Advanced Engineering Informatics 51 (2022) 101427

Contents lists available at ScienceDirect

Advanced Engineering Informatics

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

Full length article

3D CAD model retrieval based on sketch and unsupervised variational autoencoder

Feiwei Qin [a], Shi Qiu [b], Shuming Gao [b,*], Jing Bai [c]

[a] School of Computer Science and Technology, Hangzhou Dianzi University, China
[b] State Key Laboratory of CAD&CG, Zhejiang University, China
[c] School of Computer Science and Engineering, North Minzu University, China

ARTICLE INFO

Keywords:
Model retrieval
Structural semantics
Sketch
Unsupervised learning
Deep learning

ABSTRACT

How to quickly, accurately retrieve and effectively reuse 3D CAD models that conform to user's design intention has become an urgent problem in product design. However, there are several problems with the existing retrieval methods, like not being fast, or accurate, or hard to use. Hence it is difficult to meet the actual needs of the industry. In this paper, we propose a 3D CAD model retrieval approach that considers the speed, accuracy and ease of use at the same time, based on sketches and unsupervised learning. Firstly, the loop is used as the fundamental element of sketch/view, and automatic structural semantics capture algorithms are proposed to extract and construct attributed loop relation tree; Secondly, the recursive neural network based deep variational autoencoders is constructed and optimized to transform arbitrary shapes and sizes of loop relation tree into fixed length descriptor; Finally, based on the fixed length vector descriptor, the sketches and views of 3D CAD models are embedded into the same target feature space, and *k*-nearest neighbors algorithm is adopted to conduct fast CAD model matching on the feature space. In this manner, a prototype 3D CAD model retrieval system is developed. Experiments on the dataset containing about two thousand 3D CAD models validate the feasibility and effectiveness of the proposed approach.

1. Introduction

With the rapid development of the economy and the trend of globalization, the market competition among enterprises is becoming extremely fierce. In the new product development activities, if the designer can effectively reuse the existing knowledge and 3D CAD models, on this basis to modify and secondarily develop, that will greatly shorten the development cycle and reduce the development cost. This technique will help the enterprises in the cruel market competition get an invincible position [1–4]. It is a friendly and convenient way to retrieve the 3D CAD models that designers really want from the massive existing model databases by using a 2D sketch as the query input [5].

However, it is difficult to search and reuse 3D CAD models by using 2D sketches as the query input. First of all, a 2D engineering sketch and a 3D CAD model have different dimensions. Even if they both describe the same object, it is difficult to establish semantic association and mapping relationship between these two different media forms; secondly, 3D CAD models in real industrial applications have complex composition structure, arbitrary topological connection and variable shape features, and they are domain related, containing multi-granularity local details with engineering semantics [6]. Even CAD models belonging to the same category may have completely different overall shapes. It is not easy to use descriptors to conduct uniformly semantic descriptions and characterize the essential structure of 3D CAD models from multiple CAD systems in engineering [7]. In addition, two-dimensional engineering sketches drawn by different designers have different drawing styles, abstraction granularity and levels according to their personal habits and preferences. How to automatically extract key semantic features from personalized sketches is still a key issue to be solved.

This paper studies a new approach of intelligent reuse of 3D CAD model driven by sketch, aiming to implement a practical industrial application for product retrieval and reuse. By using a deep neural network, the internal structure correlation between a 2D engineering sketch and a 3D CAD model is established, which is embedded into the same feature space jointly. On this basis, the retrieval and reuse of CAD models are realized. The research carried out includes the following three points:

- Propose an automatic method to generate sketch and view's description which contains structural semantics. Firstly extract

* Corresponding author.
E-mail address: smgao@cad.zju.edu.cn (S. Gao).

https://doi.org/10.1016/j.aei.2021.101427
Received 24 March 2021; Received in revised form 28 August 2021; Accepted 22 September 2021
Available online 23 October 2021

loop structure data from the view of 3D CAD models or the sketch drawn by users. Then generate the corresponding relationship tree as an initial description, by combining these data orderly.

- Propose a sketch and view descriptor generation method based on recursive neural network (RvNN). According to the characteristics of structured view data, a RvNN based variational autoencoder is designed to automatically extract view descriptors, which can effectively compress view information meanwhile retain enough view information, so as to support rapid matching between sketch and 3D CAD models.
- A prototyping 3D CAD model retrieval system based on sketches and unsupervised autoencoder is developed, which verifies the effectiveness of the proposed method.

2. Related work

2.1. Sketch-based 3D model retrieval and reuse

In the aspect of sketch-based retrieval, many existing works generally regard the sketch as a set of two-dimensional pixels, extract the feature vector from the sketch by using image descriptor, measure the similarity between the sketch and the 3D model of the library, and then retrieve the related 3D model. This method needs to solve two major problems: one is to project the 3D model into the two-dimensional view with the best angle; the other is to select the image descriptor suitable for both the two-dimensional view and the hand-drawn sketch feature extraction.

Hou et al. [8,9] proposed a sketch-based 3D engineering part classification retrieval method. The user inputs the sketch by hand through the interactive interface. At the same time, the 3D parts in the library are projected into orthogonal three views, and then features are extracted from the sketch and three views by combining image descriptors such as spherical harmonics, Fourier transform and Zernike moment. Finally, the distance of feature vectors is calculated to complete the similarity matching between the sketch and the 3D parts. Wang et al. [10] proposed a sketch-based 3D model retrieval method using a convolutional neural network (CNN). The method trains Siamese CNN to select the best angle projection view for the three-dimensional model, with contour descriptors (e.g. closed boundaries, Suggestive Contours) being used to extract features from sketches and projected views. Eitz et al. [11–13] have done a lot of work on sketch-based retrieval for 3D models and 2D images, including selecting and improving image descriptors specifically for 2D sketches, such as Shape Context, Spark and SHoG, and using machine learning methods such as SVM to select the best angle to project 3D models and draw 2D sketches, etc. Liu et al. [14] considered the idea of a bag of words and proposed a 3D CAD model retrieval method that takes user's hand-drawn 2D sketches as input, and this method could carry out adaptive learning for each individual user. Zeng et al. [5] proposed a sketch-based retrieval and instantiation method for parametric parts. First, the feature vertices and adjacencies are extracted to form a view-dependent (VD) graph representation which is smaller and more efficient than a part's attribute face graphs. Then, automatic algorithms are designed to hierarchically search similar parametric parts from the database. In this work, a prototyping retrieval system with 50 parametric from the mechanical and furniture industries is implemented. In addition, Han et al. [15] developed a sketch-based 3D art-style face modeling system using deep learning technology.

Compared with the keyword-based or instance-based 3D model retrieval, the retrieval method based on a sketch is more convenient and practical, and conforms to the user's input habits. The existing work of 3D model retrieval based on sketch has achieved good results in general 3D model retrieval by selecting image descriptors suitable for sketch feature extraction, or improving the existing image descriptors, and using heuristic or machine learning methods to select the best angle projection of 3D model. However, the two-dimensional sketch input by engineers in the CAD field is different from the general hand-drawn sketch, which has its own constraints and specific semantics, and usually takes the loop as the basic semantic unit. The existing work of 3D model retrieval based on sketch has not been able to describe and express the engineering semantics of 2D sketch well, which should be considered in the retrieval and reuse of 3D CAD model.

2.2. Deep learning in CAD domain

Deep learning technology makes machine learning a big step toward its original goal of achieving true artificial intelligence. Since 2006, deep learning methods have made major breakthroughs [16], triggering a research boom in the field of machine learning and artificial intelligence. Deep learning technology allows machines to automatically learn multiple levels of abstraction and representation of objective objects to understand the essential meaning of sounds, text, images, videos, and three-dimensional models [17]. Deep learning technology has been successfully applied in many fields, and scholars have also introduced it into the field of CAD for related research. Currently, deep learning algorithms that have been widely used include convolutional neural networks (CNN) and its variants, recurrent neural networks (RNN) and its variants, deep belief networks (DBN) and recursive neural networks (RvNN), etc. [18].

CNN in CAD domain. Convolutional neural network has three structural characteristics: local connection, weight sharing, and spatial subsampling. It is good at processing array data (e.g. 1D vector, 2D matrix, 3D tensor, etc.) [19]. The five most widely used convolutional neural network architectures are AlexNet [20], ZFNet [21], VGGNet [22], GoogLeNet [23] and ResNet [24]. A 3D CAD model can be processed into a 3D voxel or projected into a 2D image to generate an input acceptable to CNN. Su et al. [25] designed a multi-view convolutional neural network for 3D shape recognition. The three-dimensional shape is first projected into multiple two-dimensional images, then multiple standard CNN structures are used to extract the features of the two-dimensional images from different angles independently, and finally, the image features from multiple angles are fused by the View Pooling layer to generate a compact three-dimensional shape description. On this basis, Hegde et al. [26] designed a FusionNet classifier, which combines voxelized representation and view representation of the 3D model as input at the same time to train CNN. Experimental results show that the fusion classifier has better classification performance than pixel representation or voxel representation alone. Qi et al. [27] also designed voxelized CNN and multi-view CNN with different structures, processed the 3D model into pixel and voxel representations with different resolutions as inputs, trained CNN, besides analyzed, compared the shape recognition and classification performance of various CNN structures.

