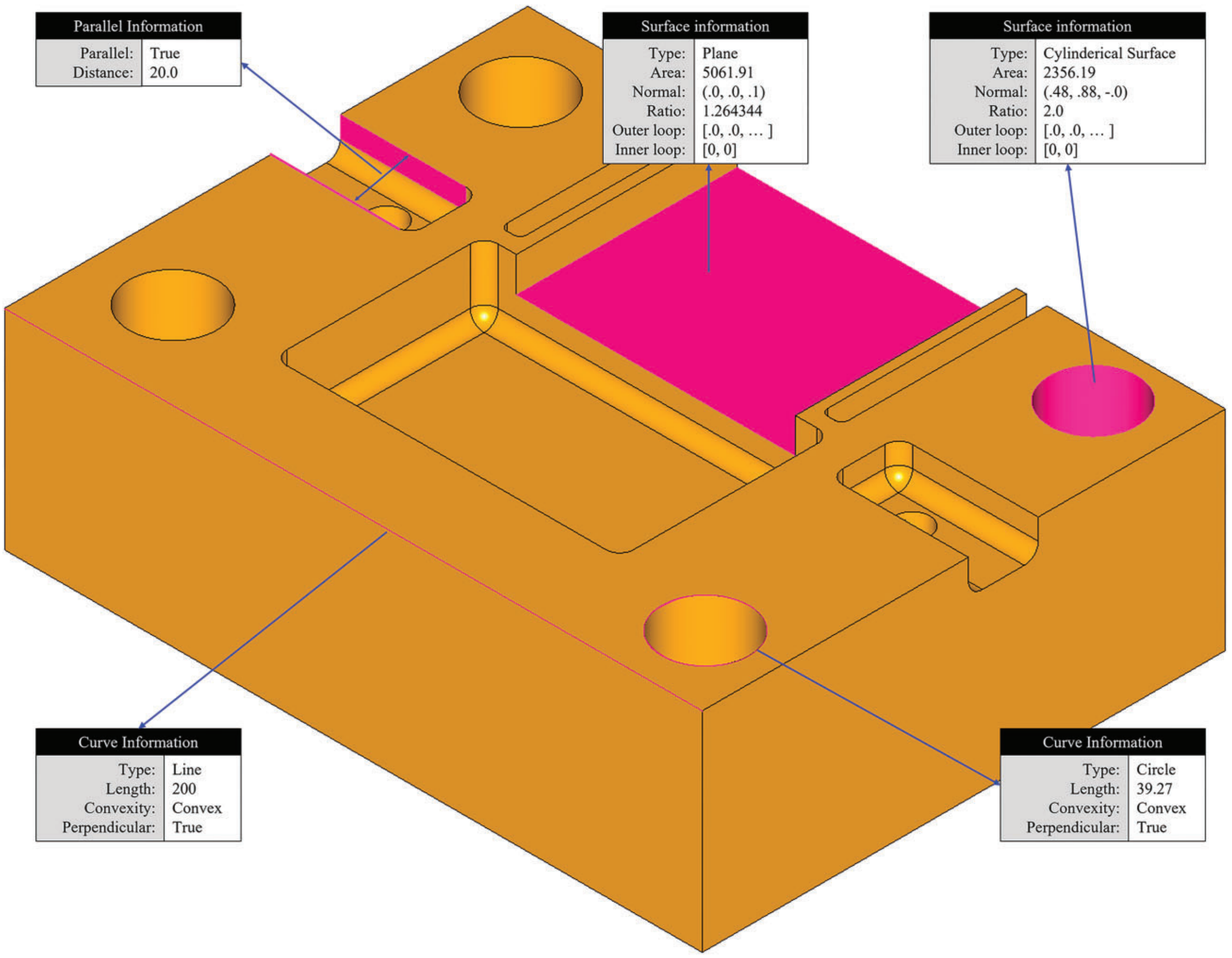


Figure 3: Information types stored in the descriptor.

CAD, 2021) consists of CAD models with various machining features applied to the stock, resulting in a variety of geometries. MFCAD model represents one of 16 types of machining features. However, MFCAD does not have non-planar machining features such as hole and o-ring rounded slot. Additionally, MFCAD exhibits few cases of intersection between machining features.

In this paper, we used MFCAD++ (Colligan *et al.*, 2022) that supplements the original MFCAD dataset. MFCAD++ dataset is composed of 59 655 models. MFCAD++ consists of CAD models

Table 1: Node feature descriptor specifications.

Descriptor item	Data type	Note
Surface type	Integer	Unknown, Bezier surface, B-spline surface, rectangular trimmed surface, conical surface, cylindrical surface, plane, spherical surface, toroidal surface, surface of linear extrusion, surface of revolution, any
Surface area	Float	
Surface normal	Float × 3	(x, y, z)
Bounding box aspect ratio	Float	Width/height
Outer loop – adjacent faces	Float array [33]	Surface type (11) × convexity (3: unknown, convex, concave)
Outer loop – C0 continuity	Float array [11]	Surface type (11)
Outer loop – perpendicular	Float array [11]	Surface type (11)
Inner loop	Float array [2]	Location and convexity

stored in the neutral format, ISO 10303. This dataset is divided into training, validation, and test with a ratio of 70% (41 766)/15% (8950)/15% (8949), respectively. Each model is based on a rectangular stock with dimensions ranging from 10 to 50 and it has multiple interacted machining features. The machining feature labels applied in MFCAD++ consist of the following 25 types: rectangular passage, rectangular pocket, rectangular through slot, rectangular blind slot, rectangular through step, rectangular blind step, triangular passage, triangular pocket, triangular through slot, triangular blind step, two-sided through step, slanted through step, chamfer, six-sided passage, six-sided pocket, through hole, blind hole, o-ring, circular through slot, circular end pocket, circular bind step, vertical circular end blind slot, horizontal circular end blind slot, round, and stock.

3.2. Machining feature labeling

In MFCAD++, the applied labels overlap from a machining perspective, meaning that even if they belong to different labels, they can be machined in the same way. Various machining tools, such as ball end mills, square end mills, rounding end mills, and drills, are employed based on their shapes. MFCAD++ defines features like triangular through slots, circular through slots, horizontal circular end blind slots, rectangular through slots, rectangular blind slots, and vertical circular end blind slots as having a consistent width from start to finish, equivalent to the tool's thickness. Triangular through slots, circular through slots, and horizontal circular end blind slots share these common characteristics but differ in the tools utilized and the closed shapes of adjacent faces. Therefore, it is justified to distinguish them using separate labels. However, rectangular through slots, rectangular blind slots, and vertical circular end blind slots, which employ the same tool and exhibit similar adjacent face shapes, were combined into a new label called "general slot".

Features such as rectangular through steps, rectangular blind steps, triangular blind steps, slanted through steps, two-sided through steps, and circular blind steps all initiate from the stock's edge end, representing stair-like shapes with at least two open faces. Moreover, the machining tools used are identical. Moreover, the machining tools used are identical, prompting the consolidation of these labels into a new label named "general step". Figure 1a displays the list of newly defined machining feature labels in this context. Figure 1b shows the difference between newly defined machining feature labels and labeling of MFCAD18++. Rectangular through slot, rectangular blind slot, and vertical circular end blind slot of MFCAD18++ are combined into general slot. Also, rectangular through steps, rectangular blind steps, triangular blind steps, slanted through steps, two-sided through steps, and circular blind steps of MFCAD18++ are combined into general step.

4. BRepGAT for Machining Feature Segmentation

4.1. Relation between B-rep and graph data structures

B-rep modes are composed of various topology entities such as vertex, edge, and face. Boundaries of face are identified by a set of edges and two ends of each edge are identified by vertices. Furthermore, faces, edges, and vertices reference surfaces, curves, and points, respectively, to represent geometric information. Typically, a graph structure is composed of node and edge. In a general graph structure, nodes represent the unit elements where data is stored, while edges specify the relationships between nodes. Node and edge can be assigned attributes, allowing for the specification of various pieces of information related to each element in the graph. The terminology "edge" in CAD data, representing the linkage between points, and "edge" in graphs, indicating the connection between nodes, might introduce potential ambiguity. To explicitly differentiate these two, in this paper, "b_edge" is employed to denote an edge in CAD data, and "g_edge" to represent an edge in a graph.

By comparing B-rep models with graph models, we find that both models are similar in terms of data representation. Therefore, we utilized the graph structure to input B-rep models into the deep learning network. When representing a B-rep model as a graph model centered around faces and b_edge, faces become nodes in the graph, and b_edges become g_edge in the graph.

