



# 'CADSketchNet' - An Annotated Sketch dataset for 3D CAD Model Retrieval with Deep Neural Networks

## ARTICLE INFO

### Article history:

Received July 3, 2021

**Keywords:** Retrieval, Search, Dataset, Deep Learning, CAD, Sketch

## ABSTRACT

Ongoing advancements in the fields of 3D modelling and digital archiving have led to an outburst in the amount of data stored digitally. Consequently, several retrieval systems have been developed depending on the type of data stored in these databases. However, unlike text data or images, performing a search for 3D models is non-trivial. Among 3D models, retrieving 3D Engineering/CAD models or mechanical components is even more challenging due to the presence of holes, volumetric features, presence of sharp edges etc., which make CAD a domain unto itself. The research work presented in this paper aims at developing a dataset suitable for building a retrieval system for 3D CAD models based on deep learning. 3D CAD models from the available CAD databases are collected, and a dataset of computer-generated sketch data, termed 'CADSketchNet', has been prepared. Additionally, hand-drawn sketches of the components are also added to CADSketchNet. Using the sketch images from this dataset, the paper also aims at evaluating the performance of various retrieval system or a search engine for 3D CAD models that accepts a sketch image as the input query. Many experimental models are constructed and tested on CADSketchNet. These experiments, along with the model architecture, choice of similarity metrics are reported along with the search results.

© 2021 Elsevier B.V. All rights reserved.

## 1. Introduction

The search or retrieval of Engineering (CAD) models is crucial for a task such as design reuse [1]. Designers spend a significant time searching for the right information and using a large percentage of existing design for a new product development [2]. [3] indicates that a large percentage (75% or greater) of design reuses existing knowledge for the new product development. This calls for the search and classification of 3D Engineering models [4]. With the wide applicability of 3D data and the increased capabilities of modelling, digital archiving, and visualization tools, the problem of searching or retrieving CAD models becomes a predominant one.

The research work presented in this paper aims at developing a well-annotated sketch dataset of 3D Engineering CAD models, that can aid in the development of deep learning-based search engines for 3D CAD models, using a sketch image as the input query. Using a sketch-based query for the search offers many

advantages. 3D shapes, unlike text documents, are not easily retrieved using textual annotations ([5]) since it is difficult to characterize what human beings see and perceive using a mere text annotation. [6, 7] show that content-based 3D shape retrieval methods (those that use the visual/shape properties of the 3D models) are shown to be more effective. It is also shown in [6, 7] that using traditional search methods for multimedia data will not yield the desired results. [8] utilizes the idea of using the feature descriptors/vectors of the 3D model for the search query. Among the available query options, a sketch-based query is shown to be very intuitive and convenient for the user ([9, 10, 11]), since it is easier for the user to learn and use such a system over using a 3D model itself as the query since it requires technical expertise and skill [12].

The Princeton Shape Benchmark (PSB) [13] was one of the earlier 3D shape databases. Consequently, large-scale datasets such as ShapeNet [14] came into being. Due to such data avail-

18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34

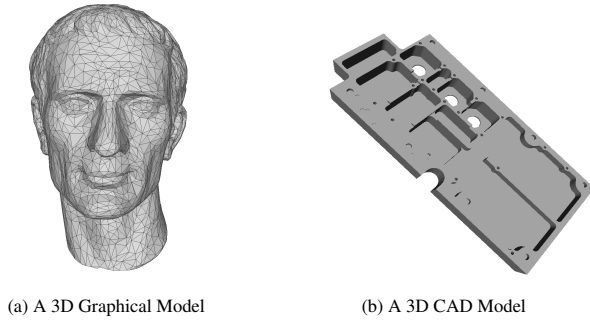


Fig. 1: Distinction between a Graphical Model and a CAD Model

ability, many machine learning-based techniques, which require a good amount of data to train the models, have been developed ([15, 16]). [17] provided the first benchmark dataset of sketches based on the 3D models in PSB. As a result of this, [18] and [19] have introduced large-scale benchmarks for sketch data for 3D shapes, including many approaches for sketch-based retrieval.

However, these datasets and methodologies only aim at generic 3D shape data or graphical models and do not contain Engineering CAD models. The presence of features such as holes (genus > 0), blind holes (genus = 0) and other machining features in an Engineering/CAD model (see Fig. 1b) calls for special treatment as opposed to a 3D graphical model where such features and holes are usually absent (see Fig. 1a). Also, sharp edges are usually found in a CAD Model unlike a graphical model that has smooth curvature throughout.

In the case of Engineering models, the people involved are required to have rich domain knowledge and experience. CAD models are typically obtained via the design process, unlike 3D shape data which are acquired via 3D scans. Since most design data are proprietary, they are not available to the public domain (e.g. [20]). Moreover, among the few datasets that are available for usage, they either lack proper annotations [21] or contain too few models ([22, 23, 24]), which are not very conducive for employing deep learning methods. Although the work presented in [25] performs a sketch-based retrieval of 3D CAD models, the dataset is proprietary. Engineering shape benchmark (ESB) [26] is a prominent dataset for CAD models having 801 models. More recently, datasets such as mechanical components benchmark (MCB) [27] and CADNET [28] have been proposed. Nonetheless, the data is in the form of 3D models, and no sketch information is present. As far as the sketch-data for Engineering models is concerned, there are no datasets available to the best of our knowledge.

Our motivation for addressing the problem of retrieval of Engineering / CAD models, therefore, comes from the following:

1. Most CAD datasets are typically contain only a few models and hence cannot be utilized by the latest technological advances such as deep neural networks.
2. Datasets having larger number of CAD models are either proprietary (not publicly available) [20] or lack classification information [21].
3. There is no existing database for sketch-data of 3D CAD models since obtaining hand-drawn sketches for CAD models is difficult.

4. The recent advances in deep learning have not been made use of, to the best of our knowledge.

The key contributions of the paper are:

1. Using the available 3D CAD models from [26] and [27], a large-scale dataset of computer-generated sketch data called ‘CADSketchNet’ is created.
2. Additional hand-drawn sketches of CAD models are also incorporated into the dataset using the models available from [26].
3. To benchmark the developed dataset, we analyzed the performance of various learning-based approaches for the sketch-based retrieval of 3D CAD models.
4. The performance of the various experimental retrieval systems on CADSketchNet are compared, and the search results are reported.

This paper makes useful contribution to the research community involving ‘mechanical components’ and allow researchers to develop new algorithms for the same. The paper is organized as follows: Section 2 discusses the related works corresponding to 3D CAD models, in addition to the literature on images and generic 3D Shapes. The dataset preparation is explained in Section 3 including the challenges involved in creating hand-drawn sketch data for 3D CAD models, the need for computer-generated sketches and the process of generating such sketches. The experiments done in order to benchmark the developed ‘CADSketchNet’ dataset are detailed in Section 4. Section 5 provides the Implementation Details. The results, limitations and possible future work are elaborated in Section 6, followed by a Conclusion (Section 7).

## 2. Related Works

Many works in recent times have focused on 3D graphical models and images. We focus more on the approaches that have been proposed for the search and classification of 3D mechanical components, which are very few. However, a few of the recent approaches used in other domains are also mentioned for the sake of completeness.

### 2.1. Images

Deep learning became ubiquitous for image-related tasks since the application of Convolutional Neural Networks (CNN) to the ImageNet Challenge [29]. For the sketch-based retrieval of image data, sketch datasets such as [30] and [31] have been introduced. The work in [32] extracts edge maps of the matching and non-matching images and uses these maps for training. [33], [34] popularize the zero-shot learning framework. [35] attempts an on-the-fly sketch-based image retrieval using reinforcement learning methodology.

### 2.2. 3D models of common objects

[14] introduced the ShapeNet dataset and also performed classification and retrieval tasks on it. Many learning-based approaches have been proposed for the tasks of classification and retrieval using the ModelNet dataset. The leader-board can

be found at [36], with the popular methods being [37, 38, 39]. However, none of them uses a sketch query. A few tracks of the Shape Retrieval Contest (SHREC) involved a sketch-based retrieval challenge for 3D shapes. SHREC'12 [40] involved a sketch-based retrieval task and introduced a benchmark dataset. Building upon the dataset by [17], SHREC'13 [18] included a Large Scale Sketch-Based 3D Shape Retrieval track with a larger dataset. [12] compares different methods for sketch-based 3D shape retrieval. SHREC'14 [19] expanded upon this further to an Extended Large Scale Sketch-Based 3D Shape Retrieval track. SHREC'17 [41] involved a sketch-based retrieval task of 3D indoor scenes.

### 2.3. 3D CAD models of Engineering shapes

The SHREC track [42] presents a retrieval challenge using the Engineering Shape Benchmark (ESB) dataset [26]. [43] uses the idea from content-based image retrieval to the domain of 3D CAD models. [44] performs a visual similarity-based retrieval using 2D engineering drawings. This method converts 2D drawings to a shape histogram and then applies the idea of spherical harmonics to obtain a rotation-invariant shape descriptor. Minkowski distance was used to measure the similarity between feature representations.

[45] uses the data from ShapeLab [46] and ESB to develop a sketch-based 3D part retrieval system. This paper uses the idea of classifier combinations to aid in the retrieval process. Engineering models are classified functionally and not visually, as opposed to the image or 3D graphical data. Taking this into account, the extracted shape descriptors (Zernike Moments and Fourier Transforms) are sent to a Support Vector Machine (SVM) classifier. A weighted combination of the classifier outputs is then used to estimate the class or category of the input query, which is then compared against the classes of the database. This is one of the earlier works that use learning-based methods for building a sketch-based retrieval system for CAD models. However, the sketch-data itself is not available.

More recently, a sketch-based semantic retrieval of 3D CAD models is presented by [25]. The CAD models and their parametric features (used in 3D modelling software) are taken. The pre-processed sketch query is first vectorized and then passed onto topology-based rules. An integrated similarity measurement strategy is used to compute the similarity between the query sketch and CAD models' database. This research work uses a dataset of 2148 CAD models and six corresponding views of each model. This dataset is proprietary and is not available. Also, the method presented here uses the classical rule-based approach over the latest advances in learning-based approaches.

### 2.4. Other related works

[47] presents a detailed study on the state-of-the-art methods and the future of sketch-based modelling and interaction. However, the discussion mainly focuses on sketch interpretation and on the development and usage of interactive sketches. There is very limited discussion on 3D engineering shape data.

[48] introduces the OpenSketch Dataset which contains annotated sketches corresponding to various product designs. A detailed study is done in order to understand the stroke time and

pressure. A taxonomy of lines is also provided and the strokes are labelled. However, the dataset contains only 107 sketches across 12 categories, which is not sufficient for developing learning based models.

The SPARE3D dataset [49] aims at understanding the spatial reasoning behind line drawings using deep neural networks. While the dataset uses 10,369 3D CAD models, it only contains line drawings of 3D objects from 8 different isometric views and does not contain any hand-drawn sketches. ProSketch3D [50] consists 1500 sketches of 3D models across 500 object categories taken from ShapeNet. All sketches corresponding to generic 3D shapes and not 3D mechanical components.