RNN in CAD domain. Through the use of self-feedback neurons, RNN is good at processing sequential data (e.g. temporal data, character stream, etc.), and has been widely used in speech recognition, machine translation, natural language generation and other tasks [28]. RNN is not suitable for the analysis and processing of 3D CAD models directly, but it can be used for natural language processing (NLP) tasks such as content analysis of design documents and automatic knowledge extraction of patent documents in CAD field.

DBN in CAD domain. Wu et al. [29] designed a deep learning model 3D ShapeNets based on a deep belief network for the recognition, shape completion and reconstruction of 3D CAD models. The model takes a $30 \times 30 \times 30$ voxel grid as input to automatically learn and extract various features of 3D shapes, and then maps and transforms the features layer by layer. The filters of each hidden layer can learn some relevant features of this layer. For example, the first hidden layer learns some surfaces and corners of three dimensional shapes; the second and third hidden layers roughly learns multiple parts which make up the three dimensional shape, such as the table top and the table foot; the

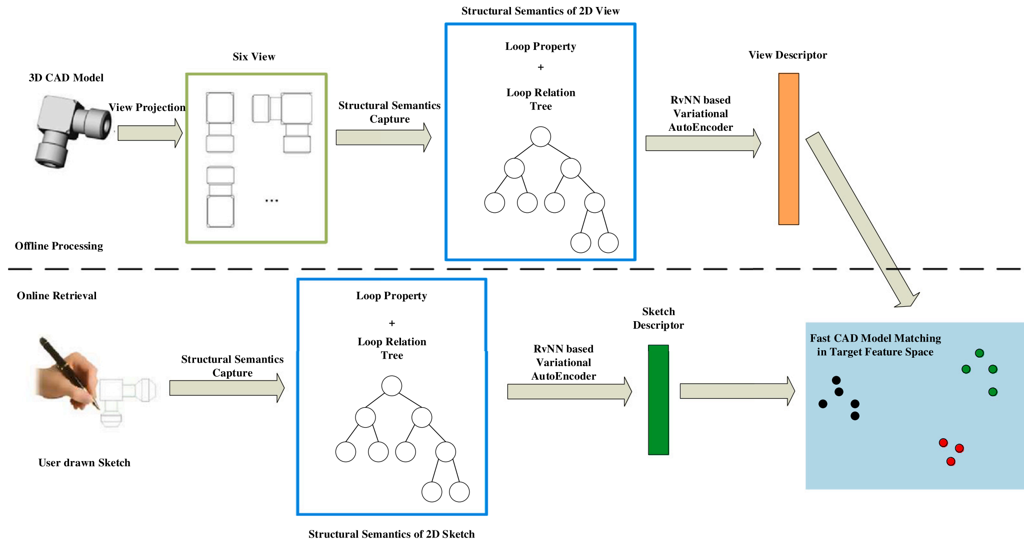


Fig. 1. Overview of the proposed approach.

fourth and fifth layers recognize the entire three dimensional shape. However, because 3D ShapeNets take 3D voxels as input, the dimension of the input vector is very large, which leads to difficulty in training the neural network model and poor generalization ability.

RvNN in CAD domain. Socher et al. [30] first proposed to use recursive neural networks to analyze natural scenes and natural languages in 2011. In this work, the text is seen as a collection of words, while images are seen as a collection of super-pixel blocks. RVNN is able to learn a parse tree and recursively merge text/image fragments. Xu et al. [31] extended RVNN and developed Generative Recursive Autoencoders for Shape Structures (GRASS) to automatically encode the semantic structure of 3D shapes. GRASS can implicitly capture the complex internal topological connection relations and symmetry relations (e.g. translational symmetry, rotational symmetry, etc.) of several kinds of three-dimensional shapes (e.g. chairs, candlesticks, vehicles, etc.), map the component layout of 3D shape to a dimensional-fixed feature vector. The feature vector can be used to retrieve, interpolate and generate the corresponding 3D shape. In essence, GRASS, based on the idea of RVNN, can recursively encode the tree/graph data structure with an arbitrary topological connection into a dimensional-fixed feature vector, which implicitly describes the internal topological connection relationship of the corresponding tree/graph data structure.

The two-dimensional sketch of engineering CAD model is composed of several sketch loops with different geometric shapes. The loop is the most basic semantic unit of engineering sketch. There are topological relationships between loops such as adjacency, disjoint, inclusion, and symmetry. We have been thinking about how to make vectorization representation (similar to pixel representation of 2D images and voxel representation of 3D shapes) of engineering sketches that contain any number of complex topological connection loops, so as to provide machine learning/deep learning algorithms for training and feature extraction. Inspired by the work of Xu et al. [31], the deep recursive neural network is used to encode and represent two-dimensional engineering sketches/views in this paper.

3. Overview of approach

Aiming at the deficiency of the existing 3D CAD model retrieval, this paper proposes a new sketch-based 3D CAD model retrieval approach combining with the characteristics of 3D CAD model, automatically extracting and using structural semantic information in sketches, and using an unsupervised autoencoder, to improve the accuracy and efficiency of 3D CAD model retrieval and reuse, as shown in Fig. 1. In reference to the behavior of engineers, this article assumes that the sketch input by the user is usually similar to one of the engineering drawings of the target 3D CAD model that the user wants to search in both topology and geometry. In general, the approach in this paper needs to solve the following three key problems.

- How to effectively and automatically extract structural semantic information from views/sketches. The 2D view projected by the 3D CAD model contains accurate geometric information and structural semantic information, which plays an important role in improving the accuracy of 3D CAD model retrieval. Since the structural semantic information is only implicit in the sketch and the view, how to extract the structural semantic information from the view/sketch completely and effectively based on the original geometric information is the first key problem to be solved.
- How to transform the extracted structural semantic information into an appropriate descriptor. On the premise that Problem 1 is solved, the structure semantic information in the view/sketch is very complex, and it is difficult to use it as a descriptor to support effective and efficient similarity calculation. Therefore, how to transform the structural semantic information into appropriate descriptors and make the descriptors preserve the structural semantic information in the original view to a large extent is the second key problem to be solved.
- How to improve the online retrieval efficiency of sketch-based 3D CAD model retrieval. Similarity comparison is one of the most important stages of 3D CAD model retrieval. Due to the dimension of the comparison data in the similarity comparison stage is uncertain, so the efficiency of online retrieval of these methods is not high. How to transform the variable-length comparison data in the similarity comparison stage into fixed-length comparison data and enable it to express the structural semantic information contained in the original views and sketches, to improve the efficiency of online retrieval is the third key problem to be solved.

CAD semantics encoding. We use a loop as the primary element to represent the structural semantics of the CAD models, and define an abstraction of loop trees, which are composed of spatial arrangements of axis align bounding boxes (AABBs). Each AABB is defined by a fixed length code to represent its geometry. The fixed length code encodes both the geometry of its child AABBs and their detailed topological grouping mechanism: adjacency, disjoint, inclusion, and symmetry (e.g. reflectional symmetry, rotational symmetry, translational symmetry).

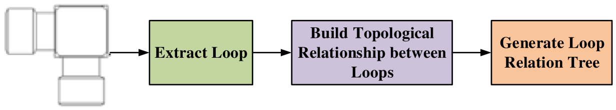


Fig. 2. The procedure for capturing structural semantics of CAD model sketch/view.

Stage1: Structural semantics capture. 3D CAD models are projected into six 2D views including *FrontView*, *RighView*, *TopView*, *BackView*, *LeftView*, *BottowView* in the offline processing module. According to the characteristics of the CAD domain, the loop is used as the fundamental element of both the projected views and user drawn sketches. Automatic semantics capture algorithms are developed to extract the loop properties and topological relationship between loops, and the attributed loop relation tree is generated from 2D views/sketches, which represents the structural semantics of 3D CAD models.

Stage2: Unsupervised autoencoder based CAD model descriptor generation. The RvNN based unsupervised autoencoders are constructed and optimized to encode the attributed loop relation tree of arbitrary shapes and sizes into the fixed length vector (80-D CAD model descriptor), which implicitly represents the structural semantics, geometric and topological information of the 3D CAD model. A 3D CAD model corresponds to multiple view descriptors, whereas a view descriptor corresponds to only the 3D CAD model that generated the view.

Stage3: Fast CAD model matching. Since both the sketches and the 2D views are represented by the fixed length vector descriptors, they would be embedded into the same target space. k-nearest neighbors (k-NN) based fast matching algorithm is used to search the most similar 3D CAD models from the dataset, inputting the user drawn sketch. The 12 CAD models with the top similarity values are returned by the online retrieval module.

