scientific reports

OPEN

Machining feature recognition based on deep neural networks to support tight integration with 3D CAD systems

Changmo Yeo[1], Byung Chul Kim[2], Sanguk Cheon[3], Jinwon Lee[1] & Duhwan Mun[1]✉

Recently, studies applying deep learning technology to recognize the machining feature of three-dimensional (3D) computer-aided design (CAD) models are increasing. Since the direct utilization of boundary representation (B-rep) models as input data for neural networks in terms of data structure is difficult, B-rep models are generally converted into a voxel, mesh, or point cloud model and used as inputs for neural networks for the application of 3D models to deep learning. However, the model's resolution decreases during the format conversion of 3D models, causing the loss of some features or difficulties in identifying areas of the converted model corresponding to a specific face of the B-rep model. To solve these problems, this study proposes a method enabling tight integration of a 3D CAD system with a deep neural network using feature descriptors as inputs to neural networks for recognizing machining features. Feature descriptor denotes an explicit representation of the main property items of a face. We constructed 2236 data to train and evaluate the deep neural network. Of these, 1430 were used for training the deep neural network, and 358 were used for validation. And 448 were used to evaluate the performance of the trained deep neural network. In addition, we conducted an experiment to recognize a total of 17 types (16 types of machining features and a non-feature) from the B-rep model, and the types for all 75 test cases were successfully recognized.

Due to structural changes in the manufacturing industry following the development of information technology, the demand for online manufacturing support has increased worldwide. Following this trend, the online manufacturing support platform industry is growing. Online manufacturing support, as shown in Fig. 1, is an intermediary service that evaluates the manufacturability of three-dimensional (3D) computer-aided design (CAD) models uploaded online by customers (individuals or businesses) and selects manufacturers. Furthermore, it calculates estimates, requests for production to manufacturers, and manages the entire process of delivering products to customers[1]. Manufacturability evaluation means the process of determining manufacturing methods, level of difficulty in manufacturing, manufacturability, and manufacturing costs for a customer-supplied CAD model[2]. Automation of the manufacturability evaluation process is essential for providing smooth online manufacturing support services. This is because it minimizes human interference and enables quick response regarding numerous manufacturing requests.

For automated manufacturability evaluation, machining features, which are necessary for making the designed shape from stock, are essential. Machining features include machining tools and processing parameters to be used to manufacture products. Using these, the manufacturing errors and manufacturing time can be identified through machining simulation and the necessary manufacturing tools, materials, and time can be ultimately estimated. However, as designers and manufacturers are different in the intermediary service platform, machining feature information is not provided for product security reasons. Therefore, for manufacturability evaluation, we have no choice but to use boundary representation (B-rep) models that contain only product shape information, not including feature information. So, it is necessary to find machining features from the B-rep model, which is possible through machining feature recognition.

[1]School of Mechanical Engineering, Korea University, 145, Anam-ro, Seongbuk-gu, Seoul 02841, South Korea. [2]School of Mechanical Engineering, Korea University of Technology and Education, 1600 Chungjeol-ro, Byeongcheon-myeon, Dongnam-gu, Cheonan-si, Chungcheongnam-do 31253, South Korea. [3]Department of Integrative Systems Engineering, Ajou University, 206, Worldcup-ro, Yeongtong-gu, Suwon 16499, South Korea. ✉email: dhmun@korea.ac.kr

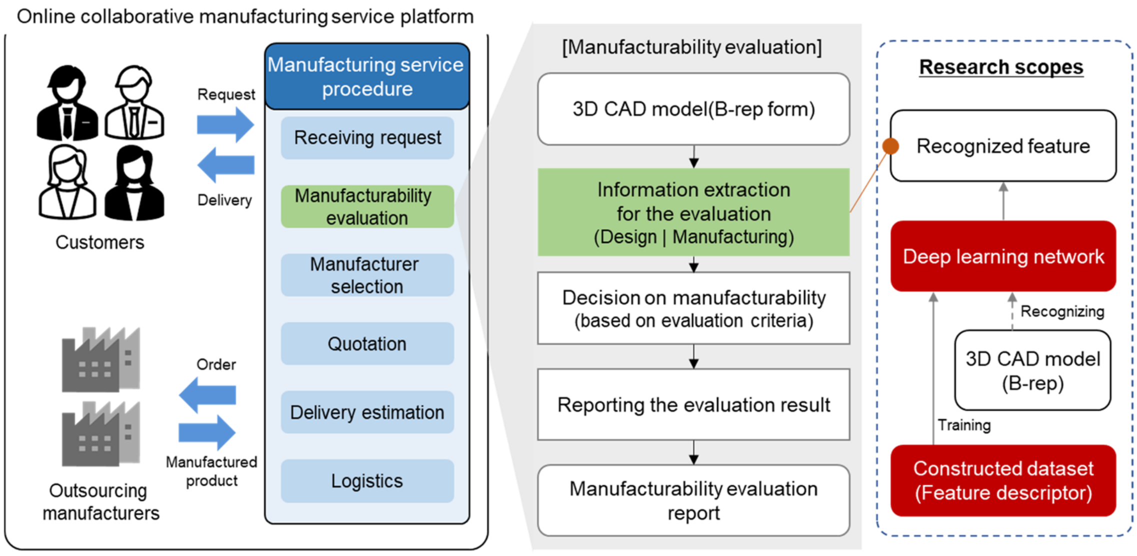

Online collaborative manufacturing service platform
Customers
Request
Delivery
Manufacturing service procedure
Receiving request
Manufacturability evaluation
Manufacturer selection
Quotation
Delivery estimation
Logistics
Outsourcing manufacturers
Order
Manufactured product
[Manufacturability evaluation]
3D CAD model(B-rep form)
Information extraction for the evaluation (Design | Manufacturing)
Decision on manufacturability (based on evaluation criteria)
Reporting the evaluation result
Manufacturability evaluation report
Research scopes
Recognized feature
Deep learning network
Recognizing
3D CAD model (B-rep)
Training
Constructed dataset (Feature descriptor)

Figure 1. Research motivation.

Common methods for recognizing the machining features of machine processing are based on algorithms including graph-based, volume decomposition, and hint-based, and similarity-based methods. For algorithm-based methods, however, the recognition rate for complicated shapes is low. Moreover, it is difficult to recognize when machining features intersect with each other, even if the shapes are not complicated. In addition, most algorithms have high complexity and long recognition time[3–16]. Recently, studies on the recognition of machining features using deep learning to compensate for the shortcomings of algorithm-based methods have been reported. When using deep learning, data should typically be represented in a lattice-structured form. Therefore, since the B-rep model cannot be used as it is, previous studies converted 3D shapes into voxel, projected images, point clouds, etc., for using the deep learning[17–26].

However, in the course of the conversion, the following problems arise. First, the geometric information of the model may be significantly lost or changed. For example, local features with a size smaller than the resolution of the converted model are lost. Moreover, in the case of formats other than mesh, accurate representation of a curved surface is difficult. Second, the association between the information obtained through deep learning and topology elements of the original B-rep model is difficult to identify. When the areas where features exist (a set of voxels) are detected through deep learning, the corresponding set of faces of the B-rep model should be found. However, due to the loss or change of the model's geometric information during the conversion, finding a set of faces corresponding to the detected area is difficult. This problem can be seen as a highly studied problem of persistent naming in the CAD field[27,28]. The above problem makes it difficult to integrate information obtained from deep learning and 3D CAD systems that have to deal with B-rep models when realizing online manufacturing support platforms.

To solve these problems, in this study, we propose a method to ensure tight integration[29] with 3D CAD systems. The proposed method defines a feature descriptor for each face of the shape. It defines and trains deep neural network models that use feature descriptors as input and feature types as outputs, and then it uses them to recognize machining features to which each face belongs. Since the proposed method recognizes machining features for each face, the recognition results of deep neural networks can be directly related to the original B-rep model.

This study has the following academic contributions. First, we propose a deep learning-based machining feature recognition architecture that enables tight integration with 3D CAD systems. Second, we propose the concept and data structure of the feature descriptor and utilize it in the feature recognition process. Third, we develop a deep neural network that classifies machining feature types by receiving the input of the feature descriptor. To the best of our knowledge, this study is the first to recognize machining features using deep learning to enable integration with 3D CAD systems.

The structure of this study is as follows. Section "Related studies" presents the related research. Section "System construction and process" proposes a deep learning-based machining feature recognition method for tight integration with 3D CAD systems. Section "Machining feature recognition using deep learning technology" explains the recognition of machining features based on deep neural networks, a key element of the proposed architecture. Section "Implementation and experimentation" discusses the implementation and experimental results. Finally, the conclusions and future research directions are presented in "Conclusions".

Related studies

Representative methods for recognizing machining features in B-rep type 3D CAD models include graph-based, convex decomposition, cell-based decomposition, hint-based, similarity-based, and artificial neural-network-based methods.

The graph-based method represents the adjacent relation between the faces and edges of the whole shape and features as a graph structure and then finds and recognizes sub-graphs of the features in the graph of the whole shape. Joshi and Chang[3] used heuristics to solve feature intersection problems in machining feature recognition. Chuang and Henderson[4] utilized the graph-based method using vertex and edge to search for patterns in machining features. Gavankar and Henderson[5] proposed a method for separating the connections after showing that the graph connection between the protrusion and depression areas is doubly constructed. Graph-based methods have the advantage of being able to easily add new features to be recognized and apply to various domains. However, it is difficult to apply when the topology structure of features is variable or features intersect. In addition, since graph search takes exponential time, it is difficult to apply to complicated shapes.

In convex decomposition and cell-based decomposition methods, features are recognized from simple shapes after complicated shapes are decomposed into simple ones. The convex decomposition method decomposes the target shape using the convex hull and delta volume. Tang and Woo[6] proposed alternating sum of volumes (ASV), which allows features to be recognized by decomposing shapes in the convex decomposition method. However, there is a problem in ASV when decomposition on a particular shape does not converge. Kim[7] proposed alternating sum of volumes with partitioning (ASVP) decomposition to solve ASP problems and used it to recognize features. The convex decomposition method can well recognize features even when they intersect, but since it cannot be applied to shapes with curved surfaces, fillets or rounds need to be removed and curved parts need to be converted to polyhedron in advance.

In cell-based decomposition, the shapes are decomposed into simple cells and the decomposed cells are recombined to form a maximum volume to find features. Sakurai and Dave[8] proposed a method of decomposing the shapes into small cells with simple shapes and recombining these cells to form large volumes. Woo[9] presented a method to perform cell-based decomposition faster than traditional cell-based decomposition. Cell-based decomposition methods can also be applied even when features intersect, and feature recognition is possible when secondary curved surfaces are included. However, it cannot be applied to complicated shapes because the process of recombining cells has high time complexity.

The hint-based method starts with minimal traces or hints for feature recognition, instead of finding complete feature patterns, and finds features through a geometric inference process for surrounding shapes. Vandenbrande and Requisha[10] developed an object-oriented feature finder (OOFF), an algorithm that explores hints from faces regarding slots, holes, and pockets. Regli[11] developed an algorithm to explore hints using edges and vertices rather than faces. Han and Requicha[12] developed incremental feature finder (IF^2), which extends the functionality of OOFF. Disadvantageously, recognition rules need to be individually defined for each feature in hint-based methods.

The similarity-based method recognizes features by examining how similar two comparison shapes are. Hong et al.[13] generated low- and high-resolution models from the B-rep model via multi-resolution modeling. They used the low-resolution model for comparing the whole shape, and the high-resolution model for comparing detailed shapes. Ohbuchi and Furuya[14] and Liu et al.[15] proposed a method to compare the similarity of the shapes contained in images after generating images of the 3D model from multiple viewpoints. Sánchez-Cruz and Bribiesca[16] compared similarities after converting 3D models to voxel formats. However, this method cannot consider the properties of faces or other properties that features have, such as adjacency relations.