## 3. Dataset Creation

As discussed in Section 2, a database of 3D models and their corresponding sketches are not available for the domain of CAD models. [51] introduces a sketch-dataset for CAD models. However, it is not useful for a traditional search problem since the sketches are based on the design workflow, i.e. based on 'how' a model is designed, rather than the model shape and geometry. [25] uses only a proprietary dataset. Therefore, we attempt to build a new sketch-dataset, termed 'CADSketchNet', using the 3D models from existing CAD databases.

### 3.1. Challenges in creating a dataset of hand-drawn sketches

Creating a hand-drawn sketch for a 3D mechanical component is much more challenging than sketching a generic 3D shape. This is because,

- It is difficult to capture the detailed information present in a CAD model, such as the presence of holes and volumetric features in a single sketch.
- Multiple viewing directions can be chosen to draw the sketch.
- The sketches need to be drawn by users with domain knowledge and experience. Gathering a set of users to contribute to building such a dataset is both time-consuming and expensive.
- Once the hand-drawn sketches are obtained, they need to be verified and validated for correctness and closeness to the input CAD model.
- Different users have different drawing styles, and consistency needs to be maintained across the hand-drawn sketches.

Due to these reasons, attempting to create a dataset of hand-drawn sketches for a large number of 3D CAD models is a tedious task.

### 3.2. Hand-drawn sketch data generation

The ESB dataset [26] is a publicly available CAD database that is also well annotated. In the ESB, there are 801 3D CAD models across 42 classes (excluding the models in the 'Misc' category). Since, the ESB is a reasonably sized dataset, we attempt to obtain hand-drawn sketches for all 801 3D CAD models of the ESB.

### 3.2.1. Gathering users for obtaining hand-drawn sketches

Around 50 users with experience of CAD and engineering drawing were selected in order to provide a hand-drawn sketch for each 3D CAD model in the ESB. The users are mostly engineering students pursuing undergraduate and postgraduate courses in design, with a few industry professionals. Because the goal is to create a dataset that can be used to develop learning-based solutions, users were asked to draw the sketches in their natural style so that the dataset might capture more variability. Furthermore, users were encouraged not to be overly precise and correct, because the learning algorithms that could use this dataset can capture input variations while staying robust to noise.

### 3.2.2. Obtaining the hand-drawn sketches

Each user was then shown the 3D object (digital) and is asked to draw a digital sketch on a hand-held tablet device. Since a 3D object can be viewed from many directions, the best viewing direction for the sketch (i.e. that which covers the entire geometry of the object) is determined by the user, based upon the domain knowledge and experience. The cases of potential ambiguity with respect to the viewing direction are resolved by a majority vote among the users.

As we only attempt to draw a single sketch for each 3D CAD model in the ESB dataset, the number of sketches obtained and the category information acquired are the same as that of the ESB. The reader can refer to the paper on ESB [26] for more information on the category information and the number of models in each class.

### 3.2.3. Processing the hand-drawn sketches

Initial cleaning of these hand-drawn sketches was done by sketch pre-processing using the software in Autodesk Sketchbook. Consequently, the sketches were then validated for their correctness and closeness (resemblance to the 3D object) by domain experts from academia and the industry. These images and the class labels (the same label as the 3D model from ESB) are stored as a database. Hence, the number of sketches is only of the order of few hundreds. Nevertheless, this dataset is stored and will be made available since it is challenging to obtain a real dataset of hand-drawn sketches. This dataset could also be used by algorithms that do not need large-scale training data.

### 3.2.4. Analyzing the hand-drawn sketches

During the course of obtaining user drawn sketches, it was observed that simple object classes like Clips, Bolt-Like Parts, Nuts and Discs were very easy to draw, and took very little time to obtain. Users encountered significant difficulty when they were required to sketch object categories with complex features, such as 90-degree elbows, Motor Bodies, Rectangular Housing, and so on. For most other object categories, the users were able to draw the sketches with a manageable level of difficulty.

There is also a need to assess the quality of the sketches in terms of the correctness of each user's choice of viewing direction. The concept of Light Field Descriptor (LFD) is proposed by [52], in which a 3D model is placed inside a regular dodecahedron and images are obtained by using each vertex as a viewing

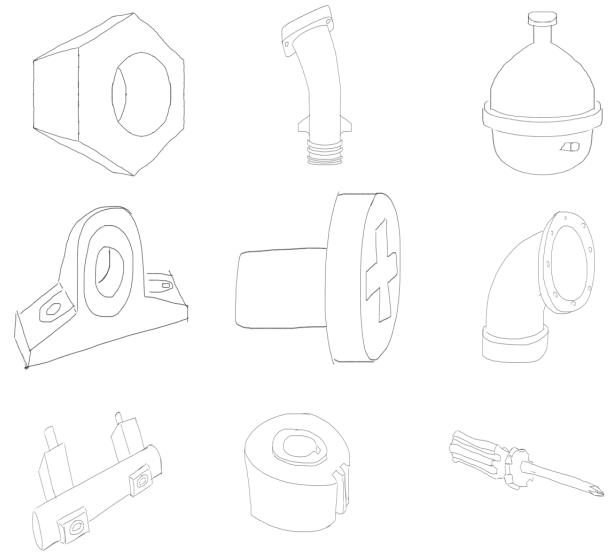


Fig. 2: Sample Hand-drawn sketches from the developed 'CADSketchNet'.

direction. Using this idea, 20 view images are obtained for each 3D object.

The obtained sketch is compared with each of the 20 view images. The view image with the highest obtained similarity score is noted, and the viewing direction is cross-checked with the user's choice of viewing direction. In majority of the cases, the gathered users were able to accurately determine the ideal viewing direction for a model. In the few circumstances where this was not the case, the user was requested to redraw the sketch. The quality of the produced hand-drawn sketch dataset is thus ensured. Figure 2 shows a few sample sketches.

## 3.3. Creating Computer-generated Sketch Data

The Mechanical Components Benchmark (MCB) [27], contains 58,696 3D CAD models across 68 classes. Hence, for our study, we utilize the 3D CAD models from the MCB to prepare a dataset of sketch images. Deep learning methodologies are data-driven, and it calls for a large amount of data. Since it is tough to obtain hand-drawn sketches for such a large-sized dataset of complex 3D mechanical components, we attempt to create computer-generated sketch images corresponding to each CAD model in the MCB.

### 3.3.1. Converting 3D model to a 2D image

A representative 2D image of every 3D model in the MCB needs to be obtained as a first step. A Python script is used to save the image of the 3D model from a particular viewing direction. Applying the idea of LFD, we obtain 20 images for every 3D object. Now it is needed to identify one representative image for each object. For this task, a group of 70 volunteers with knowledge of CAD were identified. The 3D CAD objects were split into batches. For each batch, the above procedure was applied, and the one among the resultant output images was chosen as the representative image for the 3D object. Repeating this process for every batch, one image corresponding to every 3D CAD model in the MCB was generated. The overall procedure is summarized in Algorithm 1.

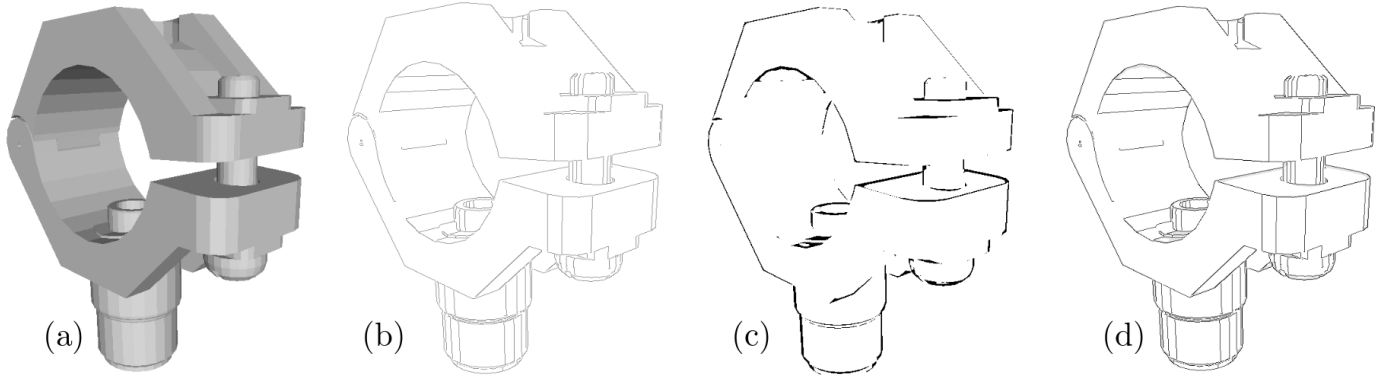


Fig. 3: (a) 2D image obtained from Algorithm 1 (b) Result of Canny edge detection (c) Result of Gaussian blurring (d) Result of weighted scheme of (b)&(c)

---

**Algorithm 1** Method to obtain a 2D image for a 3D model
 

---

**Input** Database of 3D CAD models

```

1: procedure 3DOBJ_TO_IMG
2:   Split the 3D models in the database into  $n$  batches
3:    $i = 1$ 
4:   while  $i \leq n$  do
5:     for every model in the batch do
6:       Apply LFD to obtain 20 images
7:       Users identify 1 among 20 images
  
```

**Output** 2D representative image for every 3D CAD model

---

ally, to enhance the output, we operate only in gray-scale rather than RGB. The procedure is summarized in Algorithm 2.

---

**Algorithm 2** Method to generate computer sketch from image
 

---

**Input** Images of 3D CAD models from Algorithm 1

```

1: procedure IMG_TO_SKETCH
2:    $I \leftarrow$  Read input image in RGB color space
3:    $G \leftarrow$  Convert image from RGB colorspace to Grayscale
4:    $IG \leftarrow$  Invert color values of all pixels in Grayscale
5:    $B \leftarrow$  Convolve non-uniform GF : kernel size  $k$  & SD  $\sigma$ 
6:    $IB \leftarrow$  Invert the blurred image
7:   Element-wise division of  $G$  &  $IB$  with  $scale = 256.0$ 
8:    $O1 \leftarrow$  Binary threshold the obtained image
9:    $O2 \leftarrow$  Canny Edge Detection of  $G$ 
10:   $S \leftarrow$  Weighted average over  $O1$  and  $O2$ 
  
```

**Output** Computer-generated sketches of 3D CAD models

---

### 3.3.2. Creating computer-generated sketch from the image

There is ample literature related to generating computer sketches. Edge detection of images is a fundamental approach to obtaining the object boundaries. Hence, we first begin our experiments to generate computer sketches for the 3D CAD models in the MCB dataset with the popular edge detection methods.

We first experiment with the Canny edge detection method [53], which essentially tries to identify the object boundaries present in the image using first derivative methods to identify local image features. The method is applied to the 2D images of the CAD models obtained from Algorithm 1. In order to enhance the obtained edges, we combine the Canny edge detection process with a second approach that uses the idea of image blurring, coupled with boundary shading. By using Gaussian blurring, the internal object features are suppressed. Thus, a weighted combination of the Canny edge detection and the idea of Gaussian blurring is proposed for generating the computer sketches.