4. Structural semantics capture

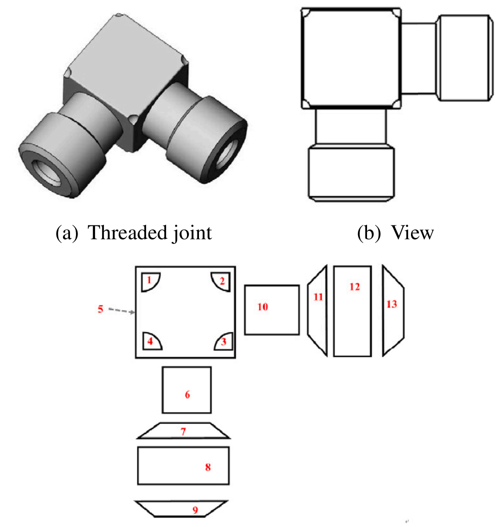
In order to effectively generate structural semantic data that can completely describe the sketch/view information, we propose a structural semantics capture algorithm. The difficulty of the structural semantics capture algorithm is how to extract the loop structure and the relationship tree of loops based on the original data of the sketch/view. Its main idea is as follows: based on the original geometric information such as lines and arcs in the 2D sketch/view, the undirected sparse graph composed of the lines and the intersections of the lines in the sketch/view is firstly constructed; then a single source search algorithm based on breadth-first traversal is used to identify the loop structure in the sketch/view; finally, apply the recursion-based algorithm to combine these loops orderly to generate the relationship tree of loops.

The algorithm includes the following steps, as shown in Fig. 2: (1) based on the definition of loop attributes, the algorithm extracts the complete loop structure from the sketch/view; (2) extract the topological relations between loops, based on the properties of loops; (3) extract the loop relation tree, with the help of the definition of the relationship between loops.

4.1. Loop

A fundamental loop is the smallest unit of meaning in a 2D sketch/view, analogous to the word component of a sentence in natural language processing. By analogy with the words in sentences, the data structure of loop property designed in this paper does not add artificial design features.

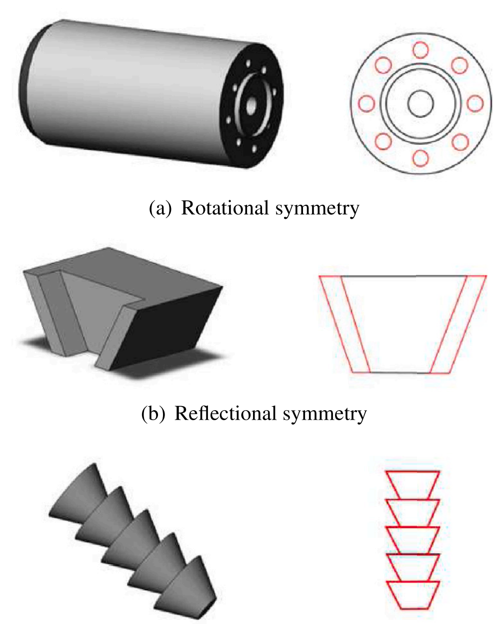
The bounding box is used to describe the loop attribute. The types of bounding boxes include spheres, AABB, Oriented Bounding Boxes (OBB), 8-DOP and convex shells. Because the geometric features of the 3D CAD model themselves are more obvious and neat, the geometric shapes of the 2D view obtained by its projection are neat and almost parallel to the coordinate axis. Hence we use AABB as the bounding box of the loop in this paper. The data structure of the loop property includes the location of the center point of the bounding box, the size of the bounding box along the axis and the type of the line taking the point as its end point in the loop. It is a 11D vector, the specific data structure is provided in Table 1. An example of extracting loops from a threaded joint model is illustrated in Fig. 3.



(a) Threaded joint (b) View

(c) Extracted loops

Fig. 3. Example of extracting loops from a threaded joint model.



(a) Rotational symmetry

(b) Reflectional symmetry

(c) Translational symmetry

Fig. 4. Examples of symmetry relation.

4.2. Topological relationship between loops

In order to make the structural semantics fully describe the view information, based on the definition of loop properties, we define the topological relationship between loops into four kinds: adjacency, disjoint, inclusion, and symmetry. There are three kinds of symmetry

Table 1
Data structure of the loop property.

Data location	Data type	Data meaning
1	Double-precision floating-point	The abscissa of the center point of the bounding box
2	Double-precision floating-point	The ordinate of the center of the bounding box
3	Double-precision floating-point	The size of the bounding box moves along the horizontal axis
4	Double-precision floating-point	The size of the bounding box moves along the longitudinal axis
5	Double-precision floating-point	The abscissa of the starting point at the lower left corner of the bounding box
6	Double-precision floating-point	The ordinate of the starting point in the lower left corner of the bounding box
7	Integer	The number of lines in the bounding box
8	Integer	The number of arcs in the bounding box
9	Integer	The number of ellipses in the bounding box
10	Integer	The number of parabolas in the bounding box
11	Integer	The number of splines in bounding box

Table 2
Data structure of the symmetry relation.

Data location	Rotational symmetry	Reflectional symmetry	Translational symmetry
1	Use 0 to identify the category	Use 1 to identify the category	Use 2 to identify the category
2	The abscissa of the rotational center point	The abscissa of the vector of symmetry axis	The abscissa of the displacement vector
3	The ordinate of the rotational center point	The ordinate of the vector of symmetry axis	The ordinate of the displacement vector
4	The number of symmetric loops	The abscissa of the starting point of symmetry axis	The abscissa of the center point of the last loop
5		The ordinate of the starting point of symmetry axis	The ordinate of the center point of the last loop

relations of loops: rotational symmetry, reflectional symmetry, and translational symmetry, as shown in Fig. 4. The specific data structure of symmetry relation is provided in Table 2.

Since the loop is described by the bounding box in this paper, the collision between objects can be quickly detected by the bounding box. In this section, based on the information of the bounding box, the geometric information of the line and the arc in the loop, the automatic extraction algorithm of the topological relationship between loops is designed. The algorithm pseudocode is shown in Algorithm 1.

Algorithm 1 The automatic extraction algorithm of the topological relationship between loops

Input: Loop list, loop center list and geometry information of each loop
Output: Topological relationship of loops

initial *relationshiplist*
for $i = 1 \rightarrow N$ **do**
  for $j = 1 \rightarrow i-1$ **do**
    if $Loop_i$ area = $Loop_j$ area **then**
      if $Loop_i$ edge list equals $Loop_j$ edge list by order **then**
        The relationship between $Loop_i$ and $Loop_j$ is $Symmetrical$
      end if
    else
      if $center_i$ in $Loop_j$ **then**
        if $Loop_i$ and $Loop_j$ have common edge **then**
          The *relationship* between $Loop_i$ and $Loop_j$ is $Adjacent$
        else
          The *relationship* between $Loop_i$ and $Loop_j$ is $Inclusion$
        end if
      else
        if $Loop_i$ and $Loop_j$ have common edge **then**
          The *relationship* between $Loop_i$ and $Loop_j$ is $Adjacent$
        else
          The *relationship* between $Loop_i$ and $Loop_j$ is $Disjoint$
        end if
      end if
    end if
  end for
end for
return *relationshiplist*

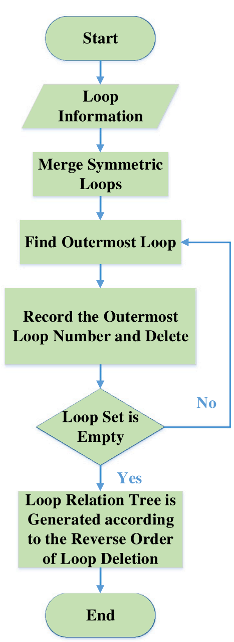


Fig. 5. Procedures for generating loop relation tree.

4.3. Loop relation tree

Based on the extracted loops and topological relationship between loops from sketch/view, in this section, the binding order of loops is processed accordingly to generate the required loop relation tree. The procedures for generating the loop relation tree are as follows, as illustrated in Fig. 5.

Step 1: Input basic information about loops and between loops. The relation between loops is obtained through geometric operation of loop bounding box, and the unprocessed loop set (hereinafter referred to as loop set) is recorded.

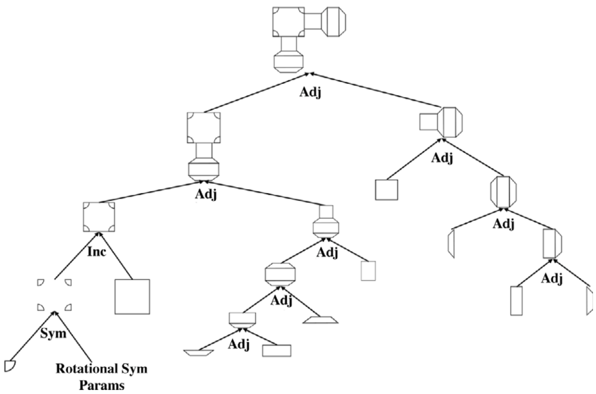


Fig. 6. The loop relation tree of threaded joint model.

Step 2: Merges loops with symmetric relations. According to statistics, the loops with symmetric relations in the two-dimensional projection view of the 3D CAD model are all fundamental loops. For the sake of simplifying the processing steps, the loops with symmetric relations are merged first, and then deleted from the loop set.

Step 3: Look for the outermost loops. Here, outermost loop refers to the loop that is not contained by any loop and includes at least one outermost boundary edge of the sketch/view in x or y direction. Taking the view in Fig. 3 as an example, the loop 9 and loop 13 are two of its outermost loops at the first. If there are more than one outermost loops, step 4 needs to be performed for each one.

Step 4: Record the outermost loop number and delete it. Remove the outermost loop from the loop set in step 3 and record its ordinal number.

