

# Sketch-based 3D Shape Retrieval using Convolutional Neural Networks

Fang Wang<sup>1</sup>

Le Kang<sup>2</sup>

Yi Li<sup>1</sup>

<sup>1</sup>NICTA and ANU

<sup>2</sup>ECE, University of Maryland at College Park

<sup>1</sup>{fang.wang, yi.li}@nicta.com.au, <sup>2</sup>le kang@umiacs.umd.edu

## Abstract

Retrieving 3D models from 2D human sketches has received considerable attention in the areas of graphics, image retrieval, and computer vision. Almost always in state of the art approaches a large amount of “best views” are computed for 3D models, with the hope that the query sketch matches one of these 2D projections of 3D models using predefined features.

We argue that this two stage approach (view selection – matching) is pragmatic but also problematic because the “best views” are subjective and ambiguous, which makes the matching inputs obscure. This imprecise nature of matching further makes it challenging to choose features manually. Instead of relying on the elusive concept of “best views” and the hand-crafted features, we propose to define our views using a minimalism approach and learn features for both sketches and views. Specifically, we drastically reduce the number of views to only two predefined directions for the whole dataset. Then, we learn two Siamese Convolutional Neural Networks (CNNs), one for the views and one for the sketches. The loss function is defined on the within-domain as well as the cross-domain similarities. Our experiments on three benchmark datasets demonstrate that our method is significantly better than state of the art approaches, and outperforms them in all conventional metrics.

## 1. Introduction

Retrieving 3D models from 2D sketches has important applications in computer graphics, information retrieval, and computer vision [9, 13, 18]. Compared to the early attempts where keywords or 3D shapes are used as queries [23], the sketch-based idea is very attractive because sketches by hand provide an easy way to input, yet they are rich enough to specify shapes.

Directly matching 2D sketches to 3D models suffers from significant differences between the 2D and 3D representations. Thus, in many state of the art methods 3D mod-

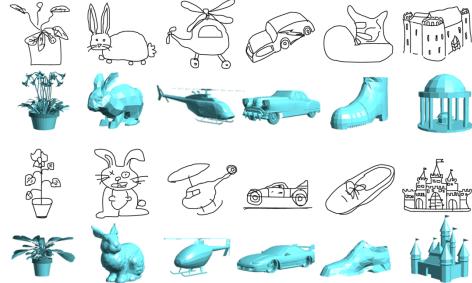


Figure 1. Examples of sketch based 3D shape retrieval.

els are projected to multiple 2D views, and a sketch matches a 3D model if it matches one of its views. Fig. 1 shows a few examples of 2D sketches and their corresponding 3D models. One can immediately see the variations in both the sketch styles and 3D models.

In almost all state of the art approaches, sketch based 3D shape retrieval amounts to finding the “best views” for 3D models and hand-crafting the right features for matching sketches and views. First, an automatic procedure is used to select the most representative views of a 3D model. Ideally, one of the viewpoints is similar to that of the query sketches. Then, 3D models are projected to 2D planes using a variety of line rendering algorithms. Subsequently, many 2D matching methods can be used for computing the similarity scores, where features are always manually defined (e.g., Gabor, dense SIFT, and GALIF [9]).

This stage-wise methodology appears pragmatic, but it also brings a number of puzzling issues. To begin with, there is no guarantee that the best views have similar viewpoints with the sketches. The inherent issue is that identifying the best views is an unsolved problem on its own, partially because the general definition of best views is elusive. In fact, many best view methods require manually selected viewpoints for training, which makes the view selection by finding “best views” a chicken-egg problem.

Further, this viewpoint uncertainty makes it dubious to match samples from two different domains without learning their metrics. Take Fig. 1 for example, even when the

viewpoints are similar the variations in sketches as well as the different characteristics between sketches and views are beyond the assumptions of many 2D matching methods.

Considering all the above issues arise when we struggle to seek the viewpoints for matching, can we bypass the stage of view selection? In this paper we demonstrate that by learning cross domain similarities, we no longer require the seemingly indispensable view similarity assumption.

Instead of relying on the elusive concept of “best views” and hand-crafted features, we propose to define our views and learn features for views and sketches. Assuming that the majority of the models are upright, we drastically reduce the number of views to two per object for the whole dataset. We also make no selections of these two directions as long as they are significantly different. Therefore, we consider this as the minimalism approach as opposed to multiple best views.

This upright assumption appears to be strong, but it turns out to be sensible for 3D datasets. Many 3D models are naturally generated upright (*e.g.*, [23]). We choose two viewpoints because it is very unlikely to get degenerated views for two significantly different viewpoints. An immediate advantage is that our matching is more efficient without the need of comparing to more views than necessary.