The reason for using these two techniques is as follows. Since the original images are obtained from CAD mesh models, they contain several mesh lines. To generate sketches that can capture the overall shape and geometry of the object, we need to use methods capable of extracting the significant edges without paying too much attention to minute details of the object. Usage of the canny filter detects prominent edges effectively by thresholding, even in a noisy environment. The minute noisy details are easily removed due to the presence of a Gaussian Filter (GF), and the required signal (prominent pixels) can be enhanced using the Canny edge detector that uses non-maximum suppression against the noise, resulting in a well-defined output. Addition-

The weight assigned for the Gaussian blurring is significantly less in order to avoid too much suppression of minute details and thus leading to loss of vital information. Hence, the output of the weighted scheme closely resembles the output of the Canny edge detection scheme. The outputs of both these methods, along with the output of the weighted scheme, are shown in Figure 3 for a sample input.

In addition to the weighted Canny edge detection method mentioned above, other popular edge detectors such as the Scharr, Prewitt, Sobel and Robert Cross operators are also experimented with. A similar weighted scheme is applied to each of these methods. The other state-of-the-art sketch generation methods such as NeuralContours [54], PhotoSketch [55], and Context-aware tracing strategy (CATS) [56], are also experimented.

### 3.3.3. Comparing hand-drawn and computer-generated sketches

Creating hand-drawn sketches for a 3D CAD model is extremely difficult, as established in Section 3.1, and since such sketch data is not available for the MCB, we cannot directly compare the computer-generated and hand-drawn sketches. However, for the ESB dataset, we have obtained hand-drawn sketches (see Section 3.2). Therefore, computer sketches are generated for the 801 models in ESB, and these are compared with the corresponding hand-drawn sketches.

Sketch-generation method	PSNR $\uparrow$	MS-SSIM $\uparrow$	IE $\uparrow$	VIF $\uparrow$	MSE $\downarrow$	UQI $\uparrow$	Conversion time (Per image in sec) $\downarrow$
plain-canny	18.0834	0.5718	1.5248	0.0034	1010.9600	0.9874	<b>0.0021</b>
weighted-scharr	21.3913	0.6327	1.3649	0.0031	472.0136	0.9948	0.0581
weighted-prewitt	21.7143	0.6412	1.3904	0.0031	438.1824	0.9952	0.0547
weighted-roberts	20.4433	0.6169	1.3498	0.0031	587.1513	0.9935	0.0517
weighted-sobel	21.6066	0.6388	1.3555	0.0031	449.1815	0.9951	0.0498
neural-contours [54]	<b>25.4318</b>	<b>0.9319</b>	1.3425	<b>0.5292</b>	<b>186.1659</b>	<b>0.9977</b>	828.50
photosketch [55]	12.5434	0.4978	<b>3.7558</b>	0.0055	3620.2508	0.9367	9.0180
CATS [56]	16.2040	0.5428	1.5796	0.0035	1558.3710	0.9788	1.7810
<b>weighted-canny (ours)</b>	24.9429	0.8208	1.6737	0.0034	209.4152	<b>0.9977</b>	0.0190

Table 1: Similarity results obtained by using various approaches for comparing the hand-drawn and the computer-generated sketches for various sketch-generation methods on the 801 CAD models of the ESB dataset.  $\uparrow$  indicates that greater value for the metric indicates higher similarity, while  $\downarrow$  indicates the opposite. The plain-canny results reported here are with non-maximal suppression. The similarity measures used are PSNR - Peak signal-to-noise ratio; MS-SSIM - Multi Scale Structural Similarity Index [57]; IE - Information Entropy; VIF - Visual Information Fidelity [58]; MSE - Mean Squared Error; UQI - Universal image Quality Index [59];

Many sketch generation methods were experimented in Section 3.3.2. To find out which among these methods result in sketches that most closely resemble the hand-drawn ones, we attempt to compare the computer generated sketch and the hand-drawn sketch of each 3D CAD model in the ESB. Various state-of-the-art similarity metrics are used, and the average similarity score across all models is obtained. A detailed comparison is reported in Table 1. From the Table, it is clear that the proposed weighted canny approach performs much better than the plain Canny edge detection. The MSE value with and without the non-maximal suppression (NMS) stage of the plain canny method are 1010.96 & 1577.43 respectively, while that of weighted canny is 209.41. The values indicate that canny without NMS performs poorly, and the weighted canny approach performs the best.

It can also be seen that for most similarity metrics, NeuralContours [54] generates a sketch closest to the hand-drawn sketch. However, the time taken to generate one sketch through the other methods is much higher than the proposed sketch generation method in Section 3.3.2. The neural contours method takes around 800 seconds to generate a single sketch image on an NVIDIA 1080Ti GPU, owing to the complex neural network pipeline. This is not suitable for generating sketches for a large number of 3D models. On the other hand, while the performance of the proposed weighted-canny method is close enough to the Neural Contours method in most of these cases, the time taken by to generate one sketch from a 3D model is just one second (including converting a 3D model to an image followed by generating a sketch), on the same hardware setup. This aids very much in generating sketches for a large dataset of 3D models such as the MCB. Hence, the proposed method of weighted canny edge detection is chosen to efficiently generate all the computer sketches of the MCB dataset.

It is important to note that the goal of creating this dataset is to aid in the development of deep learning-based CAD model

search engines. Since the end-users of the search engine are humans, and sketches drawn by an average human being are bound to have errors, the dataset needs to contain sketches that are not perfect. Only then will the learning-based methodologies that make use of this dataset become robust to input noise and errors. Figure 4 shows, for two sample cases, the image of the 3D CAD model, the hand-drawn sketch and the computer-generated sketch. It can be observed that while the computer-generated sketch contains a lot of detail and bears a close resemblance to the input, the sketch does not look realistic. On the other hand, the hand-drawn sketches provide a realistic database of query sketches that can be used to train a robust search engine. Clearly, a hand-drawn sketch dataset is more important and valuable as compared to computer-generated sketch data. Nevertheless, in the absence of a standard large-scale benchmark dataset of hand-drawn sketches, the computer-generated sketch data generated for the MCB dataset using Algorithm 2 is the best option available.

### 3.3.4. Analysis of the proposed sketch-generation method

To further understand the complexity involved in generating the sketch data, the time taken by the proposed technique is computed for each class of the MCB. Since the MCB dataset is not class-balanced, i.e. the data is unevenly distributed across classes, we compare the average time for generating a single sketch of a particular class. It is observed that complex object categories such as Helical Geared Motors, Castor, Turbine etc. take a significantly higher time compared to the other categories. Simple object classes such as Convex Washer, Cylindrical Pin, Setscrew, Washer Bolt etc., take negligible time.

As discussed earlier with respect to the hand-drawn sketches, we only attempt to generate a single sketch corresponding to every 3D CAD model in the MCB dataset and do not change anything else. Hence, the number of sketches obtained are the



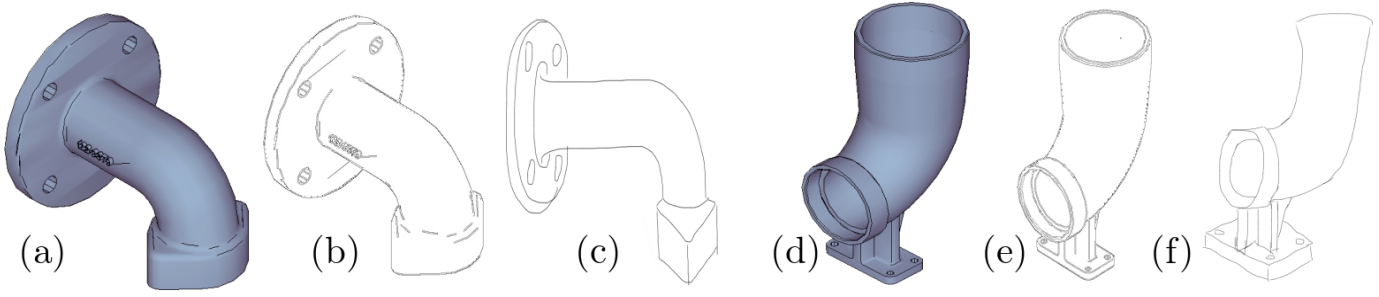


Fig. 4: (a), (d) - Sample Images extracted from two random CAD models in ESB; (b), (e) Computer generated sketch data; (c), (f) - hand-drawn sketch data; Although the computer-generated sketch has a lot of detail and resembles the input closely, the hand-drawn sketch database provides realistic query images that aid in training robust retrieval systems

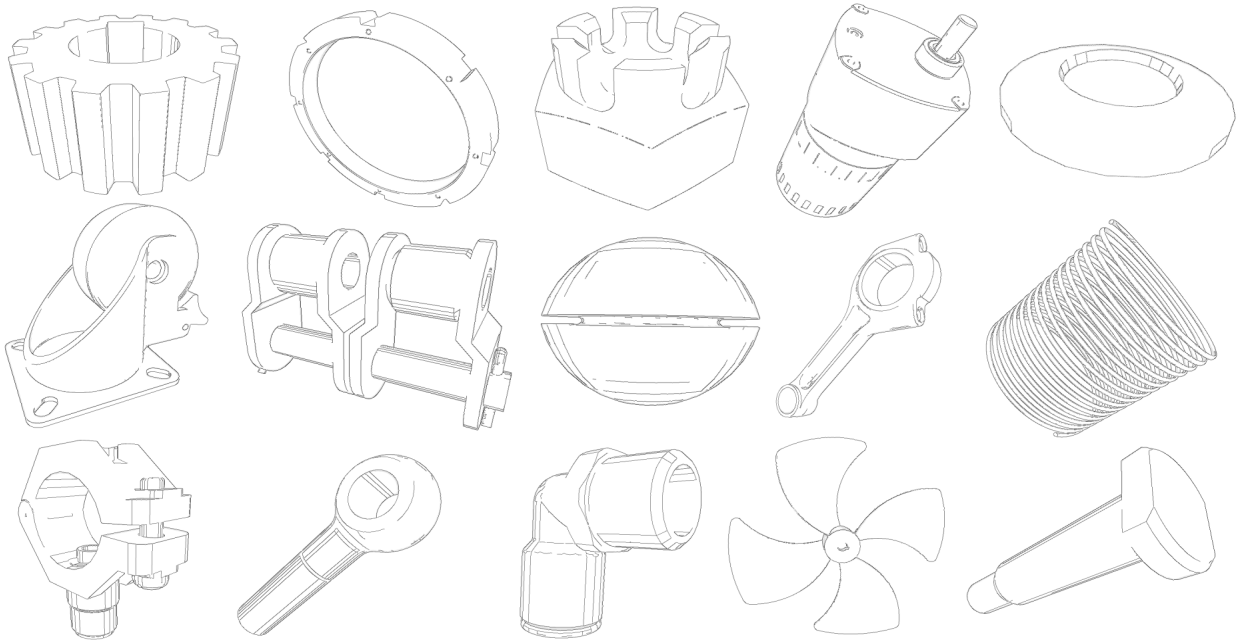


Fig. 5: Sample images from the developed 'CADSketchNet' Dataset-A: Computer-generated sketches.

same as the number of CAD models in MCB, i.e. 58696 sketches across the 68 classes of the MCB. For detailed information regarding the categories and the number of models in each class, the reader is directed to the MCB paper [27].