Recently, methods have been proposed for recognizing features in 3D models using artificial neural networks[17–21]. Jian et al.[22] proposed an improved novel bat algorithm (NBA) incorporating NBA, which was developed to complement existing neural networks with long learning time, using a graph-based method. Zhang et al.[23] recognized 24 kinds of machining features by applying 3D convolutional neural networks. Shi et al.[24] proposed MsvNet, a deep learning technology based on multiple sectional view (MSV) representation, which was used to recognize machining features. Peddireddy et al.[25] proposed a method to identify machining processes based on 3D convolutional neural networks and transfer learning. Zhang et al.[26] constructed the PointwiseNet based on 3D point clouds and showed high performance by applying the constructed deep learning model to 3D shape retrieval. However, in the artificial neural network-based method, tightly integrating the B-rep model with 3D CAD systems is difficult because the B-rep model cannot be directly used and it needs to be converted to other formats, such as a voxel. In particular, the accurate identification of the faces on the B-rep model is difficult, which corresponds to the detection of areas corresponding to features from a voxel.

System construction and process

In this study, the target range of machining methods is limited to tuning, milling, and drilling as shown in Fig. 2a. Sixteen machining features of Fig. 2b are to be recognized. The machining features to recognize are classified as five features of hole-related type, three features of slot-related type, two features of pocket-related type, two features of island-related type, two features of fillet-related type, and two features of chamfer-related type.

The proposed method using deep learning-based machining feature recognition is shown in Fig. 3, comprising online and offline processes. The online process generates feature descriptors for each face of the B-rep model loaded into a 3D CAD system, inputs them into deep neural networks, and then classifies feature types of the face. Then, the recognized type is returned to the 3D CAD system. The offline process generates feature descriptors, builds a training dataset composed of them, and then trains the deep neural networks for feature recognition.

Previous feature recognition studies identified whether the associative pattern for the face and edge that make up the B-rep model is similar to a particular feature type pattern. Here, the associative patterns between many faces and edges are used for comparison. However, in this study, we defined a base face for each feature type and recognized the target face as the feature's base face by identifying whether the target face's attributes are similar to the particular feature's base face. Here, as shown in Fig. 4, the feature descriptor explicitly represents and stores the main attributes of the face. Therefore, in this study, the machining feature recognition determines whether

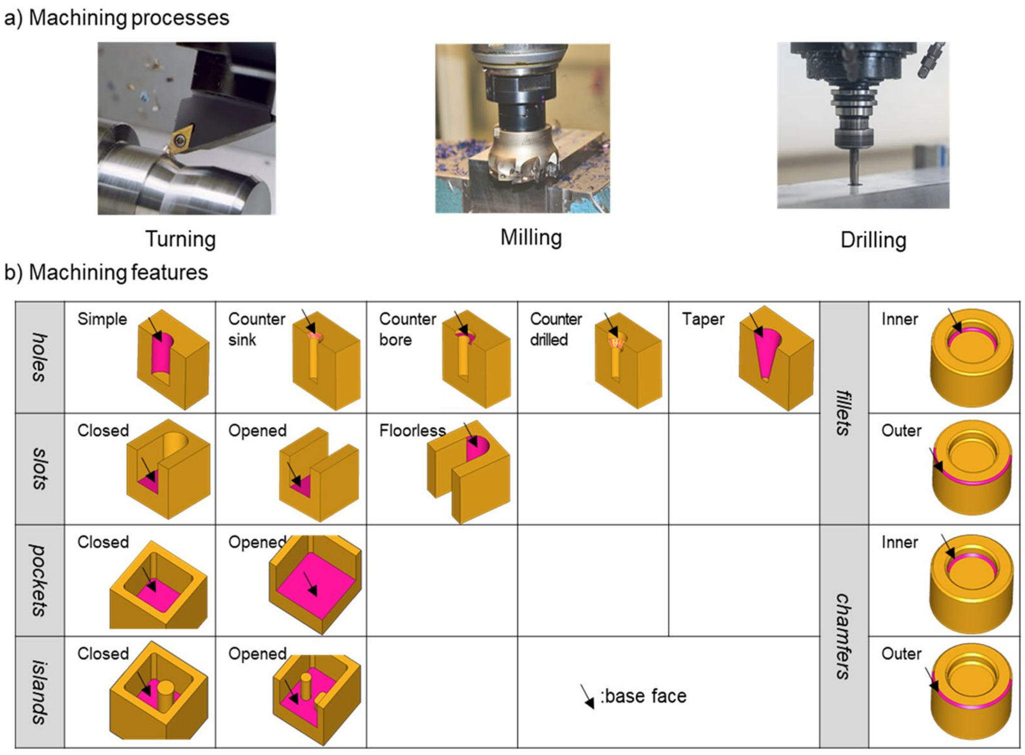
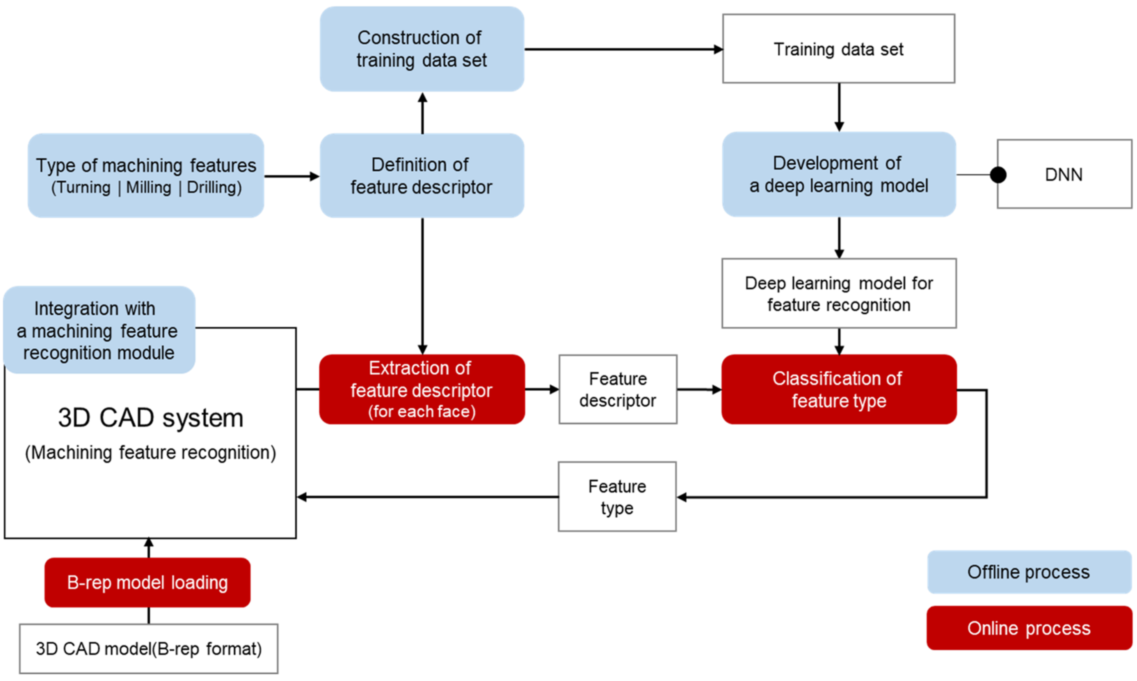


Figure 2. Machining feature types to be recognized from a B-rep model.

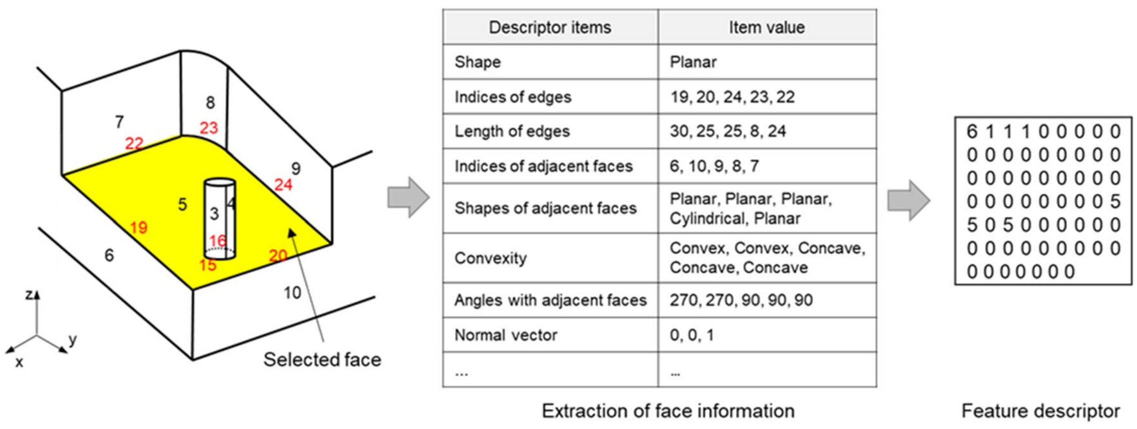


Construction of training data set
Training data set
Type of machining features (Turning | Milling | Drilling)
Definition of feature descriptor
Development of a deep learning model
DNN
Deep learning model for feature recognition
Integration with a machining feature recognition module
3D CAD system
(Machining feature recognition)
Extraction of feature descriptor (for each face)
Feature descriptor
Classification of feature type
Feature type
B-rep model loading
3D CAD model(B-rep format)
Offline process
Online process

Figure 3. Proposed method for recognizing machining features.

each face corresponds to a base face for each type of machining feature. Use of feature descriptors is effectively applicable even if interference between features makes it difficult or ambiguous to match faces or edges. Moreover, it is possible to extend the recognizable types of features by extending the descriptor.

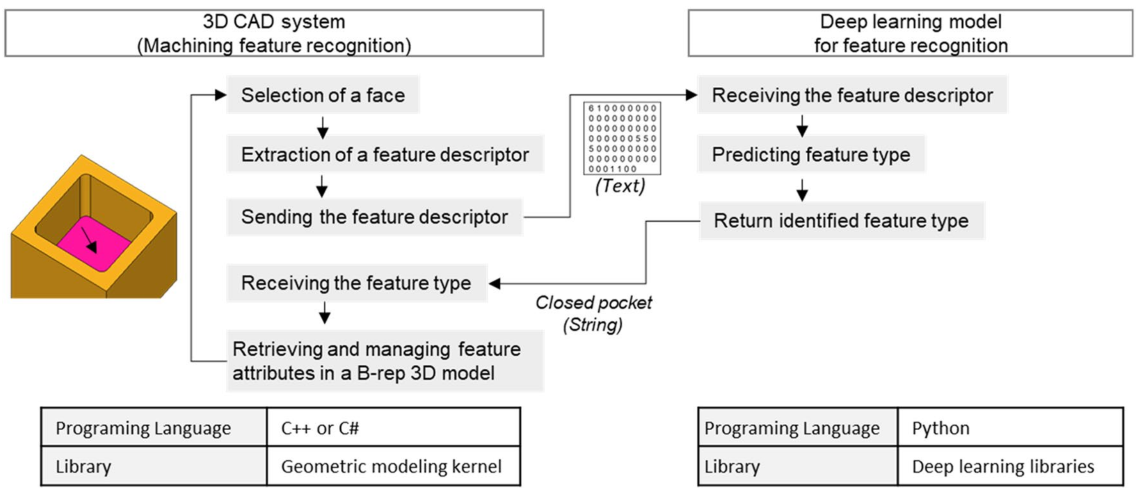
The biggest challenge encountered when developing deep neural networks that directly use B-rep models as inputs to recognize machining features is the hierarchical complexity of B-rep models and the variability of data size. Thus, previous studies used neural networks after converting B-rep models into fixed-sized voxels or multiple images. Converted voxels or images with low resolution may not properly represent curves or surfaces.



Descriptor items	Item value
Shape	Planar
Indices of edges	19, 20, 24, 23, 22
Length of edges	30, 25, 25, 8, 24
Indices of adjacent faces	6, 10, 9, 8, 7
Shapes of adjacent faces	Planar, Planar, Planar, Cylindrical, Planar
Convexity	Convex, Convex, Concave, Concave, Concave
Angles with adjacent faces	270, 270, 90, 90, 90
Normal vector	0, 0, 1
...	...

```
6 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 5
5 0 5 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0
```

Figure 4. Generation of a feature descriptor from a 3D CAD system.


3D CAD system
(Machining feature recognition)
Deep learning model
for feature recognition
Selection of a face
Extraction of a feature descriptor
Sending the feature descriptor
Receiving the feature type
Retrieving and managing feature attributes in a B-rep 3D model
(Text)
Receiving the feature descriptor
Predicting feature type
Return identified feature type
Closed pocket
(String)

Programing Language	C++ or C#
Library	Geometric modeling kernel

Programing Language	Python
Library	Deep learning libraries

Figure 5. Tight integration between a 3D CAD system and deep learning model.