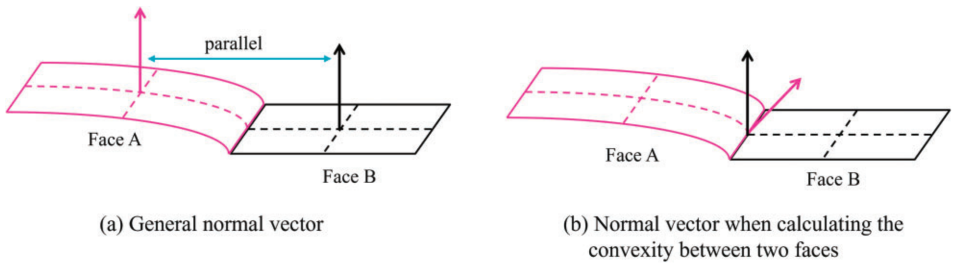

parallel
Face A
Face B
(a) General normal vector
Face A
Face B
(b) Normal vector when calculating the convexity between two faces

Figure 4: Normal vector of a face in two cases: a general case and a convexity calculation case.

Table 2: Edge feature descriptor specifications.

Descriptor item	Data type	Note
Curve type	Integer	Unknown, B-spline, Bezier curve, trimmed curve, bounded curve, circle, ellipse, hyperbola, parabola, conic, line, offset curve, extended_complex curve, any
Curve length	Float	
Convexity	Bool	T/F
Perpendicular	Bool	T/F
Parallel	Bool	T/F
Distance	Float	If parallel is true, this attribute can have value.

Figure 2 shows an example of representing a B-rep model as a graph model. Figure 2a shows a B-rep model, where F1 is surrounded by five g_edges (E1, E2, E3, E4, and E5). This is represented in the graph as node F1 connected to g_edges E1–E5, as shown in Fig. 2b. Thus, the relationship between faces and edges in the B-rep model can be easily represented in the graph model.

4.2. Feature descriptor

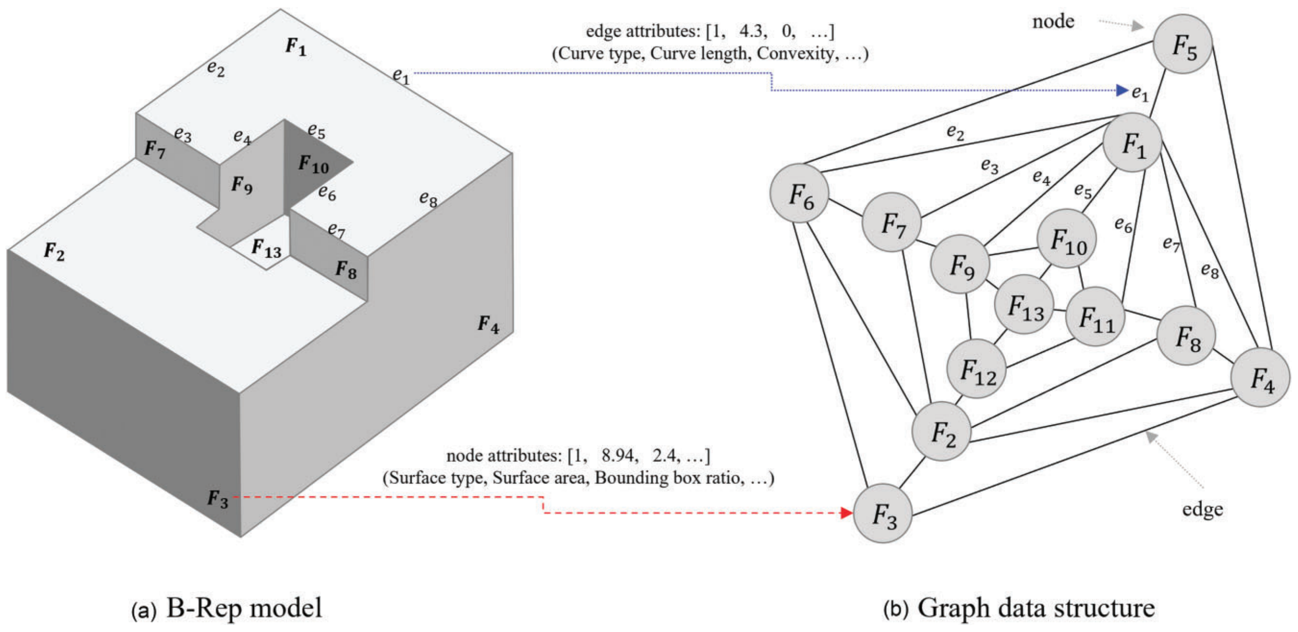
The topology and geometry, which comprise a B-rep model, are fundamental information to recognize machining features. Moreover, valuable information could be obtained by processing the topology and geometry. For example, one topological element (b_edge) can obtain basic information such as type and position from a curve that represents the geometry. By combining this information with the information of two faces adjacent to the b_edge, it is possible to extract length and convexity.

To design the network with high accuracy, it is important to determine what information is extracted from face and b_edge. Figure 3 shows three information types extracted from the B-rep model and stored in descriptor. They are face-surface, edge-curve, and parallelism. Face-surface type stores information about the face and the surface it refers to, while edge-curve type stores information about the b_edge and the curve it refers to. Parallelism stores information about the parallelism between two faces.

We defined the attribute sets of graph nodes and g_edges as node and edge feature descriptors, respectively. Table 1 shows the specifications of the node feature descriptor. The node feature descriptor includes the face-surface type information, which stores information about the B-rep face. Face-surface includes surface type, area, normal, and bounding-box ratio, as well as outer loop and inner loop. Normal vector is generally defined as the direction perpendicular to the center of a face, as shown in Fig. 4a. However, when the normals from the centers of two faces are parallel, it becomes impossible to compute the convexity despite both faces having convex properties. When calculating the convexity between the parallel of two faces, as shown in Fig. 4b, the normal is applied perpendicularly for each face based on the centroid of the edge shared by the two adjacent faces.

For the outer loop, the ratio between the number of adjacent faces belonging to each pair of surface type and convexity type and the total number of adjacent faces is stored. There are a total of 33 pairs. Secondly, the adjacent faces that have C0 continuity with a specific face are found and separated by surface type, and the ratio to the total adjacent surfaces is stored. Thirdly, the adjacent faces that are perpendicular to a specific face are found and separated by surface type, and the ratio to the total adjacent surfaces is stored. For the inner loop, the location type and convexity of the loop are stored. The bounding box ratio was determined using the UV coordinates of a surface. Firstly, the lengths in the u-direction and v-direction of the surface were computed. From these, the ratio was calculated by dividing the shorter direction length by the longer direction length.

Table 2 shows the specifications of the edge feature descriptor. The edge feature descriptor stores information on edge-curve and parallelism types. The edge-curve type stores information on the b_edge of B-rep that is adjacent to two surfaces. The edge-curve type includes curve type and length, as well as convexity type and perpendicularity between the two adjacent surfaces. The convexity of edge was computed using the normal



edge attributes: [1, 4.3, 0, …]
(Curve type, Curve length, Convexity, …)
node attributes: [1, 8.94, 2.4, …]
(Surface type, Surface area, Bounding box ratio, …)
node
edge
F_1 F_2 F_3 F_4 F_5 F_6 F_7 F_8 F_9 F_{10} F_{11} F_{12} F_{13}
e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8
(a) B-Rep model
(b) Graph data structure

Figure 5: Conversion of B-rep model into graph data.

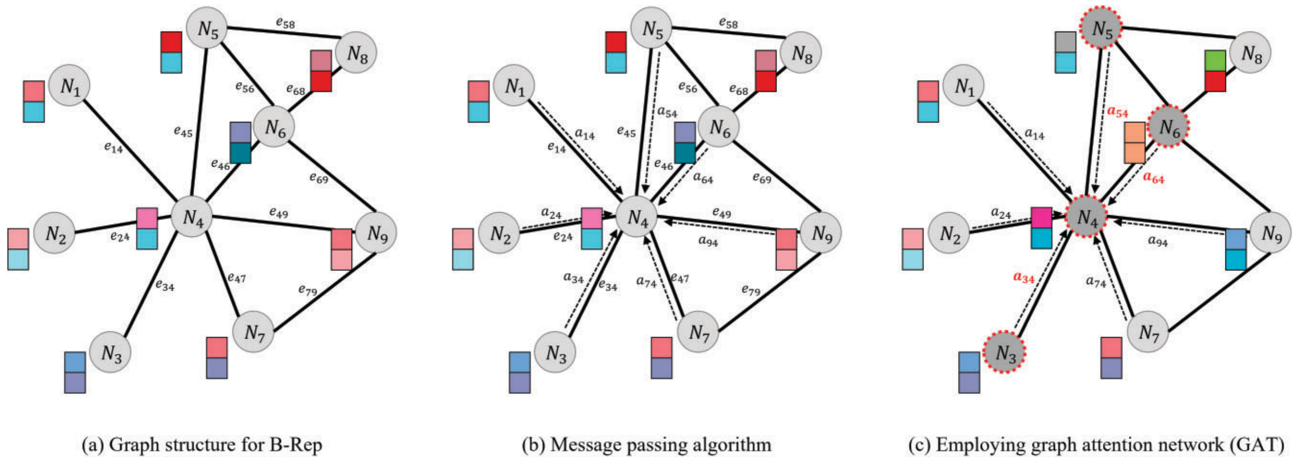

(a) Graph structure for B-Rep
(b) Message passing algorithm
(c) Employing graph attention network (GAT)

Figure 6: Massage passing neural networks and GATs processing graph data.