### 3.4. Summary and 'CADSketchNet' Details

The dataset 'CADSketchNet' contains two subsets.

- Dataset-A contains (1) one representative image for each 3D CAD model in the MCB dataset (obtained using Algorithm 1) (2) one computer-generated sketch for each representative image in the MCB dataset (obtained using Algorithm 2). This results in 58,696 computer-generated sketches across 68 categories. Some sample sketches generated by the proposed method are shown in Figure 5.
- Dataset-B contains 801 hand-drawn sketches, one for each 3D CAD model in the ESB dataset across 42 categories (as described in Section 3.2). Since the 3D CAD model data is obtained from the ESB, the same category information

applies to the Dataset-B as well. Some sample hand-drawn sketches are shown in Figure 2.

Dataset-A has all the images split into an 80-20 ratio for training and testing, respectively. This split is as per the MCB [27]. Dataset-B contains no train-test split since the size of the dataset is not as large as Dataset-A. Nonetheless, users who intend to use this data can customize the train-test ratio as needed.

## 4. Experiments

In this section, we analyze the behaviour of a few learning algorithms for 3D CAD model retrieval on the Dataset-A and Dataset-B of the CADSketchNet.

### 4.1. Experiments on Dataset-A

Methods proposed in literature mainly use point cloud representations ([15, 16]), and Voxel-grid representations ([60]). These network architectures take in point cloud inputs or graph inputs etc., and not images. Since the sketch data is available in

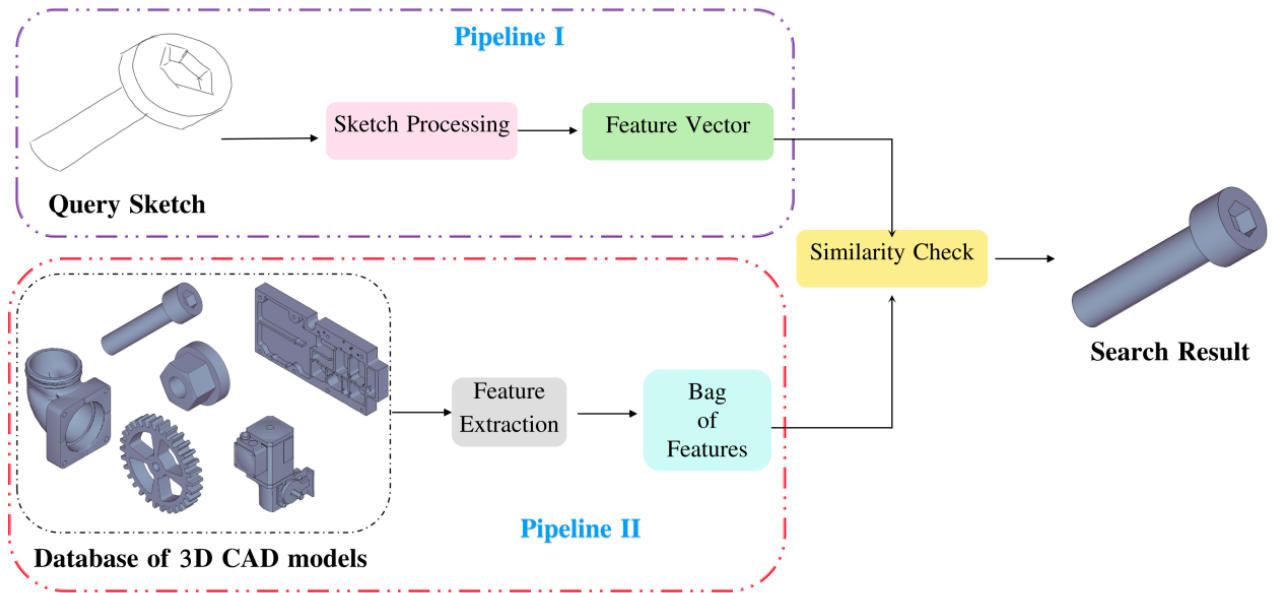


Fig. 6: A generic pipeline for a sketch-based 3D CAD model search engine. Input query - a sketch image drawn by the user. Pipeline I - input sketch pre-processing followed by feature extraction. Pipeline II - extracts features of all the CAD models in the database and stores them as a bag of features. The output feature vector from I is compared against the bag of features from II. Using similarity metrics, the object(s) that is(are) most similar to the input query is(are) retrieved.

the form of images, it is not possible to experiment with these architectures. However, a few papers use view-based representations ([37], [61]). Since, we are dealing mainly with the image representations of 3D CAD models, only the view-based methods can be experimented on the Dataset-A of ‘CADSketchNet’.

The performances of four view-based learning architectures are analysed: MVCNN [37], GVCNN [61], RotationNet [62], and MVCNN-SA [63]. For training each model, we use the code and the default settings for hyper-parameters, as mentioned in the respective papers. A short description of these papers is mentioned here for the sake of completeness and for a better understanding of the techniques experimented:

- MVCNN - Uses two camera setups (12 views and 80 views) to render 2D images from a 3D model. These views are passed to a first CNN for extracting relevant features. The obtained features are then pooled and fed into a second CNN to obtain a compact shape descriptor.
- GVCNN - The 2D views of a 3D model are generated followed by a grouping of these views resulting in different clusters with associated weights. GoogLeNet is used as the base architecture.
- RotationNet - Uses only a partial set ( $\geq 1$ ) of the full multi-view images of an object as input. For each input image, the CNN also outputs the best viewpoint along with the predicted class, since the network treats the view-images as latent variables that are optimized in the training process.
- MVCNN-SA - Uses an approach similar to MVCNN, but attempts to assign relative importance to the input views by using an additional self-attention network.

A train-test split ratio of 80%-20% is used on ‘CADSketchNet’, which is similar to that of the MCB. The results of each of these methods are summarized in Section 6.

#### 4.2. Experiments on Dataset-B

Dataset-B in itself is not a large-scale dataset, and hence, not all deep network architectures can be trained on it. However, using the LFD images (1 3D CAD model = 20 images), the amount of available training data also increases. Hence, in addition to the view-based techniques mentioned above, we also come up with a few other rule-based and learning-based approaches and analyze their performance. The overall pipeline for performing a sketch-based search for 3D CAD models using deep learning can be broadly described as follows:

1. Preparing a dataset of 3D CAD Models and their corresponding sketches suitable for training and testing a deep learning model (discussed in Section 3).
2. Extracting feature representations from the CAD model as well as the query sketch.
3. Developing the model architecture that can efficiently be trained using the extracted representation(s) as input.
4. Checking the similarity of the queried sketch and the 3D CAD models either directly or via their feature representations.
5. Retrieving the top-ranked result(s) based on the metric(s) of similarity used.

An overview of a generic sketch-based search engine for CAD models is shown in Fig. 6. In the following sub-sections, each step of the pipeline used is explained in greater detail.



#### 4.2.1. Feature Extraction and Model Architecture Details

This section discusses steps 2 & 3 of the pipeline, namely (1) extracting the feature representations of the query sketch and the 3D CAD models, and (2) using the extracted representations to build an appropriate network architecture. These two steps go hand-in-hand since the model architecture depends upon the dimensionality of the extracted features. If the extracted representation is a feature vector (1D), a deep neural network (DNN) or variational auto-encoders (VAE) can be used. If the extracted representations are images (2D), then a convolutional neural network (CNN), convolutional auto-encoder (CAE), or a Siamese Network (SN) are some possible options. In some other cases, the features are extracted by the neural network itself. The various methods experimented by us are described in this section.

**Model-1 : HOG-HOG** Histogram of Oriented Gradients (HOG) is a widely used feature descriptor in computer vision and image processing. It differs from other feature extraction methods by extracting the edges' gradient and orientation rather than extracting the edges themselves. The input image is broken down into smaller localized regions, and for each region, the gradients and orientations are computed. Using these a Histogram is computed for each region. Our Model-1 uses the HOG in both Pipelines I and II (see Fig. 6).

*Pipeline I:* The inputs to this pipeline are the sketch images from the dataset. These sketches are passed to the HOG algorithm, which generates the feature vectors for each image separately. The following configuration is used for the HOG algorithm after due experimentation: No. of pixels per cell: (8,8); No. of cells per block: (1,1); Orientations: 8; Block Normalization: L2; Feature Vector Size: 1024\*1;

*Pipeline II:* Using the idea of LFD, 20 images (256\*256) are obtained for each 3D CAD model, resulting in 801\*20 images. These images are then forwarded to the HOG block, which has a similar configuration as mentioned above, and the bag of features is obtained.

**Model-2 : HOG-AE** Model-2 uses the HOG pipeline as defined in Model-1 for Pipeline I and an auto-encoder for Pipeline II. The bag of features is obtained by training the auto-encoder (AE) on the LFD images and then extracting the encoded representation from the latent space of the AE. After various experiments, the following architecture for the AE is obtained. Encoder details: 8 conv layers with (3\*3) filter and (1,1) stride; Batch Normalization (BN) is applied after every two conv layers; 2 dense layers; Decoder details: 2 dense and 8 deconv layers; an up-sampling layer of size (2,2) is applied after every two deconv layers; Activation function: LReLU with a negative slope of 0.01. This AE is trained for 30 epochs using the Adam Optimization algorithm. Learning rate: 0.0001; Loss: Mean Squared Error (MSE);

**Model-3 : HOG-StackedAE** Model-3 uses the HOG pipeline as defined in Model-1 for Pipeline I and a stacked auto-encoder (SAE) for Pipeline II. Pipeline II is similar to the one defined in Model-2. Instead of passing all 801\*20 images as separate inputs, the 20 images of each 3D CAD model are stacked and sent as a single input. Changes from pipeline in Model-2: Number of epochs: 50; Learning-rate: 3e-5;

**Model-4 : HOG-3DCNN** Model-4 uses the HOG pipeline as defined in Model-1 for Pipeline I and a 3D convolutional neural network (3D-CNN) for Pipeline II. For Pipeline II, each 3D CAD model is passed through a 3D-CNN. The extracted representations from the final dense layer are collected together as the bag of features. The architecture used for the 3D-CNN is 18 3D conv layers with kernel size (1,1,1) and stride (1,1,1); max-pooling with kernel-size=2, stride=2 along with BN applied after every 2 conv layers; average pool layer with kernel-size=2, stride=2; two dense layers with a dropout 0.5. This network is trained for 120 epochs with a learning rate of 0.0001; LReLU activation; Loss: MSE; Optimizer: Adam;

**Model-5 : CNN-CNN / Siamese Network** Model-5 uses a convolutional neural network (CNN) for both pipelines. This architecture is also known as Siamese Network, which implements the same network architecture and weights for both pipelines. The network is trained for 10 epochs; Learning Rate: 0.0001; Optimizer: Adam; Batch Size: 2; Loss: SiameseLoss function as described in [64]; Activation function: LeakyReLU with a negative slope = 0.01.