Furthermore, it may result in the loss of features (e.g., hole, pocket, fillet, or chamfer). Even if feature areas in a converted voxel or image are detected, the recognition of features in the B-rep model is challenging because of the resolution differences between the B-rep model and converted voxel or image. Additionally, the lack of training datasets about segmented 3D models for each area of feature makes it difficult to conduct relevant research.

In the proposed method of machining feature recognition, input data of deep neural networks is the feature descriptor generated on each face of the B-rep model. Therefore, the proposed method is analogous to directly using the B-rep model information without conversion. Furthermore, this method can fix the input data format and size of deep neural networks because feature descriptors generated according to predefined structure for each face rather than those for a set of faces are used in feature recognition. The proposed method enables tight integration between a 3D CAD system and a deep learning model for machining feature recognition due to these characteristics, as shown in Fig. 5.

Machining feature recognition using deep learning technology

Feature descriptor.

Base face of a machining feature. This study introduces the concept of a base face on machining features. The existing studies[3–26] recognized most of the machining features from the relationship between the faces and the geometrical characteristics of each face. Referring to existing studies, we selected the base face for each feature in this study. Then, we devised a method to express the relationship between the faces constituting the feature and geometrical characteristics in terms of a descriptor based on this base face. The base face of a feature can be used as a reference for recognizing the features, even if the topology or geometry that makes up the feature partly changes. In other words, among the many faces that make up a feature, the base face best represents the feature characteristics. Figure 2 shows the base face for each type of feature covered in this study. For a hole, the base face depends on the number of faces that make up the hole. A simple hole or taper hole is a rotational shape, each with a cylinder and cone. A cylinder (cone) can be represented as either one cylindrical (conical) face or two half-cylindrical (half-conical) faces. This study assumes that a cylinder (cone) is represented by two half-cylindrical (half-conical) faces. The base face of a countersink hole or counterdrilled hole is the conical face. The base face of a counterbore hole is the planar face between the two cylindrical faces.

Item description	Notation
Base face type	Face type
Number of adjacent faces in an outer loop	(Face type|convexity): number
Number of adjacent faces with G1 or higher continuity in an outer loop	(Face type|convexity): number
Number of perpendicular adjacent faces in an outer loop	(Face type|convexity): number
Number of adjacent faces in an inner loop	(Face type|convexity): number
Number of adjacent faces with G1 or higher continuity in an inner loop	(Face type|convexity): number
Number of perpendicular adjacent faces in an inner loop	(Face type|convexity): number
Number of pairs of parallel adjacent faces	Number
Distance between a pair of parallel adjacent faces	Length (If the number of pairs of parallel adjacent faces is two or more, the shortest distance is chosen.)

Table 1. The feature descriptor used in a previous study[30].

Feature descriptor's item	Notation
Target face type	Face type
Curvature of target face	Curvature
Width of target face (for face-machining)	Result of comparison
Width of target face (for edge-machining)	Result of comparison
Adjacent faces in an outer loop	Face type|Convexity: number
Adjacent faces with C0 continuity in an outer loop	Face type|Convexity: number
Perpendicular adjacent faces in an outer loop	Face type|Convexity: number
Location and convexity of inner loops	Location|Convexity: number

Table 2. Definition of feature descriptor's items for deep learning.

For a slot, the base face, depending on the existence of the bottom face, is either the bottom or side face. For a pocket and an island, the base face is the bottom face of the pocket. For a fillet and chamfer, the cylindrical face or planar face is the base face.

Feature descriptor definition. The faces of the B-rep model have information about the faces themselves, information about the edges that make up the boundaries of the faces, information about the vertices that make up the edges, and relation information for the adjacent faces. The feature descriptor uses the type of face (e.g., planar face, cylindrical face, toroidal face, etc.), the normal vector, and loop type (inner or outer loop) for face information. Edges have the edge type (e.g., linear or curved type) and length. Vertices have coordinates information. Relation information for the adjacent face has the angle with the adjacent face, the convexity type, and the continuity type. Convexity is divided into concave or convex depending on the angle between the two adjacent faces. If the angle between the two adjacent faces is less than 180°, it is called concave, and if not, convex. Continuity is divided into C0, G1, C1, G2, and C2 according to the tangency and curvature between the two adjacent faces. Other continuities except C0 continuity have a tangent condition.

In a previous study[30], we proposed feature descriptors to recognize machining features based on similarity comparison. The proposed feature descriptors, as shown in Table 1, include a type of base face, information on the relation of the base face with adjacent faces, parallel information of adjacent faces, and distance between parallel adjacent faces. Methods for using feature descriptors in the previous study[30] are as follows. First, a feature descriptor D_f of the base face is defined for each feature type. Only the minimum information necessary for distinguishing features is stored in D_f. Moreover, in the feature recognition phase, feature descriptor D_i is generated for each face F_i that makes up the B-rep model. The similarity of feature descriptor D_i is compared with D_f for each feature type. If the calculated similarity value is higher than the predefined threshold value, F_i is determined as the base face of the feature.

In this study, as shown in Table 2, a new feature descriptor suitable for applying deep learning technology was constructed by referring to the feature descriptor proposed in the previous study[30]. There are ten types of faces: Bezier, BSpline, Rectangular Trimmed, Conical, Cylindrical, Planar, Spherical, Toroidal, Linear Extrusion, and Revolution. If any face is not of analytic type, we mark it as Unknown. The curvature of the target face is represented as positive (if the target face is convex in the normal direction), negative (if the target face is concave in the normal direction), or flat (if the target face is flat).

The width of a target face has two types of information, face- and edge-machining types, which mark whether the target face is longer (marked as Longer) or shorter (marked as Shorter) based on the predefined threshold. The width of the target face is the distance between two adjacent planes that are parallel to each other. If there are no adjacent faces parallel to each other, the minimum distance is calculated between adjacent faces that are not in

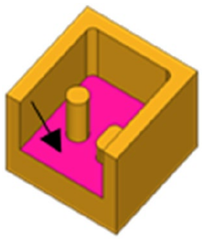
Feature type	Feature descriptor's item	Item's value
 Opened island	Target face type	Planar
	Curvature of target face	Flat
	Width of target face (for face-machining)	Longer
	Width of target face (for edge-machining)	Longer
	Adjacent faces in an outer loop	Planar|Concave: 3 Planar|Convex: 1 Cylindrical|Concave: 2
	Adjacent faces with C0 continuity in an outer loop	Planar|Concave: 3 Planar|Convex: 1 Cylindrical|Concave: 2
	Perpendicular adjacent faces in an outer loop	Planar|Concave: 3 Cylindrical|Concave: 2
	Location and convexity of inner loops	Anywhere|Concave: 2

Table 3. Feature descriptor values of the Opened island.

contact. In the face-machining type, the threshold means the maximum diameter used in machining. This item has been defined for identification between features of face-machining type (holes, slots, pockets, and islands). Among the features of face-machining types, only the width of the slot has a value smaller than the threshold[31]. If the width of the target face is less than the threshold, the target face is more likely to be determined to be a slot. In the edge-machining type, the threshold has been defined to identify the edge-machining type features (fillets and chamfers) from the face-machining type features. The threshold for edge-machining is inputted by the user, considering the following conditions. Since the width of the edge-machining is generally shorter than the width of the face-machining, the threshold of edge-machining should be smaller than the threshold of the face-machining. If the width of the target face is smaller than the threshold of the edge-machining type, the target face is more likely to be determined as an edge-machining type feature.

The adjacent face information in an outer loop shows the number of adjacent face type and convexity pairs for adjacent faces of the target face. Convexity is marked as Concave, Convex, and Unknown. If two faces have a tangent relation, the convexity is generally not considered. However, even in this case, convexity can be calculated using the cross product of the normal vectors on both the faces and direction vector of the edge shared by both sides[32]. If convexity is not calculated, it is marked as Unknown. Moreover, the adjacent face information in an outer loop also includes the number of adjacent face type and convexity pairs that have a C0 continuity relation and a perpendicular relation with the target face.

Finally, the number of location and convexity pairs represents the location of the inner loop in the target face and the convexity relation between the target face and inner loop. If the inner loop is a decomposition shape, such as a hole and pocket, it is marked as Convex. On the other hand, if the inner loop is a composition shape, such as an island, it is marked as Concave. If the inner loop's center and the target face's center are the same, it is marked as Center; otherwise, it is marked as Anywhere.

Table 3 shows the feature descriptor created in the base face of an Opened island according to the structure of the feature descriptor defined in this study. As shown in the table, the descriptor used in this study differs from those in the previous study[30] in terms of the descriptor item. When feature descriptors are created, the values for all the items that make up the descriptor are recorded.

Deep neural network for feature recognition.

Feature descriptor encoding. We used the integer encoding technique to apply feature descriptors in deep learning models. Integer encoding is a natural language processing technique, wherein the data format is converted from natural language to integer. In this section, we describe how to encode with the contents of Table 3 to aid understanding. We also describe the encoding method by dividing feature descriptor items into face, outer loop, and inner loop information.

For the type, curvature, and width items of the target face corresponding to the face information, different integer values were assigned according to the value of each descriptor item, as shown in Table 4. Table 5 shows an encoding example of the feature descriptor items regarding the target face information. As shown in the table, the descriptor's items regarding a target face represent four integer values.

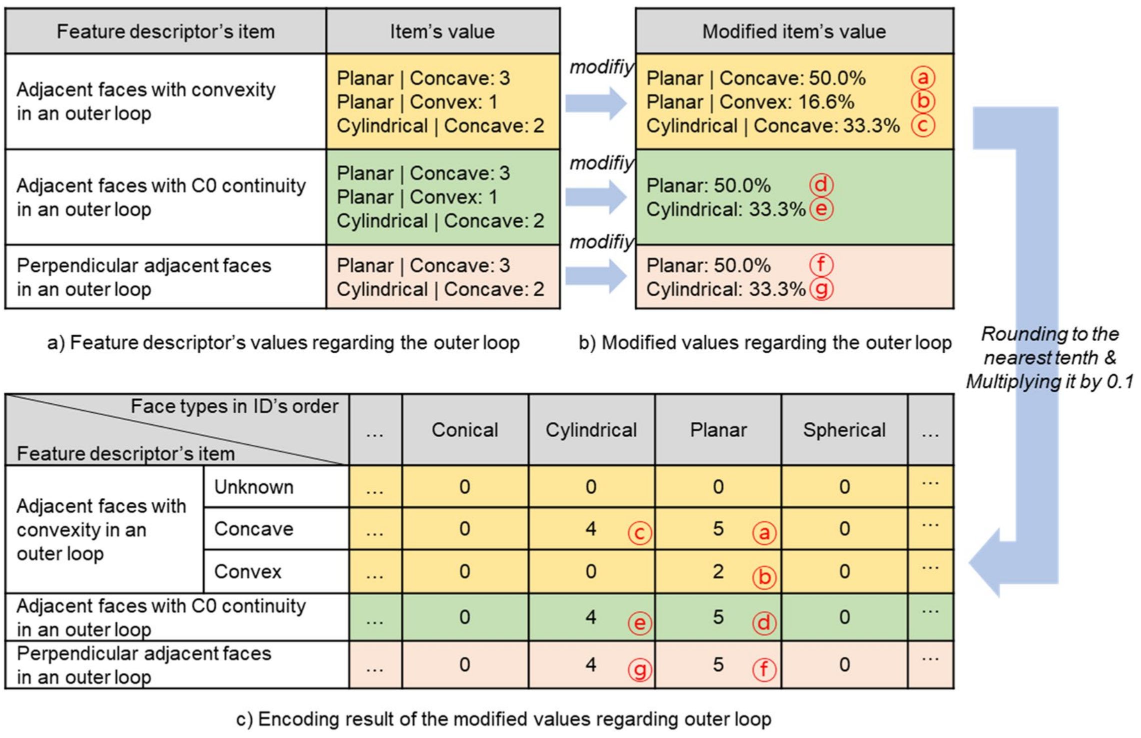
Outer loop information and descriptor item values are normalized in ratio form, as shown in Fig. 6, and then encoded as integers. The descriptor item values regarding the outer loop represent the number of adjacent faces corresponding to a specific type in the defined feature descriptor. Thus, the more adjacent faces, the larger the value of the descriptor's item naturally attains. To prevent this issue, we have normalized the number of all

	Item		
ID	Target face type	Curvature of target face	Width of target face (face-machining and edge-machining)
0	Unknown	Flat	–
1	Bezier	Positive	Longer
2	BSpline	Negative	Shorter
3	Rectangular trimmed	–	–
4	Conical	–	–
5	Cylindrical	–	–
6	Planar	–	–
7	Spherical	–	–
8	Toroidal	–	–
9	Linear Extrusion	–	–
10	Revolution	–	–

Table 4. Identifiers of descriptor item values regarding a target face.