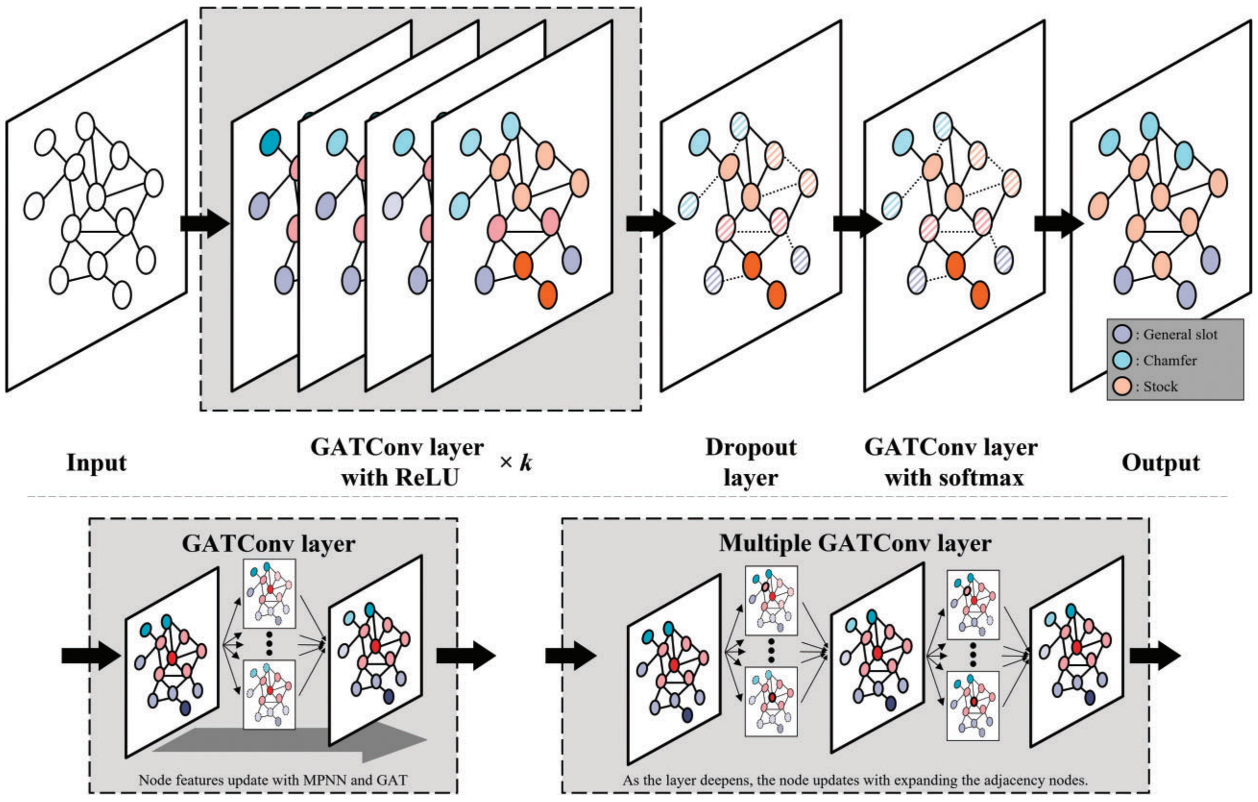

: General slot
: Chamfer
: Stock
Input
GATConv layer with ReLU × k
Dropout layer
GATConv layer with softmax
Output
GATConv layer
Node features update with MPNN and GAT
Multiple GATConv layer
As the layer deepens, the node updates with expanding the adjacency nodes.

Figure 7: BRepGAT network architecture.

vectors of the two shared faces (A and B), and the direction of the edge. When there were faces A and B, their normal were represented as $\vec{n}_A$ and $\vec{n}_B$, respectively. Firstly, the cross product of $\vec{n}_A$ and $\vec{n}_B$ is computed. Then the dot product with the direction of the shared edge is determined, taking into account that the direction of the common edge points counterclockwise relative to face A. If the dot product of the edge is less than zero, faces A and B are concave; otherwise, if greater than zero, they are convex (Lim *et al.*, 2023). When investigating machining features, many pairs of parallel faces are found. Therefore, the parallelism information between two faces is one of the important hints for recognizing machining features. The parallelism type indicates whether two faces are parallel and defines the distance between two parallel faces. Parallelism was computed by traversing every face within the CAD models. To achieve this, pairs were formed between all adjacent faces of a face. Then parallelism was calculated by checking the normal vectors of each pair.

4.3. Representation of B-rep data in graph structure

The input data must be represented in the graph structure for GNNs. The components of graph structure are node, g_edge, node attributes, and edge attributes. Therefore, the B-rep model must be transformed into graph data, and Fig. 5 illustrates this process.

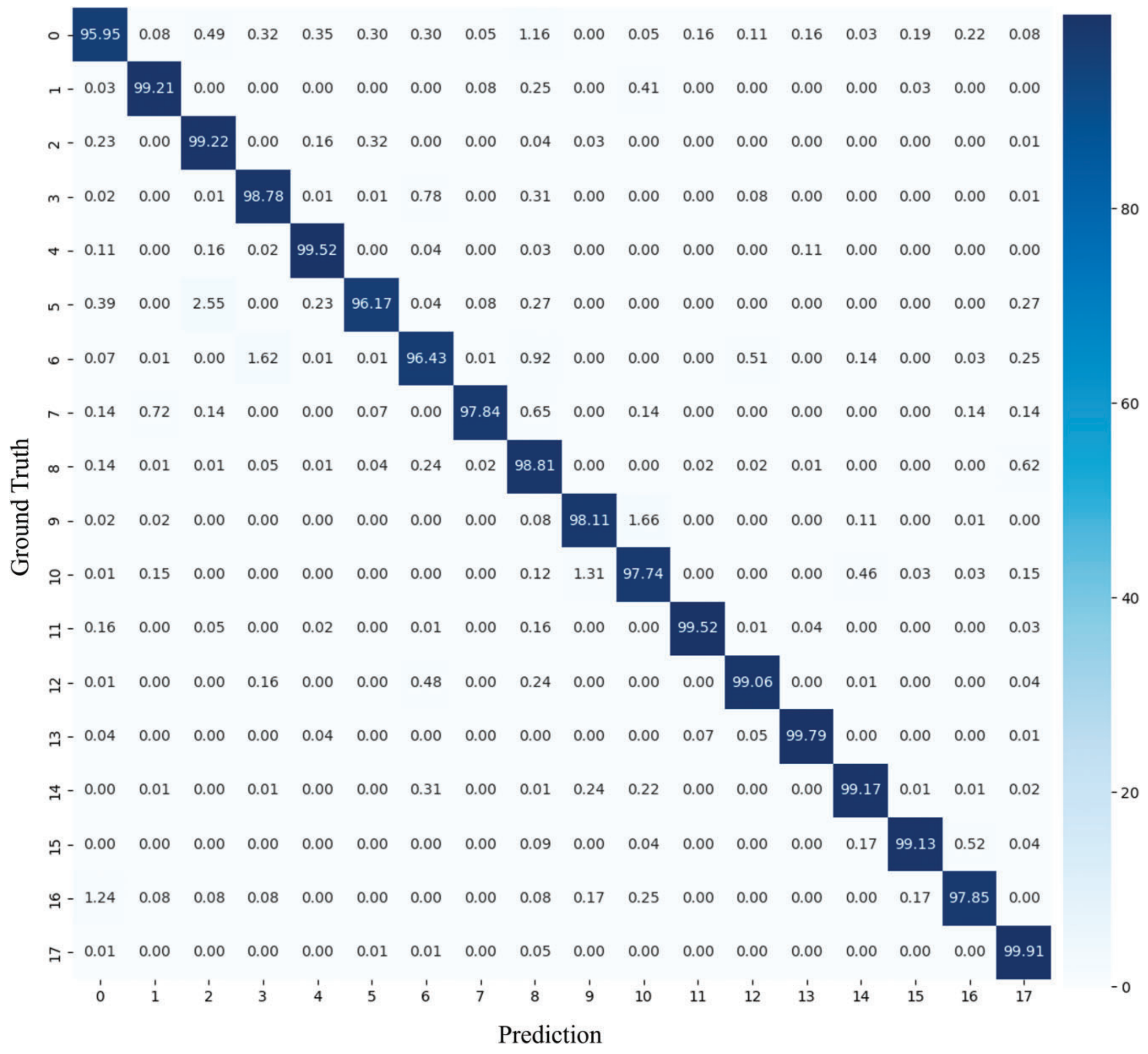


Figure 8: Confusion matrix for MFCAD18++ (accuracy, %).