This seemingly radical approach triumphs only when the features are learned properly. In principle, this can be regarded as learning representations between sketches and views by specifying similarities, which gives us a semantic level matching. To achieve this, we need comprehensive shape representations rather than the combination of shallow features that only capture low level visual information.

We learn the shape representations using Convolutional Neural Network (CNN). Our model is based on the Siamese network [5]. Since the two input sources have distinctive intrinsic properties, we use two different CNN models, one for handling the sketches and the other for the views. This two model strategy can give us more power to capture different properties in different domains.

Most importantly, we define a loss function to “align” the results of the two CNN models. This loss function couples the two input sources into the same target space, which allows us to compare the features directly using a simple distance function.

Our experiments on three large datasets show that our method significantly outperforms state of the art approaches in a number of metrics, including precision-recall and the nearest neighbor. We further demonstrate the retrievals in each domain are effective. Since our network is based on filtering, the computation is fast.

Our contributions include

- We propose to learn feature representations for sketch based shape retrieval, which bypasses the dilemma of best view selection;

- We adopt two Siamese Convolutional Neural Networks to successfully learn similarities in both the within-domain and the cross domain;
- We outperform all the state of the art methods on three large datasets significantly.

## 2. Related work

Sketch based shape retrieval has received many interests for years [10]. In this section we review three key components in sketch based shape retrieval: public available datasets, features, and similarity learning.

**Datasets** The effort of building 3D datasets can be traced back to decades ago. The Princeton Shape Benchmark (PSB) is probably one of the best known sources for 3D models [23]. There are some recent advancements for general and special objects, such as the SHREC’14 Benchmark [20] and the Bonn Architecture Benchmark [27].

2D sketches have been adopted as input in many systems [6]. However, the large scale collections are available only recently. Eitz *et al.* [9] collected sketches based on the PSB dataset. Li *et al.* [18] organized the sketches collected by [8] in their SBSR challenge.

**Features** Global shape descriptors, such as statistics of shapes [21] and distance functions [15], have been used for 3D shape retrieval [25]. Recently, local features is proposed for partial matching [11] or used in the bag-of-words model for 3D shape retrieval [3].

Boundary information together with internal structures are used for matching sketches against 2D projections. Therefore, a good representation of line drawing images is a key component for sketch based shape retrieval. Sketch representation such as shape context [1] was proposed for image based shape retrieval. Furuya *et al.* proposed BF-DSIFT feature, which is an extended SIFT feature with Bag-of-word method, to represent sketch images [12]. One recent method is the Gabor local line based feature (GALIF) by Mathias *et al.*, which builds on a bank of Gabor filters followed by a Bag-of-word method [9].

In addition to 2D shape features, some methods also explored geometry features as well as graph-based features to facilitate the 3D shape retrieval [19]. Semantic labeling is also used to bridge the gaps between different domains [14]. In this paper, we focus on view based method and only use 2D shape features.

**CNN and Siamese network** Recently deep learning has achieved great success on many computer vision tasks. Specifically, CNN has set records on standard object recognition benchmarks [16]. With a deep structure, the CNN can effectively learn complicated mappings from raw images to

the target, which requires less domain knowledge compared to handcrafted features and shallow learning frameworks.

A Siamese network [5] is a particular neural network architecture consisting of two identical sub-convolutional networks, which is used in a weakly supervised metric learning setting. The goal of the network is to make the output vectors similar if input pairs are labeled as similar, and dissimilar for the input pairs that are labeled as dissimilar. Recently, the Siamese network has been applied to text classification [28] and speech feature classification [4].

### 3. Learning feature representations for sketch based 3D shape retrieval

We first briefly introduce basic concepts in CNNs and Siamese network. Then, we present our network architecture for cross domain matching, based on the Siamese network. Given a set of view and sketch pairs, we propose to use two different Siamese networks, one for each domain. Finally, we revisit the view selection problem, and describe our minimalism approach of viewpoint definition and the line drawing rendering procedure.

#### 3.1. CNN and Siamese network

CNN is a multilayer learning framework, which consists of an input layer, a few convolutional layers and fully connected layers, as well as an output layer on which the loss function is defined. The goal of CNN is to learn a hierarchy of feature representations. Signals in each layer are convolved with a number of filters and further downsampled by pooling operations, which aggregate values in a small region by functions including max, min, and average. The learning of CNN is based on Stochastic Gradient Descent (SGD). Please refer to [17] for details.

Siamese Convolutional Neural Network has been used successfully for dimension reduction in weakly supervised metric learning. Instead of taking single sample as input, the network typically takes a pair of samples, and the loss functions are usually defined over pairs. A typical loss function of a pair has the following form:

$$L(s_1, s_2, y) = (1 - y)\alpha D_w^2 + y\beta e^{\gamma D_w}, \quad (1)$$

where  $s_1$  and  $s_2$  are two samples,  $y$  is the binary similarity label,  $D_w = \|f(s_1; w_1) - f(s_2; w_2)\|_1$  is the distance. Following [5], we set  $\alpha = \frac{1}{C_p}$ ,  $\beta = C_n$ , and  $\gamma = \frac{-2.77}{C_n}$ , where  $C_p = 0.2$  and  $C_n = 10$  are two constants.