Step 5: Determine whether the loop set is empty. If the loop set is empty, then perform step 6; if the loop set is not empty, then perform step 3.

Step 6: Generate the loop relation tree from bottom to top according to the loop order determined above.

A loop relation tree which is obtained from the threaded joint model of Fig. 3(b), by the loop relation tree generation procedures, is shown in Fig. 6.

5. Unsupervised autoencoder based CAD model sketch/view descriptor generation

In this section, we propose a method to encode CAD model sketch/view structures into a short, fixed-dimensional code. The learned encoding is fully invertible, allowing the sketch/view structures to be reconstructed from the code (CAD model sketch/ view descriptor). The main idea of this method is: use a recursive neural network (RvNN) based variational autoEncode to process the captured structural semantic information, and generate fixed length descriptor; Design RvNN based variational autoDecode to make the descriptor retain the original structural semantic information significantly; compute the loss between the captured structural semantic information and the restored semantic information, train the RvNN by back propagation through structure (BPTS) algorithm [32].

The proposed method has two procedures: (1) input the captured structural semantic information of CAD model sketch/view into the variational autoEncode to produce mean vector and standard deviation vector, then generate fixed length sketch/view descriptor. (2) restore the descriptor to sketch/view structural semantic information through the variational autoDecode, and retain the original structural semantic information as more as possible. The procedures for generating the CAD model sketch/view descriptor are shown in Fig. 7.

5.1. Variational autoencoder construction

The variational autoencoder is an artificial neural network which learns effective data code without supervision. It learns the latent representation using a variational approach which impels the encoder more effective meanwhile generates latent variables with abundant information. The variational autoencoder includes two modules: *Encode* and *Decode*. *Encode* firstly carries out dimension reduction for the input data, obtains mean and standard deviation of Gaussian distribution. Then the latent variables which are CAD model descriptors are generated by re-parameterization. *Decode* restores the latent variables and makes them as close to the original data input as possible.

5.1.1. Encode

For processing the relationship between loops such as adjacency, disjoint, inclusion and symmetry, the corresponding *Encodes* are designed in the RvNN.

The forms of these encoders are roughly similar, except that the choice of weight matrixes and bias vectors differs depending on the loop relationships. The encoder module is combined by two neural network layers which merge codes for two adjacent loops into the code for their parent node.

Adjacency, disjoint and inclusion. It has two n-D inputs and one n-D output. Its trainable parameters are a weight matrix $W \in R^{n\times 2n}$ and a bias vector $b \in R^n$, which are used to obtain the code of parent (merged) node y from children node codes x_1 and x_2 using the formula

$$y = \tanh(W \cdot [x_1; x_2] + b) \tag{1}$$

Symmetry. The symmetry encoder module differs from others in the way it converts the encoding of a symmetric child node and its symmetric mode encoding into the encoding of the parent node. The encoder for the symmetry module merges the n-D code for a generator loop of a symmetry group, as well as the m-D parameters of the symmetry itself into a single n-D output. The code for a group with generator loop x and symmetry relationship parameters p_{sym} is computed as

$$y = \tanh(W_{syme} \cdot [x; p_{sym}] + b_{syme}) \tag{2}$$

where the weight matrix $W_{syme} \in R^{n\times(n+m)}$ and the bias vector $b_{syme} \in R^n$.

In this paper, the four encode modules are two layer neural networks, the size of the hidden layer is 120, and the size of the output layer is 80, that is $n = 80$. For symmetry relationship, we set the size of symmetry relationship parameters as 5, that is $m = 5$. Furthermore, the input for RvNN is the AABB set of sketch/view. On the contrary, the aforementioned four encode modules only deal with n-D vectors. Hence a single layer *BoxEncode* module is designed for converting 11-D loop structural data to n-D vector. The encode modules based on RvNN are illustrated in Fig. 8, ellipsis dots in the figure could be the output of *BoxEncode*, *AdjEncode*, *DisEncode*, *IncEncode*, or *SymEncode* modules.

5.1.2. Decode

For containing sketch/view structural semantics in the code vector, the n-D vector produced by encode module needs to be decoded to restore the structural semantics contained in sketch/ view. Corresponding to the encode modules, four kinds of decode modules aiming for different topological relationships between loops are designed. Similar to the encoder, the structure of the decoder is also roughly the same. There is only a difference in the selection of the weight matrix and the bias. Besides, the symmetry decoder module is slightly different from the other decoder modules in that its output is the encoding of one child node and its symmetry mode encoding, while the output of the other encoding modules is the encoding of two child nodes.

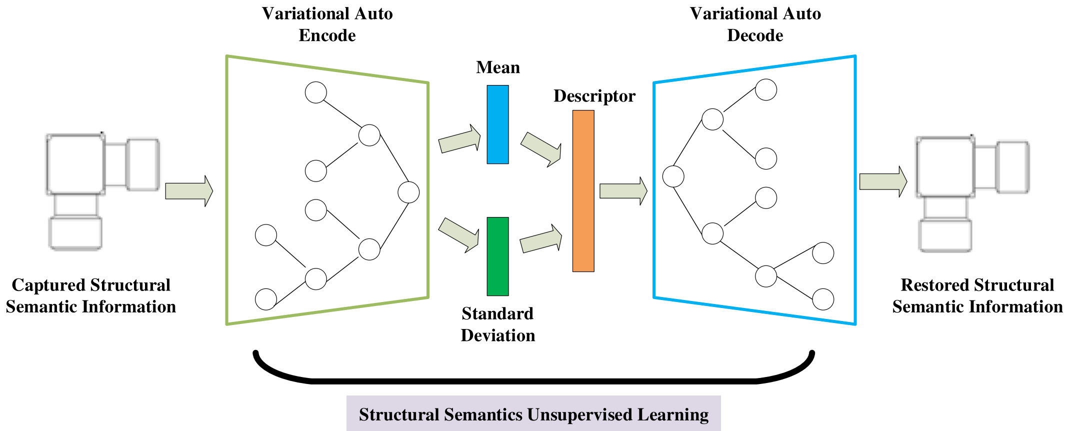


Fig. 7. Procedures of generating the CAD model sketch/view descriptor.

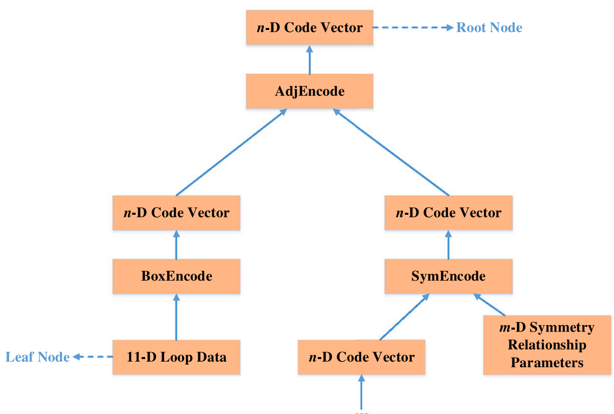


Fig. 8. The RvNN based encode modules.

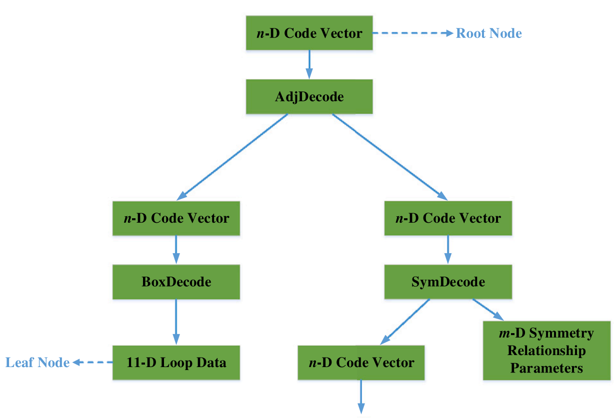


Fig. 9. The RvNN based Decode modules.

Adjacency, disjoint and inclusion. The corresponding decoder splits a parent code vector y back to chid code vectors x'_1 and x'_2, using the reverse mapping

$$[x'_1 x'_2] = \tanh(W \cdot y + b) \tag{3}$$

where the weight matrix $W \in R^{2n \times n}$ and the bias $b \in R^{2n}$.

Symmetry. The corresponding symmetry decoder recovers the generator loop semantic information and symmetry relationship parameters as

$$[x' p'_{sym}] = \tanh(W_{symd} \cdot y + b_{symd}) \tag{4}$$

where the weight matrix $W_{symd} \in R^{(n+m) \times n}$ and the bias $b_{symd} \in R^{n+m}$. The decode modules based on RvNN are illustrated in Fig. 9, "..." in the figure could be the output of *BoxDecode*, *AdjDecode*, *DisDecode*, *IncDecode*, or *SymDecode* modules.

5.2. Training

In essence, the training of a neural network is a process of solving the optimization problem of non-convex functions. And the loss function of the network is no longer a convex function of the parameter set. Therefore, it is necessary to adopt a combination of various training strategies in the training process of a neural network. Using the data augmentation method can increase the training set effectively so that the network model can achieve better generalization effect; adopting triangular learning rate policy so that the neural network can approach the global optimal solution in the feasible solution space more quickly; a training protocol is employed in order to reduce the memory overhead caused by the large amount of data and get the global optimal solution more quickly; an early stopping method is also used to prevent neural network overfitting.