Table 3: Ablation study for layer.

	Accuracy per face	Number of parameters
Layer 2	98.08%	110 386
Layer 3	98.59%	935 986
Layer 4	98.91%	1 761 586
Layer 5	99.11%	2 587 186

Each face that composes a B-rep model is traversed to generate a graph g_node for each face. Next, the node feature descriptor is created based on the face and the surface referenced by the face, which is used as the attributes of the g_node.

Each b_edge of a B-rep model is traversed, and a g_edge is created for each edge. Two adjacent faces of the b_edge are identified, and two graph nodes corresponding to these faces are connected as a g_edge. An edge feature descriptor is then created based on the b_edge and the curve referenced by the b_edge, which serves as the attributes of the g_edge. If the two adjacent faces are connected, the parallel and distance fields of the edge feature descriptor are set to 0.

During the traversal of faces comprising the B-rep model, faces parallel to the current face are examined. If a parallel face is detected, the distance d between the two faces is calculated, and the corresponding graph nodes are identified. A g_edge connecting these two graph nodes is then created. Subsequently, an edge feature descriptor is generated with a parallel field value of 1, a distance field value of d, and the remaining field values—such as curve type, length, convexity, and perpendicularity are set to 0.

4.4. BRepGAT architecture

In a GNN, the operation between nodes is performed using a MPNN that consists of aggregate and update functions (Gilmer *et al.*, 2017). The aggregate function collects attributes from the neighboring nodes connected to the central node, as shown in equation (1). The collected attributes are then used to update the attribute of the central node using an update function, as

Table 4: Ablation study for descriptor.

Descriptor		Accuracy per face	Difference from SOTA
BRepGAT	All descriptors	99.11%	
	No descriptor	22.36%	76.75%
Face descriptor	No face descriptor	22.38%	76.73%
	Without ratio	99.03%	0.08%
	Without normal	98.89%	0.22%
	Without surface type	98.86%	0.25%
	Without inner loop	98.85%	0.26%
	Without outer loop	98.56%	0.55%
Edge	No edge descriptor	97.80%	1.31%
	Without curve type	98.46%	0.65%
	Without curve length	98.36%	0.75%
	Without convexity	98.51%	0.60%
	Without perpendicular	98.49%	0.62%
	Without parallel information	98.74%	0.37%
Specific descriptor items proposed in this study	Without face descriptor (outer loop, inner loop) and edge descriptor	93.99%	5.12%
	With face descriptor (outer loop, inner loop) and edge descriptor	98.33%	0.78%

Table 5: Comparison of network performance for MFCAD18++ dataset.

Networks	Accuracy per faces (%)	Number of parameters
BRepNet	98.98	0.36M
PointNet++	96.64	1.42M
DGCNN	89.72	0.53M
Hierarchical CADNet (edge)	98.50	9.76M
BRepGAT	**99.11**	2.59M

shown in equation (2):

$$m_v^{t+1} = \sum_{w \in N(v)} M_t\left(h_v^t,\ h_w^t, e_{vw}\right) \tag{1}$$

where h_v^t and h_w^t represent the hidden state of each node v and node w at the layer t, respectively. N(v) denotes the neighbors of node v, e_{vw} represents the feature attributes of the edge connecting node v and node w, and M_t represents the message function that performs the aggregation. In addition, m_v^{t+1} represents the aggregated message at the next layer $(t + 1)$:

$$h_v^{t+1} = U_t\left(h_v^t,\ m_v^{t+1}\right) \tag{2}$$

where U_t updates the current node's hidden state (h_v^t) to the next layer's hidden state (h_v^{t+1}) by considering the aggregate message (m_v^{t+1}) obtained through the previous layer. As we pass through multiple layers, not only the information of directly connected neighbor nodes but also the information of nodes that are far apart from the center node is computed.

In the process of message passing, it is important that information about nodes containing machining features is effectively propagated from neighbors to the central node. To achieve this, we applied GATs, which assign weights to edges connecting important nodes and pay more attention to closer nodes. Equation (3) shows how the node's hidden state is calculated using the attention mechanism:

$$h_v^{t+1} = \sigma\left(\sum_{w \in N(v)} \alpha_{vw} W^t h_w^t\right) \tag{3}$$

where α_{vw} is attention weight and W^t is attention matrix in the tth layer. The attention weight is obtained by calculating the relationship between two nodes through the weight matrix W in each layer.

Figure 6 illustrates the processes of MPNN and GAT. Figure 6a represents a graph structure based on the descriptors extracted from B-rep data. As aforementioned in Fig. 5, each node N_i corresponds to a face and contains descriptors. In Fig. 6a, the two boxes mean the face descriptors. Additionally, the links connecting nodes, represented as e_i, carry the edge descriptors. Figure 6b shows the workings of the message-passing algorithm centered around node N_4. The face descriptor of node N_4 is updated by referencing the face descriptors of the nodes connected to it (i.e., $N_1,\ N_2, N_3,\ N_5,\ N_6,\ N_7, and\ N_9$). Only adjacent nodes participate in the computations. During this process, both face and edge descriptors are incorporated, which further influence the multilayer perceptron (MLP) computations. Figure 6c illustrates the GAT, where rather than applying consistent weights across all node updates, weights are only assigned to pivotal nodes involved in node classification. In Fig. 6c, centered on node N_4, weights (i.e., a_{34}, a_{54}, and a_{64}) are assigned to the crucial connected nodes $N_3,\ N_5$, and N_6.

We propose BRepGAT, a GNN for segmenting faces that belong to machining features from a B-rep model. BRepGAT combines MPNN for learning the information of the central and surrounding nodes, and GAT for assigning weights to important nodes. Figure 7 shows the proposed network (BRepGAT) architecture. Basic BRepGAT consists of six layers. We applied graph attention network convolution (GATConv) and rectified linear units (ReLU) in the 1st to 4th layers for learning, with GAT parameters of 640. In this study, we optimized the network by adjusting the number of GATConv layers to k. By applying the GATConv, not only the nearest nodes but also the surrounding nodes (adjacent nodes of adjacent nodes) were enabled to be incorporated for updating node

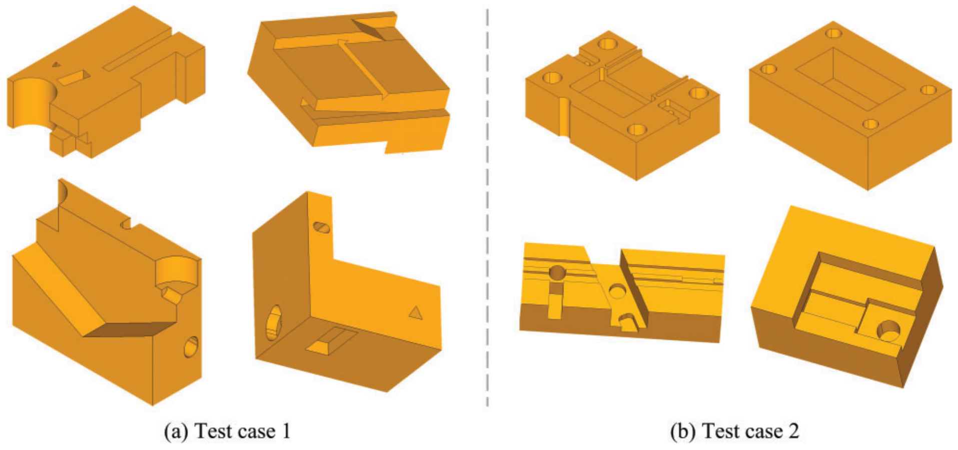


Figure 9: Test cases 1 and 2 used for network performance comparison.

Table 6: Network performance for another machining feature dataset.

Network	Dataset	Accuracy (%)	Recall (%)	Precision (%)	F1-score
Hierarchical CADNet (edge)	Test case 1	94.03	93.77	94.35	94.06
	Test case 2	15.12	17.49	15.47	16.42
BrepGAT	Test case 1	98.98	96.93	97.81	**97.37**
	Test case 2 (Layer 5)	46.71	48.55	36.24	41.50
	Test case 2 (Layer 2)	52.17	50.47	51.84	**51.15**

descriptors of a node in their computation. In the 5th layer, we performed dropout to block the signal going to the next layer with a certain probability to prevent overfitting. Finally, in the last layer, we trained a GATConv without ReLU to output segmentation result.

5. Results and Discussion

5.1. System configuration

PYTHON 3.8.13, PYTORCH 1.11.0, and PYTORCH GEOMETRIC 2.0.4 libraries were employed to train the deep learning model. All the experiments were conducted on a computer with an i9-13900K CPU (3.0 GHz), 32 GB memory, and NVIDIA RTX 4090 GPU. The gradient descent optimization used in the experiments was the Adam with a learning rate of 0.001 and cross-entropy loss function. The batch size was 64, and the epoch was 200. To prevent overfitting, early stop was applied if the validation loss did not continue to decrease in 20 epochs. The proposed network has approximately 2.57 million parameters.