To ensure that a sufficient proportion of similar and dissimilar pairs are generated, an approach similar to that of [64] is used. For each training sketch, a random number of view pairs ( $k_p$ ) in the same category and  $k_n$  view samples from other categories (dis-similar pairs) are chosen. In the current experiment, the values  $k_p = 2$ ,  $k_n = 20$  are used. This random pairing is done for each training epoch. For increasing the number of training samples, data augmentation for the sketch set is also done.

## 5. Implementation Details

### 5.1. Coding Framework and System Configuration

For implementing our neural network models, we use Python3 with PyTorch, while Python3 and sklearn were used to implement the HOG algorithm. OpenCV library is used for all implementations. All the implementations are carried out on a system running Ubuntu 18.04 Operating System. The system has an Intel Core i7-8700K CPU with 64GB RAM and an NVIDIA RTX 2080Ti GPU with 12GB RAM.

### 5.2. Hyper-parameter Tuning, Loss function & Optimization

Training a neural network is an arduous process because of the many decisions involved beforehand, such as choice of performance metrics, hyper-parameters, loss function, etc. Our choices are mainly based on heuristics ([65, 66, 67]) and are backed by experimental verification. The Weights & Biases (wandb) library is also used to assist in hyper-parameter tuning.

## 6. Results and Discussion

The results of all the Model experiments (see Section 4) are discussed here. Since there are no existing methods on the CADSketchNet dataset, we use these experimental results to compare the performance of the best retrieval system. The results of search or retrieval are subjective and cannot be precisely quantified. Nevertheless, to evaluate the retrieval system's performance, we use the standard metrics popularly used in literature.

Methodology	Training Time	Precision	Recall	Retrieval Time	<i>mAP</i>	Top k-Accuracy %
MVCNN	7h 30m	0.932	0.889	$3.73e-05$	0.894	94.03
GVCNN	7h 32m	0.959	0.904	$3.79e-05$	0.853	90.11
RotationNet	10h 14m	0.947	0.868	$3.78e-05$	0.872	92.18
MVCNN-SA	7h 04m	0.941	0.903	$3.69e-05$	0.912	94.56

Table 2: Results obtained by using various view-based approaches when trained using CADSketchNet - Dataset-A.

Methodology	Training Time	Precision	Recall	Retrieval Time	<i>mAP</i>	Top k-Accuracy %
MVCNN	5h 22m	0.949	0.902	$3.78e-05$	0.912	95.15
GVCNN	5h 25m	0.972	0.894	$3.85e-05$	0.886	92.90
RotationNet	8h 3m	0.968	0.871	$3.97e-05$	0.909	96.66
MVCNN-SA	4h 58m	0.9506	0.943	$3.63e-05$	0.947	96.23

Table 3: Results obtained by using various view-based approaches when trained using CADSketchNet - Dataset-B.

The ‘top  $k$  accuracy’ denotes how many of the  $k$ -retrieved classes match the ground truth class. For instance, if 6 out of the top 20 retrieved results match the ground truth class, the accuracy is 30%. For all the reported results, we use  $k=10$ . We also calculate the precision and recall values for the retrieval results. The mean Average Precision (mAP), which is the area under the P-R curve, is also computed.

### 6.1. Results of Experiments on Dataset-A

The results of the four view-based methods experimented in Section 4.1 are summarized in Table 2 along with the time taken for retrieval. Both the MVCNN and the MVCNN-SA methods yield similar performance when tested on Dataset-A, with MVCNN-SA performing slightly better in terms of lesser training time, lesser time taken for retrieval and the top- $k$  accuracy. Although RotationNet takes the longest time for training, it performs better than GVCNN. This could be because of the following reasons. (1) GVCNN assigns relative weights for the input view images and groups the views according to the assigned weights. It might be that the viewing direction of the sketch image does not match with that of the group with the highest weight, thus slightly affecting the performance. (2) RotationNet takes into account the alignment of the input objects. Since object orientations in the MCB Dataset, and thereby the CADSketchNet-A, are aligned, the performance of the RotationNet model is enhanced.

### 6.2. Results of Experiments on Dataset-B

The results of the four view-based methods experimented in Section 4.2 are summarized in Table 3. Also, for the experimental models described in 4.2.1, the outputs from pipelines I and II are compared using a ‘similarity check’ block. For all the models (except Model-5), the Mean Squared Error (MSE) function is used to calculate the similarity between the extracted features. For Model-5, a custom Siamese Loss function described in [64] is used to measure the similarity. These results and the time taken for retrieval are reported in Table 3.

#### 6.2.1. Discussion

Model-1 uses a simple HOG-HOG pipeline and is unable to provide the user with relevant search results. Other models use deep learning techniques and provide better results than the naive approach used in Model-1 (except Model-3). Even the time taken for retrieval is relatively higher, considering the inexpensive computational nature of an algorithmic approach instead of a data-driven approach.

Model-2 and Model-3 use a similar approach to retrieval, with the differences being Model-2 using the view images of the CAD models separately, while Model-3 uses the view images together (stacked). The time taken to train Model-3 (see Table 4) is expectedly lesser since Model-2 has to process 20 times more image data. While the results from Model-2 appear quite satisfactory (in the sense that the top- $k$  accuracy is slightly greater than 50%), Model-3 severely under-performs. This might be because the network uses a stacked set of view images, and in an attempt to capture much information, the network overfits the data and thus under-performs. Thus Model-2, which treats each view image separately, performs much better.

Model-4 uses a 3DCNN, which is computationally very intensive. This is evident from Table 4, where Model-4 takes the highest time for training and searching. This is not conducive since, in real-time scenarios, the search results need to be delivered to the user quickly. Also, the model’s performance is not satisfactory since capturing the 3D CAD models directly leads to unnecessary computations since most voxels in the input data would be empty (3D CAD models are typically sparse). This adversely affects the retrieval performance.

As it is evident from Table 4, using a Siamese network architecture (CNN in both pipelines) for the search engine yields the best results among all methods that have been experimented, for all evaluation metrics. Some sample search results are shown in Figure 7. Also, the model takes much lesser training time compared with other models. The search time depends a lot on the network architecture, the number of parameters to be trained and so forth. Also, the other methods use MSE for similarity measurement, while Model-5 uses a custom Siamese loss function which takes longer to compute. Moreover, search time is

Methodology	Training Time	Precision	Recall	Retrieval Time	<i>mAP</i>	Top k-Accuracy %
HOG-HOG	—	0.666	0.031	$3.72e-05$	0.383	40.00
HOG-AE	8h 20m	0.666	0.054	$1.81e-05$	0.526	51.25
HOG-StackedAE	4h 30m	0.666	0.020	$2.04e-05$	0.490	37.13
HOG-3DCNN	10h 55m	0.666	0.048	$5.41e-05$	0.458	41.21
Siamese Network (CNN-CNN)	1h 10m	0.970	0.784	$2.88e-05$	0.977	95.11

Table 4: Results obtained by using various models when trained on CADSketchNet - Dataset-B. The Siamese Network architecture outperforms other methods.

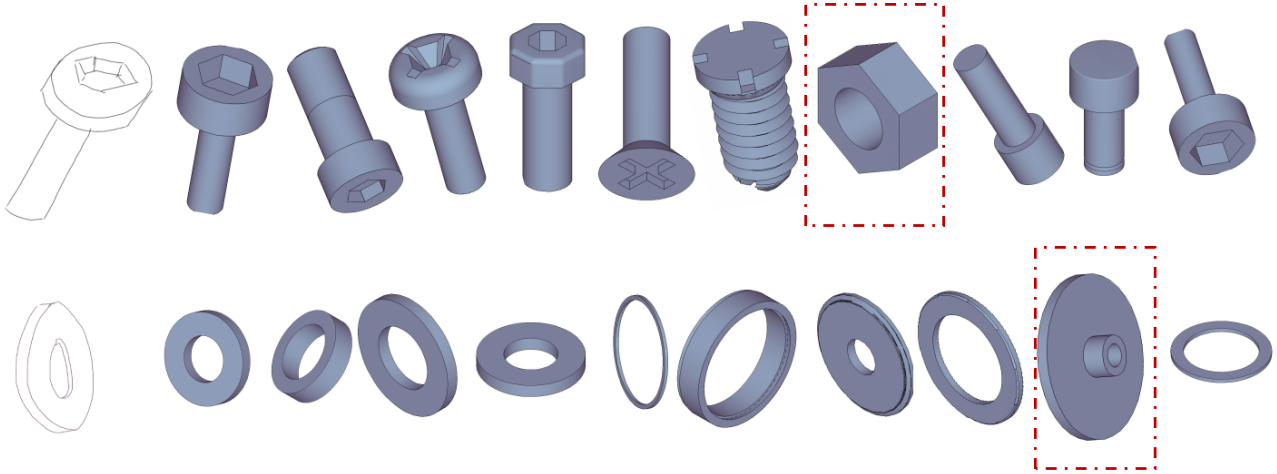


Fig. 7: Top 10 search results of the developed Siamese network model for some sample input queries. Models retrieved in red boxes are from a different category.

only a secondary measure of evaluating model performance. It is more important to obtain a good performance as opposed to obtaining inaccurate results quickly.

Since Model-5 gives the best performance, various deep learning architecture pipelines are used for the Siamese network and are analyzed for performance. These results are reported in Table 5. It can be seen that using a ResNet18 backbone yields the best retrieval accuracy, and ResNet34 backbone results in the best *mAP* value.

In addition, the class-wise results of only the Model-5 are reported in Table 7 & 8 when trained on Dataset-A. It is observed that the Siamese model performs very well on the computer-generated sketch Dataset-A. The top *k*-accuracy value for every class in the MCB dataset is in the range 97.06% to 98.50%. MCB is a large dataset and has sufficient examples in each category. Hence, the higher scores are justified.

Table 9 reports the class-wise results of the Siamese Model when trained on Dataset-B, i.e. the hand-drawn sketches, which has only 801 training samples. The results obtained differ significantly from those found for Dataset-A. The class 'Long Machine Elements,' which comprises only 15 sketches, has the lowest obtained accuracy rating of 86.67 percent. Considering the scarcity of training data, this value indicates a good performance. Some categories only have single-digit training examples, such as 'BackDoors' and 'Clips'. Some of these classes obtain a 100% accuracy, but this is only because there are insufficient number of testing examples. For a majority of the other categories, where there are a significant number of examples available to train, the

Siamese Model performs quite well.

### 6.3. Comparison with deep-learning approaches used for 3D graphical models

The results mentioned in the preceding sections are for models trained on the created CADSketchNet dataset, which focuses on 3D CAD models of mechanical components. The Section 1, Introduction, includes a description of how these CAD models differ from ordinary 3D shapes. In this section, we attempt to validate the assertion by employing deep-learning models designed for 3D graphical models. We examine the performance of the same view-based strategies that were mentioned in Section 4.1.

We use the same pipeline summarized in Figure 6 for this purpose. Pipeline I receives no changes, whereas Pipeline II employs the architecture that was pre-trained on ModelNet40 (a 3D graphical models dataset). The performance of the aforementioned approaches is then evaluated on the created CADSketchNet. Table 6 summarizes these results. Comparing these, with the methods trained on 3D CAD model data (Tables 2 & 3), it can be inferred that, the methodologies established for 3D graphical models do not translate well to 3D CAD models of engineering shapes. Network architectures trained on a dataset specific to CAD models perform significantly better. Therefore, instead of attempting to generalise the usage of approaches intended at graphical models upon engineering shapes, there is a need for developing dedicated datasets and methodologies that explicitly focus on 3D CAD models.