5.2.1. Data augmentation

Data augmentation is a way of increasing the volume of data and improving model generalization without changing data categories. In this method, operations such as translation, rotation, noise, and horizontal flipping are applied to the original data set to form more and richer data sets. More data can make the network model achieve better experimental results on the test set. Different from the data augmentation method applied to images, the data augmentation method used in this paper is as follows:

- **Rotate.** The data extracted from the two-dimensional sketch/ view is processed. The point coordinates are randomly rotated according to a certain angle, and the angle is uniformly sampled in the interval $[-20°, +20°]$.
- **Scale.** The data extracted from the two-dimensional sketch/ view is processed, and the point coordinates are randomly expanded and shrunk in a certain proportion. The scaling factors are uniformly sampled in the interval [0.6, 1.0] to obtain.
- **Translate.** The data extracted from the two-dimensional sketch/ view is processed, and the point coordinates are randomly translated according to a certain distance. The horizontal translation distance and the width of the original sketch/view have a scaling factor $R1$, and the vertical translation distance and the height of the original sketch/view have a scaling factor $R2$. $R1$ and $R2$ are uniformly sampled in the interval [0, 0.4].

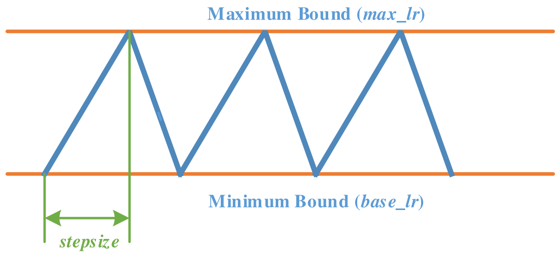


Fig. 10. The triangular learning rate policy.

Table 3
The hyper-parameter setting of learning rate.

base_lr	max_lr	stepsize	start	max_iter
0.01	0.06	100	0	1,000
0.001	0.006	50	1,000	1,700
0.0001	0.0006	30	1,700	2,000

5.2.2. *Learning rate*

This paper uses cyclical learning rates (CLRs) [33]. This method eliminates the need to find the best value through experiments and could adjust the global learning rate according to the design in advance. The method does not reduce the learning rate monotonously, but makes the learning rate change in a cycle between reasonable boundary values. The adopted triangular learning rate policy in this paper is shown in Fig. 10. Through the method mentioned in [33], the reasonable boundary of the learning rate is estimated, and the learning rate is adjusted linearly in the periodic training. The specific learning rate setting is shown in Table 3. $base\_lr$ is the initial learning rate, max_lr is the maximum learning rate, $stepsize$ is the stepsize of the linear change of learning rate, $start$ is the starting point of the cycle, and max_iter is the maximum iteration number.

5.2.3. *Training protocol*

Because neural network is a differentiable system, the gradient descent method can be used to optimize the parameters of a neural network. Different from traditional neural networks, deep neural network (DNN) and convolutional neural network (CNN) adopt back propagation (BP) algorithm to calculate the gradient; recurrent neural network (RNN) usually uses back propagation through time (BPTT) to calculate the gradient. And the variational autoencoder based on RvNN employed in this paper uses BPTS [32] algorithm to calculate the gradient.

The training protocol adopted in this paper is Mini-Batch, which is a compromise between full batch training protocol and random training protocol. A hyper parameter $batch\_size$ needs to be artificially set. The operation mechanism of the Mini-Batch training protocol is similar to that of the random gradient descent training, but the difference is that the Mini-Batch adopts multiple training samples to estimate the gradient value, which reduces the abnormal parameter fluctuations caused by a few abnormal samples. When the training data is large and the memory consumption is large, the method can still be carried out effectively under the condition of memory limitation because the data is trained in mini-batches. Typically, the batch size is 2 to the NTH power. In this paper, after several experiments were conducted, the batch size was set to 64.

5.2.4. *Early stopping*

In this paper, the training samples are divided into a training set (neural network weight is obtained through gradient descent algorithm) and a validation set (experimental results are verified with trained neural networks). The early stop method can be used to prevent neural networks from overfitting on the training set. It achieves its goal by monitoring the experimental effect of the model on the validation set. In the early stop method, the training process of the model is periodically paused (a training cycle contains multiple training sessions). And after each training cycle, the experimental effect of the model on the validation set will be tested. The specific steps for periodic estimation follow by validation are as follows:

(1) After one cycle of training, it is set to be 20 rounds in this paper. At this time, the parameter weights in the RvNN have been updated. Then, the experimental effectiveness of the current model is tested in the validation set.
(2) After the model is tested on the validation set, the training of the RvNN will start a new training cycle, and this process will continue to cycle until the experimental effect is optimal.

5.2.5. *Energy function*

During the training process, the RvNN in autoencoder is trained with the help of the loss values between the original input data and the reconstructed generated data, and the gradient calculation is performed by BPTS. The final energy function is composed of reconstruction loss and categorical cross entropy loss, and the formula is as follows.

$$Loss_{final} = \beta Loss_{recon} + (1-\beta)Loss_{crossen} \tag{5}$$

The reconstruction loss is defined as $Loss_{recon} = \|x - x'\|$, which is used to ensure that the error between the original extracted structural semantic information x and the reconstructed structural semantic information x' is continuously reduced after training. The cross entropy loss with certain weight is added to further ensure the quality of training, which is $Loss_{crossen} = -\frac{1}{2}x\log x'$. In this paper, β is set to 0.6.

6. Fast CAD model matching

For model matching, because the linear search in the target space is excessively time-consuming and reduces the user experience, we adopt k-nearest neighbors (k-NN) algorithm to improve the efficiency of online retrieval. In this paper, the k-NN algorithm is used in different ways. The first k categories of most similar descriptors are counted, and the corresponding 3D CAD models are displayed according to the frequency of view descriptor categories.

Algorithm idea of optimized k-NN. The calculation of k-NN is highly time-consuming when the training set is large. To improve the efficiency of k-NN search, special structures can be considered to store the training data in order to reduce the number of computed distances. Therefore, kd tree is used to optimize the algorithmic complexity of k-NN.

The search process for the nearest neighbor. Given a test sample, according to the construction rule of kd tree, the coordinate of the left child node is smaller than the parent node (the coordinate of the slicing point) and the coordinate of the right child node is larger than the coordinate of the parent node, then search for its nearest neighbor. The method first finds the leaf node that contains the test sample (a subregion that cannot be divided again); then starting from the leaf node, back to the parent in turn; constantly search for the node closest to the test sample, and end the algorithm when it is determined that no closer node exists.

Searching using kd tree can only be calculated on partial data set rather than the whole data, so this paper uses kNN based on kd tree algorithm to retrieve the target space to improve 3D CAD model retrieval efficiency.

7. Experimental results

We have developed a prototype system to support the proposed sketch and unsupervised learning based 3D CAD model retrieval approach. VBA (Visual Basic for Applications) is used for acquiring 2D views from 3D CAD models, C++ is used for developing automatic algorithms which capture structural semantics information from

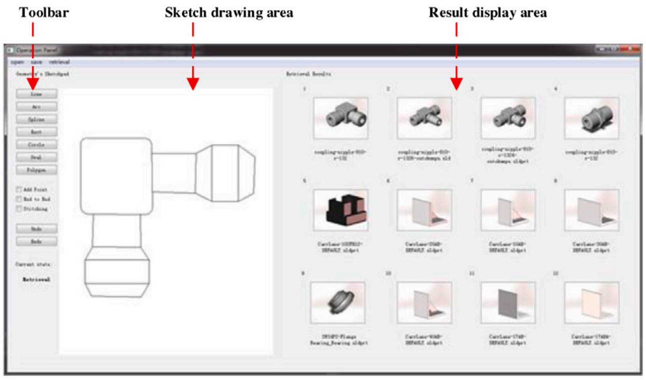


Fig. 11. UI of the proposed retrieval system.

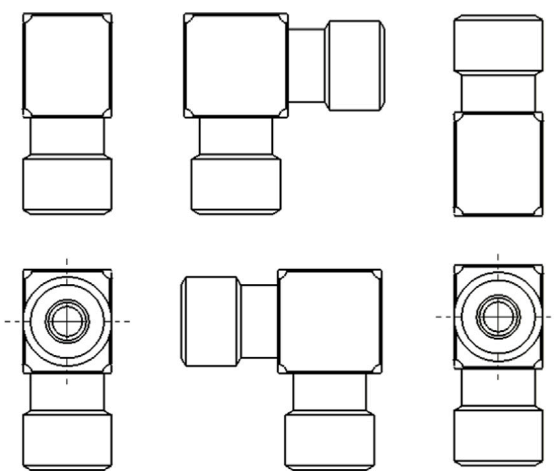


Fig. 12. Six views of a threaded joint part model.

views/sketches, moreover deep learning framework PyTorch is used for constructing the RvNN based autoencoders. The user interface (UI) of the prototype system is developed with QT. The PC used for the experiments is a desktop with 3.00 GHz Intel core i5 processor, 16GB memory, a Nvidia GeForce GTX 680 graphics card and is running on Windows 7.

