Taylor & Francis
Taylor & Francis Group

International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: www.tandfonline.com/journals/tprs20

Machine learning in feature recognition for manufacturing: taxonomy, analytical review, comparisons, trends, challenges, and outlook

Peizhi Shi, Yuchu Qin, Fanlin Meng, Paul J. Scott & Xiangqian Jiang

To cite this article: Peizhi Shi, Yuchu Qin, Fanlin Meng, Paul J. Scott & Xiangqian Jiang (23 May 2026): Machine learning in feature recognition for manufacturing: taxonomy, analytical review, comparisons, trends, challenges, and outlook, International Journal of Production Research, DOI: 10.1080/00207543.2026.2675460

To link to this article: https://doi.org/10.1080/00207543.2026.2675460

INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH
https://doi.org/10.1080/00207543.2026.2675460

Machine learning in feature recognition for manufacturing: taxonomy, analytical review, comparisons, trends, challenges, and outlook

Peizhi Shi[a], Yuchu Qin[b], Fanlin Meng[c], Paul J. Scott[b] and Xiangqian Jiang[b]

[a]Centre for Decision Research, Leeds University Business School, University of Leeds, Leeds, UK; [b]EPSRC Future Advanced Metrology Hub for Sustainable Manufacturing, University of Huddersfield, Huddersfield, UK; [c]University of Exeter Business School, University of Exeter, Exeter, UK

ABSTRACT

Feature recognition is an important topic in the area of intelligent manufacturing and production research. This technique is utilised to extract and recognise functional components and geometric shapes, such as slots, holes, and pockets, from solid models. The term 'learning-based feature recognition' refers to the feature recognition methods implemented based on machine learning. This topic was examined three decades ago but has been growing quickly in the recent five years. In this paper, a taxonomy and review of learning-based feature recognition methods are presented. This paper categorises the literature, examines basic components of a learning-based feature recognition method, identifies the technical evolution of these components, examines their impact on the overall feature recognition process, provides a broader overview of existing methods, identifies the milestone learning paradigms, makes comparisons among different approaches, assesses their practical capabilities, and identifies the gaps between research and real-world applications. By reading this paper, researchers and industry professionals can understand different learning-based feature recognition methods and their underlying principles, gain insights into design decisions made in a learning-based system, and understand their applied situations.

ARTICLE HISTORY
Received 10 June 2025
Accepted 8 May 2026

KEYWORDS
Intelligent manufacturing; production research; feature recognition; machine learning; taxonomy; analytical review

1. Introduction

In the area of geometrical product manufacturing, feature recognition is an essential topic. This technique is adopted to extract and recognise functional components and geometric shapes from raw engineering drawings or 3D models. These functional components and geometric shapes are called manufacturing features. Based on the recognised features from the 3D models, the manufacturing methods, operations, tools, and tool paths can be properly planned (Garcia et al. 2011; Y. Zhang, Zhang, Huang, et al. 2022). Automatic feature recognition has great potential to enable the automation of the manufacturing process and to increase production efficiency and adaptability. Within the broader manufacturing workflow, it functions as an integral component alongside manufacturing execution, and quality assurance (Papavasileiou et al. 2022; Papavasileiou, Michalos, and Makris 2025).

Over the past few decades, various automatic manufacturing feature recognition approaches have been proposed. In most of these methods, human experts typically encode their domain-specific knowledge into a couple of heuristic rules (Babic, Nesic, and Miljkovic 2008; Han, Pratt, and Regli 2000). Then, these handcrafted rules are utilised to analyse the topological and geometrical structures of the 3D models, and identify manufacturing features accordingly. These methods have been successfully applied in numerous cases, especially for recognising single and isolated features. However, in complicated cases, a 3D model normally comprises multiple manufacturing features that can intersect with each other. As a result, the topological and geometrical structures of these features are fundamentally changed due to feature intersection. It might be challenging to formulate robust rules to handle diverse complicated 3D models based on domain knowledge.

In the past several years, machine learning (ML) approaches become popular and have been extensively adopted in different research areas, such as production planning (Tremblet, Thevenin, and Dolgui 2024), fault diagnostics (Singh et al. 2024), quality assessment (Stavropoulos et al. 2022) as well as intelligent manufacturing (Rai et al. 2021). The ML algorithms are capable of acquiring domain knowledge automatically from the data

CONTACT Peizhi Shi p.shi@leeds.ac.uk Centre for Decision Research, Leeds University Business School, University of Leeds, Leeds, LS2 9JT, UK

Supplemental data for this article can be accessed online at http://dx.doi.org/10.1080/00207543.2026.2675460.

without too much human intervention. This data-driven nature shows the great potential to solve the inherent challenges in traditional methods, such as manual rule definition process, limited generalisation and recognition performances for complicated 3D models, and the need for domain knowledge. Although learning-based approaches become more and more popular in this topic, their full potential has not been reached, the learning paradigms have not been fully examined, and a number of practical research questions have not been addressed yet. To this end, it would be beneficial to provide a taxonomy and review of these methods to fully understand the methodology and its research trend, and outlook the future directions of the topic.

Since the 2000s, a number of good surveys have been presented to review manufacturing feature recognition methods. Babic, Nesic, and Miljkovic (2011) makes an overview of shallow learning-based feature recognition methods before 2011. Verma and Rajotia (2010) and Y. Shi et al. (2020) focus on general feature recognition algorithms where both rule- and shallow learning-based methods have been discussed. Three additional review papers (Babic, Nesic, and Miljkovic 2008; Han, Pratt, and Regli 2000; Subrahmanyam and Wozny 1995) summarise the development of traditional rule-based feature recognition methods. Deep learning-based approaches, which are state-of-the-art methods for feature recognition, have not been covered in the above review papers. To this end, this paper provides a broader overview of both deep and shallow learning-based methods before 2026. In addition to the difference in scope, there are also a lot of unaddressed questions in this research area:

- What are the basic components of a learning-based feature recognition method?
- How does each component interact with others, and affect the final feature recognition performance?
- How can the literature be categorised?
- What are the milestone methods in the past 30 years?
- What are the nature, benefits, limitations, practical capabilities, and technical evolution of different methods?
- What are the gaps between the research and real-world applications?
- What are the promising future directions of this area?

This paper aims to answer the above questions in depth. In doing so, the paper presents a taxonomy to categorise and organise existing learning-based feature recognition literature, dissects the components of learning-based methods, and examines their impacts on the overall feature recognition process. Additionally, the paper makes a comparison between different approaches, analyses the technical evolution, and discusses the milestone techniques in this area. Last but not the least, the paper identifies the gaps between existing methods and real-world applications and proposes a few research directions.

The rest of the paper is structured as follows. Section 2 presents the methodology used to conduct this review. Section 3 offers a general overview of the background of feature recognition. Section 4 dissects the components of existing learning-based feature recognition methods, and provides a taxonomy of them. Section 5 offers an analytical review of existing approaches, and discusses the technical evolutions. Section 6 discusses the remaining issues in this area and identifies a number of research directions. The topics covered in this survey are shown in Figure 1.

2. Review methodology

As discussed in the previous section, this study aims to present a taxonomy, review, and comparison of existing learning-based feature recognition approaches, identify the milestone techniques in this area, analyse the technical evolution of different approaches, and identify the gaps between existing methods and real-world applications. To attain this goal, this paper utilises a systematic approach to collect relevant papers and form insights accordingly.

In this approach, relevant papers were searched and collected from Google Scholar and Web of Science. The keywords utilised for searching are [‘manufacturing’ AND ‘feature recognition’ AND (‘learning’ OR ‘neural network’)] before 2016, and [‘manufacturing’ AND ‘feature recognition’ AND ‘machine learning’] after 2016. These different keywords are selected as researchers did not widely adopt the term ‘machine learning’ in their publications before 2016. The peer-reviewed English articles that involve using ML techniques for feature recognition were manually selected. After applying the above search criteria, this study involves 74 technical articles in total. Figure 2 illustrates the trend of publications in different time periods. From the 1990s to 2018, limited research on learning-based feature recognition was conducted. During this period, researchers normally manually designed attributes to represent solid models and utilised shallow ML models for feature recognition. The year 2018 could be regarded as a turning point, as deep learning approaches were introduced into this area. During this period, researchers started to explore more diverse data representations (e.g. voxel, image, text, point cloud, mesh, and graph) coupled with other deep learning approaches (e.g. 3D shape classification, object

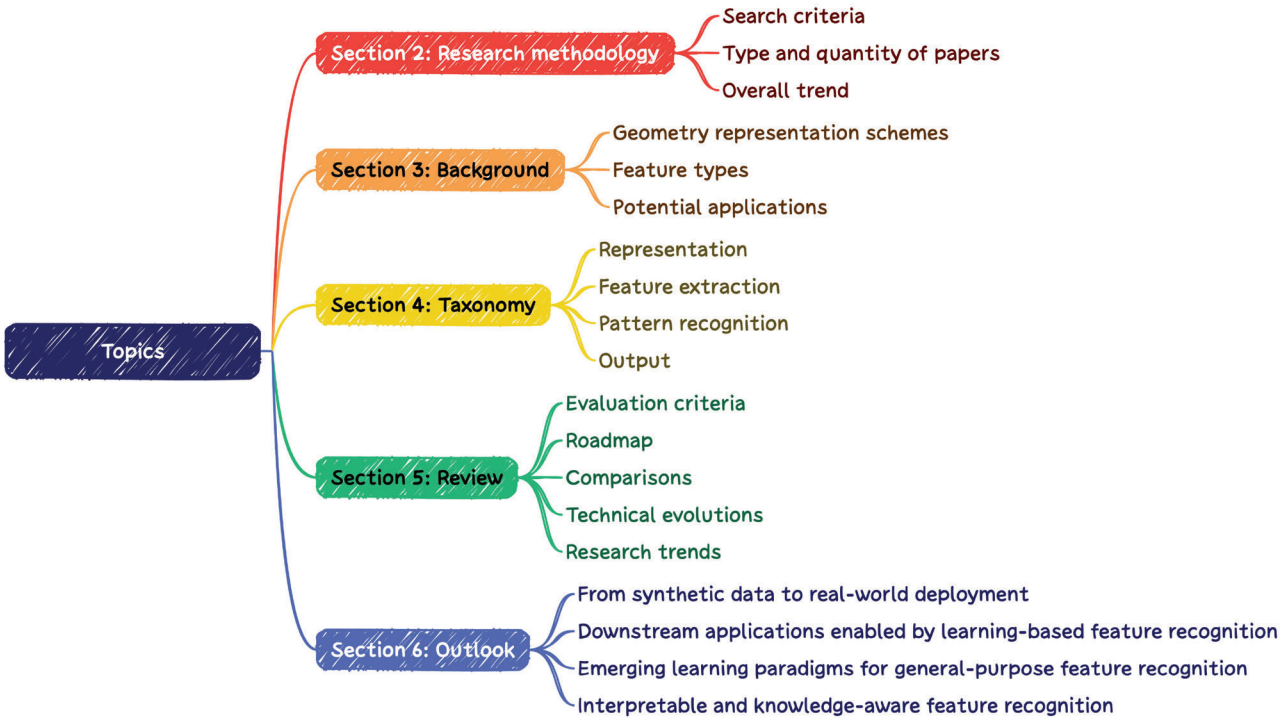

Figure 1. The topics covered in this paper.

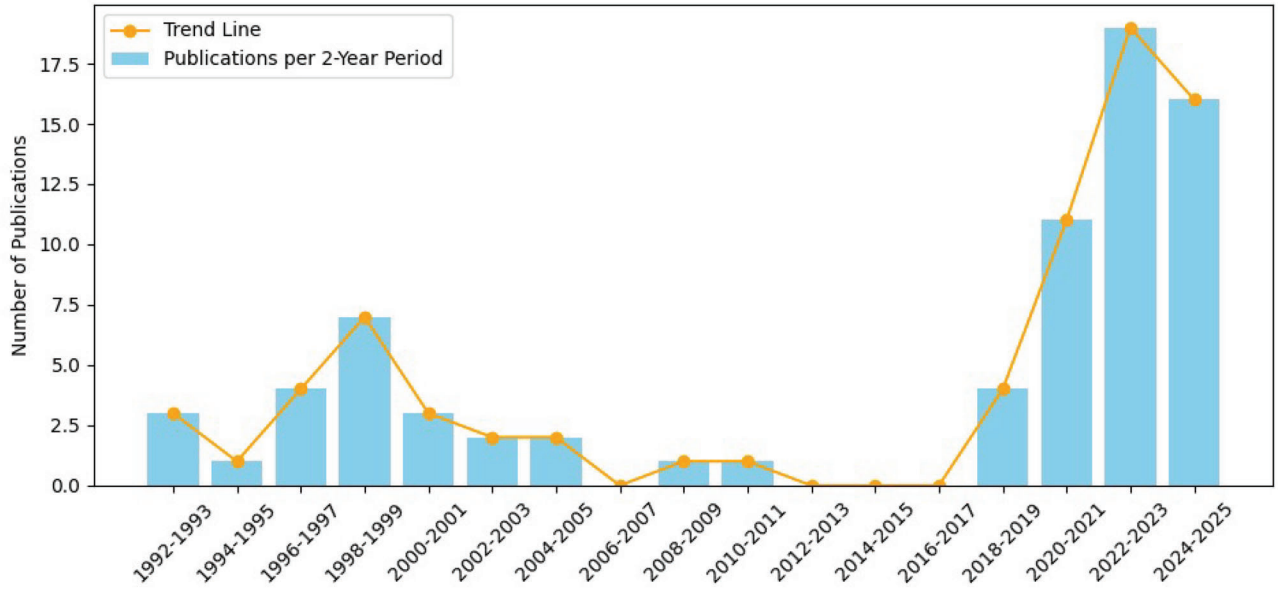

Figure 2. The number of publications per 2-year period.

detection, semantic and instance segmentation) for feature recognition. More diverse features and complicated feature intersection cases could be handled properly. It is observed from Figure 2 that the number of publications in this time period is twice that of those published before 2018, and the trend is increasing. It shows the great potential of learning-based approaches to feature recognition and their applications in industry.

The geographical distribution of publications is illustrated in Figure 3. Overall, the existing literature is mainly produced in a few regions. From a continental perspective, Asia, Europe, and North America together account for the vast majority of the published work. Among these regions, Asia has the largest number of publications, with 45 studies. Europe follows with 16 publications, and North America accounts for 10 publications. Africa reports 2 publications, and Oceania reports 1 publication.

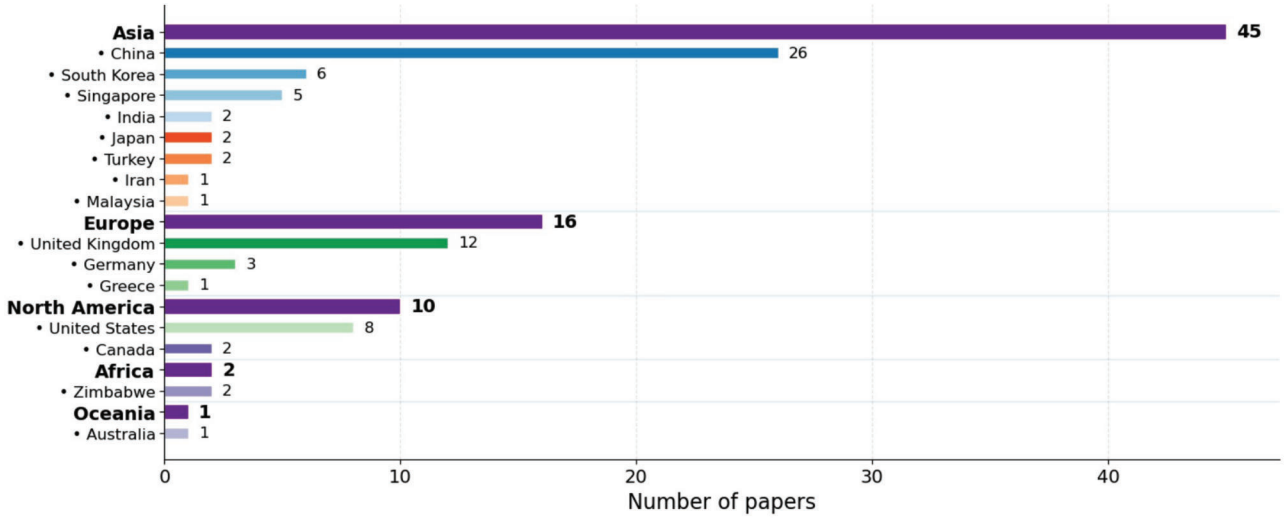

Figure 3. Geographical distribution of publications.

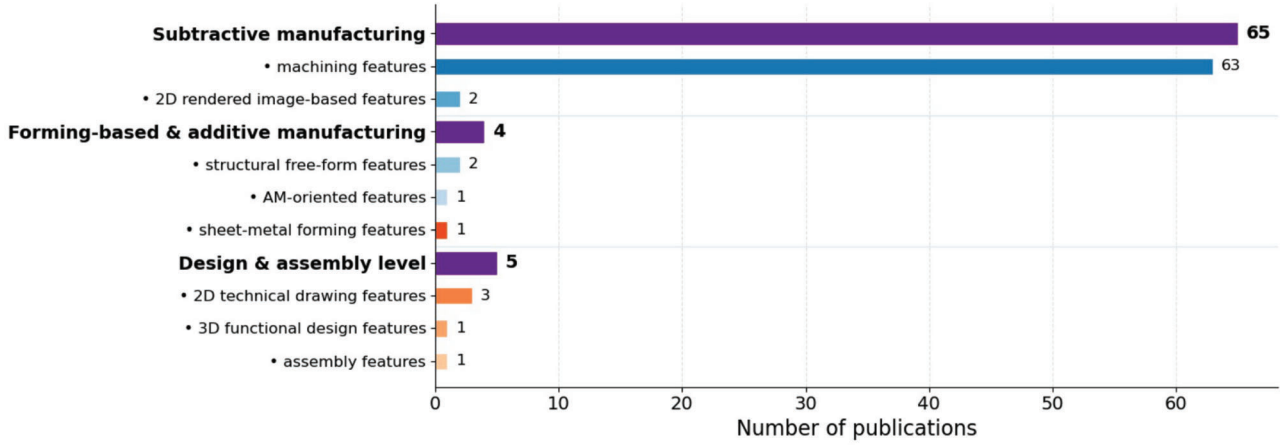

Figure 4. Distribution of publications across different geometrical product manufacturing domains.

Figure 4 shows the distribution of the reviewed studies across different geometrical product manufacturing domains using a two-level taxonomy. The first level represents the manufacturing domain, and the second level indicates the specific feature type. It is observed that most works focus on machining feature recognition in subtractive manufacturing. This is mainly because machining features, such as holes, slots, and pockets, serve as core abstractions in many downstream applications and are naturally embedded in subtractive workflows. In contrast, forming-based, additive, and other manufacturing types have received less attention. In these domains, potential reasons include the fact that decisions are often driven by global shape properties, and some features are less standardised or harder to define. Several studies also address design-level and assembly-level features. Overall, the field remains dominated by subtractive manufacturing. Other domains are still relatively underexplored.

3. Background

According to Prabhakar and Henderson (1992), features refer to the geometric and topological patterns in a solid model. The feature recognition task aims to capture the type, location, and geometric and topological information about features in a given solid model. Figure 5 illustrates the diagram of a typical feature recognition process. This section briefly discusses these relevant components of a feature recognition system.

In the area of feature recognition, an essential problem is how to represent a 3D part that is to be manufactured. The chosen geometry representation scheme is supposed to involve the essential topological and geometrical information about the 3D part. Two types of schemes have been widely utilised in existing learning-based feature recognition methods, i.e. boundary representation (BRep) and triangular mesh representation. In BRep, a 3D model is defined by its boundaries and surfaces. The basic elements of this representation include

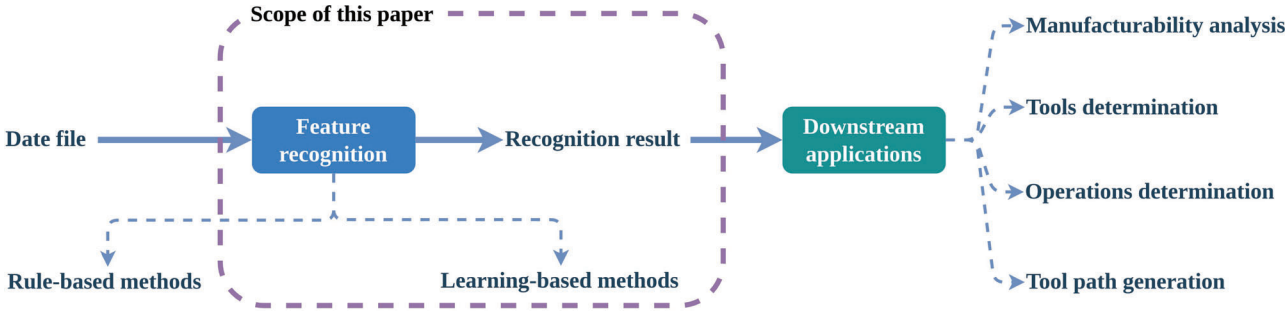

Figure 5. Feature recognition process.

items, such as vertices, edges, and faces. A vertex is represented as a 3D point, an edge is defined as a curve bounded by two vertices, and a face is defined by its boundary edges. The surfaces are defined as mathematical equations. This representation is normally stored in a STEP file. Triangular mesh representation is another scheme. It represents a 3D object as a set of triangular meshes. These meshes are employed to approximate the surfaces of the 3D object. Each triangular mesh is represented by three vertices, and each vertex is described using its 3D coordinates. This representation is normally stored in an STL or OBJ file. More comprehensive information can be found in Qin et al. (2019).