This can be regarded as a metric learning approach. Unlike methods that assign binary similarity labels to pairs, the network aims at bringing the output feature vectors closer for input pairs that are labeled as similar, or push the feature vectors away if the input pairs are labeled as dissimilar.

The Siamese network is frequently illustrated as two identical networks for two different samples. In each SGD

iteration, pairs of samples are processed using two identical networks, and the error computed by Eq. 1 is then back-propagated and the gradients are computed individually base on the two sample sets. The Siamese network is updated by the average of these two gradients.

#### 3.2. Cross-domain matching using Siamese network

In this section, we propose a method to match samples from two domains without the heavy assumption of view similarity. We first provide our motivation using an illustrated sample. Then, we propose our extension of the basic Siamese network. Specifically, we use two different networks to handle sources from different domains.

##### 3.2.1 An illustrated example

The matching problem in sketch based shape retrieval can be seen as a metric learning paradigm. In each domain, the samples are mapped to some feature vectors. The cross domain matching is successful if the features from each domain are “aligned” correctly.

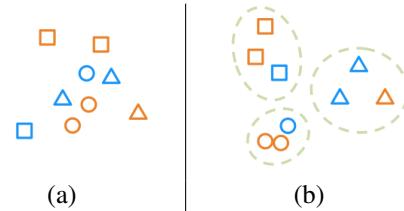


Figure 2. An illustrated example, a) the shapes in the original domain may be mixed, and b) after cross-domain metric learning, similar shapes in both domains are grouped together.

This idea is illustrated in Fig. 2. Blue denotes samples in the sketch domain, and the orange denotes the ones in the view domain. Different shapes denote different classes. Before learning, the feature points from two different domains are initially mixed together (Fig. 2a). If we learn the correct mapping using pair similarities in each domain as well as their cross-domain relations jointly, the two point sets may be correctly aligned in the feature space (Fig. 2b). After this cross domain metric learning, matching can be performed in both the same domain (sketch-sketch and view-view) and cross domain (sketch-view).

Note that, there are no explicit requirements about viewpoint similarity in this perspective (*i.e.*, whether the matched pairs are from the same viewpoints is less important). Instead, the focus is the metric between the two domains and the mapping within the same domain.

##### 3.2.2 Two networks, one loss

The basic Siamese network is commonly used for samples from the same domain. In the cross domain setting, we pro-



Figure 3. Dimension reduction using Siamese network.

pose to extend the basic version to two Siamese networks, one for the view domain and the other for the sketch domain. Then, we define the within-domain loss and the cross domain loss. This hypothesis is supported in the Sec. 4.

Assuming we have two inputs from each domain, *i.e.*,  $s_1$  and  $s_2$  are two sketches and  $v_1$  and  $v_2$  are two views. For simplicity, we assume  $s_1$  and  $v_1$  are from the same class and  $s_2$  and  $v_2$  are from the same class as well. Therefore, one label  $y$  is enough to specify their relationships.

As a result, our loss function is composed by three terms: the similarity of sketches, the similarity of views, and the cross domain similarity.

$$\begin{aligned} \mathcal{L}(s_1, s_2, v_1, v_2, y) \\ = L(s_1, s_2, y) + L(v_1, v_2, y) + L(s_1, v_1, y), \end{aligned} \quad (2)$$

where  $L(\cdot, \cdot, \cdot)$  is defined by Eq. 1. Please note that, while the category information available in the dataset can be exploited to improve the performance, we do not use the category labels in the above framework.

### 3.3. Network architecture

Fig. 3 shows the architecture of our network for the inputs being views and sketches, respectively.

We use the same network design for both networks, but they are learned separately. Our input patch size is  $100 \times 100$  for both sources. The structure of the single CNN has three convolutional layers, each with a max pooling, one fully connected layer to generate the features, and one output layer to compute the loss (Eq. 2).

The first convolutional layer followed by a  $4 \times 4$  pooling generates 32 response maps, each of size  $22 \times 22$ . The second layer and pooling outputs 64 maps of size  $8 \times 8$ . The third layer layer has 256 response maps, each pooled to a size of  $3 \times 3$ . The 2304 features generated by the final pool-

ing operation are linearly transformed to  $64 \times 1$  features in the last layer. Rectified linear units are used in all layers.