CNN Architecture	Precision	Recall	Retrieval Time	mAP	Top k-Accuracy %
LeNet5 [68]	0.9222	0.6854	2.00e-05	0.6667	60.93
AlexNet [69]	0.8757	0.7746	2.22e-05	0.7901	72.11
VGG16 [70]	0.9112	0.6990	2.24e-05	0.6860	78.81
VGG16_BN [70]	0.8989	0.7921	2.12e-05	0.7498	74.32
VGG19 [70]	0.8889	0.8100	2.20e-05	0.9004	88.80
VGG19_BN [70]	0.9498	0.9100	2.60e-05	0.8905	83.42
Inceptionv3 [71]	0.9668	0.9310	2.90e-05	0.9457	90.31
DenseNet121 [72]	0.9579	0.9330	2.60e-05	0.9001	89.65
DenseNet161 [72]	0.9660	0.9320	2.85e-05	0.9220	91.90
DenseNet169 [72]	0.9410	0.9430	2.54e-05	0.9671	96.54
DenseNet201 [72]	0.9312	0.9256	2.78e-05	0.9256	95.73
Xception [73]	0.9790	0.9555	2.44e-05	0.9780	97.20
ResNet18 [74]	0.9849	0.9610	3.56e-05	0.9660	97.50
ResNet34 [74]	0.9516	0.9810	2.09e-05	0.9851	94.98
ResNet50 [74]	0.9257	0.9066	2.67e-05	0.9234	92.30
ResNet101 [74]	0.9445	0.9255	2.93e-05	0.9222	90.34
Resnet152 [74]	0.9753	0.8923	3.34e-05	0.8231	96.80
ResNeXt50 [75]	0.9407	0.9667	2.89e-05	0.9775	92.09
ResNeXt101 [75]	0.9041	0.9433	2.85e-05	0.9432	88.91

Table 5: Results obtained by using various CNN architectures for the sketch-based retrieval of 3D CAD models when trained on the hand-drawn sketches of CADSketchNet (Dataset-B).

Method	ModelNet40	Dataset-A	Dataset-B
MVCNN	75.83%	94.03%	95.15%
GVCNN	82.46%	90.11%	92.90%
RotationNet	86.84%	92.18%	96.66%
MVCNN-SA	85.94%	94.56%	96.23%

Table 6: The top k-accuracy values of deep-learning architectures that are pre-trained on ModelNet and tested on CADSketchNet vs. when trained on the developed CADSketchNet

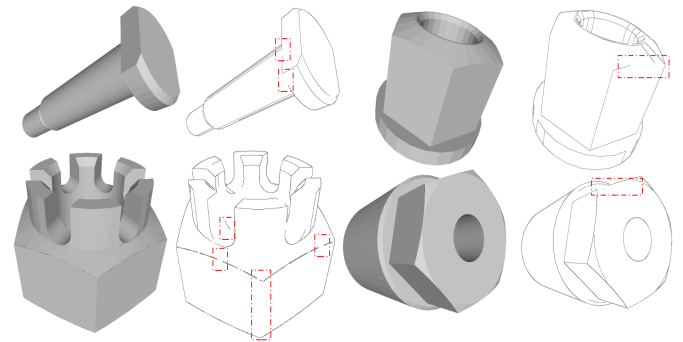


Fig. 8: Illustrating some sample cases of improperly generated computer-sketch data by the proposed method.

#### 6.4. Limitations and Possible future work

The scope of this work is confined to 3D Engineering CAD Mesh models. The current method does not retrieve other types of data, such as images or 3D point sets. It is worthwhile to investigate the possibility of creating a unified dataset with multiple input formats and developing a retrieval system that can search for multiple data formats simultaneously. Also, only sketch query inputs are handled by the proposed model and not text query or 3D model query.

Some computer-generated sketches have unintended artifacts caused by image processing algorithms and a few sketches miss out on some features (Figure 8). While additional processing or other complex sketch generation techniques might potentially be utilised for such sketches, it could considerably increase the time required to gather large-scale data.

As far as the hand-drawn sketches are concerned, since the users are primarily from the design background, the obtained sketches are in the same context. In future, gathering sketches from the perspectives of other area experts, such as assembly or machining, could be considered.

Another possible future work would be for people to contribute to the dataset by providing hand-drawn sketches, which may be used to increase the size of the dataset once it is made open. Recent 3D CAD model datasets such as [28] can also be utilised to generate more sketch images, and the dataset can be further increased. The current work could also be extended to a CAD assembly model retrieval [76].

## 7. Conclusion

A sketch dataset of computer-generated query images, called ‘CADSketchNet’, has been built using the available 3D CAD models from the MCB dataset and the ESB dataset. These images, along with each 3D CAD model’s representative images, are stored in a database. Additionally, hand-drawn sketches corresponding to CAD models from the ESB dataset are also

S No	Class	No.of Models	Precision	Recall	Retrieval Time	mAP	Top k-Accuracy
1	Articulations, eyelets and other articulated joints	1632	0.992	0.815	4.10E-05	0.986	97.58
2	Bearing accessories	107	0.990	0.803	5.60e-05	0.985	98.11
3	Bushes	764	0.988	0.820	4.20e-05	0.985	97.99
4	Cap nuts	225	0.983	0.809	4.10e-05	0.989	97.84
5	Castle nuts	226	0.979	0.820	4.80e-05	0.988	97.82
6	Castor	99	0.983	0.821	4.60e-05	0.987	98.18
7	Chain drives	100	0.974	0.814	4.00e-05	0.988	98.17
8	Clamps	155	0.979	0.807	4.60e-05	0.989	98.21
9	Collars	52	0.986	0.835	5.10e-05	0.984	97.56
10	Conventional rivets	3806	0.983	0.819	3.70e-05	0.986	97.82
11	Convex washer	91	0.981	0.818	3.40e-05	0.987	98.03
12	Cylindrical pins	1895	0.982	0.814	3.30e-05	0.985	97.69
13	Elbow fitting	383	0.992	0.823	3.10e-05	0.987	97.41
14	Eye screws	1131	0.980	0.831	5.20e-05	0.984	98.16
15	Fan	213	0.995	0.817	3.00e-05	0.989	97.15
16	Flanged block bearing	404	0.988	0.815	2.90e-05	0.988	98.03
17	Flanged plain bearings	110	0.985	0.816	4.30e-05	0.988	98.50
18	Flange nut	53	0.991	0.808	5.60e-05	0.988	97.81
19	Grooved pins	2245	0.980	0.812	3.50e-05	0.987	97.85
20	Helical geared motors	732	0.989	0.825	4.70e-05	0.986	97.83
21	Hexagonal nuts	1039	0.991	0.815	3.20e-05	0.988	97.50
22	Hinge	54	0.976	0.815	5.60e-05	0.986	97.96
23	Hook	119	0.985	0.828	5.40e-05	0.990	98.17
24	Impeller	145	0.997	0.840	4.10e-05	0.989	97.06
25	Keys and keyways, splines	4936	0.993	0.818	6.10e-05	0.985	97.71
26	Knob	644	0.988	0.809	4.20e-05	0.984	97.72
27	Lever	1032	0.972	0.816	3.90e-05	0.987	97.66
28	Locating pins	55	0.992	0.820	4.70e-05	0.989	98.14
29	Locknuts	254	0.988	0.805	5.30e-05	0.991	97.75
30	Lockwashers	434	0.979	0.817	4.60e-05	0.986	98.04
31	Nozzle	154	0.988	0.820	4.40e-05	0.988	97.77
32	Plain guidings	49	0.980	0.811	4.70e-05	0.987	98.25
33	Plates, circulate plates	365	0.985	0.813	2.20e-05	0.985	97.67
34	Plugs	169	0.983	0.815	4.60e-05	0.983	98.03
35	Pulleys	121	0.976	0.830	5.10e-05	0.988	97.99
36	Radial contact ball bearings	1199	0.981	0.824	2.30e-05	0.988	97.83

Table 7: Results obtained by the Siamese Network Architecture (Model-5) for the sketch-based retrieval of 3D CAD models when trained on the computer-generated sketches of CADSketchNet (Dataset-A): Part 1 of 2

S No	Class	No.of Models	Precision	Recall	Retrieval Time	mAP	Top k-Accuracy
37	Right angular gearings	60	0.984	0.829	5.10e-05	0.988	97.64
38	Right spur gears	430	0.991	0.815	3.80e-05	0.986	97.70
39	Rivet nut	51	0.991	0.831	4.70e-05	0.984	97.63
40	Roll pins	1597	0.990	0.814	4.20e-05	0.988	98.16
41	Screws and bolts with countersunk head	2452	0.990	0.836	5.20e-05	0.982	97.46
42	Screws and bolts with cylindrical head	3656	0.976	0.818	3.70e-05	0.988	98.06
43	Screws and bolts with hexagonal head	7058	0.983	0.819	6.10e-05	0.989	97.71
44	Setscrew	1334	0.986	0.839	3.80e-05	0.983	97.93
45	Slotted nuts	78	0.984	0.811	5.80e-05	0.989	98.03
46	Snap rings	609	0.978	0.839	4.10e-05	0.986	97.86
47	Socket	858	0.983	0.816	3.80e-05	0.992	97.78
48	Spacers	113	0.990	0.827	3.60e-05	0.987	98.03
49	Split pins	472	1.000	0.830	5.10e-05	0.985	97.53
50	Springs	328	0.986	0.819	5.30e-05	0.991	97.75
51	Spring washers	55	0.991	0.819	2.60e-05	0.984	97.52
52	Square	72	0.991	0.818	5.10e-05	0.987	97.68
53	Square nuts	53	0.987	0.821	4.40e-05	0.991	97.95
54	Standard fitting	764	0.990	0.815	4.80e-05	0.987	98.23
55	Studs	4089	0.983	0.822	5.90e-05	0.988	97.55
56	Switch	173	0.984	0.816	5.70e-05	0.987	97.51
57	Taper pins	1795	0.981	0.821	2.90e-05	0.987	98.15
58	Tapping screws	2182	0.978	0.815	4.70e-05	0.986	97.70
59	Threaded rods	1022	0.979	0.811	3.10e-05	0.987	97.99
60	Thrust washers	2333	0.992	0.835	3.20e-05	0.985	98.14
61	T-nut	101	0.980	0.823	4.40e-05	0.985	97.81
62	Toothed	47	0.989	0.815	5.30e-05	0.988	97.85
63	T-shape fitting	338	0.975	0.812	1.60e-05	0.987	98.03
64	Turbine	85	0.979	0.823	5.30e-05	0.986	97.63
65	Valve	94	0.981	0.815	3.30e-05	0.989	97.67
66	Washer bolt	912	0.982	0.837	4.80e-05	0.986	97.57
67	Wheel	243	0.979	0.826	4.20e-05	0.989	97.47
68	Wingnuts	50	0.992	0.816	5.40e-05	0.989	97.81
	Overall	58696	0.985	0.820	4.34e-05	0.987	97.83