As discussed previously, features are the geometric and topological patterns in a solid model. Topology refers to the connectivity of faces in the solid model, whereas geometry refers to the shape and appearance of the model. Therefore, another essential issue is how to categorise different feature types. One distinction among different types of features lies in their shapes. Based on the geometrical shapes, features can be categorised as regular and free-form features. Regular features contain pre-defined geometric shapes. These features are easy to manufacture using methods like machining, casting, or forging. In contrast, free-form features consist of smooth and irregular shapes and do not involve pre-defined geometric shapes. Solid models with free-form features can be produced via manufacturing technologies like 3D printing, casting, or forging.

4. Taxonomy

4.1. Overview

In the learning-based feature recognition approaches, the learning paradigms are realised through different ML system designs and processing pipelines. These design choices define how feature recognition is organised and implemented in practice. In these approaches, the ML models embedded in such systems are trained on computer aided design (CAD) data and utilised to recognise manufacturing features automatically.

Figure 6 illustrates typical learning-based feature recognition system architectures from a system design perspective. These methods take data with manufacturing features as input and produce the recognition results accordingly. It is observed that these methods could be grouped into two categories. In the first category, manufacturing features appearing in the input data are first extracted via a feature extractor and served as intermediate results. Then, a learning-based pattern recognition component is utilised to predict the type of extracted features. This approach is normally called the two-stage method. In the second category, manufacturing features are extracted and recognised simultaneously via a single ML model. This approach is normally called the one-stage method (P. Shi et al. 2020a).

A learning-based manufacturing feature recognition algorithm typically comprises four components: input data, feature extraction, pattern recognition, and output results. This section aims to dissect each component and provide a taxonomy of the literature pertaining to each component, as shown in Figure 7.

4.2. Data representation

A main design decision in a learning-based manufacturing feature recognition system is how to represent the data for feature extraction and pattern recognition tasks. From an ML system design perspective, the chosen data representation defines the interface through which geometric and topological information is made accessible to learning algorithms. In other words, the key research question is how the 3D data formats reviewed in Section 3 are mapped to representations that can effectively support learning under different feature extraction and pattern recognition formulations. A good data representation is therefore expected to preserve the information required for learning-based feature extraction and recognition tasks (Y. Shi, Zhang, and Harik 2020). This section

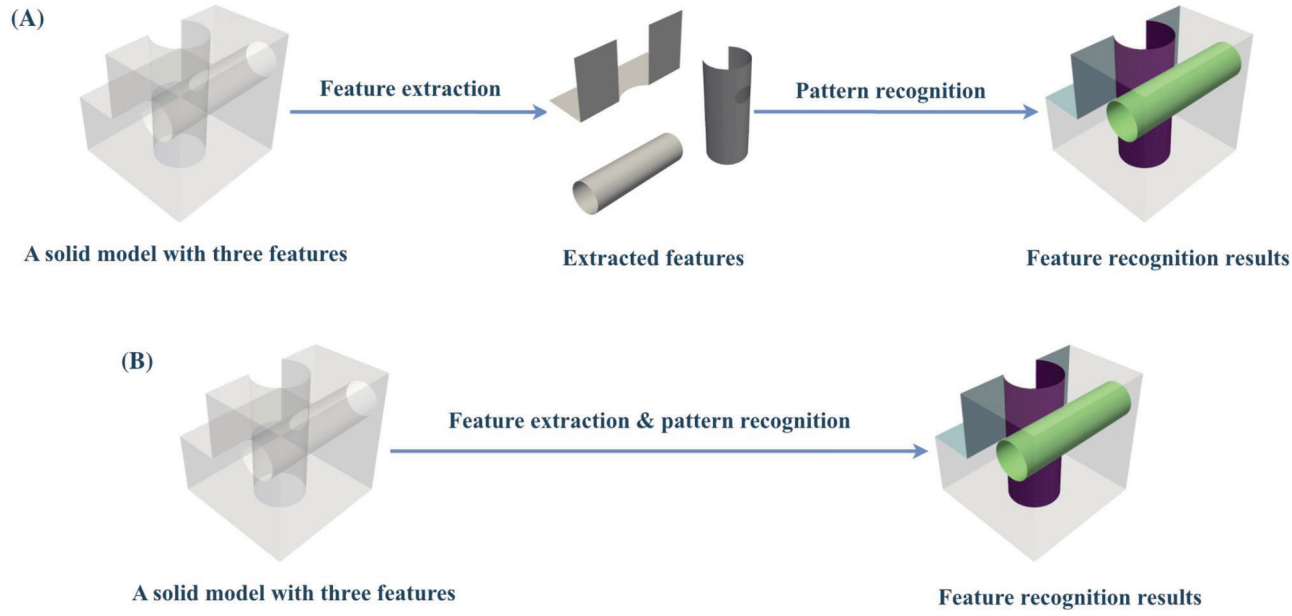

Figure 6. Two categories of learning-based feature recognition methods: (A) two-stage approaches, and (B) one-stage approaches.

provides an overview of the commonly used representation strategies.

4.2.1. Point cloud-based representation

As shown in Figure 8(a,b), a solid model can be represented as a group of unordered 3D points, known as a point cloud. A point cloud for a 3D model is normally constructed by sampling points from the surfaces of the 3D model. The sampled points can be stored in an $N_p \times k$ matrix, where N_p refers to the number of points and k refers to the number of attributes for each point. The attributes could be positional coordinates of data points ($k = 3$ as in the label of Figure 8(b)).

4.2.2. Volumetric-based representation

A 3D model can also be voxelised into a fixed-size 3D voxel grid, as shown in Figure 8(c). The voxel grid can be stored in a $Dim \times Dim \times Dim \times k$ matrix, where Dim refers to the dimension along one axis. Each cell of the matrix can either be a binary variable (i.e. $k = 1$ as in Figure 8(c)), indicating whether the cell is occupied, or an attribute vector ($k > 1$) providing more information.

4.2.3. Mesh-based representation

A 3D model can also be represented as a collection of triangular meshes, as shown in Figure 8(d). Each triangular mesh can be derived from the facets of the solid models and represented as three vertices. The meshes can be stored in an $N_m \times 9$ matrix, where N_m refers to the number of meshes, and 9 refers to the coordinates of the three vertices.

4.2.4. Graph-based representation

A 3D model is composed of a number of surfaces and edges. As shown in Figure 9, the solid model with a BRep representation includes 11 surfaces. Surfaces 3 and 7 are cylindrical faces, surfaces 2, 8, and 10 are planar faces for features, and the rest are planar stock faces. A straightforward way to encode the edge and surface information is to construct a graph to illustrate the relationships among different surfaces and edges. Each node in the graph represents a surface of the 3D model, whereas each arc represents the edge connecting two faces. An attribute vector can be associated with each node and arc of the graph. The attribute vector can indicate the properties of the corresponding surface and edge, such as surface type, concavity, and convexity between surfaces.

4.2.5. View-based representation

A 3D model can also be indirectly represented as a collection of 2D view images. Each image contains partial information about the feature(s). Figure 10 illustrates one type of view-based representation. In this format, a number of projected view images are captured from different angles of the solid model and used to represent the data. The view images can be stored in a $6 \times H \times W$ matrix, where 6 refers to six view directions, and H and W refer to the height and width of the images.

4.2.6. Attribute array-based representation

A solid model can also be indirectly represented as an attribute vector or matrix. In this representation, a number of attributes are summarised from the solid model. It could be stored in a k-D vector, where k refers to the

Figure 7. The taxonomy of learning-based feature recognition methods.

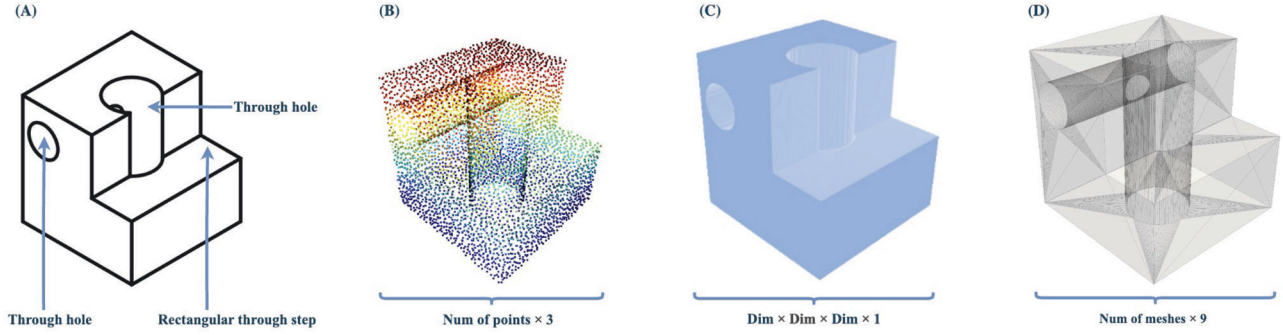

Figure 8. A solid model and four corresponding data representations: (a) a solid model with three features, (b) point cloud-, (c) volumetric-, and (d) mesh-based representation.

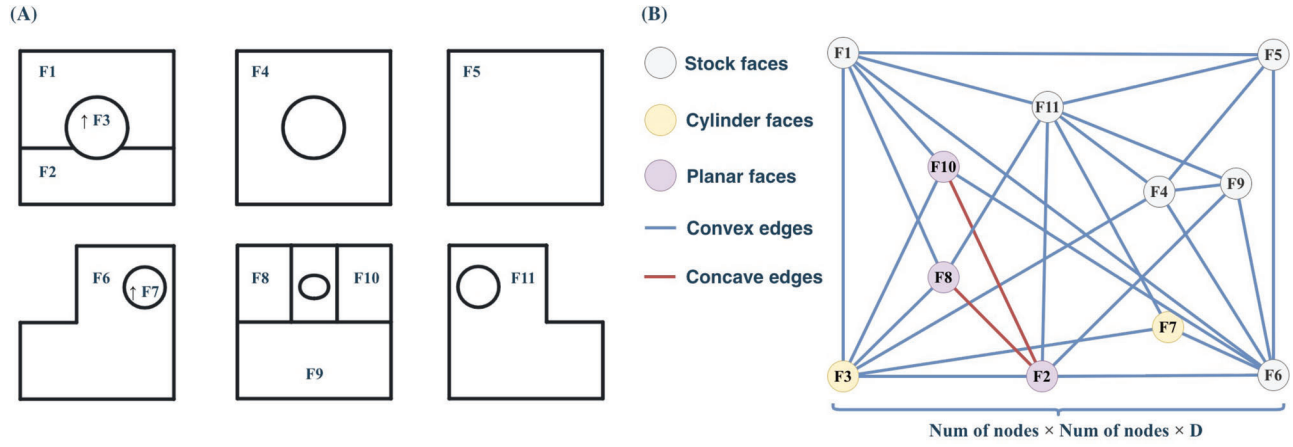

Figure 9. An example of a graph-based representation. The surfaces in the solid model are marked as F1–F11.

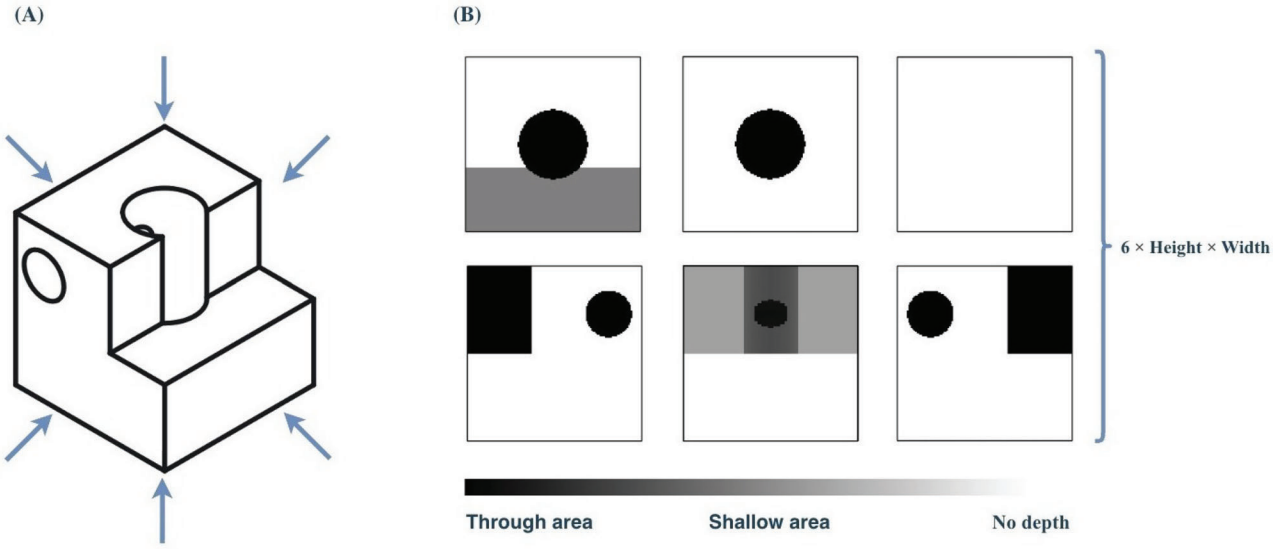

Figure 10. An example of a view-based representation.

number of attributes. Each attribute shows a key property of the manufacturing feature, such as geometric details, topological structure, and relationships among different faces.

4.2.7. Text-based representation

A solid model can also be represented using a text-based format, as exemplified in Figure 11. In this representation, different components (e.g. closed shell, face, surface, face bound, edge, circle) of a solid model are extracted from a STEP file and organised hierarchically in textual format. Each node in the tree represents a component of the 3D model. For instance, node 11 refers to the AXIS2_PLACEMENT_3D component, which defines a coordinate system in the CAD model. Nodes 12–14 refer to the origin, axis, and reference direction of the system, respectively. These four components are organised in a tree-like structure.

4.2.8. BRep-based representation

A solid model can also be represented using the raw BRep format directly. Compared with the aforementioned representations, the raw BRep format contains rich information, such as surfaces, loops, and co-edges. The surfaces of the 3D model are described using mathematical equations. Therefore, the geometric structures of the manufacturing features can be represented precisely. The relationships between different surfaces accurately reveal the topological structures of the features.

4.2.9. Brief analysis and summary

The aforementioned representations are widely utilised in learning-based feature recognition systems and exhibit distinct benefits and limitations when viewed from an ML system design perspective. These trade-offs are summarised in Table 1. In the subsequent sections, these representation-level design choices are used as a basis for

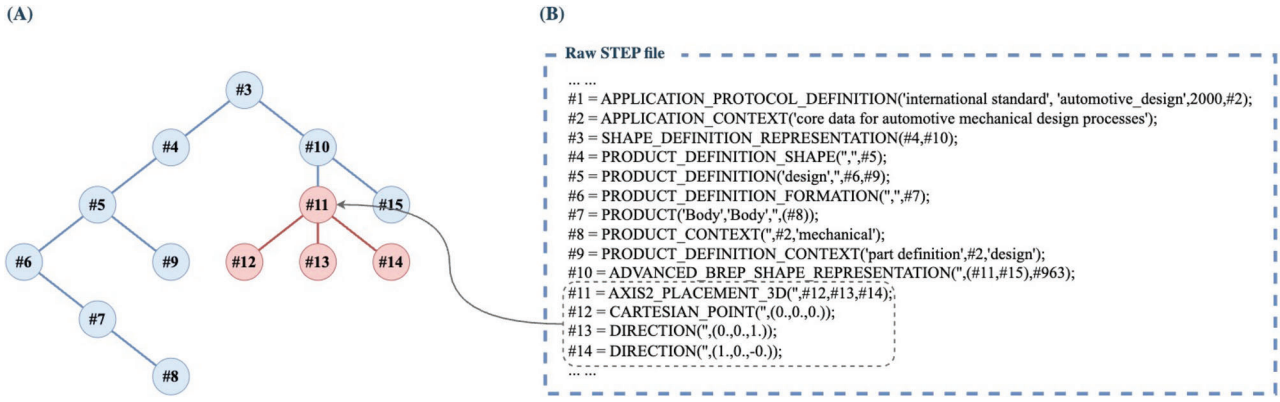

Figure 11. Text-based representations.

Table 1. Comparison of data representations from an ML system design perspective for feature recognition.

Data representation	Benefits and limitations
Attribute array	+Shallow learning model is suitable for this representation
	−Human effort and domain knowledge are needed to design attribute arrays
	−The attribute arrays are normally ad-hoc to the feature types
Graph	+Each graph node directly links to a face in the 3D model (Cao et al. 2020)
	+It contains rich topological information about the 3D features (H. Wu et al. 2024)
	+It contains useful information for feature extraction, such as the convexity and concavity between faces
	−Human effort is needed to design attribute arrays for the graph nodes and arcs
Mesh	+It contains richer geometric information than the graph-based representation
	+Each mesh directly links to a facet in the 3D model (Colligan et al. 2022)
	−It is required to determine the appropriate mesh resolution
Volumetric grid	+It is suitable to represent noisy or imperfect data (Z. Zhang, Jaiswal, and Rai 2018)
	−It lacks adequate geometric and topological details (H. Wu et al. 2024)
	−It requires a large amount of memory for data storage
	−The uniform resolution may not be sufficient to represent both small and large features
	−Tiny features have the potential to be filtered out during voxelisation
View image	+It is more memory efficient than 3D volumetric grid (P. Shi et al. 2020b)
	+Methods used in the field of computer vision can be directly applied to this representation
	−It lacks adequate geometric and topological details (H. Wu et al. 2024)
	−Projected view images are only suitable for representing 2.5D features
Text	+It is robust to features of different sizes (Miles, Giani, and Vogt 2023a)
	+Methods used in the field of natural language processing can be directly applied to this representation
	−Human effort is needed to design the structure of text (Miles, Giani, and Vogt 2023b)
	−It is unclear how to extract features from the text
Point cloud	+This representation is memory-efficient
	+It contains rich geometric information, such as shape, local structures, and fine-grained details of the features
	+Each point in the cloud directly links to a face in the 3D model (H. Zhang et al. 2022)
	−The recognition is sensitive to the sampling strategy (Vidanes et al. 2024)
	−Small features have the potential to be undersampled (Vidanes et al. 2024)
	−The point cloud is a set of unordered and unconnected 3D points, which lacks adequate topological details

Notes: '+' and '−' indicate relative strengths and limitations in supporting learning-based feature extraction and recognition.

analysing and comparing different learning-based feature recognition approaches.

4.3. Feature extraction

As reviewed in Section 4.1, a key problem in a learning-based feature recognition system is how manufacturing features are extracted from a 3D model to support subsequent pattern recognition. From an ML system design perspective, feature extraction determines how learning signals are organised and exposed to the pattern recognition component. This section therefore provides a taxonomy of the methods and strategies adopted in the feature extraction component under different learning paradigms. In this taxonomy, we assume that the feature extraction component is mainly applied to multi-feature solid models, as extracting features from single-feature models is a trivial task. It is notable that, in one-stage methods, feature extraction and pattern recognition are conducted simultaneously. Therefore, for these one-stage methods, this section focuses on their feature extraction properties.

4.3.1. Extraction method

Different feature extraction methods can be viewed as concrete instantiations of different learning paradigms. They reflect how supervision signals and inductive biases are incorporated into the extraction process. This section

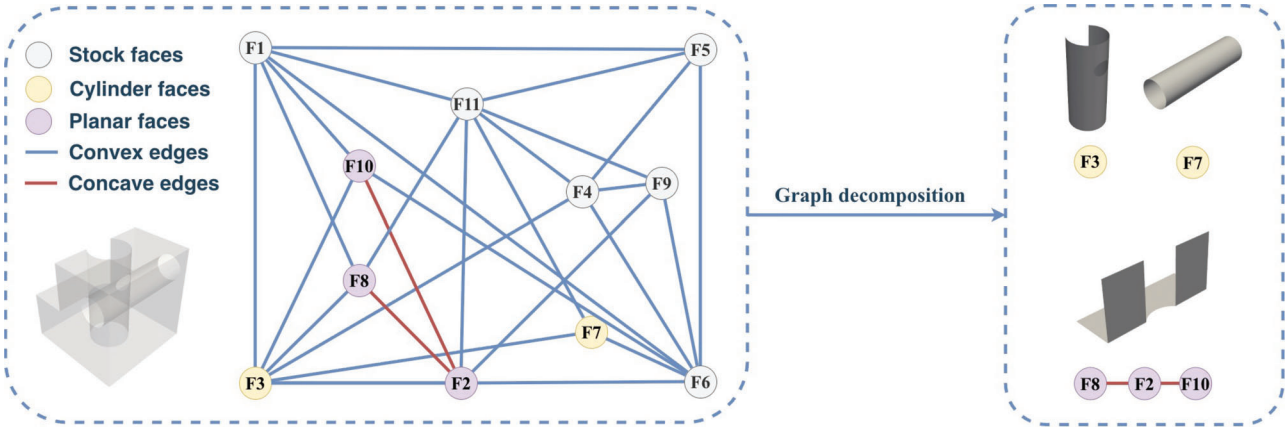

Figure 12. Feature extraction via graph decomposition rules.