### 3.4. View definitions and line drawing rendering

We present our procedure of generating viewpoints and rendering 3D models. As opposed to multiple views, we find it sufficient to use two views to characterize a 3D model because the chance that both views are degenerated is little. Following this observation, we impose the minimal assumptions on choosing views for the whole dataset:

1. Most of the 3D models in the dataset are up-right;
2. Two viewpoints are randomly generated for the whole dataset, provided that the difference in their angles is larger than 45 degrees.

Fig. 4 shows some of our views in the PSB dataset. The first row shows that the upright assumption does not require strict alignments of 3D models, because some models may not have well defined orientation. Further, while the models are upright, they can still have different rotations.



Figure 4. 3D models viewed from predefined viewpoints.

We want to stress that our approach does not eliminate the possibility of selecting more (best) views as input, but the comparisons among view selection methods are beyond the scope of this paper.

Once the viewpoints are chosen, we render the 3D models and generate 2D line drawings. Rendering line drawings that include strong abstraction and stylization effects is a very useful topic in computer graphics, computer vision, and psychology. Outer edges and internal edges both play an important role in this rendering process. Therefore, we use the following descriptors: 1) closed boundaries and 2) Suggestive Contours [7] (Fig. 5).

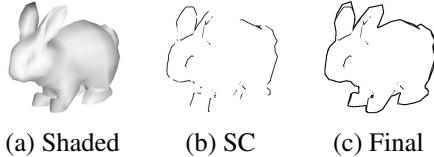


Figure 5. Rendering 3D models.

## 4. Experiments

We present our experiments on three recent large datasets in this section. In all experiments our method outperforms the state of the arts in a number of well recognized metrics. In addition to the cross-domain retrieval, we also present our within-domain retrieval results, which have not been reported in any other comparison methods. These experiments demonstrate that our Siamese network successfully learns the feature representations for both domains. The data and the code is available at <http://users.cecs.anu.edu.au/~yili/cnnsbsr/>.

### 4.1. Datasets

**PSB / SBSR dataset** The Princeton Shape Benchmark (PSB) [23] is widely used for 3D shape retrieval system evaluation, which contains 1814 3D models and is equally divided into training set and testing set.

In [9], the Shape Based Shape Retrieval (SBSR) dataset is collected based on the PSB dataset. The 1814 hand drawn sketches are collected using Amazon Mechanical Turk. In the collection process, participants are asked to draw sketches given only the name of the categories without any visual clue from the 3D models.

**SHREC'13 & '14 dataset** Although the PSB dataset is widely used in shape retrieval evaluation, there is a concern that the number of sketches for each class in the SBSR dataset is not enough. Some classes have only very few instances (27 of 90 training classes have no more than 5 instances), while some classes have dominating number of instances, *e.g.*, the “fighter jet” class and the “human” class have as many as 50 instances.

To remove the possible bias when evaluating the retrieval algorithms, Li *et al.* [18] reorganized the PSB/SBSR dataset, and proposed a SHREC’13 dataset where a subset

of PSB with 1258 models is used and the sketches in each classes has 80 instances. These sketch instances are split in two sets: 50 for training and 30 for testing. Please note, the number of models in each class still varies. For example, the largest class has 184 instances but there are 23 classes containing no more than 5 models

Recently, SHREC’14 is proposed to address some above concerns [20], which greatly enlarges the number of 3D models to 8987, and the number of classes is doubled. The large variation of this dataset makes it much more challenging, and the overall performance of all reported methods are very low (*e.g.*, the accuracy for the best algorithm is only 0.16 for the top 1 candidate). This is probably due to the fact that the models are from various sources and are arbitrarily oriented. While our performance is still superior (see Fig. 9b and Table. 3), we choose to present our results using the SHREC’13 dataset.

**Evaluation criteria** In our experiment, we use the above datasets and measure the performance using the following criteria: 1) *Precision-recall curve* is calculated for each query and linear interpolated, then the final curve is reported by averaging all precision values for fixed recall rates; 2) *Average precision (mAP)* is the area under the precision-recall curve; 3) *Nearest neighbor (NN)* is used to measure the top 1 retrieval accuracy; 4) *E-Measure (E)* is the harmonic mean of the precision and recall for the top 32 retrieval results; 5) *First/second tier (FT/ST)* and *Discounted cumulated gain (DCG)* as defined in the PSB statistics.

### 4.2. Experimental settings

**Stopping criteria** All three of the datasets had been split into training and testing sets, but no validation set was specified. Therefore, we terminated our algorithm after 50 epochs for PSB/SBSR and 20 for SHREC’13 dataset (or until convergence). Multiple runs were performed and the mean values were reported.