Feature descriptor's item	Item's value	Encoding value
Target face type	Planar	6
Curvature of target face	Flat	0
Width of target face (for face-machining)	Longer	1
Width of target face (for edge-machining)	Longer	1

Table 5. Encoding of descriptor items regarding a target face.



Feature descriptor's item	Item's value	Modified item's value
Adjacent faces with convexity in an outer loop	Planar | Concave: 3 Planar | Convex: 1 Cylindrical | Concave: 2	Planar | Concave: 50.0% ⓐ Planar | Convex: 16.6% ⓑ Cylindrical | Concave: 33.3% ⓒ
Adjacent faces with C0 continuity in an outer loop	Planar | Concave: 3 Planar | Convex: 1 Cylindrical | Concave: 2	Planar: 50.0% ⓓ Cylindrical: 33.3% ⓔ
Perpendicular adjacent faces in an outer loop	Planar | Concave: 3 Cylindrical | Concave: 2	Planar: 50.0% ⓕ Cylindrical: 33.3% ⓖ

modifiy
modifiy
modifiy
a) Feature descriptor's values regarding the outer loop
b) Modified values regarding the outer loop
Rounding to the nearest tenth & Multiplying it by 0.1

Feature descriptor's item \ Face types in ID's order		…	Conical	Cylindrical	Planar	Spherical	…
Adjacent faces with convexity in an outer loop	Unknown	…	0	0	0	0	…
	Concave	…	0	4 ⓒ	5 ⓐ	0	…
	Convex	…	0	0	2 ⓑ	0	…
Adjacent faces with C0 continuity in an outer loop		…	0	4 ⓔ	5 ⓓ	0	…
Perpendicular adjacent faces in an outer loop		…	0	4 ⓖ	5 ⓕ	0	…

c) Encoding result of the modified values regarding outer loop

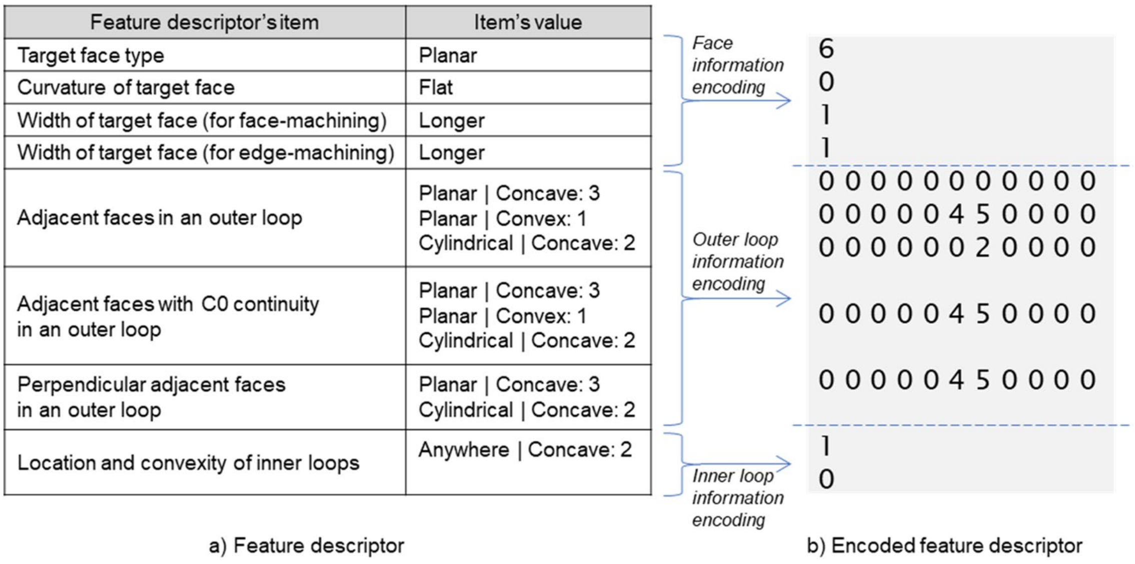
Figure 6. Encoding process of descriptor items regarding outer loop of a target face.

adjacent faces that compose the outer loop by calculating the ratio of the adjacent faces of a specific type, as shown in Fig. 6b.

It is necessary to note here that the items of adjacent faces with C0 continuity in an outer loop and perpendicular adjacent faces in an outer loop calculate only the ratio of adjacent faces with concave convexity. If two faces in contact with each other are perpendicular, convexity must be concave. Additionally, if two faces in contact with each other are in a C0 continuity relation, convexity can be either convex or concave. However,

Feature descriptor's item	Item's value	Encoding value
Location and convexity of inner loops	Anywhere|Concave: 2	1
	Center|Convex: 0	0

Table 6. Encoding of descriptor items regarding inner loops of a target face.



Feature descriptor's item	Item's value
Target face type	Planar
Curvature of target face	Flat
Width of target face (for face-machining)	Longer
Width of target face (for edge-machining)	Longer
Adjacent faces in an outer loop	Planar | Concave: 3 Planar | Convex: 1 Cylindrical | Concave: 2
Adjacent faces with C0 continuity in an outer loop	Planar | Concave: 3 Planar | Convex: 1 Cylindrical | Concave: 2
Perpendicular adjacent faces in an outer loop	Planar | Concave: 3 Cylindrical | Concave: 2
Location and convexity of inner loops	Anywhere | Concave: 2

```
6
0
1
1
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 4 5 0 0 0 0
0 0 0 0 0 0 2 0 0 0 0

0 0 0 0 0 4 5 0 0 0 0

0 0 0 0 0 4 5 0 0 0 0
1
0
```

Face information encoding
Outer loop information encoding
Inner loop information encoding
a) Feature descriptor
b) Encoded feature descriptor

Figure 7. Integer encoding result of a feature descriptor.

most machining features have concave convexity because machining features are made into shapes by removing volumes from the stock. Thus, in these items, the ratio of adjacent faces of Concave convexity is calculated. An additional point to note is to calculate the ratio of the number of faces to the total number of adjacent faces. For example, for perpendicular adjacent faces in an outer loop shown in Fig. 6b, the ratio becomes 50% because the number of Planar face types is 3 and the total number of adjacent faces is 6.

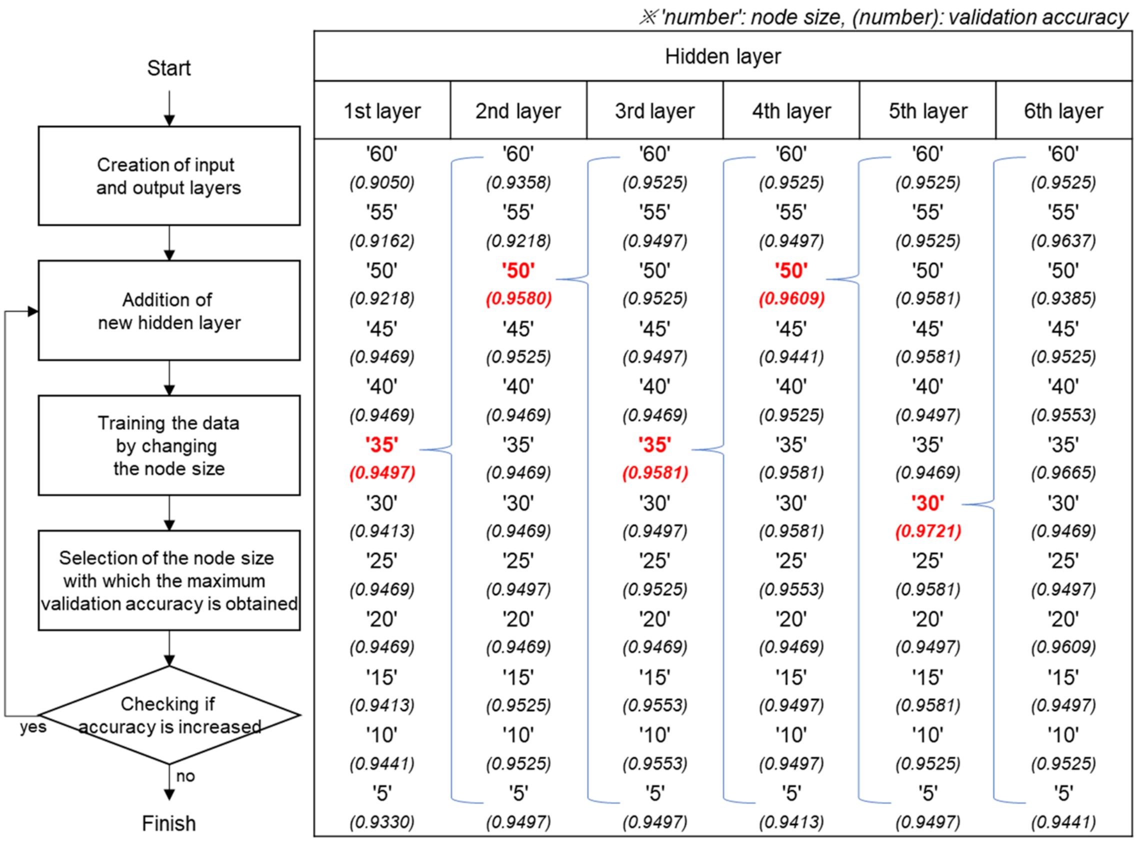
When the normalization of the feature descriptor's item values associated with outer loop information is completed, the normalized values are encoded, as shown in Fig. 6c. In the encoding process, the normalized value was rounded to the nearest tenth and then multiplied by 0.1 to ensure that the resulting value was represented as an integer value between 0 and 10. Encoding values were separately represented according to the type of faces (11 types) and the descriptor items (5 types) considering convexity, as shown in Fig. 6c. This process will result in 55 (11×5) descriptor values corresponding to the outer loop.

Features that utilize inner loop information in the machining feature classification include Islands and Counterbore holes. Counterbore holes typically have an inner loop in the form of "Center|Convex." The Islands must have an inner loop in the form of "Anywhere|Concave." Accordingly, the inner loop-related descriptor items were subdivided into "Anywhere|Concave" and "Center|Convex." In the encoding process of an inner loop-related descriptor item, as shown in Table 6, if an inner loop corresponding to the above item was present, it was recorded as 1 and as 0 if it was not present. Table 6 shows two inner loops of "Anywhere|Concave" and no inner loops of "Center|Convex" inside a target face. Thus, it was recorded as 1 and 0.

After feature descriptor encoding, a feature descriptor (Fig. 7a) is encoded into a total of 61 integer arrays (Fig. 7b) to have 4 values in face information, 55 values in outer loop information, and 2 values in inner loop information. This feature descriptor has the same size regardless of the type of faces that makes up the B-rep model, making it easy to use as input for deep neural networks.

Development of the deep neural network for feature recognition. In this study, we have developed a deep neural network of standard feed-forward fully connected method for feature recognition. Deep neural network means an artificial neural network comprising one input layer, one output layer, and n hidden layers[33–36].