As shown in Fig. 11, the UI of the proposed prototype retrieval system mainly consists of three parts: (i) toolbar, which provides some common drawing and auxiliary functions, and engineers can choose different types of brushes to draw sketches; (ii) sketch drawing area. The engineer could select the appropriate brush type to draw the sketches in the sketch drawing area; and (iii) result display area, which returns the top 12 CAD models in the 3D CAD model library that have the highest similarity to the input sketch. The thumbnail images and names of the returned models are displayed in order of similarity ranking from high to low.

3D CAD models containing design features are very valuable. They are one of the most important core assets of an enterprise. Due to the protection of intellectual property rights, it is difficult to obtain 3D CAD models from related enterprises. However, in order to conduct the experiments this paper, we have acquired more than 2000 3D CAD models from several local mechanical companies and constructed the corresponding datasets for the experiments. These 3D CAD models are created by experienced engineers with commercial CAD modeling systems (e.g. SolidWorks, CATIA). Six views including *FrontView*, *RightView*, *TopView*, *BackView*, *LeftView*, and *BottowView* are projected and generated from each 3D CAD model of the dataset. Fig. 12 shows the six views of a threaded joint part. In the future, we will further obtain 3D CAD models designed by multiple CAD systems from manufacturing companies to build a dataset with tens of thousands samples for further experiments.

7.1. Case study

To illustrate the advantages of our sketch and unsupervised variational autoencoder based retrieval approach, we compared our approach with sketch and heuristic rule based retrieval approach [34], and CNN-VAE approach of which the descriptors are generated directly by CNN, contrary to our unsupervised learning manner. CNN-VAE is a method we put forward in this paper to perform a comparison test in the experimental stage. CNN-VAE treats sketch as a 2D image composed of pixels, uses CNN to conduct feature extraction, and then feeds the extracted feature to VAE for processing. To be fair, the dimension of descriptors in CNN-VAE approach is the same as in this paper, which is 80.

Several query samples are tested. The retrieval results of approach in this paper are shown in Fig. 13 ; A part of the samples we will show in the next two methods to compare with the proposed approach. The retrieval results of sketch and heuristic rule based retrieval approach are shown in Fig. 14; And the retrieval results of CNN-VAE approach are shown in Fig. 15.

Case 1. The user draws *Sketch1* in the interactive UI. The precision/recall curves of the retrieval results of inputting *Sketch1* are shown in Fig. 16(a). It can be seen that the retrieval method in this paper is slightly better than the heuristic method on the whole and significantly better than CNN-VAE method. In fact, for *Sketch1*, the model expected to be retrieved is the model ranked 8th in the first row of Fig. 13, and *Sketch1* corresponds to its top view. The heuristic method and CNN-VAE method fail to retrieve the target model.

Case 2. The user draws *Sketch2* in the interactive UI. The precision/recall curves of the retrieval results of inputting *Sketch2* are shown in Fig. 16(b). It can be seen that the retrieval method in this paper is better than the heuristic method and the CNN-VAE method on the whole. For *Sketch2,* the model expected to be retrieved is the model ranked 1th in the second row of Fig. 13, and *Sketch2* corresponds to its top view. The right views of the models ranked 3rd and 4th in the second row of Fig. 13 have similar structural semantics with *Sketch2*. Compared with the retrieval results of the heuristic method shown in the second row of Fig. 14, the models ranked 9th and 10th are similar with *Sketch2*. But other models clearly do not match. And the retrieval results of CNN-VAE method shown in the second row of Fig. 15 are obviously not similar to the target model.

Case 3. The user draws *Sketch3* in the interactive UI. The precision/recall curves of the retrieval results of inputting *Sketch3* are shown in Fig. 16(c). It can be seen that the overall effect of the retrieval method in this paper is similar to that of the heuristic method, but obviously better than the CNN-VAE method. For *Sketch3*, the model expected to be retrieved is the model ranked 1st in the third row of Fig. 13, and *Sketch3* corresponds to its front view. Both the target model retrieved by the method in this paper and the target model retrieved by the heuristic method rank first. However, the models ranked from 2nd to 5th in the third row of Fig. 14 are similar to *Sketch3*. For the CNN-VAE method, all the retrieval results are obviously not similar to the target model.

Case 4. The user draws *Sketch4* in the interactive UI. The precision/recall curves of the retrieval results of inputting *Sketch4* are shown in Fig. 16(d). It can be seen that the retrieval method in this paper is better than the heuristic method and the CNN-VAE method on the whole. For *Sketch4*, the model expected to be retrieved is the model ranked 1st in the fourth row of Fig. 13, and also the model ranked 3 in the fourth row of Fig. 14. *Sketch4* corresponds to its front view. In the retrieval results of our method and the heuristic method, the front views of the models ranked from 1st to 4th are all similar to *Sketch4*. However, the ranking of the target model retrieved by the method in

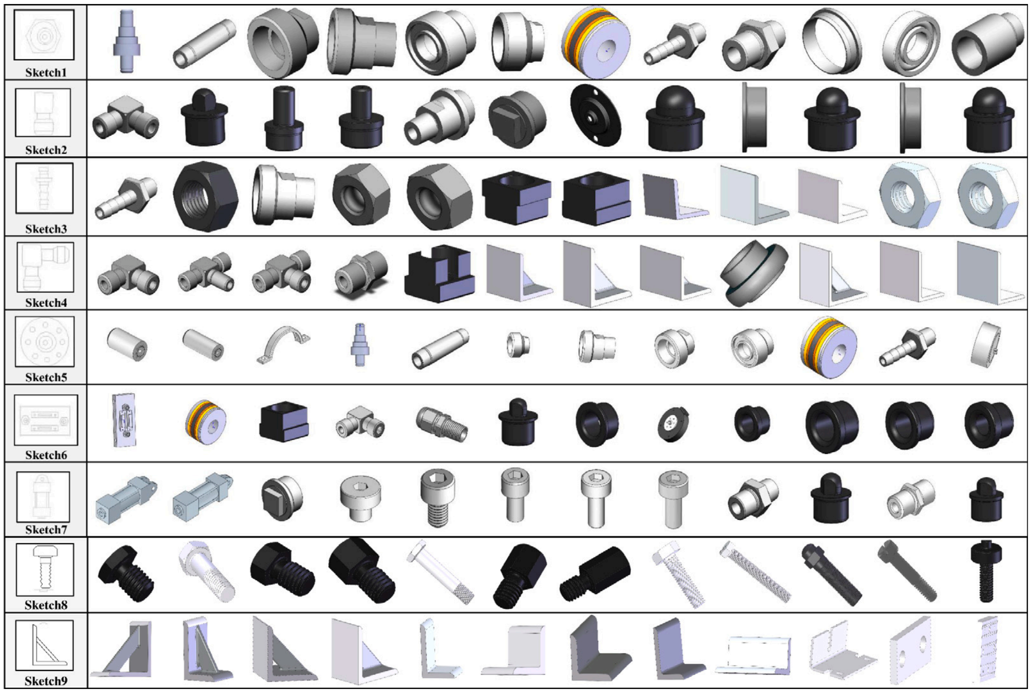


Fig. 13. The retrieval results of our approach. The sketches in the left column are input queries. Twelve models with the highest similarities are listed from the 2nd column to the 13th column in descending order.

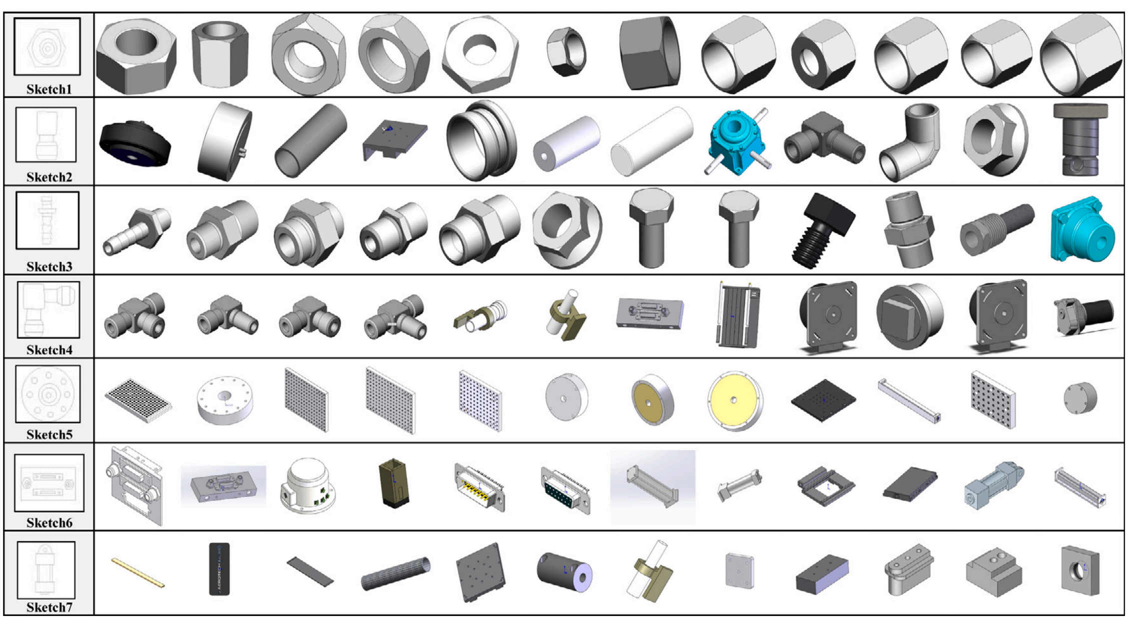


Fig. 14. The retrieval results of sketch and heuristic rule based approach. The sketches in the left column are input queries. twelve models with the highest similarities are listed from the 2nd column to the 13th column in descending order.

this paper is higher than that of the heuristic method. In the retrieval results of CNN-VAE method, there was no model similar to the target model.