**Generating pairs for Siamese network** To make sure we generate reasonable proportion of similar and dissimilar pairs, we use the following approach to generate pair sets. For each training sketch, we random select  $k_p$  view pairs in the same category (matched pairs) and  $k_n$  view samples from other categories (unmatched pairs). Usually, our dissimilar pairs are ten times more than the similar pairs for successful training. In our experiment, we use  $k_p = 2$ ,  $k_n = 20$ . We perform this random pairing for each training epoch. To increase the number of training samples, we also used data augmentation for the sketch set. To be specific, we randomly perform affine transformations on each sketch sample with small scales and angles to generate more variations. We generate two augmentations for each sketch sample in the dataset.

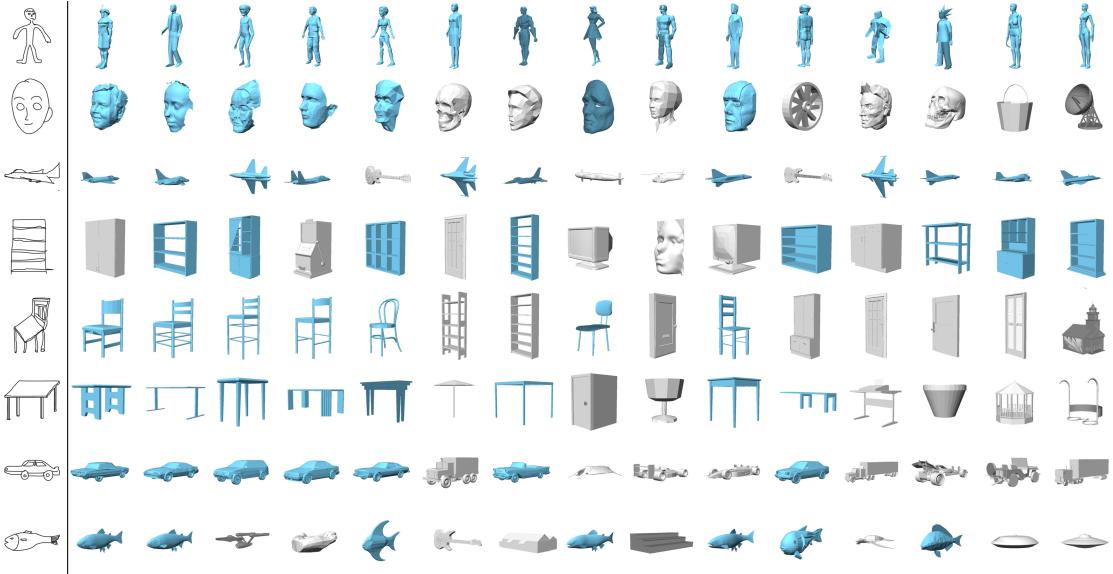


Figure 6. Retrieval examples of PSB/SBSR dataset. Cyan denotes the correct retrievals.

**Computational cost** The implementation of the proposed Siamese CNN is based on the Theano [2] library. We measure the processing time on a PC with 2.8GHz CPU and GTX 780 GPU. With preprocessed view features, the retrieval time for each query is approximately 0.002 sec on average on SHREC’13 dataset.

The training time is proportional to the total number of pairs and the number of epochs. Overall training takes approximately 2.5 hours for PSB/SBSR, 6 hours for SHREC’13, respectively. Considering the total number of pairs is large, the training time is sensible.

We test various number of views in our experiments. We find that there was no significant performance gain when we vary the view from two to ten. However, it increased the computational cost significantly when more views are used, and more importantly, the GPU memory. This motivates us to select only two views in the experiments.

### 4.3. Shape retrieval on PSB/SBSR dataset

#### 4.3.1 Examples

In this section, we test our method using the PSB/SBSR dataset. First, we show some retrieval examples in Fig. 6. The first column shows 8 queries from different classes, and each row shows the top 15 retrieval results. Cyan denotes the correct retrievals, and gray denotes incorrect ones.

Our method performs exceptionally well in popular classes such as human, face, and plane. We also find that some fine grained categorizations are difficult to distinguish. For instance, the shelf and the box differ only in a small part of the model. However, we also want to note that some of the classes only differ in semantics (*e.g.*, barn

and house only differ in function). Certainly, this semantic ambiguity is beyond the scope of this paper.

Finally, we want to stress that the importance of viewpoint is significantly decreased in our metric learning approach. Some classes may exhibit a high degree of freedom such as the plane, but the retrieval results are also excellent (as shown in Fig. 6).

#### 4.3.2 Analysis

We further show some statistics on this dataset. First, we provide the precision-recall values at fixed points in Table 1. Compared to Fig. 9 in [9], our results are approximately 10% higher. We then show six standard evaluation metrics in Table 2. Since other methods did not report the results on this dataset, we leave the comprehensive comparison to the next section. Instead, in this analysis we focus on the effectiveness of metric learning for shape retrieval.