To determine the optimal number of layers and node size of deep neural networks, training was performed by changing the number of hidden layers and node size according to the procedure given in Fig. 8a. We first fixed the node size of the input and output layers as 61 and 17, respectively. As the activation functions of the hidden and output layers, we used the ReLU and Softmax, respectively. As the loss function and optimization function of the neural network, we used Cross entropy loss and Adam optimizer, respectively. Then, we created the first hidden layer and trained while reducing the node size by an increment of 5 from 60, as shown in Fig. 8b, and selected the node size when the validation accuracy was the highest. When the number of nodes in the first hidden layer was determined, a new hidden layer was added to similarly select the optimal node size. With this


Start
Creation of input and output layers
Addition of new hidden layer
Training the data by changing the node size
Selection of the node size with which the maximum validation accuracy is obtained
Checking if accuracy is increased
yes
no
Finish
※'number': node size, (number): validation accuracy

Hidden layer					
1st layer	2nd layer	3rd layer	4th layer	5th layer	6th layer
'60'	'60'	'60'	'60'	'60'	'60'
(0.9050)	(0.9358)	(0.9525)	(0.9525)	(0.9525)	(0.9525)
'55'	'55'	'55'	'55'	'55'	'55'
(0.9162)	(0.9218)	(0.9497)	(0.9497)	(0.9525)	(0.9637)
'50'	**'50'**	'50'	**'50'**	'50'	'50'
(0.9218)	**(0.9580)**	(0.9525)	**(0.9609)**	(0.9581)	(0.9385)
'45'	'45'	'45'	'45'	'45'	'45'
(0.9469)	(0.9525)	(0.9497)	(0.9441)	(0.9581)	(0.9525)
'40'	'40'	'40'	'40'	'40'	'40'
(0.9469)	(0.9469)	(0.9469)	(0.9525)	(0.9497)	(0.9553)
'35'	'35'	**'35'**	'35'	'35'	'35'
(0.9497)	(0.9469)	**(0.9581)**	(0.9581)	(0.9469)	(0.9665)
'30'	'30'	'30'	'30'	**'30'**	'30'
(0.9413)	(0.9469)	(0.9497)	(0.9581)	**(0.9721)**	(0.9469)
'25'	'25'	'25'	'25'	'25'	'25'
(0.9469)	(0.9497)	(0.9525)	(0.9553)	(0.9581)	(0.9497)
'20'	'20'	'20'	'20'	'20'	'20'
(0.9469)	(0.9469)	(0.9469)	(0.9469)	(0.9497)	(0.9609)
'15'	'15'	'15'	'15'	'15'	'15'
(0.9413)	(0.9525)	(0.9553)	(0.9497)	(0.9581)	(0.9497)
'10'	'10'	'10'	'10'	'10'	'10'
(0.9441)	(0.9525)	(0.9553)	(0.9497)	(0.9525)	(0.9525)
'5'	'5'	'5'	'5'	'5'	'5'
(0.9330)	(0.9497)	(0.9497)	(0.9413)	(0.9497)	(0.9441)

(a) The process of optimal layer's configuration

(b) Validation accuracy according to node size

Figure 8. Experiment design to select an optimal configuration of a deep neural network.

Techniques	Hidden layer					Validation accuracy
	1st	2nd	3rd	4th	5th	
Dropout	O	–	–	–	–	0.9469
	–	O	–	–	–	0.9525
	–	–	O	–	–	0.9609
	–	–	–	O	–	0.9553
	–	–	–	–	O	0.9497
Batch normalization	O	–	–	–	–	0.9441
	–	O	–	–	–	0.9525
	–	–	O	–	–	0.9358
	–	–	–	O	–	0.9469
	–	–	–	–	O	0.9358
None	–	–	–	–	–	0.9721

Table 7. Validation accuracy when optimization techniques are applied to the deep neural network.

procedure, we repeated the generation of hidden layers until the validation accuracy did not increase. Repeated experiments found that the validation accuracy no longer increased in the sixth hidden layer. Accordingly, the deep neural network herein comprises five hidden layers, with node sizes of 35, 50, 35, 50, and 30.

To enhance the completion level of the deep neural network constructed in this study, we trained by applying dropout[37] and batch normalization[38] techniques. Table 7 shows the validation accuracy when the dropout layer and batch normalization layer are placed behind the nth hidden layer. As shown in Table 7, we can confirm that the validation accuracy is less when optimization techniques are applied than that when they are not applied. Based on Table 7, we confirmed that dropout and batch normalization techniques are not suitable for deep neural networks configured in this study. Through training experiments for the optimal configuration of neural networks, we finally developed the deep neural network configured as shown in Fig. 9.

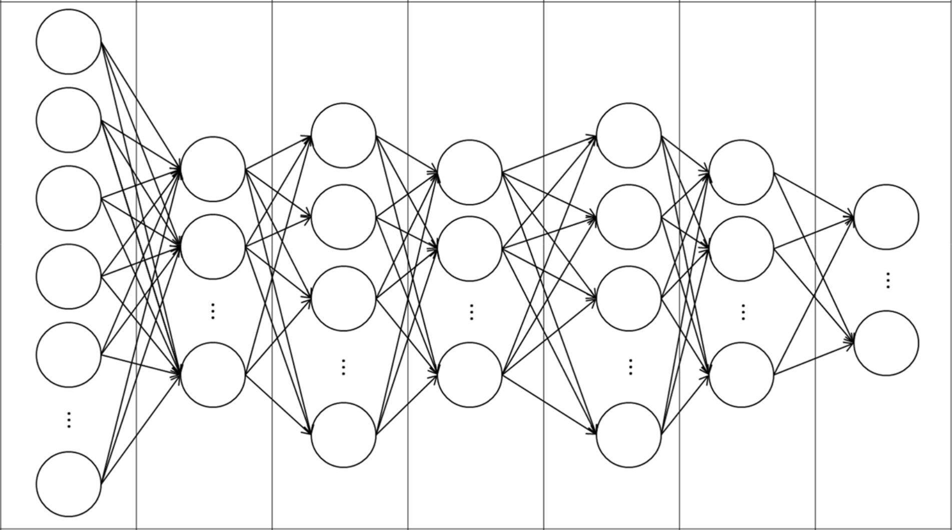
Layers	Input	Hidden_1	Hidden_2	Hidden_3	Hidden_4	Hidden_5	Output
Diagram							
Number of nodes	61	35	50	35	50	30	17
Parameters	-	2,170 (=61×35+35)	1,800 (=35×50+50)	1,785 (=50×35+35)	1,800 (=35×50+50)	1,530 (=50×30+30)	527 (=30×17+17)
Activation funtion	-	ReLU	ReLU	ReLU	ReLU	ReLU	Softmax
Loss function	Cross entropy error						
Optimizer	Adam optimizer						

Figure 9. Configuration of the deep neural network for machining feature recognition.

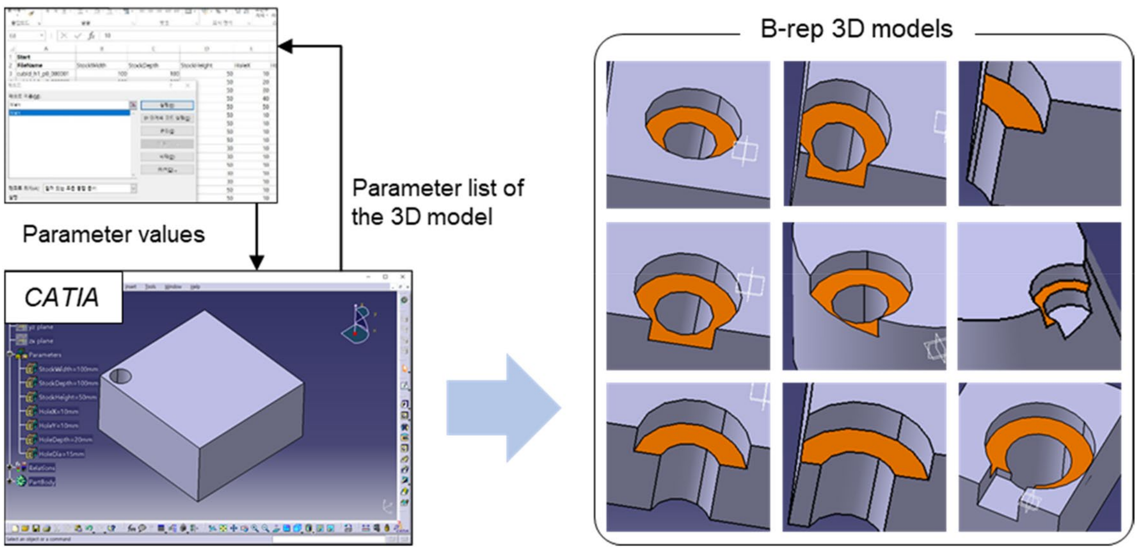
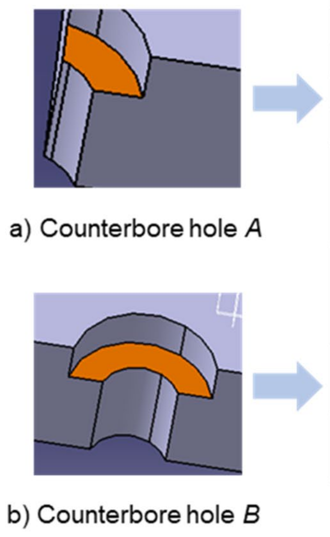


Figure 10. Parametric modeling to construct large numbers of B-rep models.

To construct training data for feature descriptors, we generated about 170,000 B-rep models through parametric modeling techniques using CATIA V5[39] and Microsoft Excel[40], as shown in Fig. 10. All B-rep models generated had one or more machining features, and the base face of each feature was given a different color. The reason for assigning different colors to the base face in the modeling process is to easily identify whether it corresponds to the base face having a feature on a particular face when creating a descriptor from the B-rep model. Since machining feature recognition aims to evaluate manufacturability, in this study, it is important



Feature descriptor's item	Item's value
Target face type	Planar
Curvature of target face	Flat
Width of target face (for face-machining)	Longer
Width of target face (for edge-machining)	Longer
Adjacent faces in an outer loop	Planar | Convex : 2 Cylindrical | Convex: 2 Cylindrical | Concave: 2
Adjacent faces with C0 continuity in an outer loop	Planar | Convex : 2 Cylindrical | Convex: 2 Cylindrical | Concave: 2
Perpendicular adjacent faces in an outer loop	Cylindrical | Concave: 2
Location and convexity of inner loops	-

c) Feature descriptor of Counterbore holes A and B

Figure 11. Different feature shapes having the same descriptor.

a) Distribution of the training dataset

Machining feature	Count	Machining feature	Count
Simple hole	72	Closed pocket	86
Countersink hole	52	Opened pocket	294
Counterbore hole	54	Closed island	66
Counterdrilled hole	120	Opened island	290
Taper hole	108	Inner fillet	114
Closed slot	254	Outer fillet	38
Opened slot	154	Inner chamfer	30
Floorless slot	314	Outer chamfer	16
Non-feature	174	Total: 2,236	

b) Ratio of the training dataset

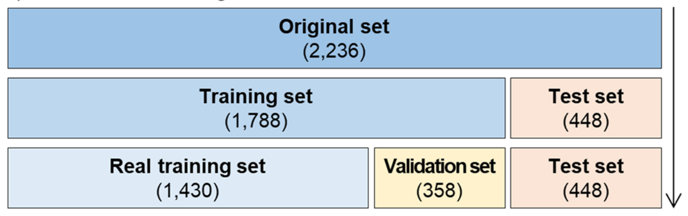


Figure 12. Overall information of the training dataset.

to distinguish between machinable features and unmachinable features. Therefore, the generated dataset also contains unrealistic B-rep models.

In the 170,000 B-rep models generated, many of them had different shapes but the same descriptors as shown in Fig. 11. We eliminated the duplicate data to prevent overfitting in the neural network's training process.

The training dataset for developing the deep neural network for machining feature recognition comprises 2236 feature descriptors, as shown in Fig. 12. This training dataset can be downloaded from the EIF lab homepage[41]. The composition of the training dataset is described in Fig. 12a. The training dataset contains feature descriptors for Simple hole (72), Countersink hole (52), Counterbore hole (54), Counterdrilled hole (120), Taper hole (108), Closed slot (254), Opened slot (154), Floorless slot (314), Closed pocket (86), Opened pocket (294), Closed island (66), Opened island (290), Inner fillet (114), Outer fillet (38), Inner chamfer (30), Outer chamfer (16), and Non-feature (174).