Table 8: Results obtained by the Siamese Network Architecture (Model-5) for the sketch-based retrieval of 3D CAD models when trained on the computer-generated sketches of CADSketchNet (Dataset-A): Part 2 of 2



S No	Class	No.of Models	Precision	Recall	Retrieval Time	mAP	Top k-Accuracy
1	90 degree elbows	41	0.964	0.853	2.21e-05	0.960	95.12
2	BackDoors	7	0.927	0.571	1.92e-05	0.957	100.00
3	Bearing Blocks	7	0.984	0.714	2.78e-05	0.967	100.00
4	Bearing Like Parts	20	0.991	0.900	2.83e-05	0.964	90.00
5	Bolt-like Parts	53	0.975	0.894	2.80e-05	0.966	92.45
6	Bracket-like Parts	18	0.972	0.778	2.68e-05	0.978	94.44
7	Clips	4	0.987	0.500	2.18e-05	0.987	100.00
8	Contact Switches	8	0.971	0.750	2.47e-05	0.971	87.50
9	Container-like Parts	10	0.981	0.700	2.76e-05	0.992	100.00
10	Contoured Surfaces	5	0.962	0.600	2.33e-05	0.986	100.00
11	Curved Housings	9	0.972	0.778	2.18e-04	0.972	100.00
12	Cylindrical Parts	43	0.980	0.930	2.88e-05	0.974	93.02
13	Discs	51	0.970	0.824	3.03e-05	0.977	94.12
14	Flange-like Parts	14	0.979	0.786	2.12e-05	0.978	92.86
15	Gear-like Parts	36	0.976	0.833	2.89e-05	0.985	97.22
16	Handles	18	0.963	0.889	1.92e-05	0.974	100.00
17	Intersecting Pipes	9	0.962	0.667	2.78e-05	0.956	100.00
18	L-Blocks	7	0.951	0.714	2.38e-05	0.979	100.00
19	Long Machine Elements	15	0.974	0.800	2.91e-05	0.978	86.67
20	Long Pins	58	0.969	0.927	2.63e-05	0.976	94.83
21	Machined Blocks	9	0.965	0.778	2.48e-05	0.966	100.00
22	Machined Plates	49	0.984	0.898	2.23e-05	0.986	91.84
23	Motor Bodies	19	0.983	0.842	1.97e-05	0.987	94.74
24	Non-90 degree elbows	8	0.973	0.875	2.61e-05	0.975	87.50
25	Nuts	19	0.971	0.737	1.99e-05	0.972	95.11
26	Oil Pans	8	0.970	0.750	2.68e-05	0.963	89.47
27	Posts	11	0.972	0.728	2.76e-05	0.984	90.91
28	Prismatic Stock	36	0.976	0.861	2.88e-05	0.977	94.44
29	Pulley-like Parts	12	0.974	0.833	2.98e-05	0.982	91.67
30	Rectangular Housings	7	0.968	0.571	2.77e-05	0.983	100.00
31	Rocker Arms	10	0.980	0.700	1.62e-05	0.991	100.00
32	Round Change At End	21	0.953	0.857	1.91e-05	0.967	95.24
33	Simple Pipes	16	0.956	0.875	2.21e-05	0.959	93.75
34	Slender Links	13	0.985	0.769	2.24e-05	0.987	100.00
35	Slender Thin Plates	12	0.971	0.750	2.13e-05	0.979	95.11
36	Small Machined Blocks	12	0.962	0.833	2.42e-05	0.985	100.00
37	Spoked Wheels	15	0.959	0.867	2.65e-05	0.978	93.33
38	T-shaped parts	15	0.954	0.800	2.48e-05	0.986	86.67
39	Thick Plates	23	0.965	0.826	1.99e-05	0.971	91.30
40	Thick Slotted plates	15	0.969	0.733	1.71e-05	0.987	93.33
41	Thin Plates	12	0.971	0.833	2.19e-05	0.995	100.00
42	U-shaped parts	25	0.972	0.800	1.82e-05	0.992	92.00
	Overall	801	0.970	0.784	2.88e-05	0.977	95.11

Table 9: Results obtained by the Siamese Network Architecture (Model-5) for the sketch-based retrieval of 3D CAD models when trained on the hand-drawn sketches of CADSketchNet (Dataset-B).

created and included in ‘CADSketchNet’. This dataset will be made open-source and could contribute to the development of image-based search engines for 3D mechanical component CAD models - using the latest advances in deep learning. The performance of standard view-based methods proposed in the literature is analyzed on CADSketchNet. Additionally, the results of a few other experiments using popular deep learning architectures are also reported. The possibilities of extending this research work to other similar problems have also been discussed.

## Acknowledgments

Thanks are due to the teams of the ESB and the MCB datasets for making their data publicly available. Thanks are also due to many users who have contributed to our CADSketchNet dataset.

## References

- [1] Bai, J, Gao, S, Tang, W, Liu, Y, Guo, S. Design reuse oriented partial retrieval of cad models. *Computer-Aided Design* 2010;42(12):1069 – 1084. doi:<https://doi.org/10.1016/j.cad.2010.07.002>.
- [2] Gunn, TG. The mechanization of design and manufacturing. *Scientific American* 1982;247(3):114–131.
- [3] Ullman, DG. The mechanical design process; vol. 2. McGraw-Hill New York; 2010. <https://www.davidullman.com/mechanical-design-process-6ed>.
- [4] Iyer, N, Jayanti, S, Lou, K, Kalyanaraman, Y, Ramani, K. Three-dimensional shape searching: state-of-the-art review and future trends. *Computer-Aided Design* 2005;37(5):509 – 530. doi:<https://doi.org/10.1016/j.cad.2004.07.002>; *geometric Modeling and Processing* 2004.
- [5] Funkhouser, T, Min, P, Kazhdan, M, Chen, J, Halderman, A, Dobkin, D, et al. A search engine for 3d models. *ACM Trans Graph* 2003;22(1):83–105. doi:10.1145/588272.588279.
- [6] Tangelder, JWH, Veltkamp, RC. A survey of content based 3d shape retrieval methods. In: *Proceedings Shape Modeling Applications*, 2004. 2004, p. 145–156. doi:10.1109/SMI.2004.1314502.
- [7] Bustos, B, Keim, D, Saupe, D, Schreck, T. Content-based 3d object retrieval. *IEEE Computer Graphics and Applications* 2007;27(4):22–27. doi:10.1109/MCG.2007.80.
- [8] Suzuki, MT. A web-based retrieval system for 3d polygonal models. In: *Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference (Cat. No. 01TH8569)*; vol. 4. 2001, p. 2271–2276 vol.4. doi:10.1109/NAFIPS.2001.944425.
- [9] Aono, M, Iwabuchi, H. 3d shape retrieval from a 2d image as query. In: *Proceedings of The 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference*. 2012, p. 1–10.
- [10] Shin, H, Igarashi, T. Magic canvas: Interactive design of a 3-d scene prototype from freehand sketches. In: *Proceedings of Graphics Interface 2007. GI’07*; New York, NY, USA: Association for Computing Machinery. ISBN 9781568813370; 2007, p. 63–70. doi:10.1145/1268517.1268530.
- [11] Lee, J, Funkhouser, T. Sketch-based search and composition of 3d models. In: *Proceedings of the Fifth Eurographics Conference on Sketch-Based Interfaces and Modeling. SBM’08*; Goslar, DEU: Eurographics Association. ISBN 9783905674071; 2008, p. 97–104.
- [12] Li, B, Lu, Y, Godil, A, Schreck, T, Bustos, B, Ferreira, A, et al. A comparison of methods for sketch-based 3d shape retrieval. *Computer Vision and Image Understanding* 2014;119:57 – 80. doi:<https://doi.org/10.1016/j.cviu.2013.11.008>.
- [13] Shilane, P, Min, P, Kazhdan, M, Funkhouser, T. The princeton shape benchmark. In: *Proceedings Shape Modeling Applications*, 2004. 2004, p. 167–178. doi:10.1109/SMI.2004.1314504.
- [14] Wu, Z, Song, S, Khosla, A, Yu, F, Zhang, L, Tang, X, et al. 3D shapenets: A deep representation for volumetric shapes. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, p. 1912–1920. doi:10.1109/CVPR.2015.7298801.
- [15] Qi, CR, Su, H, Mo, K, Guibas, LJ. Pointnet: Deep learning on point sets for 3d classification and segmentation. In: *CVPR. IEEE Computer Society*; 2017, p. 77–85.
- [16] Qi, CR, Yi, L, Su, H, Guibas, LJ. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: *NIPS*. 2017,.
- [17] Eitz, M, Richter, R, Boubekur, T, Hildebrand, K, Alexa, M. Sketch-based shape retrieval. *ACM Trans Graph* 2012;31:31:1–31:10.
- [18] Li, B, Lu, Y, Godil, A, Schreck, T, Aono, M, Johan, H, et al. SHREC’13 Track: Large Scale Sketch-Based 3D Shape Retrieval. In: *Castellani, U, Schreck, T, Biasotti, S, Pratikakis, I, Godil, A, Veltkamp, R, editors. Eurographics Workshop on 3D Object Retrieval. The Eurographics Association. ISBN 978-3-905674-44-6; 2013,doi:10.2312/3DOR/3DOR13/089-096*.
- [19] Li, B, Lu, Y, Li, C, Godil, A, Schreck, T, Aono, M, et al. Extended Large Scale Sketch-Based 3D Shape Retrieval. In: *Bustos, B, Tabia, H, Vandebrorre, JP, Veltkamp, R, editors. Eurographics Workshop on 3D Object Retrieval. The Eurographics Association. ISBN 978-3-905674-58-3; 2014,doi:10.2312/3dor.20141058*.
- [20] Qin, Fw, Li, Ly, Gao, Sm, Yang, Xi, Chen, X. A deep learning approach to the classification of 3D CAD models. *Journal of Zhejiang University SCIENCE C* 2014;15(2):91–106. doi:10.1631/jzus.C1300185.
- [21] Koch, S, Matveev, A, Jiang, Z, Williams, F, Artemov, A, Burnaev, E, et al. Abc: A big cad model dataset for geometric deep learning. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019,.
- [22] Ip, CY, Regli, WC, Sieger, L, Shokoufandeh, A. Automated learning of model classifications. In: *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications. SM ’03*; New York, NY, USA: ACM. ISBN 1-58113-706-0; 2003, p. 322–327. doi:10.1145/781606.781659.
- [23] Wu, MC, Jen, SR. A neural network approach to the classification of 3D prismatic parts. *The International Journal of Advanced Manufacturing Technology* 1996;11(5):325–335. doi:10.1007/BF01845691.
- [24] Bepalov, D, Ip, CY, Regli, WC, Shaffer, J. Benchmarking CAD search techniques. In: *Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling. SPM ’05*; New York, NY, USA: ACM. ISBN 1-59593-015-9; 2005, p. 275–286. doi:10.1145/1060244.1060275.
- [25] Qin, F, Gao, S, Yang, X, Bai, J, hong Zhao, Q. A sketch-based semantic retrieval approach for 3d cad models. *Applied Mathematics-A Journal of Chinese Universities* 2017;32:27–52.
- [26] Jayanti, S, Kalyanaraman, Y, Iyer, N, Ramani, K. Developing an engineering shape benchmark for CAD models. *Computer-Aided Design* 2006;38(9):939 – 953. doi:<https://doi.org/10.1016/j.cad.2006.06.007>; *shape Similarity Detection and Search for CAD/CAE Applications*.
- [27] Kim, S, Chi, Hg, Hu, X, Huang, Q, Ramani, K. A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks. In: *Proceedings of 16th European Conference on Computer Vision (ECCV)*. 2020,.
- [28] Manda, B, Bhaskare, P, Muthuganapathy, R. A convolutional neural network approach to the classification of engineering models. *IEEE Access* 2021;1–1doi:10.1109/ACCESS.2021.3055826.
- [29] Russakovsky, O, Deng, J, Su, H, Krause, J, Satheesh, S, Ma, S, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 2015;115(3):211–252. doi:10.1007/s11263-015-0816-y.
- [30] Sangkloy, P, Burnell, N, Ham, C, Hays, J. The sketchy database: Learning to retrieve badly drawn bunnies. *ACM Trans Graph* 2016;35(4). doi:10.1145/2897824.2925954.
- [31] Eitz, M, Hays, J, Alexa, M. How do humans sketch objects? *ACM Trans Graph* 2012;31(4). doi:10.1145/2185520.2185540.
- [32] Radenovic, F, Tolias, G, Chum, O. Deep shape matching. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018,.
- [33] Dey, S, Riba, P, Dutta, A, Lladós, JL, Song, Y. Doodle to search: Practical zero-shot sketch-based image retrieval. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, p. 2174–2183. doi:10.1109/CVPR.2019.00228.
- [34] Yelamarthi, SK, Reddy, MK, Mishra, A, Mittal, A. A zero-shot framework for sketch-based image retrieval. In: *ECCV*. 2018,.
- [35] Bhunia, A, Yang, Y, Hospedales, TM, Xiang, T, Song, YZ. Sketch less for more: On-the-fly fine-grained sketch-based image retrieval. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 2020;:9776–9785.