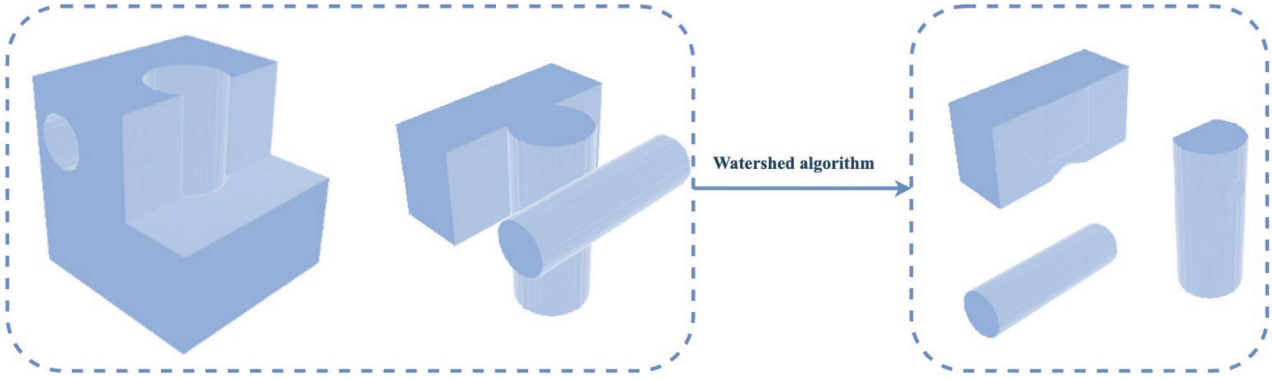

Figure 13. Feature extraction via an unsupervised learning algorithm.

makes an overview of these approaches from the perspective of ML system design.

4.3.1.1. Rule-based feature extraction. In this type of method, human experts encode their domain knowledge about manufacturing features (e.g. concavity and convexity properties of features, and feature intersections) into a number of ad-hoc rules. As exemplified in Figure 12, a solid model with multiple features is represented as a graph. Then, graph decomposition rules are adopted to extract features accordingly. The rules adopted in this example could be: (1) removing all convex edges, and (2) removing all stock faces.

4.3.1.2. Unsupervised learning-based feature extraction. Alternatively, the feature extraction task could be tackled via unsupervised learning algorithms. These methods aim to analyse the structure within the 3D model without explicit supervision. As shown in Figure 13, the solid model is represented as a 3D voxel grid. Then, its overall removal areas are captured and further segmented into different features via the watershed algorithm.

4.3.1.3. Reinforcement learning-based feature extraction. This type of method aims to uncover the structure within the 3D model by using indirect reward functions. These rewards could be designed based on different considerations, such as manufacturing process constraints (H. Zhang, Wang, Zhang, Zhang, et al. 2024). They are employed as hints, and utilised to guide the learning algorithm to segment features from the solid models.

4.3.1.4. Supervised learning-based feature extraction. In this type of method, the location and type of each feature that appears in the solid models are annotated. Then, a supervised learning algorithm is utilised to learn how to extract features by using annotated data. Once the ML model is constructed, it can extract manufacturing features from unseen data. As shown in Figure 14, the surfaces in the solid model are first annotated. Then, a supervised segmentation method is adopted to learn from the data and used to predict the label for each surface in the unseen data.

4.3.1.5. Brief analysis and summary. The above feature extraction methods have different benefits and limitations from an ML system design perspective. As

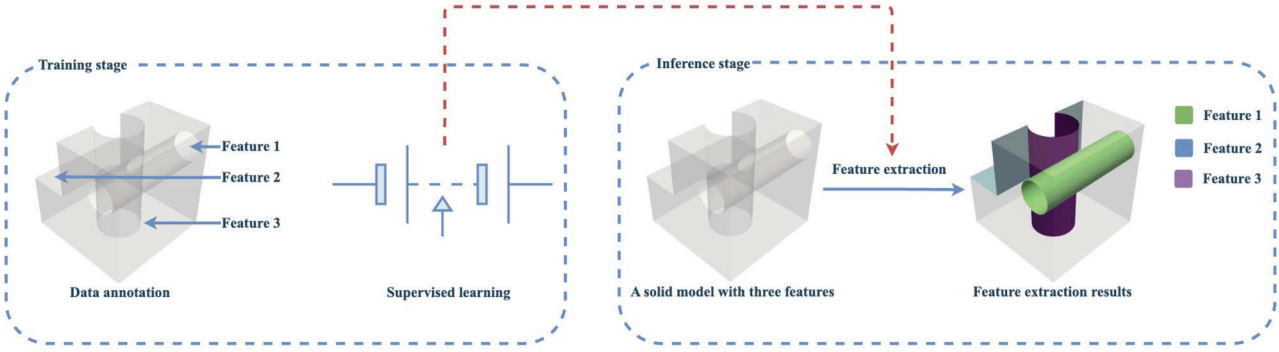

Figure 14. Feature extraction via a supervised learning algorithm.

Table 2. Comparison of feature extraction methods from an ML system design perspective.

Feature extraction methods	Supervision / human efforts	Benefits and limitations
Rule-based	Explicit human-defined rules (domain knowledge, heuristic design)	+It is suitable for extracting isolated or specific overlapped features +Designers or users have control over the feature extraction process −Domain knowledge and efforts are required to design reliable rules −Its performance on intersecting feature extraction is not satisfactory −Rule designers need to consider all scenarios on feature intersection
Unsupervised learning-based	Implicit structural signals (inductive bias, model selection)	+It is suitable for segmenting features with well-defined boundaries +It is useful for segmenting features with a small degree of overlapping +Domain knowledge is not necessarily required −Feature extraction is controlled by the inductive bias of ML model −Its performance on intersecting feature extraction is not satisfactory
Reinforcement learning-based	Indirect reward signals (reward function design)	+Feature extraction is indirectly controlled by the reward function −Domain knowledge is required to design good reward function −Training process is time-consuming
Supervised learning-based	Explicit labelled supervision (data annotation)	+Its performance on intersecting feature extraction is remarkable +Designers have control over the feature extraction results −Data annotation process is a time-consuming task

summarised in Table 2, these differences lie in supervision requirements, human effort, inductive bias, and applied situations. Rule-based methods rely on explicit domain knowledge. This property makes them suitable for features with specific topological structures. Unsupervised learning-based methods utilise inherent geometric structures in the data and are suitable for features with well-defined boundaries. Supervised methods learn from annotated data. They show strong performance on complex and intersecting feature cases. Reinforcement learning-based methods incorporate indirect supervision through reward functions that reflect manufacturing constraints. In Section 5, these extraction-level design choices are further considered when analysing complete learning-based feature recognition systems.

4.3.2. Extraction strategy

The strategies for feature extraction could be further divided into two categories, namely top-down and bottom-up approaches (see Table 3 for a comparison). In the top-down approaches, all the faces in the solid model start in one group and are then split into different groups via feature extraction algorithms. Each group is regarded as a potential feature. Examples include the graph decomposition and watershed algorithms mentioned above. In the bottom-up approaches, each face in the solid model starts in its own group, and then the label of each face is captured via the intelligent system. Faces with the same labels are grouped together. An example is the segmentation algorithm mentioned above.

Table 3. Analysis of different extraction strategies.

Feature extraction strategies	Benefits and limitations
Top-down extraction	+It is a straightforward and widely used method to segment features
Bottom-up extraction	+It shares the benefits of the segmentation-based extraction methods −A high per-face classification accuracy is required
Exhaustive extraction	+It potentially enumerates all possible features from the solid model −A post-processing step is required to eliminate redundant features
Singular extraction	+It is a straightforward and widely used method to segment features −This extraction strategy might overlook some features

Another distinction among different extraction strategies lies in whether each feature is extracted multiple times or once (see Table 3 for a comparison). The former is called exhaustive extraction in this taxonomy. It aims

to enumerate all possible features. The latter is called singular extraction in this taxonomy. It aims to have a one-to-one mapping between existing features and extracted features.

4.4. Pattern recognition

As overviewed in Section 4.1, another key problem in a learning-based feature recognition system is how the patterns of manufacturing features are recognised from the extracted geometric representations. From an ML system design perspective, pattern recognition determines how extracted features are mapped to semantic feature categories under different learning paradigms. This section reviews the methods adopted in the pattern recognition component and organises them according to their underlying learning paradigms. It is notable that feature extraction and pattern recognition tasks in one-stage methods are conducted simultaneously. Therefore, for these one-stage methods, this section focuses on their pattern recognition characteristics.

4.4.1. Shape classification-based methods

As discussed in Section 4.1, a straightforward way for pattern recognition is to construct a classifier that classifies the manufacturing feature(s) extracted from the 3D model. The input of the ML model is the extracted 3D feature obtained from the feature extraction component. The output is a label that shows the category of the extracted feature. This formulation follows a supervised learning paradigm. Its extracted features are mapped to predefined semantic categories by a classifier trained on labelled data. Such classification-based approaches are closely related to 3D shape classification methods in computer vision and are therefore most commonly adopted in two-stage feature recognition systems.

A typical example is volumetric grid classification (Z. Zhang, Jaiswal, and Rai 2018). As shown in Figure 15, the input of the ML model is the volumetric grid of the extracted feature, whereas the output is the predicted semantic label of the feature. Given the characteristics of this representation, a 3D convolutional neural network (CNN) could be used. This network involves a number of 3D convolution layers, pooling layers, flatten layers, and MLP layers. The 3D convolution layers apply the learnable filters to the volumetric grid and produce a number of feature maps that capture the 3D patterns from the input data. The pooling layer is adopted to reduce the dimensionality of the feature map, and the flatten layer is utilised to convert the 3D feature map into a 1D vector. Finally, the MLP layers are adopted to perform feature classification.

4.4.2. Object detection-based methods

In addition to the aforementioned classification-based methods, feature recognition could also be treated as an object detection problem. In this learning paradigm, the learning model takes a multi-feature solid model and predicts the bounding boxes of manufacturing features as well as their semantic labels, as shown in Figure 16. This category also follows a supervised learning paradigm. But it formulates pattern recognition as a joint localisation and classification problem.

Figure 16 illustrates a typical example. As shown in this figure, this method takes an image as input and identifies the locations of features appearing in the given 2D image. A modified version of the single shot multibox detector (SSD) (Liu et al. 2016) is adopted in this example. A number of 2D CNN blocks are adopted to extract multi-scale feature maps, and a prediction head is adopted to predict the bounding box offsets between default anchor boxes and recognised boxes, and the class label of the bounding box. Since the original SSD is designed to predict 2D boxes rather than 3D ones, the authors added another output channel to predict the depth of manufacturing features in the given image. Therefore, the final output of this approach is 2.5D bounding boxes of features.

4.4.3. Segmentation-based methods

In addition to the aforementioned methods, feature recognition could also be treated as a segmentation problem. In this learning paradigm, pattern recognition is formulated as a dense prediction problem, where supervision is provided at the face level. The ML model takes a solid model with multiple features as input and predicts the label (e.g. feature type) for each face in the solid model.

As shown in Figure 17, a typical method in this category is to utilise a graph to represent a multi-feature solid model. Each node in the graph represents a face in the solid model. Then, a graph segmentation network could be utilised for the graph node segmentation task, as illustrated in Figure 17. It takes the graph as input and predicts the feature type of each graph node. The faces sharing the same feature type will be regarded as a manufacturing feature. Graph neural networks (GNNs) are normally used to solve the graph segmentation problem. In each GNN layer, each node in the graph aggregates information from its neighbouring nodes. After several GNN layers, each node encodes information about its local region and is adopted for node classification.

4.4.4. Shape clustering-based methods

The aforementioned pattern recognition methods are constructed based on supervised learning algorithms. In

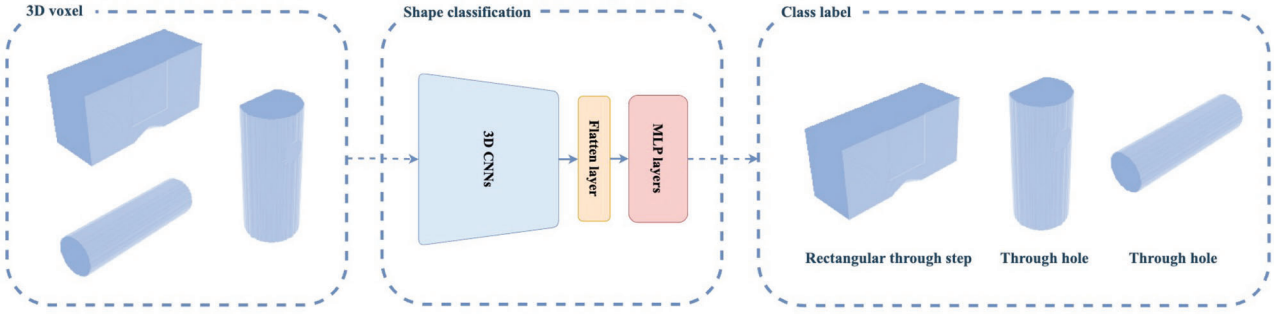

Figure 15. An example of the shape classification (adapted from Z. Zhang, Jaiswal, and Rai 2018).

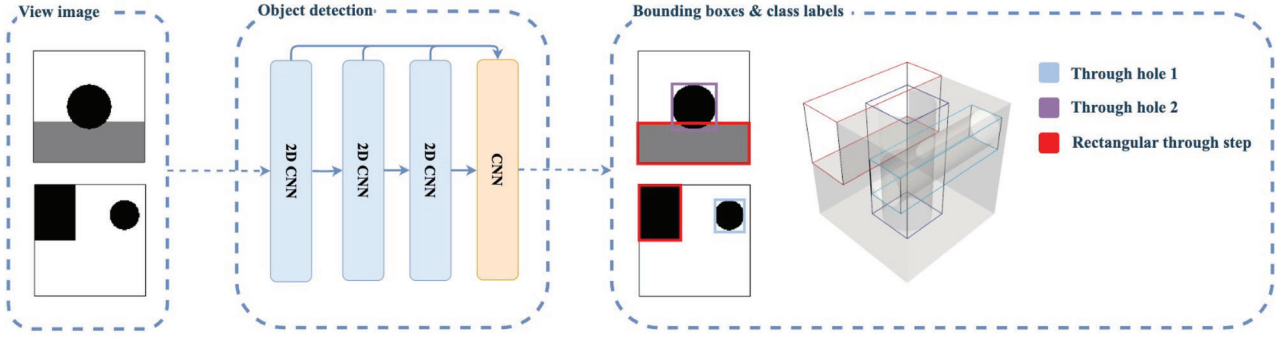

Figure 16. An example of the object detection-based approach (adapted from P. Shi et al. 2020a).

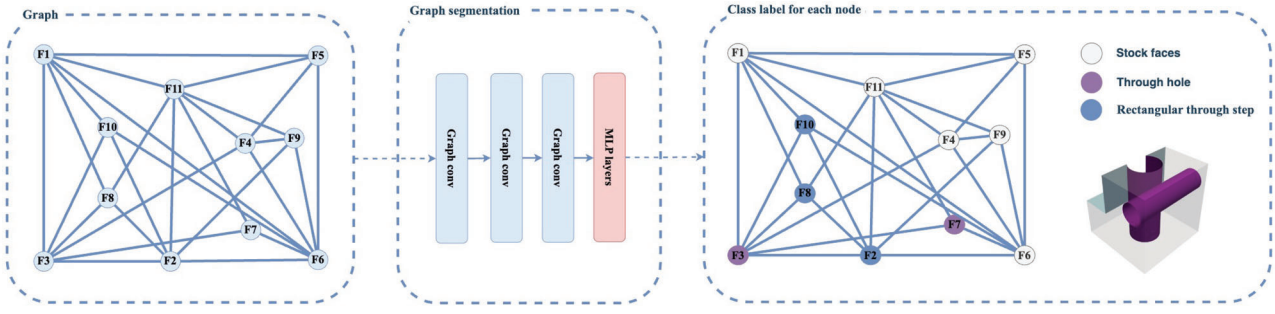

Figure 17. An example of the segmentation-based approach.

addition to these approaches, pattern recognition could also be conducted via an unsupervised method, e.g.clustering algorithms. In this learning paradigm, feature patterns are discovered based on similarity and structural regularities rather than explicit labels. The learning model takes a number of solid models as input and classifies these models into different clusters. Solid models within the same cluster are supposed to belong to the same feature class or family, as shown in Figure 18.

4.4.5. Brief summary and analysis

Shape clustering- and classification-based approaches are normally utilised in two-stage methods, while the other two approaches are widely adopted in one-stage methods. From an ML system design perspective, different pattern recognition methods correspond to different learning paradigms and supervision strategies. Table 4 summarises the main trade-offs among these paradigms. These trade-offs are discussed in terms of supervision requirements, recognition capability, and deployment challenges. The insights provided by this table are used to guide the comparative analysis presented in Section 5.

4.5. Output format

The aim of the feature recognition task is to identify the type, location, geometry, and topology of each manufacturing feature appearing in a 3D model. A proficient learning-based feature recognition system should be capable of outputting all the aforementioned information based on the given 3D model. From an ML system design perspective, the output format defines how

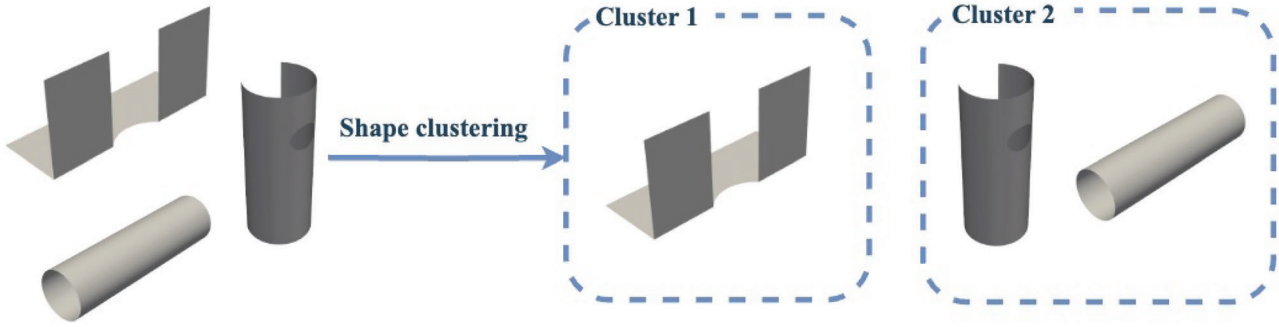

Figure 18. An example of the shape clustering-based approach.

Table 4. Comparison of pattern recognition methods from an ML system design perspective.

Pattern recognition method	Supervision / human efforts	Benefits and limitations
Shape clustering	Designing input attribute array; Choosing proper ML algorithm; Interpreting clusters	+Data annotation is not required −Result is sensitive to the input attributes and distance metrics −Assuming that models in the same cluster belong to the same class −Pattern recognition is controlled by the inductive bias of ML model
Shape classification	Annotating models with labels; Choosing proper ML algorithm	+Annotation process is easier than next two approaches +It typically needs less training samples than next two approaches −It is normally used with two-stage methods. Therefore, the feature extraction component is the bottleneck in final recognition results
Object detection	Bounding boxes annotation; Choosing proper ML algorithm	+Intersecting feature recognition performance is remarkable −The bounding box output is utilised
Segmentation	Annotating models by face class; Choosing proper ML algorithm	+Intersecting feature recognition performance is remarkable −Data annotation burden is high −Missing faces caused by feature intersection cannot be identified −High accuracy in face classification is required for deployment

learned feature semantics are represented and how recognition results interface with downstream manufacturing tasks.

The extracted manufacturing features could be represented in different forms, such as removal volume, face group, bounding box, or whole solid model. The bounding box is a cubic box that encloses a manufacturing feature appearing in a 3D model. The face group and removal volume refer to the faces and removal areas of the recognised feature. Apart from the above geometric and topological information, other meta-information about recognised features, such as feature type, geometric parameters, and tolerance requirements, is also essential. So far, most feature recognition approaches mainly output instance label, semantic label, or cluster label of the recognised features.

Figure 19 illustrates a number of output cases. From Figure 19(a), it is observed that faces of each feature and its semantic label have been extracted. It is a straightforward way to capture the location, geometry, and topology of the feature(s). However, it is difficult to obtain information about the missing faces caused by feature intersection. Additionally, features with the same type cannot be handled in this case. From Figure 19(b), both semantic and instance labels of recognised features are given. Figure 19(c) illustrates a case where cluster and instance labels of face groups are given. This approach assumes that features in the same cluster belong to the same class or family. Figure 19(d) shows a case where a 3D bounding box of each feature and its semantic and instance labels are provided. In this example, the dimension and location of features could be estimated, even for overlapping features. However, it is difficult to accurately capture the geometric and topological information about the recognised features (H. Zhang et al. 2022). Therefore, this type of output cannot be adopted in applications where accurate locations of features are needed, such as process planning. From Figure 19(e), it is observed that the removal volumes of each feature and its semantic and instance labels have been extracted. This output would be beneficial for tasks that require removal areas of features, such as manufacturability analysis. Figure 19(f) shows a case where the whole solid model is labelled with its semantic labels only. In this case, the location, geometry, and topology of the feature(s) are missing. Overall, different output formats correspond to different learning formulations and supervision granularities. Each output format reflects specific design choices. These choices lead to different trade-offs among recognition accuracy, geometric completeness, and downstream usability. Therefore, the output format should be selected together with the learning paradigm and the target manufacturing application.