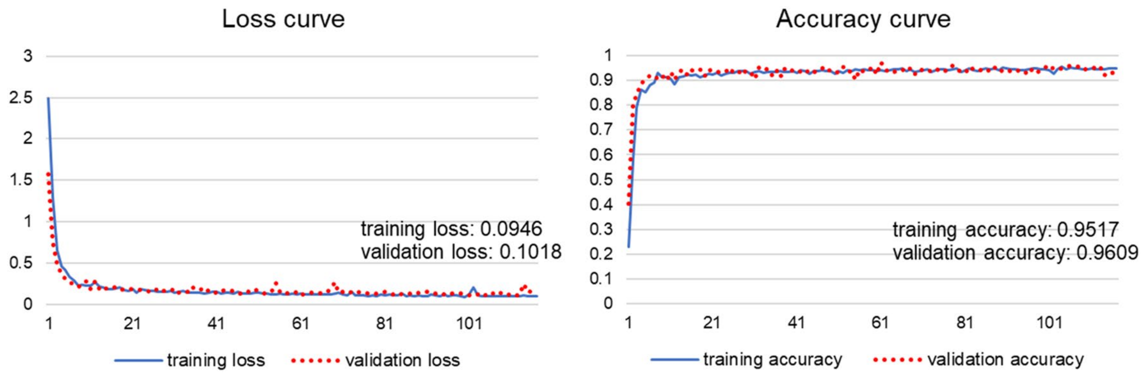


Figure 13. Learning curve of the developed deep neural network.

In the course of developing a deep neural network for the machining feature recognition, as shown in Fig. 12b, the entire dataset was divided by a ratio of 8:2, and 1788 feature descriptors were used as a training set. There is no set optimal division ratio between the training and test sets. Most of the existing studies randomly divide the training and test sets by a 7:3 or 8:2 ratio[42–45]. So, in this study, we randomly divided the entire dataset into training and test set by a ratio of 8:2. In the training set, the real training set and the validation set were randomly divided by a ratio of 8:2. As a result, the entire dataset was randomly divided into the real training set, validation set, and test set by a ratio of 64:16:20. For the training of deep neural networks, we set Batch size to 8 and Epoch to 1000. In the course of the training, deep neural networks showed a training accuracy of 0.9517, a training loss of 0.0946, a validation accuracy of 0.9609, and a validation loss of 0.1018, as shown in Fig. 13.

After the training of the deep neural networks, we validated the performance of the trained model with 448 feature descriptors that were not used in training. A confusion matrix was calculated, as shown in Fig. 14. The confusion matrix is primarily used to evaluate the performance of classification models and represents performance measures including accuracy, precision, and recall. As a result of validation, the trained deep neural network showed an accuracy of 0.9308, a mean precision of 0.9224, and a mean recall of 0.9108.

Implementation and experimentation

Implementation. According to the proposed method for machining feature recognition, a prototype system was developed, as shown in Fig. 15. We implemented a module SW for feature recognition based on the deep neural network in Python language on the Windows 10 operating system. The PyQt library was utilized to configure the module's GUI. The deep neural network was implemented using TensorFlow-based Keras and Scikit-learn libraries. In the 3D CAD system[30] for feature recognition developed in the previous study, we integrated the developed recognition module SW using the python embedding method, as shown in Fig. 15. For module development and recognition experiments, computers with Intel Core i7 CPU, 64 GB RAM, and NVIDIA GeForce GTX 760 graphics card were used.

Experimentation. The training dataset comprises descriptors that are generated from B-rep models by parametric modeling. These B-rep models have relatively simple shapes compared to the actual 3D models applied in the manufacturing field. Therefore, this experimentation tested the machining feature recognition of the complicated 3D models used in the field.

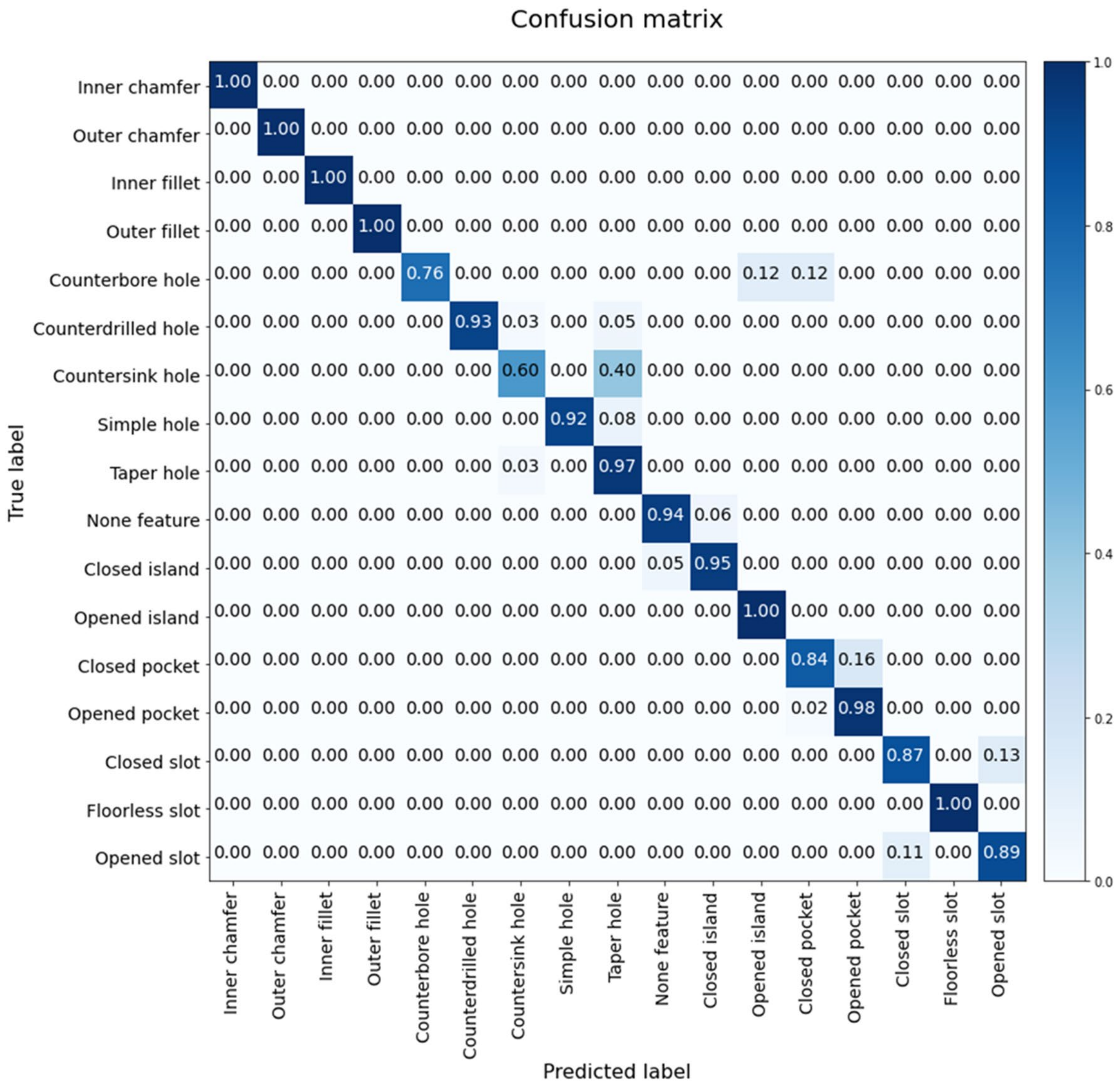
We prepared 15 B-rep models for the recognition experiment, as shown in Fig. 16, referring to the study[30,46–48] on machining feature recognition. These 3D models are parts manufactured by turning, milling, and drilling, which possess 57 machining features and 18 non-features. Accordingly, the experiment was conducted for 75 test cases. Seventy-five test cases are the generated descriptors according to the procedure explained in "Feature descriptor encoding". Nos. 15, 25, 38, and 52 of the test cases were recognition failures in the previous study[30].

Table 8 shows the results of the recognition experiment of machining features on the test cases. The probability of columns 3 and 6 in Table 8 means the highest value among the possibility of feature labels. The possibility represents the likelihood that a particular feature label is correct. The results of the recognition experiment show that the feature type, which is true, was recognized as the first priority for all 75 test cases.

Experimental results show that the deep neural network in this study calculated probabilities of over 90% on 68 of the 75 test cases. The remaining 7 test cases calculated the probability of the late 70% are Closed pockets (Nos. 3, 35, and 49), Open pockets (Nos. 4, 8, and 42), and Opened slots (No. 43). Nos. 3, 35, and 49 corresponding to Closed pockets have the same feature descriptor. Nos. 4, 8, and 42 corresponding to Opened pockets have the same feature descriptor.

To analyze features with probability of over 70%, we represented the descriptors of the two features (4 Opened pocket and 43 Opened slot), which exhibited the lowest probability, in the graph, as shown in Table 9. The graph's horizontal axis represents the index of the feature descriptor encoded by an integer, and the vertical axis represents the value at each index. Column 1 of this table represents all descriptors corresponding to a particular type in the training data. Column 2 (column 3) represents a feature descriptor after the selection of a feature with probability of 90% or more (70% or more) from the experimental results.

The feature descriptor graphs in row 2 show that sections A, B, C, and D are important for determining the recognition target face as the Opened pocket. We can confirm that all graphs for the training data and the



True label \ Predicted label	Inner chamfer	Outer chamfer	Inner fillet	Outer fillet	Counterbore hole	Counterdrilled hole	Countersink hole	Simple hole	Taper hole	None feature	Closed island	Opened island	Closed pocket	Opened pocket	Closed slot	Floorless slot	Opened slot
Inner chamfer	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Outer chamfer	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Inner fillet	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Outer fillet	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Counterbore hole	0.00	0.00	0.00	0.00	0.76	0.00	0.00	0.00	0.00	0.00	0.00	0.12	0.12	0.00	0.00	0.00	0.00
Counterdrilled hole	0.00	0.00	0.00	0.00	0.00	0.93	0.03	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Countersink hole	0.00	0.00	0.00	0.00	0.00	0.00	0.60	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Simple hole	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.92	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Taper hole	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.97	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
None feature	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.94	0.06	0.00	0.00	0.00	0.00	0.00	0.00
Closed island	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.95	0.00	0.00	0.00	0.00	0.00	0.00
Opened island	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
Closed pocket	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.84	0.16	0.00	0.00	0.00
Opened pocket	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.98	0.00	0.00	0.00
Closed slot	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.87	0.00	0.13
Floorless slot	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Opened slot	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.00	0.89

Figure 14. Confusion matrix of the developed deep neural network.

recognized faces completely correspond to each other in Section A. From the relative size of sections B, C, and D, the 1 Opened pocket tends to be more similar to the training data than the 4 Opened pocket. The graphs of the training data and the 1 Opened pocket are flat at sections B, C, and D, while the 4 Opened pocket has a relatively significant difference in the values of the sections. Consequently, the 4 Opened Pocket seemingly outputs a relatively low probability.

The feature descriptor graphs in row 3 show that sections E, F, G, and H are important for determining the recognition target face as the Opened slot. We can confirm that all graphs for the training data and the recognized faces completely correspond to each other in Section E. Sections F, G, and H show that the recognized face graphs tend to be similar to those in the training data. However, we can see that the values of sections F and H of the 43 Opened slot are located in different indexes. Consequently, the 43 Opened slot seems to output a relatively low probability, although it tends to be similar to that of the training data.

Conclusions

We proposed a method of machining feature recognition based on the deep neural network using feature descriptors to ensure tight integration with 3D CAD systems. The proposed method supports 16 types of machining feature recognition. To recognize the machining features, the proposed method generates feature descriptors from the B-rep model's face and recognizes feature types by inputting the descriptors into the deep neural network, and it returns the recognized feature types to the 3D CAD system.