PSB/SBSR is a very imbalanced dataset, where training and testing only partially overlap. Namely, there are 21 classes appear in both training and testing sets, while 71 classes are used solely for testing. This makes it an excellent dataset for investigating similarity learning, because the “unseen” classes verify the learning is not biased.

We show some examples for these unseen classes in Fig. 7 (more statistical curves are available on project website due to the space limitation). It is interesting to see that our proposed method works well even on failure cases (*e.g.*, the flower), where the retrieval returns similar shapes (“potting plant”). This demonstrates that our method learns the similarity effectively.

Table 1. Precision-recall on fixed points.

	5%	20%	40%	60%	80%	100%
	0.616	0.286	0.221	0.180	0.138	0.072

Table 2. Standard metrics on the PSB/SBSR dataset.

NN	FT	ST	E	DCG	mAP
0.223	0.177	0.271	0.173	0.451	0.218

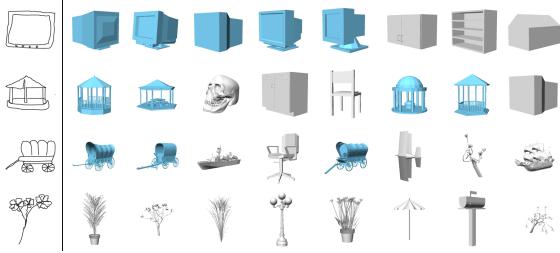


Figure 7. Retrieval examples of unseen samples in PSB/SBSR dataset. The cyan denotes the correct retrievals.

#### 4.4. Shape retrieval on SHREC’13 dataset

In this section, we use the SHREC’13 benchmark to evaluate our method. We also show the retrieval results within the same domain.

##### 4.4.1 A visualization of the learned features

First, we present a visualization of our learned features in Fig. 8. We perform PCA on the learned features and reduce the dimension to two for visualization. The green dots denote the sketches, and the yellow ones denote views. For simplicity, we only overlay the views over the point cloud. Please visit <http://users.cecs.anu.edu.au/~yili/cnnsbsr/> for an interactive demo.

While this is a coarse visualization, we can already see some interesting properties of our method. First, we can see that classes with similar shapes are grouped together automatically. On the top right, different animals are mapped to neighboring positions. On the left, various types of vehicles are grouped autonomously. Other examples include house and church, which are very similar. Note that this is an weakly supervised method. This localization suggests that the learned features are very useful for both within-domain and cross domain retrievals.

##### 4.4.2 Statistical results

We present the statistical results on SHREC’13 in this section. First, we compare the precision-recall curve against the state of the art methods reported in [18].

From the Fig. 9 we can see that our method significantly outperforms other comparison methods. On SHREC’13

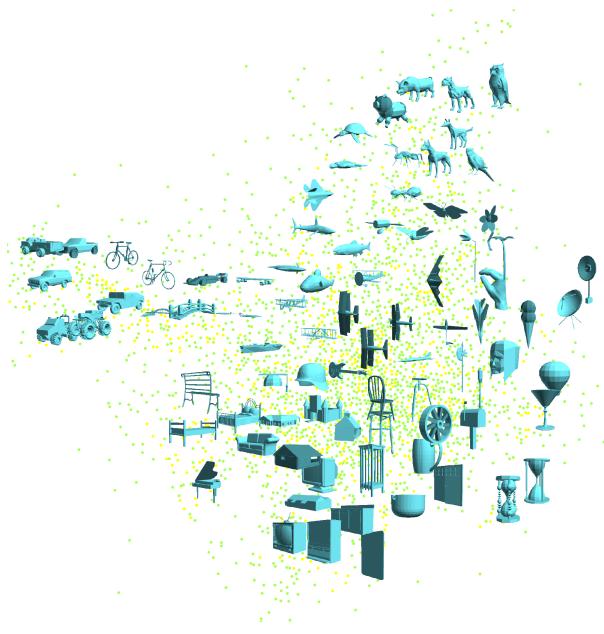


Figure 8. Visualization of feature space on SHREC’13. Sketch and view feature points are shown by green & yellow, respectively.

benchmark, the performance gain of our method is already 10% when recall is small. More importantly, the whole curve decreases much slower than other methods when the recall increases, which is desirable because it shows the method is more stable. Our method has a higher performance gain (30%) when recall reaches 1.

We note that there is a noticeable overfitting in the training when a stopping criterion is reached. It suggests the performance can be even better, if one can fine tune and explore the network structure and training procedure.

We further show the standard metrics for comparison. These metrics examine the retrieval from different perspectives. For simplicity, we only select the best method from each research group in [18]. As shown in Table 3, our method performs better in every metric on both benchmarks. This further demonstrates our method is superior.

We also compare to the case where both networks are identical, *i.e.*, both views and sketches use the same Siamese network. Fig. 9a suggests that this configuration is inferior than our proposed version, but still it is better than all other methods. This supports our hypothesis that the variations in two domains are different. This also sends a message that using the same features (hand-crafted or learned) for both domains may not be ideal.