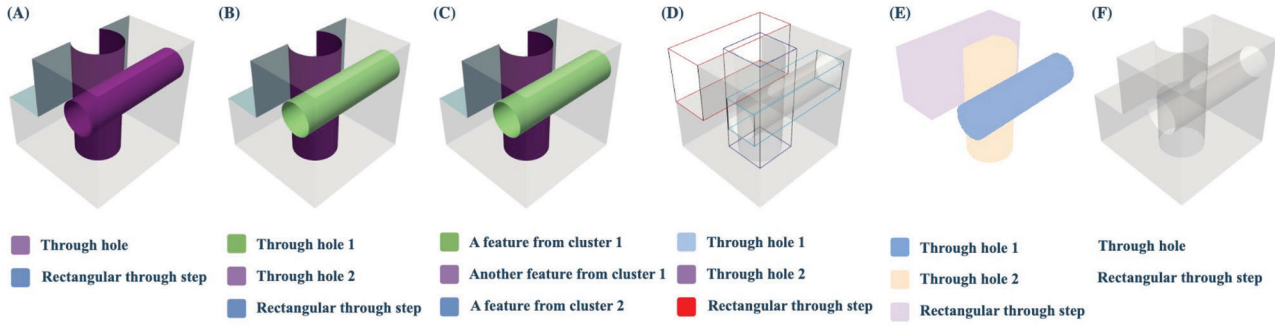

Figure 19. Output format examples: (a) face groups + semantic labels, (b) face groups + semantic & instance labels, (c) face groups + cluster & instance labels, (d) bounding boxes + semantic & instance labels, (e) removal volumes + semantic & instance labels, and (f) whole model + semantic labels.

5. Review, research trends, and comparisons

5.1. Multi-level evaluation framework and criteria

In the previous section, a taxonomy was proposed. It classifies and analyses the components of learning-based feature recognition methods from an ML system design perspective. Building upon this taxonomy, this section aims to establish a structured evaluation framework. It reviews and compares representative methods, analyses research and technical trends, and identifies their application scopes, limitations, and strengths. Due to the diversity of problem formulations, data representations, learning paradigms, output formats, and manufacturing contexts, different studies often report performance using incompatible evaluation settings. Therefore, it is generally difficult to establish a single unified quantitative benchmark that can fairly compare all existing methods.

To address this issue, this review adopts a multi-level evaluation strategy, as shown in Table 5. At the system and application level, qualitative evaluation metrics are used to characterise overall recognition capability and practical applicability. At the technical level, quantitative evaluation metrics are employed, and only within comparable technical branches where consistent problem formulations, datasets, and evaluation protocols are available. As shown in Table 5, metrics are chosen according to the output formulation and evaluation level (e.g. face-level, instance-level, or model-level). When a suitable task-specific metric is available, it is adopted for comparison. Otherwise, the metrics originally reported by the authors are used. More detailed metric definitions, together with the associated reasoning and decision process for metric selection, are provided in Appendix B.

Based on the above evaluation framework, the subsequent sections are organised as follows. Section 5.2 reviews representative learning paradigms and milestone methods under the proposed taxonomy through a series of case studies. The qualitative metrics are then used to characterise recognition scope, robustness, practical burdens, and typical application scenarios. Section 5.3 synthesises cross-case observations to abstract recurring ML design patterns and decision-relevant factors. For each pattern, the review summarises its typical strengths and limitations, and links them to the key design choices that drive the observed trade-offs. Section 5.4 then analyses research and technical evolution from an ML perspective. Within technical branches where comparable datasets, problem formulations, and evaluation protocols exist, quantitative performance metrics reported in the literature are compiled and compared to reveal relative performance trends within each branch. Finally, the combined qualitative and quantitative insights are used to identify research gaps and motivate future directions in Section 6.

5.2. Road map: milestones and case studies

5.2.1. Overview

In this paper, the evolution of learning-based feature recognition techniques is divided into three time periods: the traditional methods, deep learning-based two- and one-stage methods, as shown in Figure 20. These three approaches have entirely different properties in terms of the evaluation criteria identified in Section 5.1. This section provides an overview of each milestone learning paradigm, makes comparisons, and justifies why these approaches are classified in this way. The comparison among the three milestones is illustrated in Table 6.

The history of learning-based feature recognition can be traced back to the 1990s. Before the era of deep learning, most early approaches treated feature recognition as a shape classification (or clustering) problem. In these approaches, rule-based methods are normally adopted to extract features from the solid model. Then, shallow shape classifiers (or clusterers) are utilised in the pattern recognition component. The manually designed attributes are adopted as inputs to ML models. These

Table 5. Multi-level evaluation metrics for learning-based feature recognition systems.

Applicable output / formulation	Metric	What is evaluated
System-level qualitative metrics		
All systems	Recognition scope	Range of feature types supported
All systems	Feature intersection	Ability to recognise interacting features
All systems	Stock shape generality	Supported initial stock geometries
All systems	Robustness	Robustness to feature density and variation
All systems	Development burden	Effort required to construct the recognition system
All systems	Inference burden	Computational cost during deployment
All systems	Maintenance burden	Effort required to update or extend the system
Technical-level quantitative metrics		
Face groups + semantic label (One-stage semantic segmentation)	mIoU	Overlap between predicted and ground-truth feature regions
	Per-face accuracy	Correctness of semantic labels for individual faces
Face groups (or bounding boxes, removal volumes) + semantic & instance labels (Two-stage recognition, instance segmentation, object detection)	Instance-level F1	Correct recognition of feature instances with correct labels and sufficient overlap
Whole model + semantic label(s) (Single/multi-feature classification)	Type-level F1	Correct prediction of feature types or counts in the model
	Type-level F1, accuracy	Correct prediction of feature types or counts in the model

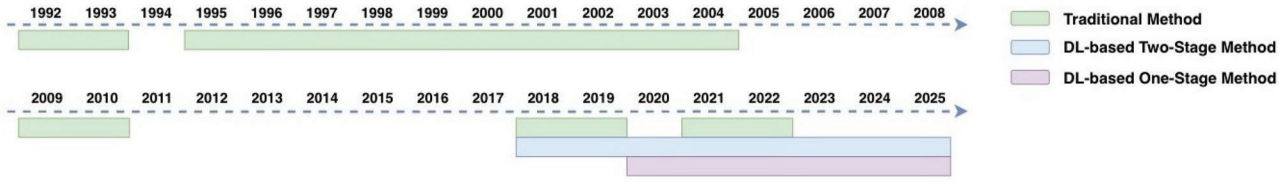

Figure 20. Evolution of three milestone learning paradigms.

Table 6. System-level comparison of different learning paradigms.

	Traditional	Deep learning-based two-stage	Deep learning-based one-stage
Common pattern	Rule-based feature extractor & manually designed input attributes & shallow ML model	Rule- or unsupervised-based extractor & raw data representation & deep ML model for classification	Raw data representation & deep ML model for recognition
Recognised features	Limited feature types	Diverse feature types	
Intersection cases	No feature intersection; Specific types of feature intersection	No feature intersection; Specific types of feature intersection; Features with a small degree of overlap	Features with arbitrary overlap
Data preparation	$\leq$ 100 single-feature models per class Manually design input attributes	$\geq$ 1000 single-feature models per class	$\geq$ 10,000 multi-feature models
Data annotation	Feature class per single-feature model		Feature class per face/bounding box
Time complexity	O(N) in theory, where N refers to the number of extracted features		O(1) in theory
Inference time	$\leq$ 100 milliseconds on CPU	Around a couple of seconds on the CPU or around a few hundred milliseconds on the GPU	
Maintenance	Label additional single-feature models; Redesign extraction rules if needed; Redesign input attributes if needed; Retrain the classifier (or clusterer)	Label additional single-feature models; Redesign extraction rules if needed; Retrain the shape classifier	Prepare multi-feature models again; Label all the data again; Retrain the ML model

approaches are called traditional methods (or shallow learning-based two-stage methods). In traditional methods, designers need to properly design attributes that can differentiate between different types of manufacturing features. Therefore, the scope of features that can be recognised is largely dependent on the input attributes crafted by designers. As traditional methods normally adopt rule-based feature extractors, isolated features or specific types of intersecting features can be recognised correctly. In traditional methods, shallow learning models normally contain a small number of parameters and therefore require less data for training. Designers could build a reliable shape classifier (or shape clusterer) using fewer than one hundred single-feature models per feature class for training. Each solid model needs to be annotated with its class label.

After entering the era of deep learning, researchers started to employ deep ML models to discover meaningful information from raw data representations (i.e. voxel, image, graph, point cloud, natural language) rather than relying on manually designed attributes. In these approaches, rule-based or unsupervised methods are normally adopted to extract features from the solid model. Then, deep shape classifiers are applied to the raw data representations for feature classification. These approaches are called deep learning-based two-stage methods. In this learning paradigm, raw data representations coupled with deep ML models are normally adopted. Thus, the scope of features in these approaches

is more diverse than that in traditional methods. In these approaches, rule- or unsupervised-based extractors have been widely adopted. Therefore, isolated features, specific types of intersecting features, features with well-defined boundaries, or features with a small degree of overlap can be recognised correctly.

As existing two-stage approaches normally rely on rule- or unsupervised learning-based methods to extract manufacturing features, it is challenging for these feature extractors to accurately segment highly intersecting features. This research gap motivates the development of the one-stage feature recognition method. Feature extraction and pattern recognition tasks in one-stage methods are conducted simultaneously in a supervised way. In one-stage approaches, features with arbitrary overlap can be recognised properly, as supervised feature extractors are adopted. Different types of feature intersections need to be considered in the training dataset.

It is observed from Table 6 that the three learning paradigms are significantly different from each other in terms of the qualitative evaluation criteria identified in Section 5.1. For each learning paradigm, researchers have put most of their efforts into designing proper representation, feature extraction, and pattern recognition methods. Within this framework, this analytical review focuses on 21 representative approaches (3 works are illustrated in the main section, while the remaining 18 cases are illustrated in Appendix C). Each approach is regarded as an anchor and can be used to compare and link other methods in the area. The comparison and taxonomy of these three milestone paradigms are illustrated in Tables 7–9 and A1–A3, respectively.

5.2.2. Traditional feature recognition

5.2.2.1. Case study: simple extraction rule + face code vector + shape classification. Sunil and Pande (2009) proposed an approach for multi-feature recognition. In this approach, the solid model with multiple features is represented as an attributed adjacency graph and further decomposed by removing the stock face nodes. For each decomposed sub-graph, each face is assigned a score value that illustrates the characteristics of the face and its relationship to adjacent faces. Then, a 12-dimensional (12D) face score vector is further extracted from each sub-graph, serving as the input of the feature classifier. The first six elements in the vector are face scores of selected faces in the sub-graph. The seventh element is the average face score. The last five elements in the vector contain topologically invariant information about features. Finally, a pre-trained shallow neural network is utilised to predict the class label of each extracted manufacturing feature. The face score vector adopted in this approach involves only translation-invariant information. Therefore, these approaches are robust to rotation and rescaling of features in the solid models. Additionally, features belonging to the same family (e.g.triangular, rectangular, or 6-sided passages) can be represented as similar score vectors. With proper training, therefore, this representation allows for representing diverse feature families with variable faces, which largely increases the feature recognition capability, as evident in Table A4. However, this representation is not sensitive to variations of features within the same family, as their representations are similar. From an application perspective, the output of this approach can also support downstream quality-related reasoning tasks. In this method, each recognised feature is represented as a face group with associated semantic and instance labels. As a result, all functional faces belonging to each feature can be explicitly identified. Based on this structured output, tolerance and surface roughness can be estimated at the feature level. For example, different feature types are often associated with typical manufacturing processes and corresponding surface quality ranges. A through hole produced by drilling may have a different achievable roughness and tolerance range compared to a milled pocket. By mapping the recognised feature type, size, and orientation to process capability data or empirical rules, an estimated tolerance grade or roughness value can be assigned to the corresponding faces. However, since the representation mainly captures topological and invariant shape information, it may be less sensitive to subtle geometric variations within the same feature family. This limitation suggests that additional geometric or process-related information may be required for this task.

5.2.2.2. Applied situations. From an ML perspective, traditional methods rely on fixed feature representations and limited learning capacity. Therefore, these approaches can handle limited feature types and intersection cases (see Tables A4 and 7). The majority of traditional methods are robust to rotation and rescaling, can handle feature-dense solid models, and can be extended to recognise features on different stocks. Their development and inference burdens are not high. These characteristics make traditional approaches suitable for situations where computational and data resources are limited, the feature types and intersections are simple, but large-scale features with variations need to be handled.

5.2.3. Deep learning-based two-stage feature recognition

5.2.3.1. Case study: volumetric grid + unsupervised extractor + volumetric grid classification. Z. Zhang, Jaiswal, and Rai (2018) proposed FeatureNet, which is

Table 7. System-level comparison of traditional methods.

	Recognition capability														Burden				
	RB					SS		ICs							DB		MB		
Reference	Non-orthogonal non-intersecting features	Features with varying ratios	Feature family with variable faces	Non-orthogonal intersecting features	Feature-dense solid model	Rectangular stock	Other stock shapes	No topology change	Face split	Edge loss	Face merge	Face merge and edge loss	Bottom merge and side split	Geometrical intersection	Assign a semantic label to each model	Assign a semantic label to each cluster	Annotate new data	Re-design or adjust attributes	Re-design extraction method
Prabhakar and Henderson (1992)	✓	✓				✓	✓	✓					✓	–	✓		–	✓	
Peters (1992)	$✓_T$	$✓_T$			–		–				–				✓		✓		–
J. Wang and Liu (1993)	✓	$✓_T$			–	✓	$✓_T$				–				✓		✓		–
Gu, Zhang, and Nee (1995)	✓	✓			✓	✓	$✓_E$	✓					✓	–	✓		✓	✓	✓
M.-C. Wu and Jen (1996)	✓	✓			–	✓	$✓_T$				–				✓		✓	✓	–
Lankalapalli, Chatterjee, and Chang (1997)	✓	✓	$✓_T$		–	✓	$✓_T$				–					✓		$✓_D$	–
Nezis and Vosniakos (1997)	✓	✓	✓		✓	✓	$✓_E$	✓		✓	✓			–	✓		✓	$✓_D$	✓
A. M. N. Fu and Yan (1997)	$✓_T$	$✓_T$			–		–				–				✓		✓		–
Chen and Lee (1998)	$✓_T$	$✓_T$			–		–				–				✓		✓		–
Zulkifli and Meeran (1999a)	$✓_T$	✓			✓	✓	$✓_E$	✓					✓	L	✓		✓	✓	✓
Marquez, Gill, and White (1999)	✓	✓	$✓_T$		–	✓	✓	✓					✓	–	✓		✓	$✓_D$	
Chuang, Wang, and Wu (1999)	✓	✓			–	✓	T				–				✓		✓		–
Onwubolu (1999a)	✓	✓	$✓_T$		–	✓	$✓_T$				–					✓		$✓_D$	–
Onwubolu (1999b)	✓	✓	$✓_T$		–	✓	$✓_T$				–				✓		✓	$✓_D$	–
Zulkifli and Meeran (1999b)	✓	✓			✓	✓	$✓_E$	✓						L	✓		✓	✓	✓
W. D. Li, Ong, and Nee (2000)	✓	✓	$✓_T$		✓	✓	✓	✓	✓	✓	✓	✓	✓	–	✓		✓	$✓_D$	✓
Jun, Raja, and Park (2001)	$✓_T$					✓	✓				–				✓		✓	$✓_D$	–
Öztürk and Öztürk (2001)	✓	✓	$✓_T$			✓	✓	✓						–	✓		✓	$✓_D$	
Meeran and Zulkifli (2002)	✓	✓		✓	✓	✓	$✓_E$	✓					✓	L	✓		✓	✓	✓
W. D. Li, Ong, and Nee (2003)	✓	✓	$✓_T$		✓	✓	✓	✓	✓	✓	✓	✓	✓	–		✓		$✓_D$	✓
Öztürk and Öztürk (2004)	✓	✓	$✓_T$			✓	✓	✓						–	✓		✓	$✓_D$	
Sunil and Pande (2009)	✓	✓	✓		✓	✓	$✓_E$							–	✓		✓	$✓_D$	
Guan, Meng, and Yuan (2010)	✓	✓	✓		✓	✓	$✓_E$	✓		✓	✓			–	✓		✓	$✓_D$	✓
Jian et al. (2018)	✓	✓			✓	✓	$✓_E$	✓					✓	–	✓		✓	✓	✓
Takaishi et al. (2019)	✓		✓		–		–				–				✓		✓		–
Muraleedharan and Muthuganapathy (2021)	✓				✓	✓	✓			–				L	✓		✓		
Y. Zhang, Zhang, He, et al. (2022)	✓	✓			✓	✓	$✓_E$	✓						–	✓		✓	✓	

Notes: 'RB', 'SS', 'ICs', 'DB', and 'MB' refer to the 'robustness', 'stack shape', 'intersection cases', 'development burdens', and 'maintenance burdens', respectively. $✓_T$ and $✓_E$ mean that the method can handle certain cases when designers provide proper training and adjust the feature extraction component, respectively. 'L' refers to a low-level geometrical feature intersection. $✓_D$ means that the method's capability depends on the type of new feature introduced into the system. The features that could be recognised by traditional methods are reviewed in Table A4.

Table 8. System-level comparison of deep learning-based two-stage methods.

	Recognition capability														Burden															
	FT				RB			SS		ICs					DB					IB										MB		
Reference	3D regular feature	2.5D negative feature	2.5D positive feature	Assembly or free-form feature	Non-orthogonal features	Features with varying ratios	Feature-dense solid model	Rectangular stock	Other stock shapes	No topology change	Face split or edge loss	Bottom merge and side split	Other topological intersections	Geometrical intersections	Prepare a single-feature set	Prepare a multi-feature set	Decompose multi-feature models	Assign semantic label(s) to each model	Assign a semantic label to each face	Voxelisation	View image capture	Point cloud capture	Graph/text capture	Mesh capture	Attribute capture	Singular extraction	Exhaustive extraction	Forward pass	Post-processing	Re-annotate a new multi-feature set	Annotate a new single-feature set	Re-design/train extraction method
Z. Zhang, Jaiswal, and Rai (2018)	✓$_T$	✓			✓$_T$			✓	✓$_E$	–				L	✓			✓		1						1		N			✓	
Y. Ma, Zhang, and Luo (2019)	✓$_T$	✓	✓$_T$		✓$_T$		–	✓	✓$_T$			–			✓			✓				1						1			✓	
P. Shi et al. (2020b)		✓			✓$_T$			✓				–		M	✓			✓		1	6					6	N	1		✓	
Y. Shi, Zhang, and Harik (2020)		✓	✓		✓		✓	✓	✓	✓	✓	✓		–		✓	✓	✓							1	1		N		✓		✓
Yeo et al. (2021)		✓	✓		✓$_T$	✓$_T$		✓	✓	✓		✓		–		✓	✓	✓							1		1	N		✓		✓
X. Fu et al. (2021)	✓$_T$	✓	✓		✓$_T$	✓	–	✓	✓$_T$			–			✓			✓		1								1			✓	
Worner, Brovkina, and Riedel (2021)				✓	–			✓	✓$_E$			–		L	✓			✓				1				1		N			✓	
Takashima and Kanai (2021)				✓	–		✓	✓	✓			–		M	✓			✓				1					1	N	1		✓	
Lee, Lee, and Mun (2022)	✓$_T$	✓	✓$_T$		✓$_T$		–	✓	✓$_T$			–				✓		✓		1								1		✓		
Ning et al. (2023)		✓	✓		✓	✓	✓	✓	✓	✓				–	✓			✓		N			1			1		N			✓	
X. Yao et al. (2023)	✓$_T$	✓	✓$_T$		✓$_T$		✓	✓	✓$_E$	✓				L	✓			✓				1				1		N			✓	
Miles, Giani, and Vogt (2023b)	✓$_T$	✓	✓$_T$		✓$_T$	✓	–	✓	✓$_T$			–			✓			✓					1					1			✓	
P. Wang, Yang, and You (2023)		✓	✓		✓	✓	✓	✓	✓	✓				–	✓			✓					1			1		6N			✓	
Jia et al. (2023)	✓$_T$	✓	✓		✓			✓	✓$_E$			–			✓			✓						N		1		N			✓	
H. Wu et al. (2023)	✓$_T$	✓	✓$_T$		✓$_T$			✓	✓$_T$			–			✓			✓		1								1			✓	
Miles, Giani, and Vogt (2023a)	✓$_T$	✓	✓$_T$		✓$_T$	✓	✓$_T$	✓	T	✓	✓	✓	✓	–		✓		✓					1					1		✓		
H. Zhang, Wang, Zhang, Zhang, et al. (2024)	✓$_T$	✓	✓$_T$		✓$_T$	✓$_T$	✓	✓	✓	✓	✓$_*$	✓	✓$_*$	–		✓							1				1	kN				✓
Huang et al. (2024)	✓$_T$	✓	✓$_T$		✓$_T$		–	✓	✓$_T$			–			✓			✓		1								1			✓	
Khan et al. (2024a)	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	–		✓					1							1				
Vatandoust et al. (2025)	✓$_T$	✓	✓$_T$		✓$_T$		–	✓	✓$_T$			–			✓			✓				1						1			✓	
J. Yang et al. (2025)	✓$_*$	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	–		✓	✓		✓				1			1		N	1	✓		✓

Notes: 'FT' and 'IB' refer to the 'feature type', and 'inference burdens', respectively. 'M' refers to a medium-level geometrical feature intersection. ✓$_*$ means that the method can handle certain cases.

a pioneering work in this category. In this approach, the solid model with multiple features is voxelised and represented as a 3D voxel. Subsequently, the watershed algorithm is adopted to extract manufacturing features from a 3D multi-feature voxel. Finally, a pre-trained 3D feature classifier based on a 3D CNN is utilised to predict the class label of each extracted incomplete manufacturing feature. A strength of this method is that its raw representation can represent diverse features without imposing any additional burden on human developers. Due to the nature of the volumetric-based representation and the watershed extraction algorithm, this approach could be adopted to handle all types of 2.5D negative features and also has potential to address 3D negative or non-orthogonal features if the shape classifier is further trained with additional data. In addition, a voxelisation step is required for pre-processing the data. Tiny features have the potential to be voxelised out and be missing. Therefore, this approach is not suitable for features with large varying ratios. Furthermore, voxelisation is also memory- and time-consuming, which makes this approach difficult to handle extremely feature-dense models. Moreover, this approach was designed and tested on rectangular or cubic stock. If developers want the method to be suitable for handling parts on other

Table 9. System-level comparison of deep learning-based one-stage methods.