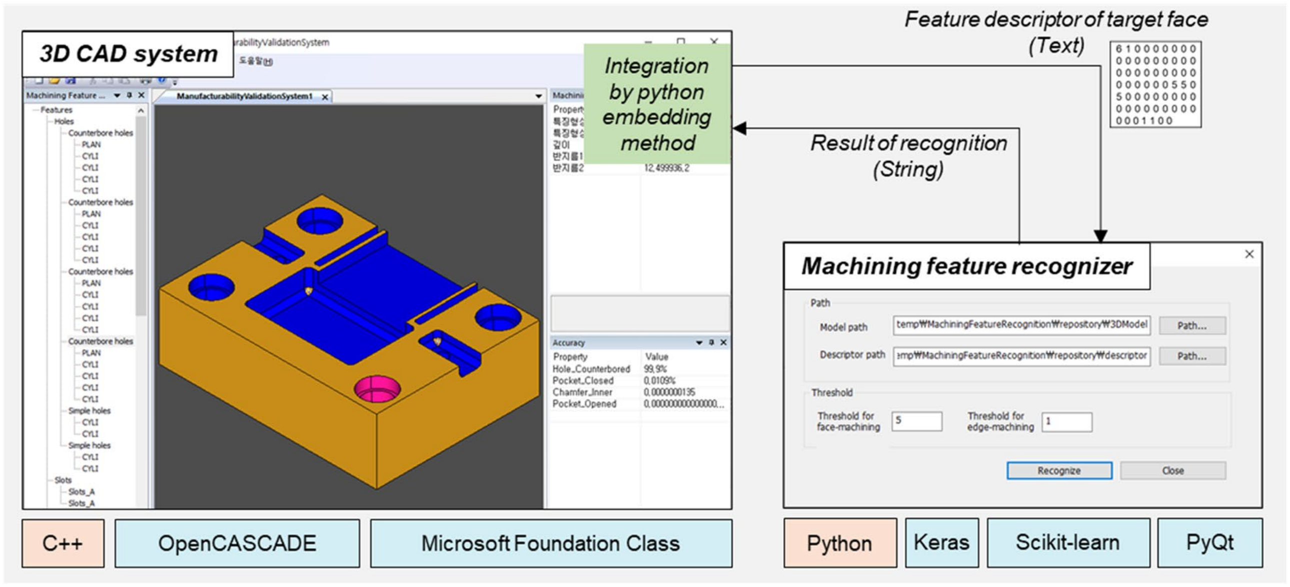

3D CAD system
Integration by python embedding method
Feature descriptor of target face (Text)
Result of recognition (String)
Machining feature recognizer
Model path
Descriptor path
Path...
Threshold for face-machining
Threshold for edge-machining
Recognize
Close
Accuracy
Hole_Counterbored 99.9%
Pocket_Closed 0.0109%
Chamfer_Inner
Pocket_Opened
C++
OpenCASCADE
Microsoft Foundation Class
Python
Keras
Scikit-learn
PyQt

Figure 15. Prototype system development through integration of a 3D CAD system with machining feature recognition based on the deep neural network.

We defined new feature descriptors suitable for the application of deep learning technology, referring to the descriptors proposed in a previous study[30]. Moreover, we used the integer encoding technique to apply the feature descriptor to the deep learning model. Since this technique can create feature descriptors having the same structure and size for each face composing the B-rep model, it can be easily used as input for the deep neural network.

The standard feed-forward fully connected method was applied to develop the deep neural network for machining feature recognition. The deep neural network has five hidden layers in addition to input and output layers. As the activation functions of the hidden and output layers, we used the ReLU and Softmax, respectively. In addition, as loss and optimization functions of the neural network, we used Cross entropy loss and Adam optimizer, respectively.

The training dataset used in the development of the deep neural network has a total of 2236 feature descriptors. We used 1788 training data for the learning of the deep neural network. We then tested the performance of the model with 448 feature descriptors that were not used for training. Consequently, the trained deep neural network showed an accuracy of 0.9308, a mean precision of 0.9224, and a mean recall of 0.9108.

In the experiment, we prepared 75 test cases for 15 B-rep models, referring to existing machining feature recognition studies[30,46–48]. In the recognition experiment for the test cases, 68 cases recognized true feature types with over 90% probability as the first priority, while the remaining seven cases recognized true feature types with over 70% probability as the first priority.

It is not easy to prove that the data for training is sufficient. So, for most DNN training, researchers generate as much different data as possible and ensure that the resulting data distribution is uniform. After creating 3D CAD models in this study, we generated the various data by transshaping the models through the parametric modeling method. The generated data distribution was made uniform by adjusting the case that transshapes for each 3D CAD model type. After training the proposed deep neural network with the generated dataset, we experimented with recognizing features with the proposed deep neural network for 75 test cases of Fig. 16. Through this experiment, we confirmed that the deep neural network trained well.

In the future, we will further subdivide the items that constitute the feature descriptor to expand recognizable machining feature types. We will also improve the probability of three feature types (Closed pockets, Opened pockets, and Opened slots) that show relatively low probability of over 70%. According to the subdivision of the items constituting the feature descriptors, we will increase the number of training data because the number of training data (2236) in this study may be insufficient. Finally, we plan to conduct a study to recognize machining features by applying a deep learning model, such as a convolutional neural network or a recurrent neural network, with higher performance than the deep neural network used in this study.

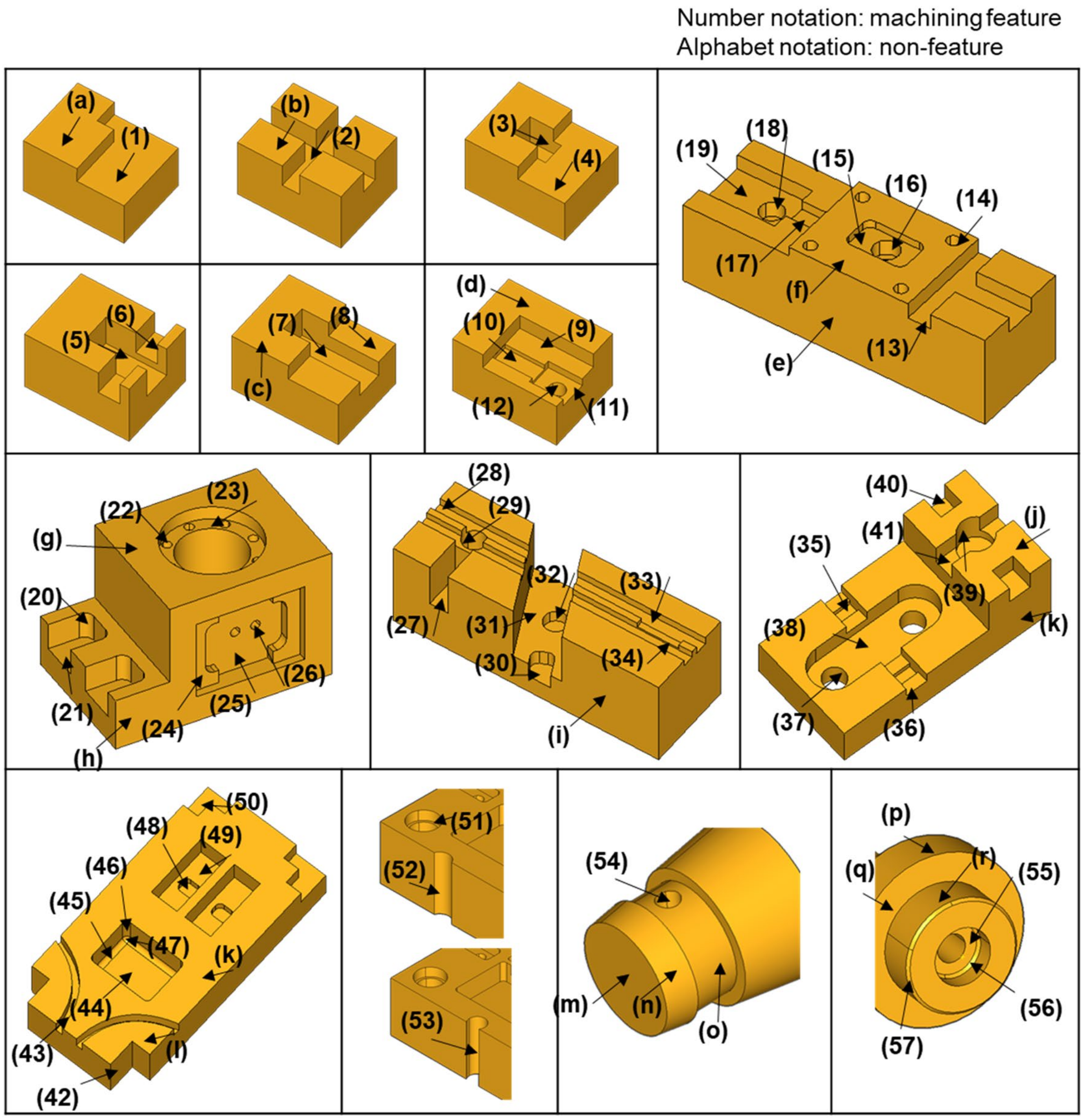


Figure 16. Test cases used in an experiment.

No.	Feature	Probability	No.	Feature	Probability
1	Opened pocket	99.9858	39	Simple hole	99.8326
2	Opened slot	99.1164	40	Opened pocket	99.9794
3	Closed pocket	78.1835	41	Opened pocket	99.9362
4	Opened pocket	77.4747	42	Opened pocket	77.4747
5	Opened pocket	99.9794	43	Opened slot	77.8519
6	Opened slot	99.4024	44	Closed pocket	97.2565
7	Opened pocket	99.9794	45	Inner fillet	100.0000
8	Opened pocket	77.4747	46	Inner fillet	100.0000
9	Opened pocket	98.8221	47	Inner fillet	99.5128
10	Opened pocket	99.9794	48	Closed pocket	99.4155
11	Opened pocket	99.9794	49	Closed pocket	78.1835
12	Simple hole	99.9977	50	Opened pocket	99.9893
13	Opened slot	99.3274	51	Counterbore hole	99.9996
14	Simple hole	99.9977	52	Simple hole	97.9496
15	Closed pocket	99.4155	53	Simple hole	99.9627
16	Counterbore hole	99.9996	54	Simple hole	99.9254
17	Opened pocket	99.9794	55	Counterbore hole	99.9526
18	Simple hole	99.9977	56	Inner chamfer	99.7046
19	Opened pocket	99.9794	57	Outer chamfer	99.2645
20	Inner fillet	100.0000	a	Non-feature	99.9615
21	Opened pocket	94.2895	b	Non-feature	99.9615
22	Simple hole	99.9977	c	Non-feature	99.9615
23	Counterbore hole	99.9996	d	Non-feature	99.9615
24	Opened pocket	99.9333	e	Non-feature	99.9615
25	Closed pocket	98.9804	f	Non-feature	99.9615
26	Simple hole	99.9977	g	Non-feature	99.9615
27	Opened pocket	99.9794	h	Non-feature	99.9615
28	Opened slot	95.4172	i	Non-feature	99.9615
29	Simple hole	99.9848	j	Non-feature	99.9188
30	Opened pocket	94.2895	k	Non-feature	99.9615
31	Opened pocket	99.0065	l	Non-feature	99.7054
32	Simple hole	99.9977	m	Non-feature	98.3473
33	Opened slot	99.4024	n	Non-feature	99.9921
34	Opened pocket	99.9858	o	Non-feature	94.0457
35	Closed pocket	78.1835	p	Non-feature	99.9995
36	Opened pocket	99.9858	q	Non-feature	99.9962
37	Simple hole	99.9627	r	Non-feature	99.9998
38	Closed pocket	99.4155			

Table 8. Machining feature recognition result for the test case.

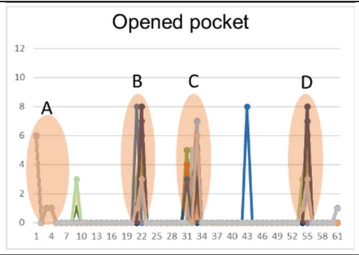
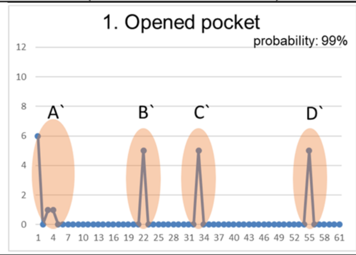
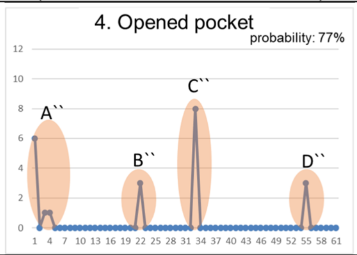
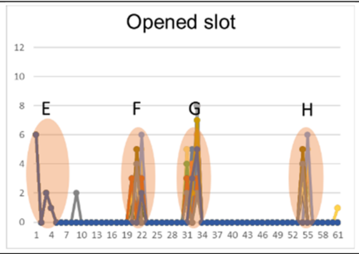
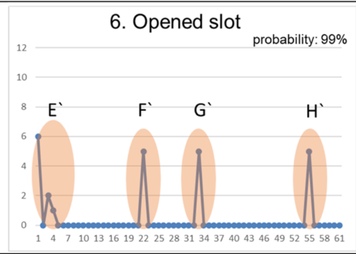
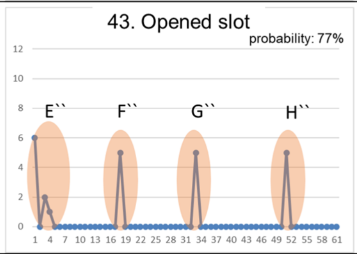
Training data	High probability feature (more than 90%)	Low probability feature (between 75% and 80%)
		
		

Table 9. Analyses of machining features showing low probability.