5.2. Experimental results and comparison with other networks

To evaluate how accurately BRepGAT segments machining features, we conducted experiments on the MFCAD18++ dataset. As mentioned earlier, the MFCAD18++ dataset is divided into 70:15:15 training/validation/test ratios. BRepGAT showed an accuracy of 99.10% in segmenting machining features in the test dataset. Figure 8 shows the recognition accuracy for individual classes. The horizontal axis in Fig. 6 represents the prediction, and the vertical axis represents the ground truth. High accuracy of over 97.5% was achieved for most classes. However, there were about 4% recognition failures in classes #0 (chamfer), #5 (triangular through slot), and #6 (general slot).

To evaluate the performance of our proposed network, we conducted an ablation study by modifying the number of GAT layers, excluding batch normalization and dropout layers. Table 3 shows the accuracy according to the number of GAT layers. In GNNs, as the depth of layers increases, the probability of oversmoothing increases, which can lead to a decrease in recognition performance (Chen *et al.*, 2020). Therefore, we limited the number of layers for our experiments to be between 2 and 5.

Beyond the ablation study for the layers, we performed various ablation studies to assess the influence of face and edge descriptors on the performance of our network. There was a noticeable change in performance when certain features were removed, as shown in Table 4. Notably, the absence of descriptors resulted in the lowest accuracy, measuring at 22.36%, while the exclusion of the face descriptor resulted in the second-lowest accuracy, at 22.38%. However, when using only the face descriptor (without the edge descriptor), the accuracy improved significantly, reaching 97.80%. This demonstrates the critical role played by the face descriptor in face segmentation. Upon introducing the edge descriptor alongside the face descriptor, we observed a performance improvement of 1.31%. Furthermore, the inclusion of specific descriptor items proposed in this study, such as outer loop, inner loop, and edge descriptor, led to a substantial performance boost of 5.12%. This clearly illustrates the effectiveness of the descriptor items we introduced in enhancing overall performance.

We performed new labeling of the MFCAD18++ dataset to fit the machining operations and compared the performance of Hierarchical CADNet and our proposed network on the labeled MFCAD18++ dataset. Table 5 presents the segmentation accuracy results: our proposed network attained the highest accuracy at 99.10%, followed by BrepNet at 98.98%, Hierarchical CAD-

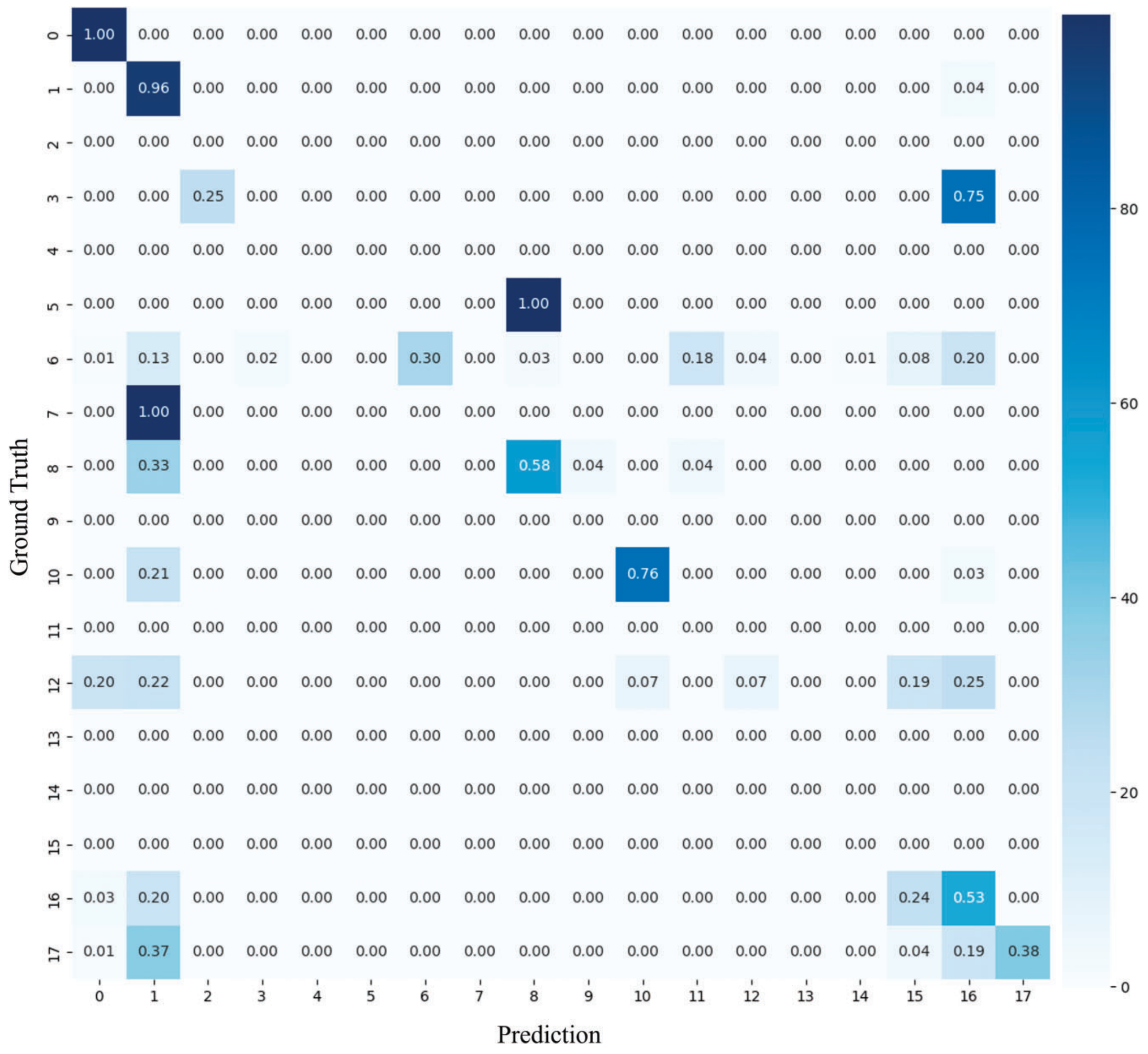


Figure 10: Confusion matrix for test case 2 (accuracy, %).

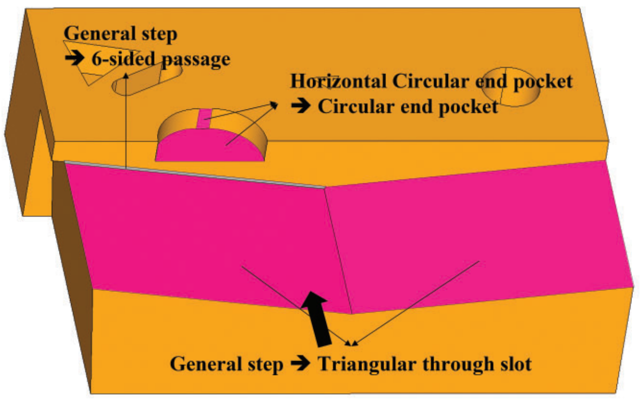
Table 7: Segmentation accuracy per face and number of parameters for Fusion360 gallery segmentation dataset.

Networks		Accuracy per faces (%)	Number of parameters
BRepNet		<u>92.52</u>	0.36M
Hierarchical CADNet	Adj	86.30	4.49M
	Edge	88.46	6.60M
BRepGAT	Layer 2	80.32	0.11M
	Layer 5	80.29	2.59M

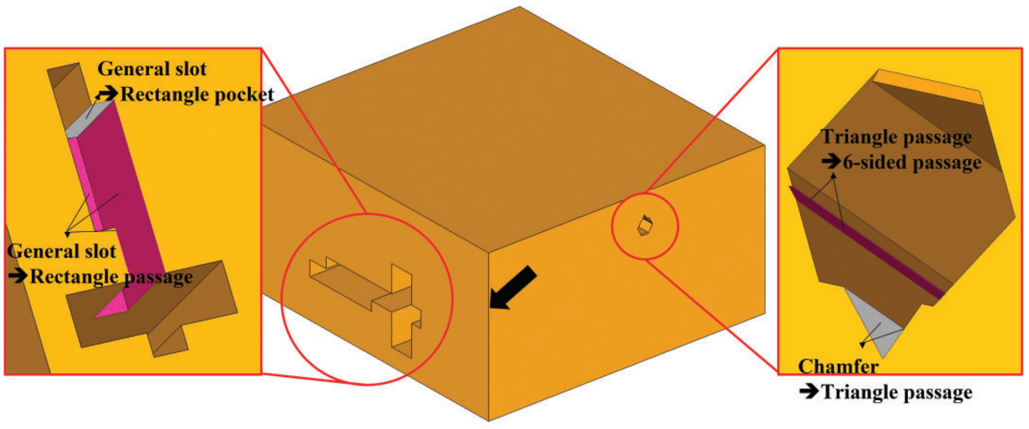
Net at 98.50%, PointNet++ at 96.64%, and DGCNN at 89.72%. In the MFCAD18++ dataset, our network demonstrated SOTA performance. We thought that incorporating descriptor items such as parallelism, distance, and convexity improved the recognition accuracy for machining features. Moreover, our proposed BRepGAT had the least number of network parameters.