##### 4.4.3 Within-domain retrieval

Finally, we show the retrievals in the same domain. This interesting experiment shall be straightforward to report because the data is readily available, but was not shown before

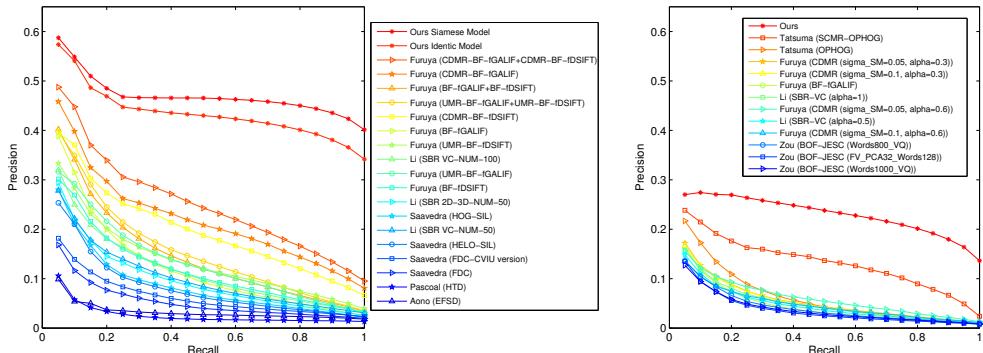


Figure 9. Performance comparison on SHREC'13 & '14. Please refer to [18] and [20] for the descriptions of the compared methods.

Table 3. Comparison on SHREC’13 & ’14 dataset. The best results are shown in red, and the second best results are shown in blue.

SHREC'13						
	NN	FT	ST	E	DCG	mAP
Ours	0.405	0.403	0.548	0.287	0.607	0.469
Identic	0.389	0.364	0.516	0.272	0.588	0.434
[13]	0.279	0.203	0.296	0.166	0.458	0.250
[18]	0.164	0.097	0.149	0.085	0.348	0.116
[24]	0.017	0.016	0.031	0.018	0.240	0.026
[22]	0.110	0.069	0.107	0.061	0.307	0.086
SHREC'14						
	NN	FT	ST	E	DCG	mAP
Ours	0.239	0.212	0.316	0.140	0.496	0.228
[26]	0.160	0.115	0.170	0.079	0.376	0.131
[13]	0.109	0.057	0.089	0.041	0.328	0.054
[18]	0.095	0.050	0.081	0.037	0.319	0.050

in any literature. Since this is a “by-product” of our method, we do not tune up any parameter or re-train the system.

Figs. 10 and 11 visualize some retrieval results in each domain, respectively. Table 4 further reports the statistics. The retrieval results demonstrate our method is powerful in learning the features for both within-domain and cross-domain. From these figures, one can see that the view domain is much more consistent than the sketch domain. Comparing Table 4 to Table 3, we conclude that the inconsistency in sketch is the most challenging issue in the sketch-based 3D shape retrieval.

## 5. Conclusion

In this paper we propose to learn feature presentations for sketch based 3D shape retrieval. Instead of comput-

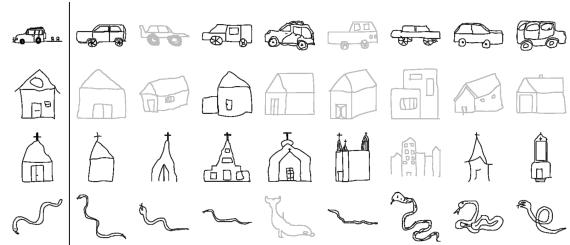


Figure 10. Sketch-sketch retrieval for SHREC’13. The incorrect retrievals are marked as light gray.

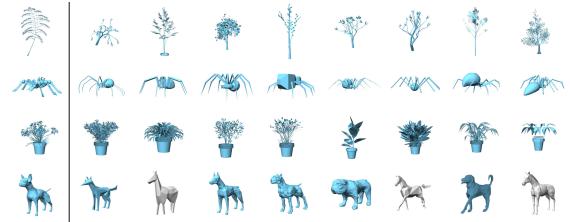


Figure 11. View-view retrieval for SHREC’13. The cyan denotes the correct retrievals.

Table 4. Standard metrics for the within-domain retrieval on SHREC'13.

	NN	FT	ST	E	DCG	mAP
view	0.965	0.877	0.982	0.536	0.971	0.909
sketch	0.431	0.352	0.514	0.298	0.679	0.373

ing “best views” and match them against queries, we use predefined viewpoints for the whole dataset and adopt two Siamese CNNs, one for views and one for sketches. Our experiments on three large datasets demonstrated that our method is superior.