Case 5. The user draws *Sketch5* in the interactive UI. The precision/recall curves of the retrieval results of inputting *Sketch5* are shown in Fig. 16(e). It can be seen that the retrieval method in this paper is better than the heuristic method and CNN-VAE method on the whole. For *Sketch5*, the model expected to be retrieved is the models ranked 1st and 2nd in the fifth row of Fig. 13, and *Sketch5* is their right view. The symmetry features in *Sketch5* is rotational symmetry, and the model feature in the retrieval results of the method in this paper is rotational symmetry. However, the features of the models retrieved by the heuristic method are mostly translational symmetry, and the models with rotational symmetry feature rank low. In the retrieval results of CNN-VAE method, the number of models with rotational symmetry is small, and they are not similar to the target model.

Case 6. The user draws *Sketch6* in the interactive UI. The precision/recall curves of the retrieval results of inputting *Sketch6* are shown in Fig. 16(f). It can be seen that the retrieval method in this paper is not as good as the heuristic method on the whole, but better than CNN-VAE method. For *Sketch6*, the model expected to be retrieved is the model ranked 1st in the sixth row of Fig. 13, and *Sketch6* is the front view of the model. In the retrieval results of the heuristic method, the target model ranked 2nd, and one of the views of the models ranked

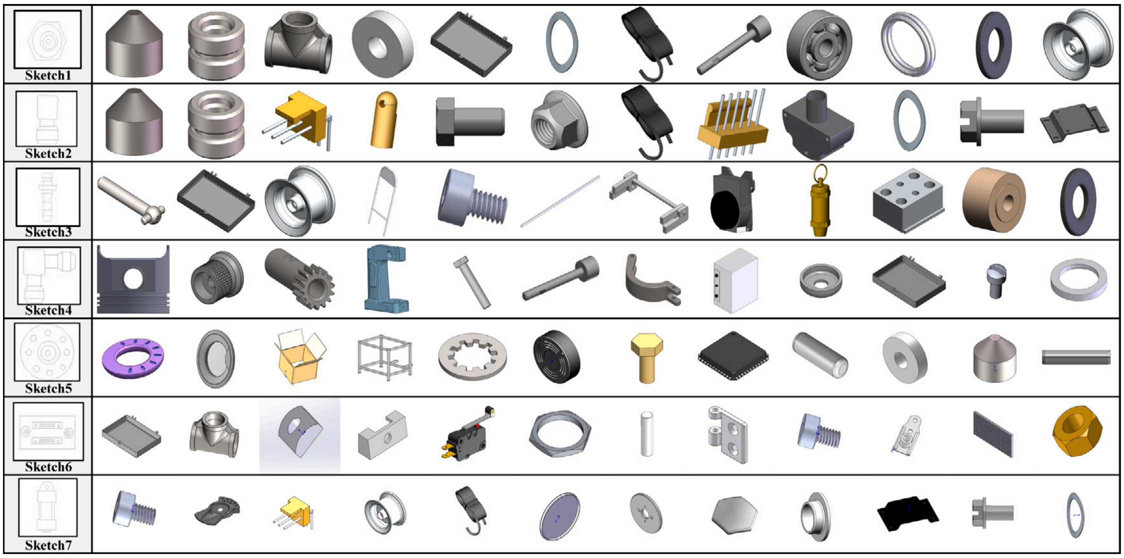


Fig. 15. The retrieval results of CNN-VAE approach. The sketches in the left column are input queries. twelve models with the highest similarities are listed from the 2nd column to the 13th column in descending order.

1st, 5th and 6th is similar to *Sketch6*. The results of CNN-VAE method are obviously not similar to the target model.

Case 7. The user draws *Sketch7* in the interactive UI. The precision/recall curves of the retrieval results of inputting *Sketch7* are shown in Fig. 16(g). It can be seen that the retrieval method in this paper is better than the heuristic method and CNN-VAE method on the whole. For *Sketch7*, the model expected to be retrieved is the models ranked 1st and 2nd in the seventh row of Fig. 13, and *Sketch7* is the right view of the model. However, the retrieval results of the heuristic method and CNN-VAE method are obviously not similar to the target model.

7.2. Comparative analysis

In the first four case studies, the sketch retrieval target of *Case 1* and *Case 3* is the same model, and the sketch retrieval target of *Case 2* and *Case 4* is the same model. *Case 1* and *Case 3*, *Case 2* and *Case 4* depict 3D CAD models with input sketches of different complexity from different perspectives. Although the sketch input in *Case 1* is simpler than that in *Case 3*, and the sketch input in *Case 2* is simpler than that in *Case 4*, the retrieval results of *Case 1* and *Case 2* contain the target model, and the ranking of the proposed method is relatively high. From the comparative analysis of *Case 1* and *Case 2*, and *Case 3* and *Case 4*, it can be seen that when the target model can be described with sketches of different complexity, the method in this paper can handle both relatively simple and complex sketch inputs.

From the point of view of engineers' retrieval of the target model, the expected 3D CAD model has views from multiple perspectives, and the complexity of the views is different, that is, the difficulty of drawing the sketch is different. The method proposed in this paper can query the target model by inputting relatively simple sketches instead of complex sketches, shorten the drawing time of input sketches and improve the efficiency of product design.

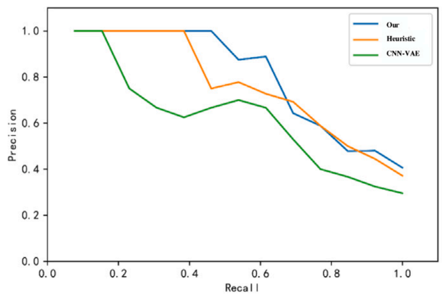
In *Case 3* and *Case 6*, the method presented in this paper is slightly inferior to the heuristic method. Because our approach currently just supports normal retrieval mode where model matching is based on the models' complete information including detailed information, the retrieval effect of our approach is sometimes not as good as that of the heuristic method if the input sketch lacks some details. Sketch 3 and Sketch 6 in Fig. 13 are two examples. In the future, we will improve the retrieval effect of our approach for such situation by supporting coarse-grained retrieval mode where model matching is only based on the models' coarse-grained information.

In summary, the retrieval effect of this method is significantly better than that of CNN-VAE method in the seven case studies. The retrieval effect of this method in *Cases 1*, *2*, *4*, *5* and *7* is significantly better than that of the heuristic method, but the retrieval effect in *Cases 3* and *6* is slightly inferior to that of the heuristic method. In addition, under the same experimental environment, the average retrieval time of the proposed method on the data set is about 52 ms, which is similar to CNN-VAE method and nearly 2 times faster than that of the heuristic. The advantages and disadvantages of this method are as follows:

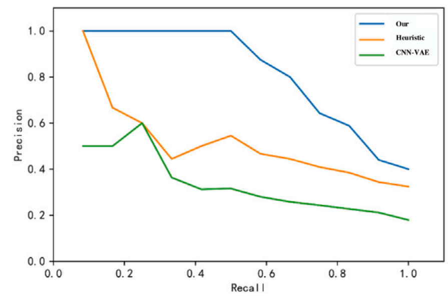
- The method of retrieving CAD models based on structure semantics is more effective than CNN-VAE method. The experiment proves that it is unreasonable to extract descriptors from CAD model view and sketch by using CNN from the perspective of image. The proposed method combines the structural semantics of two-dimensional view and sketches to achieve a high improvement in retrieval effect.
- When the target CAD model has different complexity of sketch input, we can obtain a better retrieval effect than the heuristic method and CNN-VAE method by input the sketch with lower complexity.
- The method in this paper improves the retrieval efficiency while ensuring the accuracy of model retrieval.
- The method in this paper is sensitive to the shape of the loop relation tree. When the input sketch is too complex, the retrieval effect of the method in this paper will be reduced because the engineer will lose some details when drawing.

7.3. Efficiency analysis of the k-NN based on kd-tree

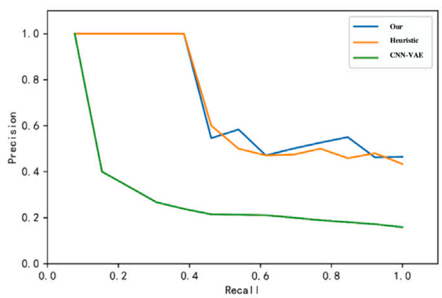
In our approach, to improve the efficiency of searching the similar 3D CAD models from the library by matching the feature vector of the input sketch with the feature vectors of 3D CAD models in the library, we use the k-NN based on kd-tree to efficiently find out 12 most similar 3D CAD models according to the input sketch. And thus, we construct a kd-tree for the feature vectors of 2000 3D CAD models in the library while setting up the 3D CAD model library. In our current experiments, it is about 100 times faster using k-NN based on kd-tree than using k-NN without help of kd-tree. In fact, the larger the 3D CAD model library, the greater the efficiency improvement (see Table 4).