We conducted experiments on other machining feature datasets. Figure 9 shows examples of CAD models included in test cases 1 and 2. Test case 1 (Fig. 9a) and test case 2 (Fig. 9b) consist of 40 and 24 models, respectively. Test case 1 consists of a dataset randomly sampled from the MFCAD18++ dataset (Fig. A.1) Test case 2 is a dataset modeled using the same 3D CAD software by three different designers (Fig. A.2). These designs were inspired by representative sample models from machining feature recognition studies (Gupta *et al.*, 2019; Han *et al.*, 2000; Ning *et al.*, 2020; Wang & Yu, 2014; Yeo, Cheon, *et al.*, 2021; Zhang, Luo, *et al.*, 2017). Given that the three designers utilized their individual design expertise, even identical shapes might have different B-rep structures. Refer to Appendix 1 for the details of test cases 1 and 2.

Table 6 presents the recognition results of BRepGAT and Hierarchical CADNet on test cases 1 and 2. In test case 1, both networks showed performance indicators similar to those of the MFCAD18++ dataset. However, in test case 2, both networks showed low recognition accuracy. In particular, Hierarchical CADNet showed a very low accuracy of 15.12% on the test case 2

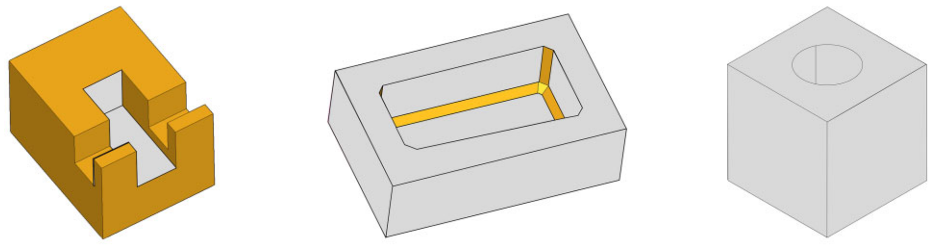

General step
➔ 6-sided passage
Horizontal Circular end pocket
➔ Circular end pocket
General step ➔ Triangular through slot

(a) Result of recognition for model 8397 (# of error face: 5)



(b) Result of recognition for model 43144 (# of error face: 8)

Figure 11: Recognition failure cases in test case 1.



(a) Designer A's model (b) Designer B's model (c) Designer C's model

Figure 12: Recognition failure cases in test case 2 (yellow: recognition success, gray: recognition failure).

dataset. As shown in Table 6, as the number of layers increased, the performance on test case 1, a subset of MFAD18++, improved. However, overfitting occurred for test case 2, another dataset. It means that deepen layer reduced the robustness of the network to the external dataset. Consequently, we limited our testing to only up to five layers.

Figure 10 shows the recognition results of BRepGAT on test case 2 as a confusion matrix. As shown in Fig. 10, although high accuracy was achieved in some classes, biased recognition results were observed in some classes.

To evaluate the performance of BRepGAT, we utilized the Fusion 360 gallery segmentation dataset for testing. Table 7 shows the segmentation accuracy per face and the number of parameters for Fusion360 gallery segmentation dataset. BRepNet demonstrated the highest accuracy of 92.52% on this dataset, while our network achieved an accuracy of 80.32%. The Fusion 360 gallery includes features like fillets and chamfers, which are also present in MFCAD18++, but it also contains shapes distinct from machining features, such as resolve-side and resolve-end. The descriptors employed in this study encompassed the geometry of each face, including surface type and surface convexity, as well as the relationships with adjacent faces. These relationships involved assessing convexity between the target face and adjacent faces, measuring angles between the target face and adjacent faces, and counting adjacent faces. Additionally, the bounding box ratio emerged as a critical descriptor in machining feature recognition, particularly for distinguishing between steps and slots. However, in the context of face segmentation based on BRepNet's labels, the descriptors used in our study presented significant challenges, with effective differentiation achieved only for fillet and chamfer among BRepNet's labels. The specific descriptor items proposed in this study played a minimal role in the Fusion 360 dataset. They failed to effectively distinguish between cut-side, cut-end, extrude-side, and extrude-end, and in some cases, may have even introduced noise. Consequently, relying solely on these descriptors may result in suboptimal recognition performance for the Fusion 360 dataset. We believe that performance can be further enhanced by extracting descriptor items fitting the new labels in the Fusion 360 gallery segmentation dataset and training the GNN accordingly.

6. Discussion

Our proposed BRepGAT showed SOTA performance in recognizing machining features from the MFCAD18++ dataset. Hierarchical CADNet, a network that recognizes machining features using B-rep models and mesh data, achieved an accuracy of 98.50%. While CADNet used 7.18M more parameters than BRepGAT, it showed 0.60% lower accuracy. The computational load increased because CADNet performed graph convolutional layers while embedding and transferring facet layers and B-rep layers. In addition, feature attributes that were insufficient for distinguishing between rectangles and slanted in machining feature recognition, such as through step, were used. Furthermore, information may have been lost when embedding B-rep models and mesh data, resulting in a decrease in accuracy.

Using BRepGAT, the recognition results for test case 1 were as follows: all faces were correctly recognized for 26 out of 40 models, one face recognition failure occurred for seven models, two face recognition failures occurred for two models, two face recognition failures occurred for three models, five face recognition failures occurred for one model, and eight face recognition failures occurred for one model. The two models with the most recognition failures (five errors and eight errors) are shown in Fig. 11. As seen in Fig. 11a, a general step was misrecognized as a triangular through slot. It is possible to recognize it a triangular through slot when the model is viewed at the arrow direction. However, additional training is required in that it was recognized as a through slot although the obstructing face in the back. The cases where a horizontal circular end pocket was recognized as a circular end pocket and a general step was recognized as a six-sided passage are clear recognition failures. As seen in Fig. 11b, the case where a general slot was recognized as a rectangular passage is thought to be due to oversmoothing occurring over several layers in deep learning. The case where a triangular passage was recognized as a six-sided passage is thought to occur because this feature is applied to a six-sided passage and is very small compared with the surrounding shapes.

Test case 2 consists of 24 models created by three designers using the same 3D CAD system based on their own design know-how. The recognition accuracy of BRepGAT for seven models created by designer A was relatively high at 88.97%. Figure 12a shows the model created by designer A, which failed to recognize three out of a total of 18 faces. These were faces that could be misrecognized depending on the direction from which they were viewed from a machining perspective. On the other hand, eight models created by designer B showed a worse accuracy of 39.86%, and nine models created by designer C showed the worst accuracy of 22.90%. Although there were some simple models created by designers B and C, most of them failed to recognize properly, as shown in Fig. 12b and c.

Our proposed network was trained with CAD models in the MFCAD18++ dataset; thus, it is plausible that its performance was suboptimal on models created by users not included in the training set. As mentioned in Table 5, networks with fewer layers (models using Layer 2) showed a relatively high accuracy (52.17%) on external datasets. This suggests that our proposed network may have been overfitting to the MFCAD18++ dataset. In addition, when converting native models from 3D CAD systems to neutral models in ISO 10303, the internal topological structure of B-rep models may have differed from that of B-rep models in the MFCAD18++ dataset. For example, the topological structure used to represent cylinders can vary between CAD systems. Nonetheless, BRepGAT demonstrated relatively robust performance on other datasets compared with Hierarchical CADNet. In future study, it will be necessary to train the network on CAD datasets that include user-generated models with machining features to improve its adaptability.

7. Conclusions

We proposed a graph-based deep learning network called BRepGAT for segmenting machining features in B-rep models. The topology and geometry in a B-rep model were converted into a graph structure, enabling its input into the deep learning network. Furthermore, we generated node and edge feature descriptors that include face, surface, edge, curve, and parallelism information of the B-rep model and stored them as attributes of graph nodes and edges. The proposed network achieved SOTA recognition accuracy of 99.10% with 2.58 million parameters on the MFCAD18++ dataset. It also performed well on external datasets compared with other networks.

However, since BRepGAT was trained on the MFCAD18++ dataset, it showed lower performance on external CAD datasets that contain machining features. Therefore, additional training on CAD datasets with machining features is needed to increase the network's robustness. In addition, we plan to customize BRepGAT to recognize functional features contained in 3D CAD models.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (Nos. 2021R1I1A1A01050440 and NRF-2022R1A2C2005879), by the Industrial Technology Innovation Program (No. 20012462) funded by the Korean Government (MOTIE), by Liquid Air Energy Storage (LAES) and Utilization System Development Program (RS-2022–00143652) funded by the Ministry of Land, Infrastructure and Transport, and by the MSIT (Ministry of Science and ICT), Korea, under the Innovative Human Resource Development for Local Intellectualization Support Program (IITP-2023-RS-2023–00260267) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation.