Consent to participate. All authors consent for participation.

Consent for publication. All authors consent for publication.

Data availability

Dataset used in this study is available from[41].

Received: 28 July 2021; Accepted: 26 October 2021
Published online: 12 November 2021

References

1. Cheng, K., Pan, P. Y. & Harrison, D. K. Web-based design and manufacturing support systems: Implementation perspectives. *Int. J. Comput. Integr. Manuf.* **14**(1), 14–27. https://doi.org/10.1080/09511920150214875 (2001).
2. Amrina, E., & Yusof, S. M. (2011). Key performance indicators for sustainable manufacturing evaluation in automotive companies. in *2011 IEEE international conference on industrial engineering and engineering management*, 1093–1097. IEEE.
3. Joshi, S. & Chang, T. C. Graph-based heuristics for recognition of machined features from a 3D solid model. *Comput. Aided Des.* **20**(2), 58–66. https://doi.org/10.1016/0010-4485(88)90050-4 (1988).
4. Chuang, S. H. & Henderson, M. R. Three-dimensional shape pattern recognition using vertex classification and vertex-edge graphs. *Comput. Aided Des.* **22**(6), 377–387. https://doi.org/10.1016/0010-4485(90)90088-T (1990).
5. Gavankar, P. & Henderson, M. R. Graph-based extraction of protrusions and depressions from boundary representations. *Comput. Aided Des.* **22**(7), 442–450. https://doi.org/10.1016/0010-4485(90)90109-P (1990).
6. Tang, K. & Woo, T. O. N. Y. Algorithmic aspects of alternating sum of volumes. Part 1: Data structure and difference operation. *Comput.-Aided Design.* **23**(5), 357–366. https://doi.org/10.1016/0010-4485(91)90029-V (1991).
7. Kim, Y. S. Recognition of form features using convex decomposition. *Comput. Aided Des.* **24**(9), 461–476. https://doi.org/10.1016/0010-4485(92)90027-8 (1992).
8. Sakurai, H. & Dave, P. Volume decomposition and feature recognition, Part II: Curved objects. *Comput. Aided Des.* **28**(6–7), 519–537. https://doi.org/10.1016/0010-4485(95)00067-4 (1996).
9. Woo, Y. Fast cell-based decomposition and applications to solid modeling. *Comput. Aided Des.* **35**(11), 969–977. https://doi.org/10.1016/S0010-4485(02)00144-6 (2003).
10. Vandenbrande, J. H. & Requicha, A. A. Spatial reasoning for the automatic recognition of machinable features in solid models. *IEEE Trans. Pattern Anal. Mach. Intell.* **15**(12), 1269–1285. https://doi.org/10.1109/34.250845 (1993).
11. Regli III, W. C. (1995). Geometric algorithms for recognition of features from solid models (Doctoral dissertation).
12. Han, J. & Requicha, A. A. Feature recognition from CAD models. *IEEE Comput. Graph. Appl.* **18**(2), 80–94. https://doi.org/10.1109/38.656791 (1998).
13. Hong, T., Lee, K. & Kim, S. Similarity comparison of mechanical parts to reuse existing designs. *Comput. Aided Des.* **38**(9), 973–984. https://doi.org/10.1016/j.cad.2006.05.004 (2006).
14. Ohbuchi, R., & Furuya, T. (2009). Scale-weighted dense bag of visual features for 3D model retrieval from a partial view 3D model. in *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 63–70. IEEE.
15. Liu, Y. J. *et al.* User-adaptive sketch-based 3-D CAD model retrieval. *IEEE Trans. Autom. Sci. Eng.* **10**(3), 783–795. https://doi.org/10.1109/TASE.2012.2228481 (2013).
16. Sánchez-Cruz, H. & Bribiesca, E. A method of optimum transformation of 3D objects used as a measure of shape dissimilarity. *Image Vis. Comput.* **21**(12), 1027–1036. https://doi.org/10.1016/S0262-8856(03)00119-7 (2003).
17. Prabhakar, S. & Henderson, M. R. Automatic form-feature recognition using neural-network-based techniques on boundary representations of solid models. *Comput. Aided Des.* **24**(7), 381–393. https://doi.org/10.1016/0010-4485(92)90064-H (1992).
18. Onwubolu, G. C. Manufacturing features recognition using backpropagation neural networks. *J. Intell. Manuf.* **10**(3), 289–299 (1999).

19. Öztürk, N. & Öztürk, F. Neural network based non-standard feature recognition to integrate CAD and CAM. *Comput. Ind.* **45**(2), 123–135. https://doi.org/10.1016/S0166-3615(01)00090-2 (2001).
20. Sunil, V. B. & Pande, S. S. Automatic recognition of machining features using artificial neural networks. *Int. J. Adv. Manuf. Technol.* **41**(9), 932–947. https://doi.org/10.1007/s00170-008-1536-z (2009).
21. Marchetta, M. G. & Forradellas, R. Q. An artificial intelligence planning approach to manufacturing feature recognition. *Comput. Aided Des.* **42**(3), 248–256. https://doi.org/10.1016/j.cad.2009.11.007 (2010).
22. Jian, C., Li, M., Qiu, K. & Zhang, M. An improved NBA-based STEP design intention feature recognition. *Futur. Gener. Comput. Syst.* **88**, 357–362. https://doi.org/10.1016/j.future.2018.05.033 (2018).
23. Zhang, Z., Jaiswal, P. & Rai, R. Featurenet: Machining feature recognition based on 3d convolution neural network. *Comput. Aided Des.* **101**, 12–22. https://doi.org/10.1016/j.cad.2018.03.006 (2018).
24. Shi, P., Qi, Q., Qin, Y., Scott, P. J. & Jiang, X. A novel learning-based feature recognition method using multiple sectional view representation. *J. Intell. Manuf.* **31**(5), 1291–1309. https://doi.org/10.1007/s10845-020-01533-w (2020).
25. Peddireddy, D. *et al.* Deep learning based approach for identifying conventional machining processes from CAD data. *Procedia Manufact.* **48**, 915–925. https://doi.org/10.1016/j.promfg.2020.05.130 (2020).
26. Zhang, D., He, F., Tu, Z., Zou, L. & Chen, Y. Pointwise geometric and semantic learning network on 3D point clouds. *Integr. Comput.-Aided Eng.* **27**(1), 57–75. https://doi.org/10.3233/ICA-190608 (2020).
27. Cheon, S. U., Mun, D., Han, S. & Kim, B. C. Name matching method using topology merging and splitting history for exchange of feature-based CAD models. *J. Mech. Sci. Technol.* **26**(10), 3201–3212. https://doi.org/10.1007/s12206-012-0827-3 (2012).
28. Mun, D. & Han, S. Identification of topological entities and naming mapping for parametric CAD model exchanges. *Int. J. CAD/CAM* **5**(1), 69–81 (2005).
29. Kim, H., Yeo, C., Lee, I. D. & Mun, D. Deep-learning-based retrieval of piping component catalogs for plant 3D CAD model reconstruction. *Comput. Ind.* **123**, 103320. https://doi.org/10.1016/j.compind.2020.103320 (2020).
30. Yeo, C., Cheon, S. & Mun, D. Manufacturability evaluation of parts using descriptor-based machining feature recognition. *Int. J. Comput. Integr. Manufact. Online First*. https://doi.org/10.1080/0951192X.2021.1963483 (2021).
31. Abrahamsen, M. Spiral tool paths for high-speed machining of 2D pockets with or without islands. *J. Comput. Design Eng.* **6**(1), 105–117. https://doi.org/10.1016/j.jcde.2018.01.003 (2019).
32. Asiabanpour, B., Mokhtar, A., Hayasi, M., Kamrani, A. & Nasr, E. A. An overview on five approaches for translating cad data into manufacturing information. *J. Adv. Manuf. Syst.* **8**(01), 89–114. https://doi.org/10.1142/S0219686709001687 (2009).
33. Chen, G., Parada, C., & Heigold, G. (2014). Small-footprint keyword spotting using deep neural networks. in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4087–4091. IEEE.
34. Bengio, Y., Courville, A. & Vincent, P. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 1798–1828. https://doi.org/10.1109/TPAMI.2013.50 (2013).
35. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **61**, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003 (2015).
36. Zhang, D., Zou, L., Zhou, X. & He, F. Integrating feature selection and feature extraction methods with deep learning to predict clinical outcome of breast cancer. *IEEE Access* **6**, 28936–28944. https://doi.org/10.1109/ACCESS.2018.2837654 (2018).
37. Baldi, P. & Sadowski, P. J. Understanding dropout. *Adv. Neural. Inf. Process. Syst.* **26**, 2814–2822 (2013).
38. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. in *International conference on machine learning*, 448–456. PMLR.
39. Dassault Systèmes. CATIA V5. https://www.3ds.com/. Accessed 1 March 2021
40. Microsoft. Microsoft Excel. https://www.microsoft.com/. Accessed 1 March 2021
41. EIF laboratory at KU. Training dataset for descriptor-based machining feature recognition. http://www.dhmun.net/home/Research_Data. Accessed 1 March 2021
42. Havasi, M., Hernández-Lobato, J. M., & Murillo-Fuentes, J. J. (2018). Inference in deep gaussian processes using stochastic gradient hamiltonian monte carlo.
43. Rácz, A., Bajusz, D. & Héberger, K. Effect of dataset size and train/test split ratios in QSAR/QSPR multiclass classification. *Molecules* **26**(4), 1111. https://doi.org/10.3390/molecules26041111 (2021).
44. Yadav, A., Jain, T., Verma, V. K., & Pal, V. (2021). Evaluation of machine learning algorithms for the detection of fake bank currency. in *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 810–815. IEEE.
45. Basnet, B., Chun, H. & Bang, J. An intelligent fault detection model for fault detection in photovoltaic systems. *J. Sens.* https://doi.org/10.1155/2020/6960328 (2020).
46. Ning, F., Shi, Y., Cai, M., Xu, W. & Zhang, X. Manufacturing cost estimation based on the machining process and deep-learning method. *J. Manuf. Syst.* **56**, 11–22. https://doi.org/10.1016/j.jmsy.2020.04.011 (2020).
47. Han, J., Pratt, M. & Regli, W. C. Manufacturing feature recognition from solid models: a status report. *IEEE Trans. Robot. Autom.* **16**(6), 782–796. https://doi.org/10.1109/70.897789 (2000).
48. Wang, Q. & Yu, X. Ontology based automatic feature recognition framework. *Comput. Ind.* **65**(7), 1041–1052. https://doi.org/10.1016/j.compind.2014.04.004 (2014).

Author contributions

C.Y.: methodology, software, writing—original draft preparation, writing—reviewing and editing. B.C.K.: writing—reviewing and editing. S.C.: software. Writing—original draft preparation. J.L.: software. D.M.: conceptualization, methodology, supervision, writing—original draft preparation, writing—reviewing and editing.

Funding

This research was supported by the Basic Science Research Program (Project ID: NRF-2019R1F1A1053542) through the National Research Foundation of Korea (NRF) funded by the Korea government (MSIT), by the AI-based gasoil plant O&M core technology development program (Project ID: 21ATOG-C161932-01) funded by the Korea government (MOLIT), by the Industrial Technology Innovation Program (Project ID: 20012462) funded by the Korean government (MOTIE), and by the Korea University Grant 2020.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to D.M.