	Recognition capability											Burden														
	FT				RB		SS		ICs			DB						IB							MB	
Reference	3D regular feature	2.5D negative feature	2.5D positive feature	Drawing or camera image	Non-orthogonal features	Features with varying ratios	Rectangular stock	Other stock shapes	Face-shared intersections	Same-type intersections	Other topological intersections	Prepare a single-feature set	Prepare multi-feature set(s)	Assign a semantic label to each model	Assign a semantic label to each face	Assign an instance label to each face	Assign a bounding box to feature	Voxelisation	View image capture	Point cloud capture	Graph/BRep capture	Mesh capture	Forward pass	Non-maximum suppression	Re-annotate a new multi-feature set	Annotate a new single-feature set
P. Shi et al. (2020a)		✓	$✓_T$		$✓_T$		✓		✓	✓	✓	✓		✓				1	6				6	1		✓
Cao et al. (2020)		$✓_*$	$✓_*$		$✓_T$	$✓_T$	✓	$✓_T$			✓		✓		✓						1		1		✓	
Jayaraman et al. (2021)	$✓_T$	✓	✓		$✓_T$		✓	$✓_T$			✓		✓		✓					1	1		1		✓	
Lambourne et al. (2021)	$✓_T$	✓	✓		✓	$✓_T$	✓	$✓_T$			✓		✓		✓						1		1		✓	
P. Shi et al. (2022)		✓	$✓_T$		$✓_T$		✓		✓	✓	✓	✓		✓				1	6				6	1		✓
Colligan et al. (2022)	$✓_T$	✓	$✓_T$		$✓_T$	$✓_L$	✓	$✓_T$			✓		✓		✓						1	1	1		✓	
Mohammadi and Nategh (2022)				✓	✓		✓	$✓_T$	✓	✓	✓		✓				✓		1				1	1	✓	
H. Zhang et al. (2022)	$✓_T$	✓	✓		$✓_T$	$✓_L$	✓	$✓_T$		✓	✓		✓		✓	✓				1			1		✓	
Lei, Wu, and Peng (2022)	✓	✓	✓		$✓_T$		✓	✓			✓		✓		✓					1			1		✓	
Hilbig et al. (2023)	✓	✓	✓		$✓_T$		✓	✓			✓		✓		✓			1					1		✓	
X. Fu et al. (2023)	$✓_T$	✓	$✓_T$		$✓_T$	$✓_L$	✓	$✓_T$			✓		✓		✓						1		1		✓	
Lee et al. (2023)	$✓_T$	✓	$✓_T$		$✓_T$	$✓_T$	✓	$✓_T$			✓		✓		✓						1		1		✓	
Cha and Chul Kim (2023)	$✓_T$	✓	$✓_T$		✓	$✓_T$	✓	$✓_T$			✓		✓		✓						1		1		✓	
Lou et al. (2023)	$✓_T$	✓	✓		$✓_T$	$✓_L$	✓	✓			✓		✓		✓						1		1		✓	
Vidanes et al. (2024)	$✓_T$	✓	$✓_T$		$✓_T$	$✓_L$	✓	$✓_T$			✓		✓		✓					1			1		✓	
H. Wu et al. (2024)	$✓_T$	✓	$✓_T$		$✓_T$	$✓_L$	✓	$✓_T$		✓	✓		✓		✓	✓					1		1		✓	
H. Zhang, Wang, Zhang, Wang, et al. (2024)	$✓_T$	✓	✓		$✓_T$	$✓_L$	✓	$✓_T$		✓	✓		✓		✓	✓				1			1		✓	
S. Zhang et al. (2024)	$✓_T$	✓	✓		$✓_T$	✓	✓	✓			✓		✓		✓						1		1		✓	
L. Ma and Yang (2024)	✓	✓	$✓_T$		$✓_T$	$✓_T$	✓	$✓_T$			✓		✓		✓						1		1		✓	
Khan et al. (2024b)	$✓_T$	✓	$✓_T$		$✓_T$	$✓_L$	✓	$✓_T$			✓		✓		✓						1	1	1		✓	
Kesler and Slama (2024)	✓	✓	✓		$✓_T$	$✓_L$	✓	✓			✓		✓		✓						1	1	1		✓	
D. Yang et al. (2025)		✓			$✓_T$	$✓_L$	✓		✓	✓	✓		✓			✓					1		1		✓	
Xia, Zhao, and Hu (2024)	$✓_T$	✓	✓		$✓_T$	$✓_L$	✓	✓		✓	✓		✓		✓	✓					1		1		✓	
Y. Li et al. (2025)	$✓_T$	✓	$✓_T$		$✓_T$	$✓_L$	✓	$✓_T$		✓	✓		✓			✓					1		1		✓	
Park et al. (2025)	✓	✓	✓		$✓_T$	$✓_T$	✓	✓			✓		✓		✓						1		1		✓	
Dai et al. (2025)	$✓_T$	✓	✓		$✓_T$	$✓_L$	✓	✓			✓		✓		✓						1		1		✓	

Notes: $✓_L$ means that the method can handle certain cases when designers provide proper training, but may cause information loss (L).

stocks, they need to properly adjust its feature extraction method. Due to the use of singular unsupervised learning-based extraction, manufacturing features with a low degree of geometrical overlapping could be handled (P. Shi et al. 2020b). The final output of this approach is each feature's 3D removal volume (i.e. voxel group) and corresponding semantic and instance labels. This means that each recognised feature is represented as a voxel group rather than an explicit set of boundary faces. From a surface quality and tolerance reasoning perspective, the output representation of this approach also introduces certain limitations. As a result, if surface roughness or tolerance needs to be estimated at the feature level, an additional step is required to map the voxel-based feature back to the original geometric surfaces. This mapping is necessary to identify the actual functional faces and to associate them with appropriate surface finish or tolerance specifications. Without this reconstruction step, reasoning directly on the voxel representation may lead to coarse estimations. This issue is more serious for features with small geometric variations or precise functional requirements. Therefore, while the volumetric representation is suitable for tasks such as coarse feature-level reasoning, it is less appropriate for applications that require accurate surface-level information, such as process planning, tool path generation, tolerance or surface

roughness design. This observation suggests that additional geometric reconstruction or face-level reasoning modules may be required.

5.2.3.2. Applied situations. From a learning system design perspective, deep learning-based two-stage methods decouple feature extraction and pattern recognition. This paradigm allows learning signals and inductive biases to be introduced at different stages of the pipeline. Most deep learning-based two-stage approaches can handle diverse feature types but are limited in managing complex intersection cases (see Table 8). The majority of methods are capable of processing feature-dense solid models and can be extended to recognise features on different stock geometries. However, their computational burden at various stages is higher than that of traditional methods. These characteristics make such approaches suitable for scenarios where users need to recognise diverse feature types with simple intersections.

5.2.4. Deep learning-based one-stage feature recognition

5.2.4.1. Case study: volumetric grid + semantic segmentation. Hilbig et al. (2023) proposed a one-stage feature recognition approach based on a volumetric grid and 3D convolutional neural networks. Its main application scenario is adaptive slicing in additive manufacturing. In this approach, the input CAD model is represented as a volumetric representation. The volumetric grid is then processed by a 3D CNN to perform voxel-wise semantic segmentation. The network predicts the geometric primitive type associated with each voxel, e.g. planes, cylinders, spheres, and cones. The recognised primitive regions are used to support adaptive slicing in additive manufacturing. Traditional adaptive slicing methods are usually rule-based. They rely on local geometric metrics, such as surface slope, curvature, or thickness, and apply predefined thresholds to adjust layer thickness or toolpath parameters. When complex non-planar surfaces intersect, local rules may produce inconsistent or suboptimal slicing decisions. In contrast, the learning-based approach attempts to recognise geometric primitives across the entire volume. By recognising primitive types globally, the method provides structured geometric information for downstream slicing decisions. This enables slicing strategies to be conditioned on semantic geometric regions rather than relying solely on local thresholds. This strategy has the potential to lead to more informed and globally consistent slicing decisions for complex structures in additive manufacturing. However, the recognised primitives in this approach are still defined at the surface level, where each voxel is assigned a primitive type. The method focuses on primitive segmentation rather than higher-level manufacturing features or slicing-oriented functional regions. As a result, the recognised entities may not directly correspond to slicing-relevant features, such as thin-walled regions, overhang zones, or process-critical areas. Therefore, although this approach provides a learning-based alternative to rule-based adaptive slicing, the recognised primitives are still geometric entities rather than process-aware slicing features. This indicates a remaining gap between primitive-level segmentation and slicing-oriented feature modelling, and further research is needed to bridge this gap.

5.2.4.2. Applied situations. Deep learning-based one-stage methods represent an end-to-end learning paradigm. The feature localisation and recognition are jointly optimised under a unified learning objective. Most one-stage approaches can handle diverse feature types and complex intersection cases. Since this type of approach adopts the whole solid model as input to the ML model, it does not have strong ability in processing feature-dense solid models. Additionally, their development and maintenance burden at various stages is higher than that of two-stage methods. These characteristics make such approaches suitable for scenarios where users need to recognise diverse features with highly overlapping intersections.

5.3. Linking learning paradigms to evaluation criteria

The case studies discussed in the previous subsection and Appendix C reveal that existing methods have different benefits and limitations due to their ML system design paradigms. These paradigms reflect different design choices, and directly influence the system-level evaluation metrics defined in Section 5.1. Based on these observations, several recurring ML design patterns can be identified. These patterns represent typical combinations of representation, learning paradigms in feature extraction and pattern recognition, and output formulation. Figure 21 illustrates the conceptual relationship between these ML system design choices and the system-level evaluation criteria. Table 10 further summarises the representative design patterns and their typical strengths and limitations under these criteria. It needs to be noted that single-feature recognition has been extensively studied and is relatively mature. Therefore, the following paradigm-level comparison focuses on multi-feature recognition scenarios, as they better reflect practical manufacturing challenges.

As shown in cases (1)–(10), traditional methods and deep learning-based two-stage methods generally follow

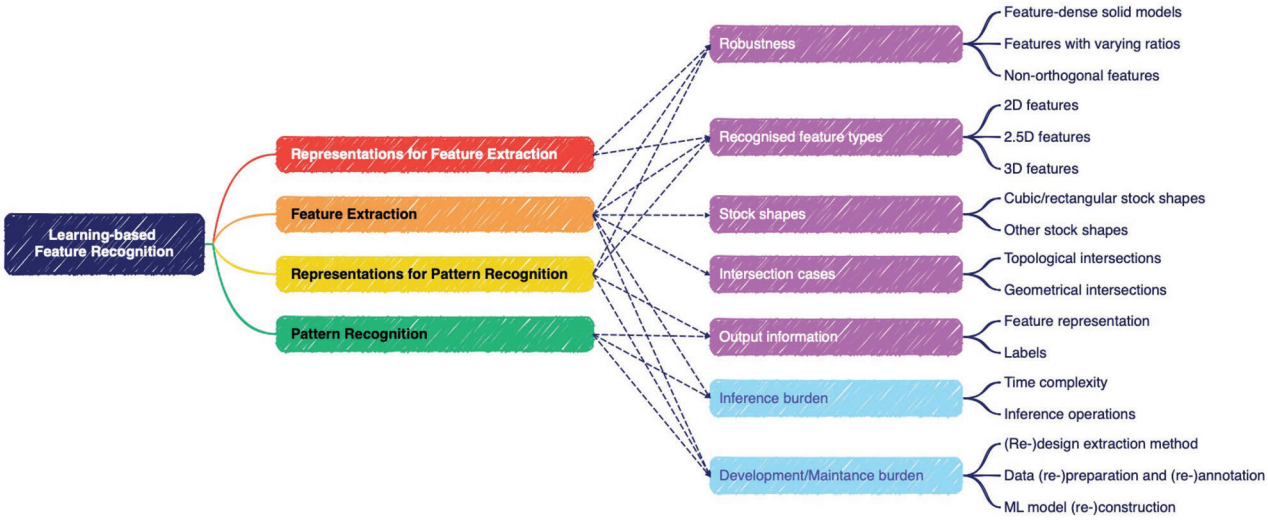

Figure 21. The relationship between ML system design and evaluation criteria.

a similar pipeline. In most cases, these approaches first extract feature candidates and then perform shape classification to recognise the feature types. Although a few exceptions exist (e.g. cases (4) and (8)), most methods in this category follow this extraction–classification formulation. From the feature extraction perspective, many methods rely on graph- or B-Rep-based representations, which explicitly encode the topological structure of the model. When rule-based strategies are used, the systems are usually robust to variations in the number of features (e.g. cases (2)–(4), (7)–(9)). Their performance is therefore less sensitive to feature-dense models. The complexity of the extraction rules also affects the ability to handle feature interactions. Simple rules are easier to design but can only handle limited intersections. More complex rules can address more complicated interactions, but they are harder to implement. Unsupervised extraction methods rely on geometric regularities and can handle certain levels of geometric intersections. Another design choice is whether the extraction is exhaustive or singular. Exhaustive strategies improve coverage of interacting features. However, they may also generate redundant candidates. From the pattern recognition perspective, the shift from shallow learning to deep learning mainly changes the feature scope and data requirements. Shallow learning methods usually rely on low-dimensional, hand-crafted descriptors. This limits the range of feature types that can be recognised. However, many of these descriptors are designed to capture topological properties, so they are often less sensitive to feature scale or size ratios. In contrast, deep learning-based classifiers use higher-dimensional and more expressive representations. This usually enables a broader feature scope, but it also requires more training data. Within the two-stage paradigm, several branches can be observed. One branch replaces classification with shape clustering, as shown in case (4). This reduces the annotation burden but may lead to less stable recognition performance. Another branch replaces the second stage with graph-based segmentation, forming an extraction–segmentation pipeline, as shown in case (8). This can improve recognition for complex feature interactions, but it increases the annotation burden because face-level labels are required. These variants show the trade-offs among supervision level, recognition capability, and annotation cost within the two-stage design space. The quantitative differences among the main paradigms are discussed in the following section.

Segmentation-based methods represent the dominant paradigm in recent one-stage learning-based feature recognition research, as illustrated by Cases 11–15. Instead of classifying pre-extracted feature candidates, these methods treat feature recognition as a dense prediction problem. In these paradigms, each face or local region is assigned a semantic or instance label. Different works adopt different geometric representations, such as graph or B-Rep structures, point clouds, or volumetric grids. Despite these representational differences, segmentation-based paradigms share a strong capability to handle models with highly overlapping or intersecting features. However, when applied to extremely feature-dense solid models, their performance and efficiency may be less favourable than certain two-stage pipelines (J. Yang et al. 2025). The choice of representation directly influences system-level robustness. For example, different representations may exhibit different sensitivities to features with varying size ratios or geometric scales, as discussed in the qualitative analysis in the previous section. In addition, the combination of representation and network architecture leads

Table 10. Relationship between representative ML design paradigms and evaluation criteria.

Case	ML system design paradigm: FR: Volumetric grid	FR: Graph or BRep	FR: View image(s)	FR: Low dimensional array	FR: Point cloud	ES: Exhaustive extraction	ES: Singular extraction	FE: Simple extraction rules	FE: Complicated extraction rules	PR: Unsupervised extraction	PR: Shape classification	PR: Shape clustering	PR: Object detection	PR: Semantic segmentation	PR: Instance segmentation	LP: Traditional methods	LP: Deep learning-based two-stage	LP: Deep learning-based one-stage	Recognition capability: FT: 3D regular feature	FT: 2.5D negative feature	FT: 2.5D positive feature	FT: Free-form feature	RB: Features on varying stocks	RB: Features with varying ratios	RB: Feature-dense solid model	ICs: Topological intersections	ICs: Geometrical intersections
(1)		E		P		✓		✓			✓					✓				✓*	✓*		✓	✓		M	–
(2)		E		P			✓	✓			✓					✓				✓*	✓*		✓	✓	✓	L	–
(3)		E		P			✓		✓		✓					✓				✓*	✓*		✓	✓	✓	H	–
(4)		E		P			✓		✓			✓				✓				✓*	✓*		✓	✓	✓	H	–
(5)	✓						✓			✓	✓						✓		✓T	✓			✓E			–	L
(6)			✓			✓				✓	✓						✓			✓						–	M
(7)					✓		✓	✓		✓	✓						✓		✓T	✓	✓T		✓E		✓	L	L
(8)		✓					✓		✓					✓			✓		✓*	✓	✓		✓	✓	✓	H	–
(9)	P	E					✓	✓			✓						✓			✓	✓		✓E	✓	✓	L	–
(10)					✓	✓				✓	✓						✓					✓	✓E	✓	✓	–	M
(11)		✓					✓							✓				✓	✓T	✓	✓		✓T	✓*		H	H
(12)					✓		✓							✓				✓	✓T	✓	✓		✓T			H	H
(13)	✓						✓							✓				✓	✓	✓	✓		✓T			H	H
(14)		✓					✓								✓			✓	✓T	✓	✓		✓T	✓*		H	H
(15)					✓		✓								✓			✓	✓T	✓	✓		✓T			H	H
(16)			✓			✓							✓					✓		✓	✓T					H	H

Case	Burden: DB: Prepare a single-feature set	DB: Prepare a multi-feature set	DB: Number of training samples	DB: Assign a semantic label to each model	DB: Assign a semantic label to each face	DB: Assign an instance label to each face	DB: Assign a semantic label to each cluster	IB: Inference time	IB: GPU preferred	Output: FL: Instance label	FL: Semantic label	FL: Cluster index	OR: Face group	OR: Removal volume	OR: Bounding box	Quant.
(1)	✓		L	✓				L		✓	✓		✓			–
(2)	✓		L	✓				L		✓	✓		✓			–
(3)	✓		L	✓				L		✓	✓		✓			–
(4)	✓		L				✓	L		✓		✓	✓			Section 5.4.3+
(5)	✓		M	✓				H	✓	✓	✓			✓		Section 5.4.2
(6)	✓		M	✓				H	✓	✓	✓				✓	Appendix D2
(7)	✓		M	✓				H	✓	✓	✓		✓			–
(8)		✓	M		✓			H	✓	✓	✓		✓			Section 5.4.2
(9)	✓		M	✓				H	✓	✓	✓		✓			–
(10)	✓		M	✓				H	✓	✓	✓		✓			–
(11)		✓	H		✓			M	✓		✓		✓			Section 5.4.1
(12)		✓	H		✓			M	✓		✓		✓			Section 5.4.1
(13)		✓	H		✓			M	✓		✓		✓			–
(14)		✓	H		✓	✓		M	✓	✓	✓		✓			Section 5.4.1
(15)		✓	H		✓	✓		M	✓	✓	✓		✓			Section 5.4.1
(16)	✓		M	✓				M	✓	✓	✓				✓	Appendix D2

Notes: 'FR', 'ES', 'FE', 'PR', and 'LP' refer to the 'feature representation', 'extraction strategy', 'feature extraction', 'pattern recognition', and 'learning paradigm', respectively. 'FL' and 'OR' refer to the 'feature label' and 'output representation', respectively. 'L', 'M', and 'H' refer to a low, medium, and high, respectively. 'E' and 'P' mean that the representation is adopted in feature extraction and pattern recognition stages, respectively. 'Quant.' refers to the quantitative evaluation metrics. '+' indicates that the effect of clustering is analysed under a different experimental setting.

to different recognition performances across methods. These are analysed quantitatively within comparable settings in the following section. From a data and supervision perspective, segmentation-based methods typically require multi-feature training datasets with face-level annotations. Each face/point/voxel cell needs to be labelled according to its semantic category or instance membership. This leads to a relatively high annotation burden. In return, these methods provide detailed output structures, making them suitable for applications that require precise per-face labels or explicit instance boundaries. Within this paradigm, two main branches can be identified, i.e. semantic segmentation and instance segmentation. Semantic segmentation assigns a category label to each face or point. This strategy focuses on the functional type of local regions. Instance segmentation

further groups faces or points into individual feature instances. It enables explicit modelling of feature boundaries and interactions. These two branches reflect different output requirements and annotation granularities. In addition to segmentation-based paradigms, object detection-based methods also represent a one-stage formulation, as shown in case (16). These approaches share the benefit of end-to-end learning (e.g. being effective at handling interacting feature cases) and typically require a lower annotation burden. However, due to their coarse output granularity and the commonly adopted view-based representations, they are mainly suited to 2.5D feature types rather than fully complex 3D features. Additionally, this output is generally less suitable for applications that require surface-level labels, such as process planning, and tolerance and roughness design.