Conflict of interest statement

The authors declared no potential conflicts of interest with respect to the research, authorship, and publication of this article.

References

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 *Proceedings of the ASME 2020 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. https://doi.org/10.1115/DETC2020-22355.

Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., & Sun, X. (2020). Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. *Proceedings of the AAAI Conference on Artificial Intelligence*, **34**, 3438–3445. https://doi.org/10.1609/aaai.v34i04.5747.

Colligan, A. R., Robinson, T. T., Nolan, D. C., Hua, Y., & Cao, W. (2022). Hierarchical CADNet: Learning from B-reps for machining feature recognition. *Computer-Aided Design*, **147**, 103226. https://doi.org/10.1016/j.cad.2022.103226.

Elinson, A., Nau, D. S., & Regli, W. C. (1997). Feature-based similarity assessment of solid models. In *Proceedings of the Fourth ACM Symposium on Solid Modeling and Applications* (pp. 297–310). Association for Computing Machinery. https://doi.org/10.1145/267734.267806.

Fu, X., Zhou, F., Peddireddy, D., Kang, Z., Jun, M. B.-G, & Aggarwal, V. (2023). An finite element analysis surrogate model with boundary oriented graph embedding approach for rapid design. *Journal of Computational Design and Engineering*, **10**, 1026–1046. https://doi.org/10.1093/jcde/qwad025.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *Proceedings of the International Conference on Machine Learning* (pp. 1263–1272).

Gupta, M. K., Swain, A. K., & Jain, P. K. (2019). A novel approach to recognize interacting features for manufacturability evaluation of prismatic parts with orthogonal features. *The International Journal of Advanced Manufacturing Technology*, **105**, 343–373. https://doi.org/10.1007/s00170-019-04073-7.

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*.

Han, J., Pratt, M., & Regli, W. C. (2000). Manufacturing feature recognition from solid models: A status report. *IEEE Transactions on Robotics and Automation*, **16**, 782–796. https://doi.org/10.1109/70.897789.

Hilbig, A., Vogt, L., Holtzhausen, S., & Paetzold, K. (2023). Enhancing three-dimensional convolutional neural network-based geometric feature recognition for adaptive additive manufacturing: A signed distance field data approach. *Journal of Computational Design and Engineering*, **10**, 992–1009. https://doi.org/10.1093/jcde/qwad027.

Hwang, J., Mun, D., & Han, S. (2009). Representation and propagation of engineering change information in collaborative product development using a neutral reference model. *Concurrent Engineering*, **17**, 147–157. https://doi.org/10.1177/1063293×09105339.

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). IEEE.

Jeon, H., Lee, J., & Yang, J. (2016). A touch-probe path generation method through similarity analysis between the feature vectors in new and old models. *Journal of Mechanical Science and Technology*, **30**, 4705–4716. https://doi.org/10.1007/s12206-016-0941-8.

Kim, B. C., Kim, H., Moon, Y., Lee, G., & Mun, D. (2022). End-to-end digitization of image format piping and instrumentation diagrams at an industrially applicable level. *Journal of Computational Design and Engineering*, **9**, 1298–1326. https://doi.org/10.1093/jcde/qwac056.

Kim, B. C., Mun, D., Han, S., & Pratt, M. J. (2011). A method to exchange procedurally represented 2D CAD model data using ISO 10303 STEP. *Computer-Aided Design*, **43**, 1717–1728. https://doi.org/10.1016/j.cad.2011.07.006.

Kim, H., Yeo, C., Lee, I. D., & Mun, D. (2020). Deep-learning-based retrieval of piping component catalogs for plant 3D CAD model reconstruction. *Computers in Industry*, **123**, 103320. https://doi.org/10.1016/j.compind.2020.103320.

Kim, S., Chi, H. G., Hu, X., Huang, Q., & Ramani, K. (2020). A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks. In *Proceedings of the European Conference on Computer Vision*. https://doi.org/10.1007/978-3-030-58523-5_11.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *preprint*(arXiv:1609.02907). https://doi.org/10.48550/arXiv.1609.02907.

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). IEEE.

Lee, H., Lee, J., Kim, H., & Mun, D. (2022a). Dataset and method for deep learning-based reconstruction of 3D CAD models containing machining features for mechanical parts. *Journal of Computational Design and Engineering*, **9**, 114–127. https://doi.org/10.1093/jcde/qwab072.

Lee, J., Lee, H., & Mun, D. (2022b). 3D convolutional neural network for machining feature recognition with gradient-based visual explanations from 3D CAD models. *Scientific Reports*, **12**, 14864. https://doi.org/10.1038/s41598-022-19212-6.

Li, M., Zhu, S., Li, C., & Zhao, W. (2021). Target unbiased meta-learning for graph classification. *Journal of Computational Design and Engineering*, **8**, 1355–1366. https://doi.org/10.1093/jcde/qwab050.

Liang, Y., He, F., Zeng, X., & Yu, B. (2022). Feature-preserved convolutional neural network for 3D mesh recognition. *Applied Soft Computing*, **128**, 109500. https://doi.org/10.1016/j.asoc.2022.109500.

Lim, S., Yeo, C., He, F., Lee, J., & Mun, D. (2023). Machining feature recognition using descriptors with range constraints for mechanical 3D models. *International Journal of Precision Engineering and Manufacturing*, **24**, 1865–1888. https://doi.org/10.1007/s12541-023-00836-1.

MFCAD: A dataset of 3D CAD models with machining feature labels. (2021, June 14). GitHub, Inc. https://github.com/hducg/MFCAD.

Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., & Bronstein, M. M. (2017). Geometric deep learning on graphs and manifolds using mixture model CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5115–5124). IEEE.

Ning, F., Shi, Y., Cai, M., Xu, W., & Zhang, X. (2020). Manufacturing cost estimation based on the machining process and deep-learning method. *Journal of Manufacturing Systems*, **56**, 11–22. https://doi.org/10.1016/j.jmsy.2020.04.011

Peddireddy, D., Fu, X., Wang, H., Joung, B. G., Aggarwal, V., Sutherland, J. W., & Jun, M. B.-G. (2020). Deep learning based approach for identifying conventional machining processes from CAD data. *Procedia Manufacturing*, **48**, 915–925. https://doi.org/10.1016/j.promfg.2020.05.130.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2008). The graph neural network model. *IEEE Transactions on Neural Networks*, **20**, 61–80. https://doi.org/10.1109/TNN.2008.2005605.

Shi, P., Qi, Q., Qin, Y., Scott, P. J., & Jiang, X. (2020). A novel learning-based feature recognition method using multiple sectional view representation. *Journal of Intelligent Manufacturing*, **31**, 1291–1309. https://doi.org/10.1007/s10845-020-01533-w.

Son, Y. H., Kim, G.-Y., Kim, H. C., Jun, C., & Noh, S. D. (2022). Past, present, and future research of digital twin for smart manufacturing. *Journal of Computational Design and Engineering*, **9**, 1–23. https://doi.org/10.1093/jcde/qwab067.

Vandenbrande, J. H., & Requicha, A. A. (1993). Spatial reasoning for the automatic recognition of machinable features in solid models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**, 1269–1285. https://doi.org/10.1109/34.250845.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. preprint (arXiv:1710.10903). https://doi.org/10.48550/arXiv.1710.10903.

Wang, Q., & Yu, X. (2014). Ontology based automatic feature recognition framework. *Computers in Industry*, **65**, 1041–1052. https://doi.org/10.1016/j.compind.2014.04.004.

Woo, Y. (2003). Fast cell-based decomposition and applications to solid modeling. *Computer-Aided Design*, **35**, 969–977. https://doi.org/10.1016/S0010-4485(02)00144-6.

Yeo, C., Cheon, S., & Mun, D. (2021). Manufacturability evaluation of parts using descriptor-based machining feature recognition. *International Journal of Computer Integrated Manufacturing*, **34**, 1196–1222. https://doi.org/10.1080/0951192X.2021.1963483.

Yeo, C., Kim, B. C., Cheon, S., Lee, J., & Mun, D. (2021). Machining feature recognition based on deep neural networks to support tight integration with 3D CAD systems. *Scientific Reports*, **11**, 22147. https://doi.org/10.1038/s41598-021-01313-3.

Yigang Wang, E. S., Hong, L. i, Wanbin, P., Weijuan, C., Jie, M., & Xiaofei, A. i. (2023). An intelligent identification approach of assembly interface for CAD models. *Computer Modeling in Engineering & Sciences*, **137**, 859–878. https://doi.org/10.32604/cmes.2023.027320.