- [36] Labs, PVR. Modelnet benchmark leaderboard. 2021. URL: <http://modelnet.cs.princeton.edu/>.
- [37] Su, H, Maji, S, Kalogerakis, E, Learned-Miller, EG. Multi-view convolutional neural networks for 3d shape recognition. 2015 IEEE International Conference on Computer Vision (ICCV) 2015;:945–953.
- [38] Sfikas, K, Theoharis, T, Pratikakis, I. Exploiting the PANORAMA Representation for Convolutional Neural Network Classification and Retrieval. In: Pratikakis, I, Dupont, F, Ovsjanikov, M, editors. Eurographics Workshop on 3D Object Retrieval. The Eurographics Association. ISBN 978-3-03868-030-7; 2017;doi:10.2312/3dor.20171045.
- [39] Maturana, D, Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2015, p. 922–928. doi:10.1109/IROS.2015.7353481.
- [40] Li, B, Schreck, T, Godil, A, Alexa, M, Boubekeur, T, Bustos, B, et al. SHREC'12 Track: Sketch-Based 3D Shape Retrieval. In: Spagnuolo, M, Bronstein, M, Bronstein, A, Ferreira, A, editors. Eurographics Workshop on 3D Object Retrieval. The Eurographics Association. ISBN 978-3-905674-36-1; 2012;doi:10.2312/3DOR/3DOR12/109-118.
- [41] Hua, BS, Truong, QT, Tran, MK, Pham, QH, Kanazaki, A, Lee, T, et al. RGB-D to CAD Retrieval with ObjectNN Dataset. In: Pratikakis, I, Dupont, F, Ovsjanikov, M, editors. Eurographics Workshop on 3D Object Retrieval. The Eurographics Association. ISBN 978-3-03868-030-7; 2017;doi:10.2312/3dor.20171048.
- [42] Muthuganapathy, R, Ramani, K. Shape retrieval contest 2008: Cad models. In: 2008 IEEE International Conference on Shape Modeling and Applications. 2008, p. 221–222. doi:10.1109/SMI.2008.4547977.
- [43] Jain, A, Muthuganapathy, R, Ramani, K. Content-based image retrieval using shape and depth from an engineering database. In: Proceedings of the 3rd International Conference on Advances in Visual Computing - Volume Part II. ISVC'07; Berlin, Heidelberg: Springer-Verlag. ISBN 3540768556; 2007, p. 255–264.
- [44] Pu, J, Ramani, K. On visual similarity based 2d drawing retrieval. Computer-Aided Design 2006;38(3):249 – 259. doi:<https://doi.org/10.1016/j.cad.2005.10.009>.
- [45] Hou, S, Ramani, K. Calligraphic interfaces: Classifier combination for sketch-based 3d part retrieval. Comput Graph 2007;31(4):598–609. doi:10.1016/j.cag.2007.04.005.
- [46] Pu, J, Ramani, K. A 3d model retrieval method using 2d freehand sketches. In: Sunderam, VS, van Albada, GD, Sloat, PMA, Dongarra, JJ, editors. Computational Science – ICCS 2005. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-540-32114-9; 2005, p. 343–346.
- [47] Bonnici, A, Akman, A, Calleja, G, Camilleri, KP, Fehling, P, Ferreira, A, et al. Sketch-based interaction and modeling: where do we stand? Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AI EDAM 2019;33(4):370–388.
- [48] Gryaditskaya, Y, Sytsesteyn, M, Hoftijzer, JW, Pont, S, Durand, F, Bousseau, A. Opensketch: A richly-annotated dataset of product design sketches. ACM Transactions on Graphics (Proc SIGGRAPH Asia) 2019;38.
- [49] Han, W, Xiang, S, Liu, C, Wang, R, Feng, C. Spare3d: A dataset for spatial reasoning on three-view line drawings. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2020, p. 14678–14687. doi:10.1109/CVPR42600.2020.01470.
- [50] Zhong, Y, Qi, Y, Gryaditskaya, Y, Zhang, H, Song, YZ. Towards practical sketch-based 3d shape generation: The role of professional sketches. IEEE Transactions on Circuits and Systems for Video Technology 2020;:1–1doi:10.1109/TCSVT.2020.3040900.
- [51] Seff, A, Ovadia, Y, Zhou, W, Adams, R. Sketchgraphs: A large-scale dataset for modeling relational geometry in computer-aided design. ArXiv 2020;abs/2007.08506.
- [52] Chen, DY, Tian, XP, Shen, YT, Ouhyoung, M. On visual similarity based 3D model retrieval. Computer Graphics Forum 2003;22(3):223–232. doi:10.1111/1467-8659.00669.
- [53] Canny, J. A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 1986;PAMI-8(6):679–698. doi:10.1109/TPAMI.1986.4767851.
- [54] Liu, D, Nabail, M, Hertzmann, A, Kalogerakis, E. Neural contours: Learning to draw lines from 3d shapes. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2020, p. 5427–5435. doi:10.1109/CVPR42600.2020.00547.
- [55] Li, M, Lin, Z, Mech, R, Yumer, E, Ramanan, D. Photo-sketching: Inferring contour drawings from images. In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). 2019, p. 1403–1412. doi:10.1109/WACV.2019.00154.
- [56] Huan, L, Zheng, X, Xue, N, He, W, Gong, J, Xia, G. Unmixing convolutional features for crisp edge detection. CoRR 2020;abs/2011.09808. arXiv:2011.09808.
- [57] Wang, Z, Simoncelli, E, Bovik, A. Multiscale structural similarity for image quality assessment. In: The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003; vol. 2. 2003, p. 1398–1402 Vol.2. doi:10.1109/ACSSC.2003.1292216.
- [58] Sheikh, H, Bovik, A. Image information and visual quality. IEEE Transactions on Image Processing 2006;15(2):430–444. doi:10.1109/TIP.2005.859378.
- [59] Wang, Z, Bovik, A. A universal image quality index. IEEE Signal Processing Letters 2002;9(3):81–84. doi:10.1109/97.995823.
- [60] Maturana, D, Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2015;:922–928.
- [61] Feng, Y, Zhang, Z, Zhao, X, Ji, R, Gao, Y. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition 2018;:264–272.
- [62] Kanazaki, A, Matsushita, Y, Nishida, Y. Rotationnet for joint object categorization and unsupervised pose estimation from multi-view images. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 2021;43(1):269–283. doi:10.1109/TPAMI.2019.2922640.
- [63] Shajahan, DA, Nayel, V, Muthuganapathy, R. Roof classification from 3-d lidar point clouds using multiview cnn with self-attention. IEEE Geoscience and Remote Sensing Letters 2020;17(8):1465–1469. doi:10.1109/LGRS.2019.2945886.
- [64] Fang Wang, , Le Kang, , Yi Li, . Sketch-based 3d shape retrieval using convolutional neural networks. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015, p. 1875–1883. doi:10.1109/CVPR.2015.7298797.
- [65] Goodfellow, I, Bengio, Y, Courville, A. Deep Learning, Chapter - Practical Methodology. MIT Press; 2016. <http://www.deeplearningbook.org>.
- [66] Bengio, Y. Practical recommendations for gradient-based training of deep architectures. CoRR 2012;abs/1206.5533.
- [67] Larochelle, H, Bengio, Y, Louradour, J, Lamblin, P. Exploring strategies for training deep neural networks. Journal of Machine Learning Research 2009;10:1–40.
- [68] Lecun, Y, Bottou, L, Bengio, Y, Haffner, P. Gradient-based learning applied to document recognition. Proceedings of the IEEE 1998;86(11):2278–2324. doi:10.1109/5.726791.
- [69] Krizhevsky, A, Sutskever, I, Hinton, GE. Imagenet classification with deep convolutional neural networks. In: Pereira, F, Burges, CJC, Bottou, L, Weinberger, KQ, editors. Advances in Neural Information Processing Systems 25. Curran Associates, Inc.; 2012, p. 1097–1105.
- [70] Simonyan, K, Zisserman, A. Very deep convolutional networks for large-scale image recognition. CoRR 2014;abs/1409.1556. arXiv:1409.1556.
- [71] Szegedy, C, Vanhoucke, V, Ioffe, S, Shlens, J, Wojna, Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, p. 2818–2826.
- [72] Huang, G, Liu, Z, Van Der Maaten, L, Weinberger, KQ. Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 4700–4708.
- [73] Chollet, F. Xception: deep learning with depthwise separable convolutions (2016). arXiv preprint arXiv:161002357 2016;.
- [74] He, K, Zhang, X, Ren, S, Sun, J. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016, p. 770–778. doi:10.1109/CVPR.2016.90.
- [75] Xie, S, Girshick, R, Dollár, P, Tu, Z, He, K. Aggregated residual transformations for deep neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017, p. 5987–5995. doi:10.1109/CVPR.2017.634.
- [76] Han, Z, Mo, R, Yang, H, Hao, L. CAD assembly model retrieval based on multi-source semantics information and weighted bipartite graph. Computers in Industry 2018;96:54 – 65. doi:<https://doi.org/10.1016/j.compind.2018.01.003>.