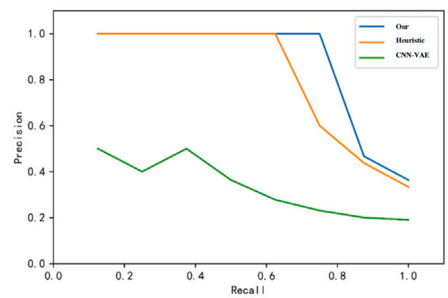
(a) *Case 1*



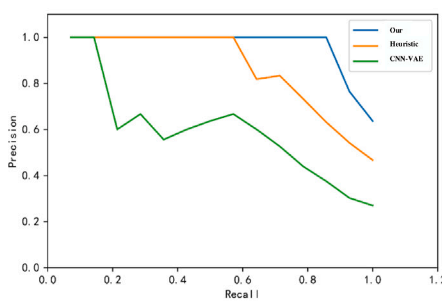
(b) *Case 2*



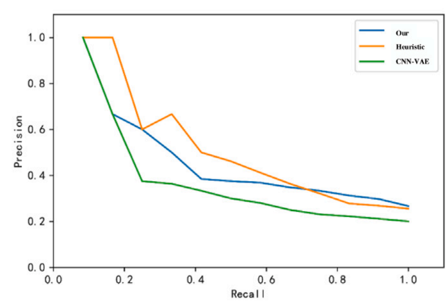
(c) *Case 3*



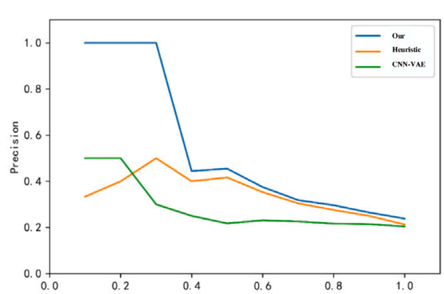
(d) *Case 4*



(e) *Case 5*



(f) *Case 6*



(g) *Case 7*

Fig. 16. Precision/Recall curves of the retrieval results by query cases. Blue: our approach; Red: sketch and heuristic rule based approach; Green: CNN-VAE approach.

8. Conclusion and future work

Aiming at the existing problems of the 3D CAD model retrieval approach based on sketches, this paper proposes a 3D CAD model retrieval approach based on sketches and unsupervised learning. This approach uses structural semantics and variational autoencoder to improve the accuracy and the efficiency of 3D CAD model retrieval: (i) The structural semantics based sketch/view description and automatic semantics capture algorithms are proposed; (ii) The unsupervised autoencoder based sketch/view descriptor generation algorithm is proposed, which transforms arbitrary shape and sizes of attributed loop relation trees to fixed length vector representation; (iii) A prototype system for 3D CAD model retrieval is developed. A model database consisting of nearly 2000 3D CAD models is constructed, and *k*-NN algorithm is used to improve the efficiency of online retrieval.

In the future, we will extend our 3D CAD model library and conduct more experiments so as to sufficiently verify the proposed approach. In addition, sketches used at present are more accurate, and the replacement of complex sketches with brief strokes will become the development trend of future product design. Brief strokes pay more attention to the topological relationship between drawing graphs, but

Table 4
Comparison of k-NN optimized by kd tree and linear search k-NN.

Retrieval strategy	Retrieval time (s)
Optimized k-NN	0.2520
Linear search k-NN	1.7362

weakens the geometric elements of drawing graphs. Therefore, it is necessary to further optimize the neural network based on the structural semantics, and to improve the network structure appropriately to meet the development trend of sketch input simplification. Currently our retrieval system just support one retrieval mode, that is the normal retrieval mode. In the future work, we will develop more different kinds of retrieval modes including coarse-grained retrieval mode and fine-grained retrieval mode so that the different abstraction granularity and levels can be handled more effectively.

Declaration of competing interest

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

Acknowledgments

This work was supported by the National Key R&D Program of China (No. 2020YFB1709200), National Natural Science Foundation of China (No. 61972121) and Zhejiang Provincial Natural Science Foundation of China (No. LY21F020015). The authors would like to thank the reviewers for their comments and suggestions in advance.

References

[1] B.C. Kim, Y. Jeon, S. Park, H. Teijgeler, D. Leal, D. Mun, Toward standardized exchange of plant 3D CAD models using iso 15926, Comput. Aided Des. 83 (2017) 80–95.
[2] D. Zhang, F. He, Z. Tu, L. Zou, Y. Chen, Pointwise geometric and semantic learning network on 3D point clouds, Integr. Comput.-Aided Eng. 27 (1) (2020) 57–75.
[3] A. Llewelyn, Review of CAD/cam, Comput. Aided Des. 21 (5) (1989) 297–302.
[4] T.G. Gunn, The mechanization of design and manufacturing, Sci. Am. 247 (3) (1982) 114–131.
[5] L. Zeng, Z.-k. Dong, J.-y. Yu, J. Hong, H.-y. Wang, Sketch-based retrieval and instantiation of parametric parts, Comput. Aided Des. 113 (2019) 82–95.
[6] Y. Cheng, F. He, X. Lv, W. Cai, On the role of generating textual description for design intent communication in feature-based 3D collaborative design, Adv. Eng. Inf. 39 (2019) 331–346.
[7] Y. Wu, F. He, D. Zhang, X. Li, Service-oriented feature-based data exchange for cloud-based design and manufacturing, IEEE Trans. Serv. Comput. 11 (2) (2015) 341–353.
[8] S. Hou, K. Ramani, Classifier combination for sketch-based 3D part retrieval, Comput. Graph. 31 (4) (2007) 598–609.
[9] S. Hou, K. Ramani, Sketch-based 3D engineering part class browsing and retrieval, in: SBM, 2006, pp. 131–138.
[10] F. Wang, L. Kang, Y. Li, Sketch-based 3D shape retrieval using convolutional neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, 1875–1883..
[11] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, M. Alexa, Sketch-based shape retrieval, ACM Trans. Graph. 31 (4) (2012) 1–10.
[12] M. Eitz, J. Hays, M. Alexa, How do humans sketch objects? ACM Trans. Graph. 31 (4) (2012) 1–10.
[13] M. Eitz, K. Hildebrand, T. Boubekeur, M. Alexa, Sketch-based image retrieval: Benchmark and bag-of-features descriptors, IEEE Trans. Vis. Comput. Graphics 17 (11) (2010) 1624–1636.
[14] Y.-J. Liu, X. Luo, A. Joneja, C.-X. Ma, X.-L. Fu, D. Song, User-adaptive sketch-based 3-D CAD model retrieval, IEEE Trans. Autom. Sci. Eng. 10 (3) (2013) 783–795.
[15] X. Han, C. Gao, Y. Yu, Deepsketch2face: a deep learning based sketching system for 3D face and caricature modeling, ACM Trans. Graph. 36 (4) (2017) 1–12.
[16] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[17] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Netw. 61 (2015) 85–117.
[18] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[19] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, et al., Recent advances in convolutional neural networks, Pattern Recognit. 77 (2018) 354–377.
[20] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90.
[21] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European Conference on Computer Vision, Springer, 2014, pp. 818–833.
[22] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, arXiv preprint arXiv:1409.1556.
[23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[24] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[25] H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller, Multi-view convolutional neural networks for 3D shape recognition, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 945–953.
[26] V. Hegde, R. Zadeh, Fusionnet: 3D object classification using multiple data representations, 2016, arXiv preprint arXiv:1607.05695.
[27] C.R. Qi, H. Su, M. Nieß ner, A. Dai, M. Yan, L.J. Guibas, Volumetric and multi-view cnns for object classification on 3D data, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5648–5656.
[28] H. Wang, B. Raj, On the origin of deep learning, 2017, arXiv preprint arXiv:1702.07800.
[29] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3D shapenets: A deep representation for volumetric shapes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1912–1920.
[30] R. Socher, C.C.-Y. Lin, A.Y. Ng, C.D. Manning, Parsing natural scenes and natural language with recursive neural networks, in: ICML, 2011.
[31] J. Li, K. Xu, S. Chaudhuri, E. Yumer, H. Zhang, L. Guibas, Grass: Generative recursive autoencoders for shape structures, ACM Trans. Graph. 36 (4) (2017) 1–14.
[32] C. Goller, A. Kuchler, Learning task-dependent distributed representations by backpropagation through structure, in: Proceedings of International Conference on Neural Networks (ICNN'96), vol. 1, IEEE, 1996, pp. 347–352.
[33] L.N. Smith, Cyclical learning rates for training neural networks, in: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2017, pp. 464–472.
[34] F.-w. Qin, S.-m. Gao, X.-l. Yang, J. Bai, Q.-h. Zhao, A sketch-based semantic retrieval approach for 3D CAD models, Appl. Math.-A J. Chinese Univ. 32 (1) (2017) 27–52.