Zhang, D., He, F., Tu, Z., Zou, L., & Chen, Y. (2020). Pointwise geometric and semantic learning network on 3D point clouds. *Integrated Computer-Aided Engineering*, **27**, 57–75. https://doi.org/10.3233/ICA-190608.

Zhang, Y., Luo, X., Zhang, B., & Zhang, S. (2017). Semantic approach to the automatic recognition of machining features. *The International Journal of Advanced Manufacturing Technology*, **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. *Computer-Aided Design*, **101**, 12–22. https://doi.org/10.1016/j.cad.2018.03.006.

Appendix 1. Test Cases 1 and 2

Test case 1 consists of a dataset randomly sampled from the MFCAD18++ dataset. Test case 2 is a dataset modeled using the same design software, CATIA, by three different designers. These designs were inspired by representative sample models from machining feature recognition studies (Gupta *et al.*, 2019; Han *et al.*, 2000; Ning *et al.*, 2020; Wang & Yu, 2014; Yeo, Cheon, *et al.*, 2021; Zhang, Luo, *et al.*, 2017). Given that the three designers utilized their individual design expertise, even identical shapes might have different B-rep structures.

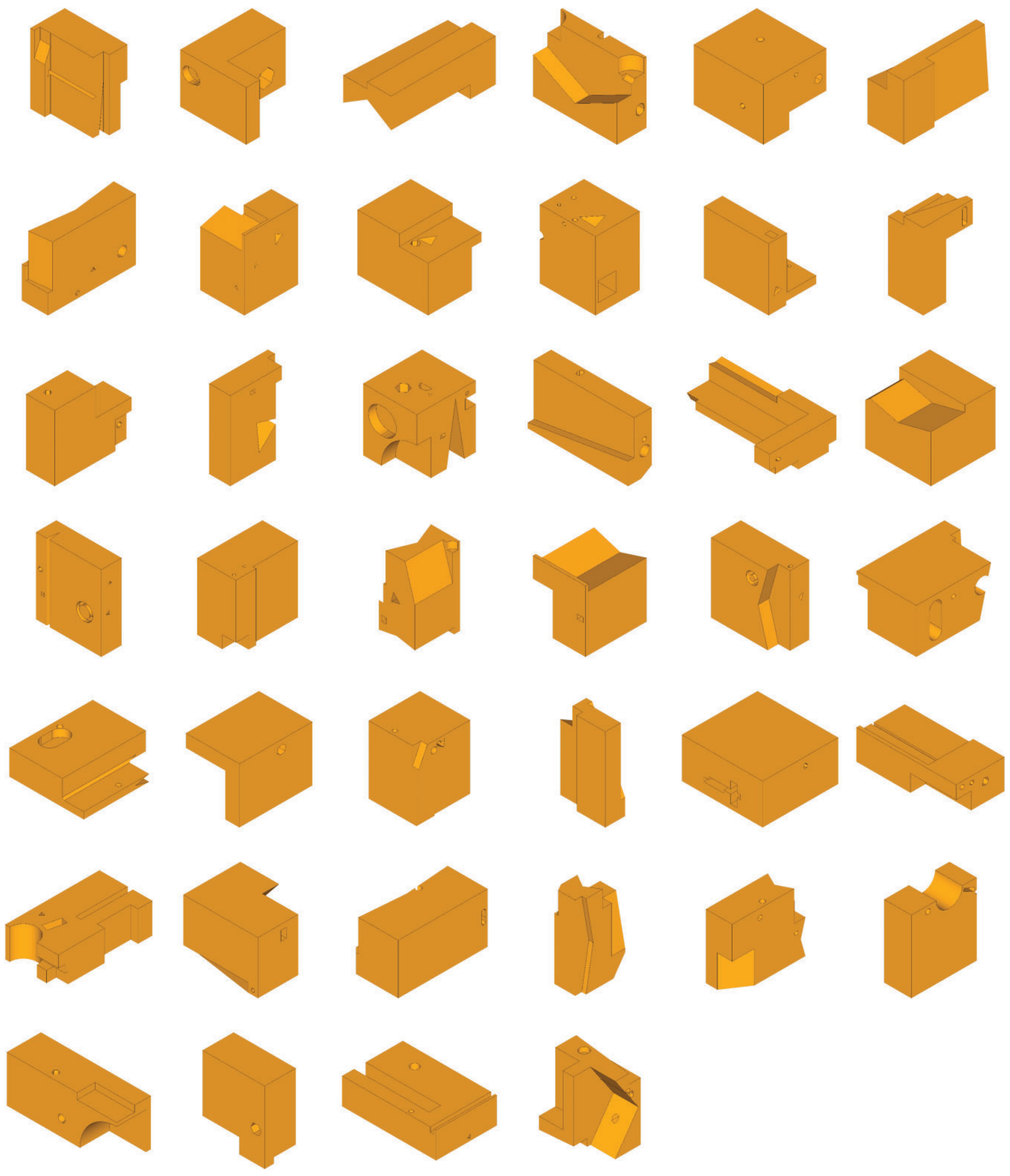


Figure A.1. Test case 1.

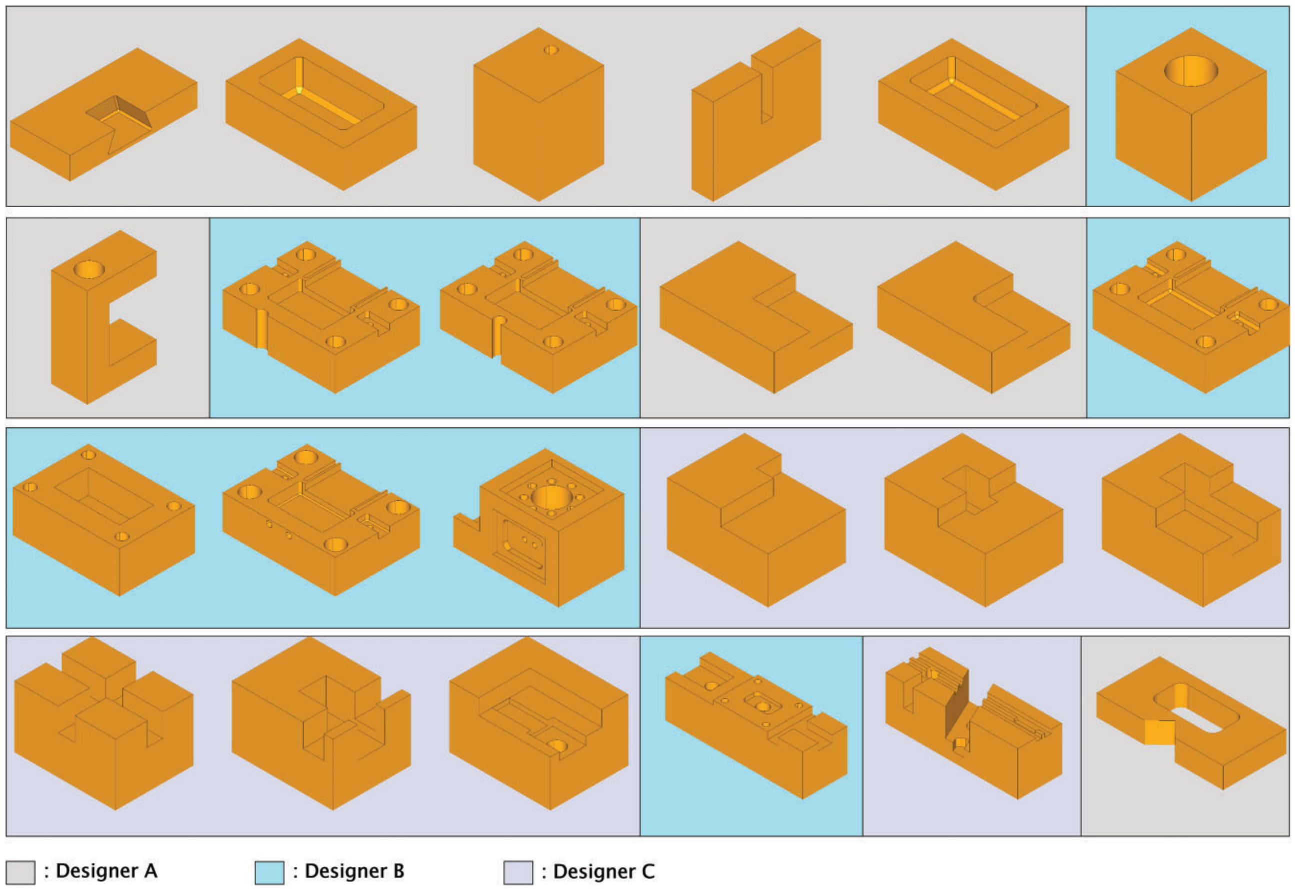


Figure A.2. Test case 2.

Received: May 26, 2023. **Revised:** November 18, 2023. **Accepted:** November 19, 2023