Overall, these observations suggest that the performance differences are strongly influenced by the underlying ML system design choices. The qualitative evaluation metrics introduced in Section 5.1 therefore provide a suitable framework to interpret these differences at a system level. Building upon these insights, the next subsection analyses the research trends and technical evolution of learning-based feature recognition methods. Within comparable technical branches, quantitative evaluation metrics reported in the literature are further examined to illustrate performance improvements and emerging research directions.

5.4. ML technical evolution and research trends

In the previous sections, three milestone learning-based feature recognition paradigms and a taxonomy have been presented. While the previous sections mainly focussed on qualitative system-level comparison, this section further examines recent developments from an ML perspective under comparable technical settings. Specifically, three ML-driven directions are summarised: segmentation-based end-to-end learning, scalable and domain-resilient learning, and annotation-efficient learning. Additional trends are provided in Appendix D.

5.4.1. Segmentation-based end-to-end learning

As reviewed in Section 5.2.4 and Appendix C.3, the state-of-the-art approaches prefer to treat feature recognition as a semantic or instance segmentation problem. This reflects a clear shift from extraction-classification pipelines toward segmentation-based end-to-end learning paradigms, although other approaches are still explored in parallel. In this section, the MFCAD++, Fusion 360, MFInstSeg, and MFTRCAD datasets are adopted as benchmarks for comparison. The first two sets are used to compare performance (i.e. face classification accuracy) for semantic segmentation-based approaches, while the last two sets are used to evaluate performance (i.e. F1 score) for instance segmentation-based methods. These sets are selected as they are widely utilised as benchmarks and allow for capturing insights from case studies.

From an ML perspective, the aim of a semantic segmentation network is to assign a feature type to each surface in the solid model. Therefore, the information captured from each surface is essential to guide the network in distinguishing between different feature types. This surface information could be encoded in nodes of a graph-based representation or points in a point cloud-based representation. As shown in cases (1) and (7) in Table 11, more relevant surface attributes can improve recognition performance (Vidanes et al. 2024).[1] It should be noted that these attributes might be ad hoc to specific types of datasets or features (Lee et al. 2023). This could potentially result in good recognition performance on one set, and less favourable results on another, as evident in case (5). Additionally, there is a trend toward learning surface (or edge) embeddings rather than relying on handcrafted attributes. The embeddings transform raw attributes into meaningful vector representations. Surface embeddings tend to encode more implicit information than handcrafted attributes. It therefore avoids the bias introduced by human designers. In the embedded space, for instance, surfaces with similar geometry or functionalities could be located near each other, while dissimilar surfaces could be located farther apart. These characteristics make it easier for the network to classify the surface (Jayaraman et al. 2021; Lou et al. 2023), as shown in cases (2), (6), and (8).

In a solid model, an edge can be used to connect adjacent surfaces. Therefore, the edge can implicitly reveal the topological relationship between surfaces and also encode the geometrical information of a feature. In point cloud-based segmentation methods, such as cases (1) and (7) in Table 11 and case (1) in Table 12, this information is missing. This could be one potential reason why a point cloud-based segmentation approach typically produces less favourable results than its graph-based counterpart. For the remaining approaches, the edge information could be handcrafted or automatically embedded via learning.

In segmentation-based feature recognition, one essential problem is how to transfer information among different surfaces or edges. For instance, a single planar surface could be classified as a triangular passage type not only because of its own geometry, but also because this surface is connected with another two angled side faces. This

Table 11. Case studies on semantic segmentation-based approaches.

			Surface								Edge						Hop reach				Bias			Acc. (%)	
Case	Method	Reference	Coordinate	Normal vector	Surface type	Surface area	2D UV-grid	Mesh coefficients	Other attributes	Surface embedding	Convexity	Length	Type	1D UV-grid	Other attributes	Edge embedding	Local region	1-hop	Multi-hop	All-hop	Implicit (spatial)	Explicit (adjacency)	Injected (encodings)	MFCAD++	Fusion 360
(1)	PointNet++	Qi et al. (2017)	✓	✓													✓				✓			85.9	70.8
(2)	UV-Net	Jayaraman et al. (2021)					*			✓				*		✓		✓				✓		99.5	92.3
(3)	BRepNet	Lambourne et al. (2021)			✓	✓					✓	✓	✓		✓				✓			✓		99.4	94.3
(4)	Hierarchical CADNet	Colligan et al. (2022)	✓		✓	✓		✓			✓							✓				✓		97.4	86.3
(5)	BRepGAT	Lee et al. (2023)		✓	✓	✓			✓		✓	✓	✓		✓			✓				✓		99.1	80.3
(6)	BRep-BERT	Lou et al. (2023)			*	*	*		*	✓				*		✓				✓			✓	–	95.1
(7)	Modified PointNet++	Vidanes et al. (2024)	✓	✓	✓				✓								✓				✓			97.8	84.5
(8)	BrepMFR	S. Zhang et al. (2024)			*	*	*		*	✓	*	*	*	*	*	✓				✓			✓	99.8	–
(9)	Sheet-metalNet	L. Ma and Yang (2024)		✓	✓	✓			✓		✓	✓	✓					✓				✓		98.4	–

Notes: '*' means the information is used to learn surface or edge embeddings.

Table 12. Case studies on instance segmentation-based approaches.

			Surface			Edge			Hop reach			Param. sharing				Sem. seg.		Ins. seg.	
Case	Method	Reference	2D UV-grid	Attributes	Surface embedding	1D UV-grid	Attributes	Edge embedding	0-hop	1-hop	All-hop	Shared encoder	Task-specific encoder	Gating mechanisms	Inter-task transfer	MFInstSeg	MFTRCAD	MFInstSeg	MFTRCAD
(1)	ASIN	H. Zhang et al. (2022)		*	✓				✓			*	✓			86.5	66.2	73.2	72.6
(2)	AAGNet	H. Wu et al. (2024)	*	*	✓	*	*	✓		✓		✓	✓			99.2	75.8	98.8	75.0
(3)	MFTReNet	Xia, Zhao, and Hu (2024)	*	*	✓	*	*	✓		✓		✓	✓	✓	✓	99.6	89.9	98.9	85.4

Notes: '*' under the surface and edge columns means that the information is used to learn surface and edge embeddings. '*' under the parameter sharing column means the network is only used for learning embeddings.

example shows the importance of topological relationships among different surfaces (or edges) in surface type classification. Table 11 illustrates the hop reach per layer in different networks. In the PointNet++ series, i.e. cases (1) and (7), one point on a surface can aggregate information from its Euclidean neighbours (Qi et al. 2017). This approach only relies on local Euclidean radius rather than topological information of a solid model. In the majority of approaches that rely on a graph neural network, e.g. cases (2), (4), (5), and (9), each surface aggregates information from its 1-hop neighbour surface. The neighbourhood relation is determined via edge connections. This approach can effectively utilise the topological information of manufacturing features. In the case of long-term dependency–e.g. when a surface's type is related to another k-hop-away surface's geometry–GNNs with 1-hop message passing per layer may produce less favourable results (Akansha 2025). In case (3), multiple entities (i.e. surface, edge, and coedge) from a BRep file have been considered. Then, a set of pre-defined 1- or 2-hop topological walks has been adopted to aggregate information to each coedge. These walks are specifically designed for BRep data and capture diverse relationships between entities. Experiments show that this design can produce good recognition results even when the attributes in surfaces and edges are relatively simple (Lambourne et al. 2021). In cases (6) and (8), a transformer has been introduced. This network enables all-hop message passing per layer and allows each surface to aggregate information from all other surfaces. The topological structures of the BRep data can be injected into the transformer's attention mechanism. These attention biases can guide a surface to aggregate information from more relevant entities.

From an ML perspective, instance segmentation represents a further evolution of the segmentation paradigm. The insights gained from the previous discussions, e.g. surface and edge information, hop reach per layer, and structural bias, are still applicable (see Table 12 for details). As these methods normally adopt a multi-task

Table 13. Four domain adaptation cases (adapted from S. Zhang et al. 2024).

			Recognition performance (%)	
Case	Source set	Target set	Direct transfer	Domain adaptation
(1)	MFCAD	CADSynth	22.91	85.71
(2)	MFCAD	MFCAD++	54.80	87.08
(3)	MFCAD++	CADSynth	61.82	92.74
(4)	CADSynth	MFCAD++	81.50	90.32

learning paradigm, how to share knowledge among different tasks is essential (Xia, Zhao, and Hu 2024). In the experiments conducted in case (3), three components–i.e. a shared encoder that learns task-invariant knowledge, another encoder that learns task-specific knowledge, and a gating network that dynamically fuses shared and specific knowledge–have been proven effective. Additionally, adding a semantic link among different tasks could also improve recognition performance.

5.4.2. Scalable and domain-resilient learning

A learning-based feature recognition method normally works well within its recognition scope, but might not generalise effectively beyond this scope. Therefore, an emerging research trend is to implement a more scalable method that is robust to geometric and topological changes. This section discusses methods that are robust to changes in stock shapes, feature numbers, and feature size ratios.

An ML direction is to adopt domain adaptation methods to learn domain-invariant information for feature recognition. This type of approach aims to transfer the knowledge learned from the source domain to a related but slightly different target domain. A good example is the BRepMFR proposed by S. Zhang et al. (2024). In this approach, three different datasets–i.e. MFCAD, MFCAD++, and CADSynth–have been adopted to verify the concepts (see Table E1 for details). The domain adaptation approach has been used to learn domain-invariant information between each pair of datasets, as shown in Table 13. It is observed that a significant performance drop can occur when ML models pre-trained on the source domain are applied to recognise features in the target domain. By adopting domain adaptation, the performances have been largely improved. It is also observed that the more similar two domains are (e.g. MFCAD++ → CADSynth), the better the domain adaptation performance achieved.

Another strategy is to adopt translation-invariant data representation to make the recognition method more robust to feature size changes. As discussed in Sections 5.2.3 and 5.2.4, both volumetric and view-based representations are not suitable for representing tiny features. A point cloud-based representation's capabilities depend on its sampling strategies. In contrast, graph- and text-based representations can encode more transformation-invariant information, which potentially produces more robust results. A representative case study is the experiment conducted in Miles, Giani, and Vogt (2023a). The robustness of text- and view-based representations in multi-feature classification has been verified, as summarised in Table 14. In this experiment, three different methods were tested on four test sets with features of different scale ratios on cubic stocks. Experimental results show that text-based representation effectively encodes transformation-invariant information, and the shape classification task primarily relies on this transformation-invariant information for good results.

In addition to the aforementioned approaches, a properly designed two-stage approach could handle solid models with a varying number of features. A representative example is the experiment conducted in J. Yang et al. (2025). In this paper, a new AircraftMF dataset has been constructed and divided into training and testing sets. The number of features appearing in the solid models in the testing set is far larger than those in the training set. Additionally, the solid models in the two sets also have large variations in stock shapes, feature ratios, and interacting structures. Table 15 illustrates a number of methods tested on the AircraftMF set. It is observed from cases (1)–(4) that one-stage graph segmentation-based approaches produce less favourable results. From cases (6)–(8), it is observed that properly designed two-stage approaches can produce good performance. In these cases, practitioners use heuristic rules to remove the stock faces and decompose the solid model into a number of manageable feature primitives. Then, a graph segmentation network can be further utilised to handle small-scale graphs effectively.

Overall, these studies indicate a broader ML trend toward improving generalisation across domains, feature scales, and model complexities. This direction is particularly important for real-world applications, where domain shifts are common.

5.4.3. Annotation-efficient learning

ML aims to enable algorithms to learn from experiences and perform better at tasks with the knowledge gained from these experiences. These experiences are encoded in the data. When building a learning-based intelligent system, an appropriate amount of data is required. In different approaches, data annotation is normally required and can potentially impose additional burdens on practitioners. Therefore, an important ML trend is to develop

Table 14. Methods tested with different feature ratios (adapted from Miles, Giani, and Vogt 2023a).

Case	Method	Reference	Representation	Recognition performance			
				Scale 1	Scale 2	Scale 3	Scale 4
(1)	MsvNet	P. Shi et al. (2020b)	View-based	71.56%	61.01%	60.23%	57.13%
(2)	SsdNet	P. Shi et al. (2020a)	View-based	92.17%	83.67%	76.57%	74.13%
(3)	Tree-LSTM	Miles, Giani, and Vogt (2023a)	Text-based	89.09%	89.76%	88.31%	87.80%

Table 15. Eight case studies conducted on the AircraftMF dataset (adapted from J. Yang et al. 2025).

Case	Method	Reference	Paradigm	Accuracy
(1)	UV-Net	Jayaraman et al. (2021)	One-stage	40.54%
(2)	Hierarchical CADNet	Colligan et al. (2022)	One-stage	43.78%
(3)	BRepGAT	Lee et al. (2023)	One-stage	45.38%
(4)	AAGNet	H. Wu et al. (2024)	One-stage	49.60%
(5)	Unsupervised extraction + FeatureNet	Z. Zhang, Jaiswal, and Rai (2018)	Two-stage	81.23%
(6)	Extraction rules + DeepFeature	P. Wang, Yang, and You (2023)	Two-stage	94.66%
(7)	Extraction rules + AAGNet	H. Wu et al. (2024)	Two-stage	95.75%
(8)	Extraction rules + FPCNet	J. Yang et al. (2025)	Two-stage	98.23%

annotation-efficient learning strategies (see Table A5 for a review).

Shape clustering-based approaches have been adopted for decades. This type of method is an unsupervised learning-based approach, which can avoid the burden of data annotation. An example is the approach proposed by Jayaraman et al. (2021). In this approach, a self-supervised learning method is adopted to learn meaningful shape embeddings, and then a K-Means clustering algorithm is used to categorise 3D shapes. Case (1) in Table 16 shows that this approach completely bypasses data annotation. However, a significant classification accuracy drop occurs in comparison to supervised shape classification. This performance gap can be regarded as a limitation of shape clustering-based approaches. In general, the final clustering results are not stable and are sensitive to the input attributes (i.e. shape embedding in this case).

Data augmentation is an approach that aims to generate additional training data by applying transformations to the existing data. This approach can significantly increase the diversity of the data and hence improve the generalisation ability of an ML model. A typical example is the RDetNet proposed by P. Shi et al. (2022), as shown in case (2) in Table 16. In this approach, voxel and image manipulation operations are adopted to enrich the training set. This strategy can minimise the burden of data collection and annotation and produce competitive results when only 7% of the training samples are used.

Transductive transfer learning is another approach that aims to build an ML model using labelled data from the source domain and unlabelled data from the target domain. The domain adaptation approach reviewed in the previous section is a typical example of transductive transfer learning. In case (3) in Table 16, the source and target datasets are the MFCAD++ and CADSynth sets, respectively. It is observed that this approach completely bypasses annotation in the target domain. However, this approach assumes that a labelled source dataset is available. Additionally, the source and target domains must share the same learning objectives and label spaces. The final recognition performance on the target domain is largely affected by the similarity between the two domains.

Inductive transfer learning is another transfer learning approach. It aims to pre-train an ML model on a source dataset and fine-tune the model on the labelled target feature recognition dataset. In this paradigm, the source dataset could be either labelled or unlabelled, but the target domain needs to be labelled. Lou et al. (2023) proposed an approach named BRep-BERT that belongs to this category. In this method, a self-supervised learning technique is adopted to pre-train the network on the Fusion360 dataset. Then, the pre-trained network is fine-tuned on the SolidLetters dataset to enable few-shot learning. From case (4) in Table 16, this strategy reduces the data preparation and annotation burdens.

Semi-supervised learning is another typical method that aims to build a reliable ML model by using a large amount of unlabelled data and a small amount of labelled data. A typical example is the method proposed by H. Zhang, Wang, Zhang, Wang, et al. (2024). In this method, an autoencoder is pre-trained with the unlabelled ASIN dataset and further fine-tuned with a small amount of labelled data from the ASIN dataset. As shown in case (5) in Table 16, this strategy can minimise the data annotation burden and produce competitive results when only 12.5% of the labels are used.

A recent ML direction is the exploration of pre-trained large language models (LLMs) for feature recognition. In this approach, practitioners only need to define prompt templates for each type of feature. These prompts are

Table 16. Six case studies on annotation burden reduction strategies.

Case	Reference	Strategy	Target set	Data preparation	Data annotation	Recognition performance
(1)	Jayaraman et al. (2021)	Shape embedding + clustering	SolidLetters	96K → 96K	96K → 0	Acc: 97.24% → 58.17%
(2)	P. Shi et al. (2022)	Data augmentation	MsvNet set	12.29K → 768	12.29K → 768	F1: 95.38% → 93.17%
(3)	S. Zhang et al. (2024)	Transductive transfer learning	CADSynth	80K → 80K	80K → 0	Acc: 99.96% → 92.74%
(4)	Lou et al. (2023)	Inductive transfer learning	SolidLetters	96K → 20-shot	96K → 20-shot	Acc: 97.97% → 75.92%
(5)	H. Zhang, Wang, Zhang, Wang, et al. (2024)	Semi-supervised learning	ASIN	40K → 40K	40K → 5K	mIoU: 96.0% → 91.3%
(6)	Khan et al. (2024a)	Vision-language model	View images	-	Zero-shot	Acc: 75%

Notes: The values before the right arrow refer to the data preparation, annotation burdens, and recognition performance without using the strategies, while the values after the arrow refer to the burdens/performances after using the strategies.

used to guide LLMs to recognise features in the given file. As shown in case (6), Khan et al. (2024a) utilises the reasoning capabilities of LLMs to recognise a wide range of features in different domains on view images via prompt-based inference. Experiments show that Claude-3.5-Sonnet achieves 75% feature name recognition accuracy. These approaches reflect an emerging direction toward foundation-model-based and zero-shot feature recognition.

6. Open challenges, outlooks, and future directions

6.1. From synthetic data to real-world deployment

6.1.1. Distribution shift and generalisation limits

As reviewed in the previous sections, the training data utilised in many feature recognition methods are predominantly synthesised. This type of training set might mainly contain randomly placed regular features on predefined topological structures on cubic, prism, cylinder, cone, sphere, or mixed stocks.

In real-world samples, however, the stock might have an arbitrary shape, and the features appearing in the stock might be a mix of regular and free-form features. It is also possible that real-world samples contain new feature types and much smaller features, and these may not have been seen in the training set. The feature distributions in real samples might also differ from those in the training samples. Therefore, the knowledge learned from the training samples may not be successfully transferred to recognise real-world samples. This distribution discrepancy is one of the major sources of errors when deploying feature recognition models in practice, and remains a key bottleneck for real-world adoption.

6.1.2. Digital twins as a bridge between simulation and reality

One possible research direction to minimise the above discrepancy between synthetic training data and real-world deployment is the use of Twins. They incorporate manufacturing-related constraints and process knowledge within the system. Therefore, they are able to provide a link between design models and practical manufacturing conditions (J.-F. Yao et al. 2023).

From a feature recognition perspective, Twins will not be adopted to introduce new design features nor redefine feature semantics. Their main role could be to constrain synthetic training data generation and to filter candidate geometries based on manufacturability considerations. By embedding manufacturing constraints (e.g. process knowledge and feedback from physical systems) into simulation environments, Twins can help restrict training data to more realistic and engineering-feasible design spaces. Therefore, this solution has the potential to reduce the discrepancy between synthetic training datasets and real-world samples. They could serve as an enabling mechanism to incorporate domain knowledge into the data preparation stage. In addition, such knowledge may also be incorporated at the learning stage. For example, constraint-aware loss functions or regularisation terms could be used. However, these directions are still uncertain and require further investigation.

6.1.3. Learning strategies for robust transfer

Based on the insights from Section 5.4.2, a robust method may use multiple strategies. First, minimal heuristic rules can be used to remove stock faces and decompose a solid model into manageable groups. Then, supervised learning-based methods (e.g. graph segmentation) can handle complex intersections (J. Yang et al. 2025). Second, training objectives can be designed to encourage transformation-invariant representations, for example through additional regularisation losses. Third, domain adaptation can be used to learn domain-invariant information across datasets (S. Zhang et al. 2024). For unseen features, open-set domain adaptation is a promising direction (Choe et al. 2024).

6.1.4. ML benchmarking and evaluation protocols

Another issue is that there are no shared benchmarks for comparison. As reviewed in the previous section, existing studies often differ in feature scope, output formulation, and train–test settings. These differences

make direct numerical comparison difficult. As a result, this review adopts a multi-level evaluation strategy as a practical compromise. Future research could focus on constructing shared benchmarks with controlled feature scopes, unified output definitions, and deployment-oriented evaluation protocols. Such benchmarks are essential for fair comparison and for measuring real-world generalisation.

6.2. Downstream applications enabled by learning-based feature recognition

6.2.1. Feature-level parameter prediction

In state-of-the-art approaches, the location and type of features appearing in the synthesised solid models can be predicted accurately. As manufacturing features are commonly associated with candidate machining tools within conventional process planning frameworks, tool selection strategies can be implemented accordingly. Apart from tool selection, however, there are other process planning tasks that require more information about features. Therefore, how to accurately predict the parameters of recognised features is a research question. The answer to this question would allow for building a reliable process planning algorithm and supporting manufacturing automation.

As discussed in Appendix D.3, research has been conducted to estimate the stock size, radius, length, centre, and depth of recognised features (Khan et al. 2024b). For isolated features present in solid models, parameter prediction is relatively straightforward, as the topology of isolated features is complete. However, in solid models with overlapping features, the topology and geometry of features are compromised. Predicting their parameters is still a challenging issue (Khan et al. 2024b). The current bottleneck of this research topic is how to restore information about the missing faces caused by feature intersection. One potential solution is to design heuristic reconstruction rules or automatically learn a reconstruction model to restore the missing faces.