## 6. Acknowledgement

ICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

## References

- [1] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522, Apr. 2002.
- [2] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, Jun. 2010.
- [3] A. M. Bronstein, M. M. Bronstein, L. J. Guibas, and M. Ovsjanikov. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Trans. Graph.*, 30(1):1:1–1:20, Feb. 2011.
- [4] K. Chen and A. Salman. Extracting speaker-specific information with a regularized siamese deep network. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *NIPS 2011*, pages 298–306. 2011.
- [5] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR 2005*, volume 1, pages 539–546. IEEE, 2005.
- [6] P. Daras and A. Axenopoulos. A 3d shape retrieval framework supporting multimodal queries. *International Journal of Computer Vision*, 89(2-3):229–247, 2010.
- [7] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. *ACM Trans on Graphics*, 22(3):848–855, July 2003.
- [8] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Trans. on Graphics*, 31(4):44:1–44:10, 2012.
- [9] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa. Sketch-based shape retrieval. *ACM Trans. Graphics*, 31(4):31:1–31:10, 2012.
- [10] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs. A search engine for 3D models. *ACM Transactions on Graphics*, 22(1):83–105, Jan. 2003.
- [11] T. Funkhouser and P. Shilane. Partial matching of 3d shapes with priority-driven search. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, pages 131–142. Eurographics Association, 2006.
- [12] T. Furuya and R. Ohbuchi. Dense sampling and fast encoding for 3d model retrieval using bag-of-visual features. In *Proceedings of the ACM international conference on image and video retrieval*, page 26. ACM, 2009.
- [13] T. Furuya and R. Ohbuchi. Ranking on cross-domain manifold for sketch-based 3d model retrieval. In *International Conference on Cyberworlds 2013*, pages 274–281. IEEE, 2013.
- [14] B. Gong, J. Liu, X. Wang, and X. Tang. Learning semantic signatures for 3d object retrieval. *Multimedia, IEEE Transactions on*, 15(2):369–377, 2013.
- [15] M. M. Kazhdan, B. Chazelle, D. P. Dobkin, A. Finkelstein, and T. A. Funkhouser. A reflective symmetry descriptor. *ECCV ’02*, pages 642–656, London, UK, UK, 2002. Springer-Verlag.
- [16] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
- [17] Y. LeCun and Y. Bengio. The handbook of brain theory and neural networks. chapter Convolutional networks for images, speech, and time series, pages 255–258. MIT Press, Cambridge, MA, USA, 1998.
- [18] B. Li, Y. Lu, A. Godil, T. Schreck, B. Bustos, A. Ferreira, T. Furuya, M. J. Fonseca, H. Johan, T. Matsuda, et al. A comparison of methods for sketch-based 3d shape retrieval. *Computer Vision and Image Understanding*, 119:57–80, 2014.
- [19] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, Q. Chen, N. K. Chowdhury, B. Fang, et al. A comparison of 3d shape retrieval methods based on a large-scale benchmark supporting multimodal queries. *Computer Vision and Image Understanding*, 131:1–27, 2015.
- [20] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, Q. Chen, N. Chowdhury, B. Fang, T. Furuya, H. Johan, R. Kosaka, H. Koyanagi, R. Ohbuchi, and A. Tatsuma. SHREC14 track: Comprehensive 3d shape retrieval. In *Proc. EG Workshop on 3D Object Retrieval*, 2014.
- [21] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape distributions. *ACM Transactions on Graphics*, 21(4):807–832, Oct. 2002.
- [22] J. M. Saavedra, B. Bustos, T. Schreck, S. Yoon, and M. Scherer. Sketch-based 3d model retrieval using keyshapes for global and local representation. In *EG 3DOR’12*, pages 47–50. Eurographics Association, 2012.
- [23] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The princeton shape benchmark. In *Shape modeling applications, 2004. Proceedings*, pages 167–178. IEEE, 2004.
- [24] P. M. A. Sousa and M. J. Fonseca. Sketch-based retrieval of drawings using spatial proximity. *J. Vis. Lang. Comput.*, 21(2):69–80, 2010.
- [25] J. W. Tangelander and R. C. Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia Tools Appl.*, 39(3):441–471, Sept. 2008.
- [26] A. Tatsuma, H. Koyanagi, and M. Aono. A large-scale shape benchmark for 3d object retrieval: Toyohashi shape benchmark. *Signal and Information Processing Association*, pages 3–6, 2012.
- [27] R. Wessel, I. Blümel, and R. Klein. A 3d shape benchmark for retrieval and automatic classification of architectural data. In *3DOR ’09*, pages 53–56. Eurographics Association, 2009.
- [28] W. Yih, K. Toutanova, J. C. Platt, and C. Meek. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning, CoNLL ’11*, pages 247–256, 2011.