6.2.2. Feature-aware slicing in additive manufacturing

Slicing is an important process planning step in additive manufacturing. It determines how a 3D model is decomposed into layers. Although slicing itself is not a feature recognition task, feature-level information can guide slicing strategies. In uniform slicing, a constant layer thickness is applied. This approach is simple to implement, but could introduce staircase effects on curved or inclined surfaces. Geometry-aware adaptive slicing methods have therefore been proposed (Y. Wu et al. 2026; Xiao, Lee, and Feldhausen 2025). In these approaches, layer thickness or orientation is adjusted based on local metrics such as surface slope, curvature, or thin protrusions. However, real industrial components often contain non-planar adjoining surfaces with complex transitions. In such regions, local geometric rules may produce inconsistent layer thickness or unstable slicing boundaries. Therefore, it is difficult for rule-based methods to maintain global geometric continuity. Most adaptive slicing approaches still rely on predefined geometric heuristics and local thresholds (Hu et al. 2026). These rules are derived mainly from geometric discontinuities rather than manufacturing semantics, such as tolerance-critical regions or overhang-sensitive zones. As a result, regions with different manufacturing intents may be treated similarly.

The case study discussed earlier suggests that learning-based feature recognition approaches (Hilbig et al. 2023) make it possible to infer region-level surface semantics from volumetric data. However, these approaches are still at an early stage. A research gap therefore lies in first defining manufacturing-relevant feature semantics that are directly related to slicing decisions, and then integrating such learned feature information into slicing workflows. Future strategies may move toward feature-aware and learning-guided slicing methods for complex industrial parts. In addition, most existing feature recognition benchmarks are developed for subtractive manufacturing scenarios. Additive manufacturing parts often contain freeform surfaces, which are not well represented in current datasets. Therefore, future work should construct AM-oriented benchmarks and develop feature semantics tailored to slicing, support planning, and build orientation tasks, enabling more generalisable learning-based process planning systems.

6.2.3. Surface quality and tolerance-related applications

In the area of manufacturing, surface roughness and tolerance are critical considerations. They capture functional and quality-related requirements. In practical manufacturing workflows, tolerance and surface finish requirements may be encoded through manufacturing annotations (Jia et al. 2023; Xu, Li, and Chen 2022). However, in many real industrial CAD models, such information is incomplete or not preserved during data exchange. Therefore, downstream tasks that design functional surfaces with appropriate tolerance and roughness are required.

Learning-based feature recognition can support this process by identifying manufacturing features, feature shape, physical size, functional surfaces, and their spatial relationships. The structured information obtained

from feature recognition therefore provides a basis for tolerance and surface roughness design. Since tolerance types are often strongly coupled with specific design feature categories and their functional roles, such feature-level information can support the inference of appropriate tolerance specifications. Recent studies have explored data-driven prediction of surface quality for AM parts (Islahudin et al. 2025). This suggests that quality reasoning is possible, and feature-level semantics may provide a more structured basis. When combined with design intent, assembly conditions, process capabilities and domain knowledge, this structured information can facilitate semi-automatic tolerance design and surface finish planning. In this context, tolerance and surface finish requirements derived from design features and functional surfaces may further guide the selection or generation of corresponding machining features and process plans. For complex features with non-planar or freeform faces, tolerance and surface roughness considerations become even more challenging. Unlike planar surfaces, non-planar faces involve continuously varying curvature and local geometric conditions. It makes the functional relevance of surface deviations highly location-dependent. Consequently, a single global tolerance or roughness specification is often insufficient. In these cases, additional reasoning over functional surfaces, local geometry variation, and manufacturing effects is required.

6.3. Emerging learning paradigms for general-purpose feature recognition

6.3.1. Multimodal representation learning

As reviewed in the previous sections, different data representations have unique benefits and limitations. To utilise complementary strengths, researchers have explored the fusion between different representations, such as graph and attribute vector (H. Wu et al. 2024), graph and sampled points (Jayaraman et al. 2021), graph and mesh (Colligan et al. 2022), and point cloud and attribute vector (Vidanes et al. 2024). The complementary geometric, topological, and semantic information is jointly encoded into a shared feature space. Such learning paradigms aim to reduce the limitations of single-representation systems and improve robustness across varying feature configurations.

One emerging trend in computer vision is multimodal learning, which aims to build ML models by using different representations. A landmark research effort is called contrastive language-image pre-training (CLIP) (Radford et al. 2021). This approach aims to build a visual model under the supervision of natural language and learn a joint embedding space between images and language. This pre-trained visual model can be transferred to other downstream tasks without additional task-specific training. Analogously, feature recognition can also be formulated as a multimodal learning problem, where geometric representations (e.g. B-Rep, mesh, or point cloud) are combined with semantic or process-related information. To enable such approaches, solid models and their corresponding text descriptions could be collected to form multimodal training datasets. Then, a multi-modal network that relies on graph and text embeddings could be trained. Such models may support multiple downstream tasks, such as multi-label classification, region-level retrieval, or segmentation, with reduced task-specific training.

6.3.2. Foundation models and zero-shot learning

In most feature recognition tasks, developers need to collect and annotate the task-specific dataset, which imposes greater burdens on human developers. As reviewed in Section 5.4.3, a number of learning tricks have been utilised to minimise the amount of labelled data.

From a learning paradigm perspective, this challenge motivates the exploration of foundation models and pre-trained representations that can be transferred across tasks and manufacturing scenarios. Recent research has adopted pre-trained foundation models (e.g. ChatGPT) and prompt engineering for feature recognition. Khan et al. (2024a) utilises the reasoning capabilities of large language models to recognise a wide range of features in different domains on view images via prompt-based inference. Such approaches represent a shift from task-specific supervised learning towards prompt-driven or zero-shot inference, where feature semantics are described in natural language rather than encoded solely through labelled geometric datasets. The current bottleneck of this research topic is how to improve the recognition performance and generate precise locations of recognised features. In particular, existing foundation-model-based approach (Khan et al. 2024a) operates at the classification or reasoning level, while geometric localisation and precise boundary extraction remain open challenges.

6.4. Interpretable and knowledge-aware feature recognition

6.4.1. Incorporation of domain knowledge in learning

The feature recognition task is normally treated as a computer vision task, e.g. shape classification, object detection, semantic or instance segmentation. Unlike instances/objects (e.g. cats, dogs, and cars) in computer vision tasks, a manufacturing feature normally has well-defined topological and geometrical structures. For

example, a blind hole has cylindrical and planar surfaces; a triangular pocket includes three interior walls and one bottom face. These definitions contain rich information. From an ML learning perspective, this structured geometric and topological knowledge represents an important source of inductive bias that is often under-utilised in purely data-driven models. Traditional rule-based systems encode such knowledge explicitly, whereas many learning-based approaches rely primarily on statistical patterns in training data.

Research has explored injecting domain knowledge into the learning process. One approach is called a physics-informed neural network (Cuomo et al. 2022). This type of approach aims to encode physical laws into the neural network and loss function, and has attracted great attention in different domains. Another approach is neural-symbolic learning (Yu et al. 2023), which also encodes logical rules or symbolic definitions into the training process. These approaches illustrate a broader learning paradigm in which prior knowledge is incorporated as constraints, regularisation terms, or auxiliary objectives. By using such structural or logical priors, the learning process can potentially reduce data requirements and improve generalisation. In a feature recognition task, researchers could encode the properties of features into the loss function and an auxiliary network. Such knowledge-aware learning strategies may enable models to capture domain-invariant geometric relationships.

6.4.2. Interpretability of learning-based feature recognition

ML is not an error-free methodology. The final learning performance can be affected by various factors, such as the learning paradigm, the gap between the data distribution in training and testing data, the inductive bias of the selected ML model, and the quality of the training data. When utilising ML in feature recognition, unexpected errors will normally appear, no matter how good the ML model is. Since most ML models are black boxes, it is difficult to detect the source of these errors. This lack of transparency has been identified as a key limitation of many learning-based feature recognition systems. Since a learning-based feature recognition approach is typically utilised in process planning and automatic manufacturing, unexpected recognition errors have the potential to cause catastrophic failure in the entire system, and debugging these errors is challenging.

A response to the above challenge is to utilise explainable ML (XML) techniques to enhance the transparency and interpretability of models (Marcinkevičs and Vogt 2023). XML focuses on opening the black box of ML models and making the prediction process of ML models more understandable to humans. This approach has been utilised in the area of feature recognition (Lee, Lee, and Mun 2022) to provide a visual explanation for the 3D feature classification task. Apart from this, other types of explanations may also be useful in manufacturing. One possible direction is post-hoc rule mining. Rule-based feature recognition has a long history. Therefore, extracting human-interpretable rules from trained models may help bridge data-driven learning and deterministic reasoning. Another possible direction is to generate verbal explanations of the ML model. Engineers and designers could ask follow-up questions about certain predictions. They could interact directly with AI tools. This approach makes the system more accessible to non-AI technical users (Ruiz, Torres, and del Pozo 2023).

Note

1. The performance is also affected by its stratified sampling strategy, and facewise and pointwise loss functions.

Acknowledgments

The authors would like to thank the editors and anonymous reviewers for their insightful and constructive feedback on our submission. The overall quality of the manuscript has significantly improved as a result of their comments and suggestions. During the writing of this paper, the authors used a large language model (i.e. ChatGPT) to detect grammatical issues and assist with proofreading. The authors have reviewed the content and take full responsibility for the final version of the publication.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Peizhi Shi is currently a Lecturer in Applied Artificial Intelligence with the University of Leeds, UK. He received his master's degree in Software Engineering from the University of Science and Technology of China in 2013, and his PhD degree in Computer Science from the University of Manchester, UK, in 2019. His research interests include machine learning, computer vision, artificial intelligence, and their real-world applications, particularly in sustainable manufacturing, design for additive manufacturing, and production research. He has been conducting research in the area of machine learning for around 14 years and has authored or co-authored 20 papers in several leading journals in his research areas, such as IEEE Transactions on Industrial Informatics, Robotics and Computer-Integrated Manufacturing, Virtual and Physical Prototyping, Journal of Intelligent Manufacturing, and Computers & Industrial Engineering.

Yuchu Qin is currently a Senior Research Fellow with the EPSRC Future Advanced Metrology Hub for Sustainable Manufacturing, University of Huddersfield, UK. He received the DEng degree in measurement technology and instrument from the Huazhong University of Science and Technology, China in 2017 and the PhD degree in advanced manufacturing and precision engineering from the University of Huddersfield, UK in 2021. His research interests include design for additive manufacturing, artificial intelligence in manufacturing, machine learning, computational intelligence, ontology, and applied category theory. He is leading a Royal Society Grant which aims to develop an artificial intelligence-assisted approach for laser powder bed fusion product specification and verification. He has authored or co-authored 60+ journal papers. He is a fellow of the Higher Education Academy, a member of the EPSRC Peer Review College, an expert reviewer of EPSRC funding applications, an expert reviewer of AHRC funding applications, a review expert of funding applications from an international funding body, an editorial team member of 6 journals, and a reviewer of 80+ journals.

Fanlin Meng is currently a Senior Lecturer in Operations and Analytics at the University of Exeter Business School, University of Exeter. He previously held academic positions in Data Science at the University of Manchester and University of Essex. He received the B.Sc. degree in automation from the China University of Mining and Technology, Xuzhou, China, in 2008, the M.Sc. degree in systems engineering from Xiamen University, Xiamen, China, in 2011, and the Ph.D. degree in computer science from the University of Manchester, Manchester, U.K., in 2015. His primary research interests include energy market, carbon market, smart energy and mobility, operations research, machine learning, and game theory. He is a Fellow of British Computer Society and a Senior Member of IEEE.

Paul J. Scott is currently a Professor with the EPSRC Future Advanced Metrology Hub, University of Huddersfield, UK. He received the DSc degree in computational geometry from the University of Huddersfield, UK in 2021 and the honours degree in mathematics, the MSc degree in statistics, and the PhD degree in statistics from the Imperial College London, UK in 1979, 1980, and 1983, respectively. He was the Project Leader for 20 published ISO standards and is currently working on four new ISO documents. His research interests include manufacturing informatics, geometrical product specifications and verification, philosophy of the measurement of product geometry, and foundations of specifying and characterising solutions for real world industrial problems. He is a fellow of Royal Statistical Society, an EPSRC Fellow of Manufacturing, a leading member of ISO TC 213, a founder member of the strategic group AG1 and the technical review group AG2 of ISO TC 213, a convenor of the working group WG15 (Filtration and Extraction) and the advisory group AG12 (Mathematics for Geometrical Product Specifications) of ISO TC 213, a core member of BSI TDW4, a convenor of BSI TDW4/-/9, a Visiting Industrial Professor of Taylor Hobson Ltd, and the Taylor Hobson Chair for Computational Geometry.

Xiangqian Jiang is currently the Chair Professor and the Director of the EPSRC Future Advanced Metrology Hub, University of Huddersfield, UK and the Royal Academy of Engineering and Renishaw Chair in Precision Metrology. She received the honorary DSc degree in precision engineering from the City, University of London, UK in 2019, the DSc degree in precision metrology from the University of Huddersfield, UK in 2007, and the PhD degree in surface metrology from the Huazhong University of Science and Technology, China in 1995. She was awarded a Royal Society Wolfson Merit Award and an Outstanding Woman of Achievement Award in 2006, becoming the first Royal Society Wolfson Research Fellow from a 92' new University. She was made a Dame Commander (DBE) of the Order of the British Empire for services to Engineering and Manufacturing in 2017. Her research interests include surface measurement, precision engineering, and advanced manufacturing technologies. She is a fellow of the Royal Academy of Engineering, a fellow of the Royal Society of Arts, a fellow of the Institute of Engineering Technologies, a fellow of the International Academy of Production Research, a fellow of the International Society for Nanomanufacturing, a principle member of ISO TC 213 and BSI TW/4, an Impact Assessor in the Research Assessment Framework, a member of the Engineering and Physical Science Research Council, an advisory member for UK national measurement system, and the UK Chairman of the International Academy of Production Research.

Data availability statement

The authors confirm that the data supporting the findings of this study are available within the article.

References

Akansha, Singh. 2025. "Over-squashing in Graph Neural Networks: A Comprehensive Survey." *Neurocomputing* 642: 130389. https://doi.org/10.1016/j.neucom.2025.130389.

Babic, Bojan, Nenad Nesic, and Zoran Miljkovic. 2008. "A Review of Automated Feature Recognition with Rule-Based Pattern Recognition." *Computers in Industry* 59 (4): 321–337. https://doi.org/10.1016/j.compind.2007.09.001.

Babic, Bojan, Nenad Nesic, and Zoran Miljkovic. 2011. "Automatic Feature Recognition Using Artificial Neural Networks to Integrate Design and Manufacturing: Review of Automatic Feature Recognition Systems." *AI EDAM* 25 (3): 289–304.

Cao, Weijuan, Trevor Robinson, Yang Hua, Flavien Boussuge, Andrew R Colligan, and Wanbin Pan. 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*, Vol. 84003. V11AT11A003. American Society of Mechanical Engineers.

Cha, Min Hyeok, and Byung Chul Kim. 2023. "Machining Feature Recognition Using BRepNet." *Journal of Mechanical Science and Technology* 37 (12): 6103–6113.

Chen, Y. H., and H. M. Lee. 1998. "A Neural Network System for Two-Dimensional Feature Recognition." *International Journal of Computer Integrated Manufacturing* 11 (2): 111–117. https://doi.org/10.1080/095119298130859.

Choe, Seun-An, Ah-Hyung Shin, Keon-Hee Park, Jinwoo Choi, and Gyeong-Moon Park. 2024. "Open-Set Domain Adaptation for Semantic Segmentation." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 23943–23953. Seattle.

Chuang, J.-H., P.-H. Wang, and M.-C. Wu. 1999. "Automatic Classification of Block-Shaped Parts Based on Their 2D Projections." *Computers & Industrial Engineering* 36 (3): 697–718. https://doi.org/10.1016/S0360-8352(99)00160-6.

Colligan, Andrew R., Trevor T. Robinson, Declan C. Nolan, Yang Hua, and Weijuan Cao. 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.

Cuomo, Salvatore, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. 2022. "Scientific Machine Learning through Physics–informed Neural Networks: Where We Are and What's Next." *Journal of Scientific Computing* 92 (3): 88. https://doi.org/10.1007/s10915-022-01939-z.

Dai, Yongkang, Xiaoshui Huang, Yunpeng Bai, Hao Guo, Hongping Gan, Ling Yang, and Yilei Shi. 2025. "BRepFormer: Transformer-Based B-rep Geometric Feature Recognition." arXiv preprint arXiv:2504.07378.

Fu, Alan M. N., and Hong Yan. 1997. "Object Representation Based on Contour Features and Recognition by a Hopfield-Amari Network." *Neurocomputing* 16 (2): 127–138. https://doi.org/10.1016/S0925-2312(97)00026-X.

Fu, Xingyu, Dheeraj Peddireddy, Vaneet Aggarwal, and Martin Byung-Guk Jun. 2021. "Improved Dexel Representation: A 3-D CNN Geometry Descriptor for Manufacturing CAD." *IEEE Transactions on Industrial Informatics* 18 (9): 5882–5892. https://doi.org/10.1109/TII.2021.3136167.

Fu, Xingyu, Dheeraj Peddireddy, Fengfeng Zhou, Yuting Xi, Vaneet Aggarwal, Xingyu Li, and Martin Byung-Guk Jun. 2023. "Boundary Representation Compatible Feature Recognition for Manufacturing CAD Models." *Manufacturing Letters* 35:895–903. https://doi.org/10.1016/j.mfglet.2023.07.025.

Garcia, Fernando, Minna Lanz, Eeva Järvenpää, and Reijo Tuokko. 2011. "Process Planning Based on Feature Recognition Method." In *2011 IEEE International Symposium on Assembly and Manufacturing (ISAM)*, 1–5. IEEE.

Gu, Z, Y. F. Zhang, and Andrew Y. C. Nee. 1995. "Generic Form Feature Recognition and Operation Selection Using Connectionist Modelling." *Journal of Intelligent Manufacturing* 6:263–273. https://doi.org/10.1007/BF00128649.

Guan, Xuesong, Guangwei Meng, and Xiuhua Yuan. 2010. "Machining Feature Recognition of Part from STEP File Based on ANN." In *2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering*, Vol. 1, 54–57. IEEE.

Han, JungHyun, Mike Pratt, and William C. Regli. 2000. "Manufacturing Feature Recognition from Solid Models: A Status Report." *IEEE Transactions on Robotics and Automation* 16 (6): 782–796. https://doi.org/10.1109/70.897789.

Hilbig, Arthur, Lucas Vogt, Stefan Holtzhausen, and Kristin Paetzold. 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 (3): 992–1009. https://doi.org/10.1093/jcde/qwad027.

Hu, Zeqi, Yongshuo She, Lin Hua, Kang Dong, Mingzhang Chen, Yitong Wang, and Xunpeng Qin. 2026. "Non-planar Additive Manufacturing: A Comprehensive Review of Path Planning, System Integration, Process Control, and Applications." *Additive Manufacturing* 115:105059. https://doi.org/10.1016/j.addma.2025.105059.

Huang, Xunzhuo, Lin Wang, Ming Chen, Dingxiao Huang, Bing He, and Cheng Jiang. 2024. "Automated Recognition of Machining Features in Mechanical Components Utilizing 3D Convolutional Neural Networks." In *2024 8th International Conference on Electrical, Mechanical and Computer Engineering*, 1101–1105. IEEE.

Islahudin, Nur, Dony Satriyo Nugroho, Dewa Kusuma Wijaya, Herwin Suprijono, Turnad Lenggo Ginta, Muizuddin Azka, Helmy Rahadian. 2025. "Machine Learning-Driven Optimization for Surface Roughness Prediction of Vertical Orientation Measurements on 3D Printed Components." *Cleaner Engineering and Technology* 28: 101046. https://doi.org/10.1016/j.clet.2025.101046.

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

Jia, Jia-Le, Sheng-Wen Zhang, You-Ren Cao, Xiao-Long Qi, We Zhu. 2023. "Machining Feature Recognition Method Based on Improved Mesh Neural Network." *Iranian Journal of Science and Technology, Transactions of Mechanical Engineering* 47 (4): 2045–2058.

Jian, Chengfeng, Miao Li, Keyi Qiu, and Meiyu Zhang. 2018. "An Improved NBA-based STEP Design Intention Feature Recognition." *Future Generation Computer Systems* 88:357–362. https://doi.org/10.1016/j.future.2018.05.033.

Jun, Y., V. Raja, and S. Park. 2001. "Geometric Feature Recognition for Reverse Engineering Using Neural Networks." *The International Journal of Advanced Manufacturing Technology* 17 (6): 462–470. https://doi.org/10.1007/s001700170164.

Kesler, Selin, and Donia Slama. 2024. "GraSTNet: Graph Neural Networks and Set Transformer for Machining Feature Recognition." In *2024 International Conference on Control, Automation and Diagnosis (ICCAD)*, 1–6. IEEE.

Khan, Muhammad Tayyab, Lequn Chen, Ye Han Ng, Wenhe Feng, Nicholas Yew Jin Tan, and Seung Ki Moon. 2024a. "Leveraging Vision-Language Models for Manufacturing Feature Recognition in CAD Designs." *arXiv preprint arXiv:2411.02810*.

Khan, Muhammad Tayyab, Wenhe Feng, Lequn Chen, Ye Han Ng, Nicholas Yew Jin Tan, and Seung Ki Moon. 2024b. "Automatic Feature Recognition and Dimensional Attributes Extraction from Cad Models for Hybrid Additive-Subtractive Manufacturing." In *International Design Engineering Technical Conferences and Computers and Information*

in Engineering Conference, Vol. 88346, V02AT02A035. American Society of Mechanical Engineers.

Lambourne, Joseph G., Karl D. D. Willis, Pradeep Kumar Jayaraman, Aditya Sanghi, Peter Meltzer, and Hooman Shayani. 2021. "Brepnet: A Topological Message Passing System for Solid Models." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12773–12782.

Lankalapalli, Kishore, Subrata Chatterjee, and T. C. Chang. 1997. "Feature Recognition Using ART2: A Self-organizing Neural Network." *Journal of Intelligent Manufacturing* 8 (3): 203–214. https://doi.org/10.1023/A:1018521207901.

Lee, Jinwon, Hyunoh Lee, and Duhwan Mun. 2022. "3D Convolutional Neural Network for Machining Feature Recognition with Gradient-Based Visual Explanations from 3D CAD Models." *Scientific Reports* 12 (1): 1–14.

Lee, Jinwon, Changmo Yeo, Sang-Uk Cheon, Jun Hwan Park, and Duhwan Mun. 2023. "BRepGAT: Graph Neural Network to Segment Machining Feature Faces in a B-rep Model." *Journal of Computational Design and Engineering* 10 (6): 2384–2400. https://doi.org/10.1093/jcde/qwad106.

Lei, Ruoshan, Hongjin Wu, and Yibing Peng. 2022. "MFPointNet: A Point Cloud-Based Neural Network Using Selective Downsampling Layer for Machining Feature Fecognition." *Machines* 10 (12): 1165. https://doi.org/10.3390/machines10121165.

Li, W. D., S. K. Ong, and A. Y. C. Nee. 2000. "Recognition of Overlapping Machining Features Based on Hybrid Artificial Intelligent Techniques." *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 214 (8): 739–744. https://doi.org/10.1243/0954405001518107.

Li, W. D., S. K. Ong, and A. Y. C. Nee. 2003. "A Hybrid Method for Recognizing Interacting Machining Features." *International Journal of Production Research* 41 (9): 1887–1908. https://doi.org/10.1080/0020754031000123868.

Li, Yang, Eugene Li, Michael Lenover, Stephen Mann, and Sanjeev Bedi. 2025. "Edge Adjacency Graph and Neural Network Architecture for Machining Feature Recognition." *The International Journal of Advanced Manufacturing Technology* 136 (2): 897–908. https://doi.org/10.1007/s00170-024-14903-y.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. "Ssd: Single Shot Multibox Detector." In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, the Netherlands, October 11–14, 2016, Proceedings, Part I 14*, 21–37. Springer.

Lou, Yunzhong, Xueyang Li, Haotian Chen, and Xiangdong Zhou. 2023. "Brep-Bert: Pre-training Boundary Representation BERT with sub-graph Node Contrastive Learning." In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 1657–1666. Birmingham.

Ma, Liuhuan, and Jiong Yang. 2024. "Adaptive Recognition of Machining Features in Sheet Metal Parts Based on a Graph Class-Incremental Learning Strategy." *Scientific Reports* 14 (1): 10656. https://doi.org/10.1038/s41598-024-61443-2.

Ma, Yaolong, Yingzhong Zhang, and Xiaofang Luo. 2019. "Automatic Recognition of Machining Features Based on Point Cloud Data Using Convolution Neural Networks." In *Proceedings of the 2019 International Conference on Artificial Intelligence and Computer Science*, 229–235. Wuhan.

Marcinkevičs, Ričards, and Julia E. Vogt. 2023. "Interpretable and Explainable Machine Learning: A Methods-Centric Overview with Concrete Examples." *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 13 (3): e1493.

Marquez, M., R. Gill, and A. White. 1999. "Application of Neural Networks in Feature Recognition of Mould Reinforced Plastic Parts." *Concurrent Engineering* 7 (2): 115–122. https://doi.org/10.1177/1063293X9900700204.

Meeran, Sheik, and A. H. Zulkifli. 2002. "Recognition of Simple and Complex Interacting Non-orthogonal Features." *Pattern Recognition* 35 (11): 2341–2353. https://doi.org/10.1016/S0031-3203(01)00223-0.

Miles, Victoria, Stefano Giani, and Oliver Vogt. 2023a. "Approaching STEP File Analysis as a Language Processing Task: A Robust and Scale-Invariant Solution for Machining Feature Recognition." *Journal of Computational and Applied Mathematics* 427:115166. https://doi.org/10.1016/j.cam.2023.115166.

Miles, Victoria, Stefano Giani, and Oliver Vogt. 2023b. "Recursive Encoder Network for the Automatic Analysis of STEP Files." *Journal of Intelligent Manufacturing* 34 (1): 181–196. https://doi.org/10.1007/s10845-022-01998-x.

Mohammadi, Naser, and Mohammad Javad Nategh. 2022. "Development of a Deep Learning Machining Feature Recognition Network for Recognition of Four Pilot Machining Features." *The International Journal of Advanced Manufacturing Technology* 121 (11-12): 7451–7462. https://doi.org/10.1007/s00170-022-09839-0.

Muraleedharan, Lakshmi Priya, and Ramanathan Muthuganapathy. 2021. "A Learning-Based Approach to Feature Recognition of Engineering Shapes." arXiv preprint arXiv:2112.07962.

Nezis, Konstantinos, and George Vosniakos. 1997. "Recognizing 21/2D Shape Features Using a Neural Network and Heuristics." *Computer-Aided Design*29 (7): 523–539. https://doi.org/10.1016/S0010-4485(97)00003-1.

Ning, Fangwei, Yan Shi, Maolin Cai, and Weiqing Xu. 2023. "Part Machining Feature Recognition Based on a Deep Learning Method." *Journal of Intelligent Manufacturing* 34 (2): 809–821. https://doi.org/10.1007/s10845-021-01827-7.

Onwubolu, Godfrey C. 1999a. "Design of Parts for Cellular Manufacturing Using Neural Network-Based Approach." *Journal of Intelligent Manufacturing* 10:251–265. https://doi.org/10.1023/A:1008947824050.

Onwubolu, Godfrey C. 1999b. "Manufacturing Features Recognition Using Backpropagation Neural Networks." *Journal of Intelligent Manufacturing* 10 (3): 289–299. https://doi.org/10.1023/A:1008904109029.

Öztürk, Nursel, and Ferruh Öztürk. 2001. "Neural Network Based Non-standard Feature Recognition to Integrate CAD and CAM." *Computers in Industry* 45 (2): 123–135. https://doi.org/10.1016/S0166-3615(01)00090-2.

Öztürk, Nursel, and Ferruh Öztürk. 2004. "Hybrid Neural Network and Genetic Algorithm Based Machining Feature Recognition." *Journal of Intelligent Manufacturing* 15 (3): 287–298. https://doi.org/10.1023/B:JIMS.0000026567.63397.d5.

Papavasileiou, A., G. Michalos, and S. Makris. 2025. "Quality Control in Manufacturing–review and Challenges on Robotic Applications." *International Journal of Computer Integrated Manufacturing* 38 (1): 79–115. https://doi.org/10.1080/0951192X.2024.2314789.

Papavasileiou, Apostolis, Panagiotis Aivaliotis, Sotiris Aivaliotis, and Sotiris Makris. 2022. "An Optical System for Identifying and Classifying Defects of Metal Parts." *International Journal of Computer Integrated Manufacturing* 35 (3): 326–340. https://doi.org/10.1080/0951192X.2021.1992660.

Park, Jun Hwan, Seungeun Lim, Changmo Yeo, Youn-Kyoung Joung, and Duhwan Mun. 2025. "DFGAT for Recognizing Design Features from a B-rep Model for Mechanical Parts." *Robotics and Computer-Integrated Manufacturing* 93:102938. https://doi.org/10.1016/j.rcim.2024.102938.

Peters, Thomas J. 1992. "Encoding Mechanical Design Features for Recognition via Neural Nets." *Research in Engineering Design* 4 (2): 67–74. https://doi.org/10.1007/BF01580145.

Prabhakar, Shashikanth, and Mark R. Henderson. 1992. "Automatic Form-Feature Recognition Using Neural-Network-based Techniques on Boundary Representations of Solid Models." *Computer-Aided Design* 24 (7): 381–393. https://doi.org/10.1016/0010-4485(92)90064-H.

Qi, Charles Ruizhongtai, Li Yi, Hao Su, and Leonidas J Guibas. 2017. "Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space." *Advances in Neural Information Processing Systems*30:5105–5114.

Qin, Yuchu, Qunfen Qi, Paul J. Scott, and Xiangqian Jiang. 2019. "Status, Comparison, and Future of the Representations of Additive Manufacturing Data." *Computer-Aided Design* 111:44–64. https://doi.org/10.1016/j.cad.2019.02.004.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, et al. 2021. "Learning Transferable Visual Models from Natural Language Supervision." In *International Conference on Machine Learning*, 8748–8763. PMLR.

Rai, Rahul, Manoj Kumar Tiwari, Dmitry Ivanov, and Alexandre Dolgui. 2021. "Machine Learning in Manufacturing and Industry 4.0 Applications." *International Journal of Production Research* 59 (16): 4773–4778. https://doi.org/10.1080/00207543.2021.1956675.

Ruiz, Eneko, María Inés Torres, and Arantza del Pozo. 2023. "Question Answering Models for Human–machine Interaction in the Manufacturing Industry." *Computers in Industry* 151:103988. https://doi.org/10.1016/j.compind.2023.103988.

Shi, Peizhi, Qunfen Qi, Yuchu Qin, Paul J. Scott, and Xiangqian Jiang. 2020a. "Intersecting Machining Feature Localization and Recognition via Single Shot Multibox Detector." *IEEE Transactions on Industrial Informatics* 17 (5): 3292–3302. https://doi.org/10.1109/TII.9424.

Shi, Peizhi, Qunfen Qi, Yuchu Qin, Paul J. Scott, and Xiangqian Jiang. 2020b. "A Novel Learning-Based Feature Recognition Method Using Multiple Sectional View Representation." *Journal of Intelligent Manufacturing* 31 (5): 1291–1309. https://doi.org/10.1007/s10845-020-01533-w.

Shi, Peizhi, Qunfen Qi, Yuchu Qin, Paul J. Scott, and Xiangqian Jiang. 2022. "Highly Interacting Machining Feature Recognition via Small Sample Learning." *Robotics and Computer-Integrated Manufacturing* 73:102260. https://doi.org/10.1016/j.rcim.2021.102260.

Shi, Yang, Yicha Zhang, and Ramy Harik. 2020. "Manufacturing Feature Recognition with a 2D Convolutional Neural Network." *CIRP Journal of Manufacturing Science and Technology* 30:36–57. https://doi.org/10.1016/j.cirpj.2020.04.001.

Shi, Yang, Yicha Zhang, Kaishu Xia, and Ramy Harik. 2020. "A Critical Review of Feature Recognition Techniques." *Computer-Aided Design and Applications* 17 (5): 861–899. https://doi.org/10.14733/cadaps.

Singh, Shubhendu Kumar, Raj Pradip Khawale, Subhashis Hazarika, Ankur Bhatt, Brian Gainey, Benjamin Lawler, and Rahul Rai. 2024. "Hybrid Physics-Infused 1D-CNN Based Deep Learning Framework for Diesel Engine Fault Diagnostics." *Neural Computing and Applications* 36 (28): 17511–17539.

Stavropoulos, Panagiotis, Kyriakos Sabatakakis, Alexios Papacharalampopoulos, and Dimitris Mourtzis. 2022. "Infrared (IR) Quality Assessment of Robotized Resistance Spot Welding Based on Machine Learning." *The International Journal of Advanced Manufacturing Technology* 119 (3): 1785–1806. https://doi.org/10.1007/s00170-021-08320-8.

Subrahmanyam, Somashekar, and Michael Wozny. 1995. "An Overview of Automatic Feature Recognition Techniques for Computer-Aided Process Planning." *Computers in Industry* 26 (1): 1–21. https://doi.org/10.1016/0166-3615(95)80003-4.

Sunil, V. B., and S. S. Pande. 2009. "Automatic Recognition of Machining Features Using Artificial Neural Networks." *The International Journal of Advanced Manufacturing Technology* 41 (9): 932–947. https://doi.org/10.1007/s00170-008-1536-z.

Takaishi, Ippei, Satoshi Kanai, Hiroaki Date, and Hideyoshi Takashima. 2019. "Free-Form Feature Classification for Finite Element Meshing Based on Shape Descriptors and Machine Learning." *Proceedings of CAD* 19:414–419. https://doi.org/10.14733/cadconfP.2019.414-419.

Takashima, Hideyoshi, and Satoshi Kanai. 2021. "Recognition of Free-Form Features for Finite Element Meshing Using Deep Learning." *Computer-Aided Design and Applications* 19 (4): 677–693. https://doi.org/10.14733/cadaps.

Tremblet, David, Simon Thevenin, and Alexandre Dolgui. 2024. "Makespan Estimation in a Flexible Job-Shop Scheduling Environment Using Machine Learning." *International Journal of Production Research* 62 (10): 3654–3670. https://doi.org/10.1080/00207543.2023.2245918.

Vatandoust, Farzad, Xiaoliang Yan, David Rosen, and Shreyes N. Melkote. 2025. "Manufacturing Feature Recognition with a Sparse Voxel-Based Convolutional Neural Network." *Journal of Computing and Information Science in Engineering* 25 (3): 031002. https://doi.org/10.1115/1.4067334.

Verma, Arvind Kumar, and Sunil Rajotia. 2010. "A Review of Machining Feature Recognition Methodologies." *International Journal of Computer Integrated Manufacturing* 23 (4): 353–368. https://doi.org/10.1080/09511921003642121.

Vidanes, Gerico, David Toal, Xu Zhang, Andy Keane, Jon Gregory, and Marco Nunez. 2024. "Extending Point-Based Deep Learning Approaches for Better Semantic Segmentation in CAD." *Computer-Aided Design* 166:103629. https://doi.org/10.1016/j.cad.2023.103629.

Wang, Jianxiang, and Shenquan Liu. 1993. "Hopfield Neural Network-Based Automatic Recognition for 3-D Features." In

Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan), Vol. 3, 2121–2124. IEEE.

Wang, PengYu, Wen-An Yang, and YouPeng You. 2023. "A Hybrid Learning Framework for Manufacturing Feature Recognition Using Graph Neural Networks." *Journal of Manufacturing Processes*85:387–404. https://doi.org/10.1016/j.jmapro.2022.10.075.

Worner, Jonathan M., Daniella Brovkina, and Oliver Riedel. 2021. "Feature Recognition for Graph-Based Assembly Product Representation Using Machine Learning." In *2021 21st International Conference on Control, Automation and Systems (ICCAS)*, 629–635. IEEE.

Wu, Hongjin, Ruoshan Lei, Pei Huang, and Yibing Peng. 2023. "A Semi-supervised Learning Framework for Machining Feature Recognition on Small Labeled Sample." *Applied Sciences* 13 (5): 3181. https://doi.org/10.3390/app13053181.

Wu, Hongjin, Ruoshan Lei, Yibing Peng, and Liang Gao. 2024. "AAGNet: A Graph Neural Network towards Multi-task Machining Feature Recognition." *Robotics and Computer-Integrated Manufacturing* 86:102661. https://doi.org/10.1016/j.rcim.2023.102661.

Wu, Muh-Cherng, and S. R. Jen. 1996. "A Neural Network Approach to the Classification of 3D Prismatic Parts." *The International Journal of Advanced Manufacturing Technology* 11 (5): 325–335. https://doi.org/10.1007/BF01845691.

Wu, Yan, Jiale Hu, Hongyu Gao, Xiaoshuai Chen, Muchun Fan, and Feng Hong. 2026. "Five-Axis Support-Free Adaptive Slicing Optimization for STL Model Detail Features." *Progress in Additive Manufacturing* 11 (1): 611–625. https://doi.org/10.1007/s40964-025-01367-z.

Xia, Mingyuan, Xianwen Zhao, and Xiaofeng Hu. 2024. "Machining Feature and Topological Relationship Recognition Based on a Multi-task Graph Neural Network." *Advanced Engineering Informatics*62:102721. https://doi.org/10.1016/j.aei.2024.102721.

Xiao, Xinyi, Yousub Lee, and Thomas Feldhausen. 2025. "Autonomous Direct Freeform Fabrication Strategy for Multi-Axis Additive Manufacturing." *The International Journal of Advanced Manufacturing Technology* 137 (7): 3525–3539. https://doi.org/10.1007/s00170-025-15329-w.

Xu, Tongming, Jianxun Li, and Zhuoning Chen. 2022. "Automatic Machining Feature Recognition Based on MBD and Process Semantics." *Computers in Industry* 142:103736. https://doi.org/10.1016/j.compind.2022.103736.

Yang, Dingye, Yingguang Li, Xu Liu, Tianchi Deng, Changqing Liu, and Ke Xu. 2025. "A Data-Driven Approach for Volume Feature Recognition Based on Cell Graph." *International Journal of Computer Integrated Manufacturing* 38 (6): 844–860.

Yang, Jianping, Qiaoyun Wu, Yuan Zhang, Jiajia Dai, and Jun Wang. 2025. "A Hybrid Recognition Framework for Highly Interacting Machining Features Based on Primitive Decomposition, Learning and Reconstruction." *Computer-Aided Design* 179:103813. https://doi.org/10.1016/j.cad.2024.103813.

Yao, Jun-Feng, Yong Yang, Xue-Cheng Wang, and Xiao-Peng Zhang. 2023. "Systematic Review of Digital Twin Technology and Applications." *Visual Computing for Industry, Biomedicine, and Art* 6 (1): 10. https://doi.org/10.1186/s42492-023-00137-4.

Yao, Xinhua, Di Wang, Tao Yu, Congcong Luan, and Jianzhong Fu. 2023. "A Machining Feature Recognition Approach Based on Hierarchical Neural Network for Multi-feature Point Cloud Models." *Journal of Intelligent Manufacturing* 34 (6): 2599–2610. https://doi.org/10.1007/s10845-022-01939-8.

Yeo, Changmo, Byung Chul Kim, Sanguk Cheon, Jinwon Lee, and Duhwan Mun. 2021. "Machining Feature Recognition Based on Deep Neural Networks to Support Tight Integration with 3D CAD Systems." *Scientific Reports* 11 (1): 1–20. https://doi.org/10.1038/s41598-021-01313-3.

Yu, Dongran, Bo Yang, Dayou Liu, Hui Wang, and Shirui Pan. 2023. "A Survey on Neural-Symbolic Learning Systems." *Neural Networks* 166:105–126. https://doi.org/10.1016/j.neunet.2023.06.028.

Zhang, Hang, Wenhu Wang, Shusheng Zhang, Zhen Wang, Yajun Zhang, Jingtao Zhou, and Bo Huang. 2024. "Point Cloud Self-supervised Learning for Machining Feature Recognition." *Journal of Manufacturing Systems* 77:78–95. https://doi.org/10.1016/j.jmsy.2024.08.029.

Zhang, Hang, Wenhu Wang, Shusheng Zhang, Yajun Zhang, Jingtao Zhou, Zhen Wang, Bo Huang, and Rui Huang. 2024. "A Novel Method for Intersecting Machining Feature Segmentation via Deep Reinforcement Learning." *Advanced Engineering Informatics* 59:102256. https://doi.org/10.1016/j.aei.2023.102256.

Zhang, Hang, Shusheng Zhang, Yajun Zhang, Jiachen Liang, and Zhen Wang. 2022. "Machining Feature Recognition Based on a Novel Multi-task Deep Learning Network." *Robotics and Computer-Integrated Manufacturing* 77:102369. https://doi.org/10.1016/j.rcim.2022.102369.

Zhang, Shuming, Zhidong Guan, Hao Jiang, Xiaodong Wang, and Pingan Tan. 2024. "BrepMFR: Enhancing Machining Feature Recognition in B-rep Models through Deep Learning and Domain Adaptation." *Computer Aided Geometric Design*111:102318. https://doi.org/10.1016/j.cagd.2024.102318.

Zhang, Yajun, Shusheng Zhang, Rui Huang, Bo Huang, Jiachen Liang, Hang Zhang, and Zheng Wang. 2022. "Combining Deep Learning with Knowledge Graph for Macro Process Planning." *Computers in Industry* 140:103668. https://doi.org/10.1016/j.compind.2022.103668.

Zhang, Yu, Yongsheng Zhang, Kaiwen He, Dongsheng Li, Xun Xu, and Yadong Gong. 2022. "Intelligent Feature Recognition for STEP-NC-compliant Manufacturing Based on Artificial Bee Colony Algorithm and Back Propagation Neural Network." *Journal of Manufacturing Systems* 62:792–799. https://doi.org/10.1016/j.jmsy.2021.01.018.

Zhang, Zhibo, Prakhar Jaiswal, and Rahul Rai. 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.

Zulkifli, A. H., and S. Meeran. 1999a. "Decomposition of Interacting Features Using a Kohonen Self-organizing Feature Map Neural Network." *Engineering Applications of Artificial Intelligence* 12 (1): 59–78. https://doi.org/10.1016/S0952-1976(98)00050-5.

Zulkifli, A. H., and S. Meeran. 1999b. "Feature Patterns in Recognizing Non-interacting and Interacting Primitive, Circular and Slanting Features Using a Neural Network." *International Journal of Production Research* 37 (13): 3063–3100. https://doi.org/10.1080/002075499190428.