Deep Sketch-Shape Hashing with Segmented 3D Stochastic Viewing

Jiaxin Chen[1*], Jie Qin[1*†], Li Liu[1], Fan Zhu[1], Fumin Shen[2], Jin Xie[3] and Ling Shao[1]
[1]Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, UAE
[2]University of Electronic Science and Technology of China, Chengdu, China
[3]Nanjing University of Science and Technology, Nanjing, China
{firstname.lastname}@inceptioniai.org, fumin.shen@gmail.com, csjxie@njust.edu.cn

Abstract

Sketch-based 3D shape retrieval has been extensively studied in recent works, most of which focus on improving the retrieval accuracy, whilst neglecting the efficiency. In this paper, we propose a novel framework for efficient sketch-based 3D shape retrieval, i.e., Deep Sketch-Shape Hashing (DSSH), which tackles the challenging problem from two perspectives. Firstly, we propose an intuitive 3D shape representation method to deal with unaligned shapes with arbitrary poses. Specifically, the proposed Segmented Stochastic-viewing Shape Network models discriminative 3D representations by a set of 2D images rendered from multiple views, which are stochastically selected from non-overlapping spatial segments of a 3D sphere. Secondly, Batch-Hard Binary Coding (BHBC) is developed to learn semantics-preserving compact binary codes by mining the hardest samples. The overall framework is jointly learned by developing an alternating iteration algorithm. Extensive experimental results on three benchmarks show that DSSH improves both the retrieval efficiency and accuracy remarkably, compared to the state-of-the-art methods.

1. Introduction

Recently, sketch-based 3D shape retrieval has drawn a significant amount of attention from the computer vision community [41, 14, 26, 50, 58, 12, 53, 48, 37, 7], owing to the succinctness of free-hand sketches and the increasing demands from real applications. This task aims to search for semantically relevant 3D shapes queried by 2D sketches, which is very challenging due to the large divergences between the two modalities.

Numerous efforts have been devoted to this task, and they typically aim at improving the retrieval accuracy by learning discriminative representations for both sketches and shapes [48, 53, 7] or developing ranking/distance metrics robust to cross-modality variations [50, 15, 12, 27, 37].

* indicates equal contributions; † indicates corresponding author.

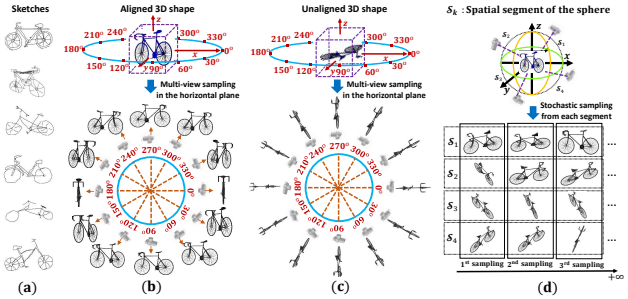


Figure 1. (a) 2D sketches; (b) Sampling 2D images from the aligned 3D shape; (c) Sampling 2D images from the unaligned 3D shape; (d) Our segmented stochastic sampling strategy.

In general, one of the critical issues in this task is how to model the meshed surface of a 3D shape. Most state-of-the-art methods adopt the projection-based model [45], in which a 3D shape is projected into a set of rendered 2D images. Through aborative observations on the existing 3D shape datasets, we find that most shapes are stored in the upright position, as also pointed out by [50, 11]. Hence, if we select the rendering views horizontally (see Fig. 1 (b)), we can always obtain informative 2D images to learn robust representations. The above strategy is commonly adopted by most existing approaches. However, realistic 3D shapes often lack alignment with arbitrary poses[1]. In such cases, conventional methods may fail to acquire useful 2D images. For instance, as shown in Fig. 1 (c), if the 3D shape is stored horizontally, the rendered 2D images will hardly contain any useful information. Note that sketches are often drawn from the side view, so the retrieval task will become intractable given the significant discrepancies between the sketches and 2D images rendered from 3D shapes.

In this work, we first propose a novel stochastic sampling method, namely *Segmented Stochastic-viewing Shape Network (S^{3}N)*, to tackle the above challenge of 3D shape representation. S^{3}N randomly samples rendering views from the sphere around a 3D shape. Concretely, it first divides the

[1]Though shapes can be aligned beforehand, 3D shape alignment is non-trivial and will induce considerable computational time additionally.

sphere into K non-overlapping segments, and then stochastically samples one view from each spatial segment. Furthermore, an attention network is proposed to exploit the importance of different views. Despite its simplicity, the proposed strategy has the following advantages: **1)** K is typically set to a small value (*e.g.*, 4). A 3D shape is thus represented by a set of limited 2D images, making the sampling procedure *computationally efficient*. **2)** Since the spatial segments are non-overlapping and the views are sampled randomly, S^{3}N *avoids* sampling *completely non-informative* views (see Fig. 1 (d)). **3)** If we sample a 3D shape multiple times, a sequence of K rendering views will be generated. Therefore, given sufficient sampling times, S^{3}N can capture as comprehensive information as possible, resulting in much more discriminative representations.

In addition, most existing sketch-based 3D shape retrieval approaches require high computational costs and large memory load. As a result, they are not capable of providing real-time responses for efficient retrieval, especially when dealing with large-scale 3D shape data. There is thus a pressing need for 3D shape retrieval systems that can store a large number of 3D shapes with low memory costs, whilst accomplishing fast and accurate retrieval. Moreover, with the prevalence of portable/wearable devices, which have limited computational capabilities and storage space, the demands for real-time applications in handling large-scale data is rising. To deal with this, inspired by recent advances in binary coding, we aim to project high-dimensional representations to low-dimensional Hamming space, where the distances between the binary codes of sketches and shapes can be computed extremely fast using XOR operations. To this end, a *Batch-Hard Binary Coding (BHBC)* scheme is proposed to learn semantics-preserving discriminative binary codes by mining the hardest positive/negative samples.

Finally, by jointly learning the above two modules and the *Sketch Network* (as shown in Fig. 2), we propose a novel framework, *i.e.*, *Deep Sketch-Shape Hashing (DSSH)*, for efficient and accurate sketch-based 3D shape retrieval. Our main contributions are three-fold:

1) We propose a novel binary coding approach for efficient sketch-based 3D shape retrieval. DSSH learns to embed both sketches and shapes into compact binary codes, which can significantly reduce memory storage and boost computational efficiency. To the best of our knowledge, this is the first work that addresses the efficiency issue of sketch-based 3D shape retrieval, whilst achieving competitive accuracies with the state-of-the-art methods.

2) A new projection-based method (*i.e.*, S^{3}N) is proposed for learning effective 3D representations, even when 3D shapes lack alignment. S^{3}N represents a 3D shape as a set of 2D images rendered from segmented stochastic rendering views. Furthermore, S^{3}N incorporates an attention network that exploits the importance of different views.

3) A novel binary coding strategy (*i.e.*, BHBC) is developed to learn discriminative binary codes by mining the hardest samples across different modalities. More importantly, BHBC, S^{3}N, and the Sketch Network are learned jointly via an alternative iteration optimization algorithm to fulfill the ultimate goal of efficient and accurate retrieval.

2. Related Work

Sketch-based 3D Shape Retrieval. A lot of efforts have been devoted to 3D shape retrieval [31, 4, 52, 2, 45, 39, 54, 3, 11, 1]. Since free-hand sketches are more convenient to acquire than 3D models, sketch-based 3D shape retrieval has attracted more and more attention in the last few years.

Existing works mainly focus on learning modality-specific representations for sketches and 3D shapes [28, 26, 55, 56, 58, 27], or designing effective matching methods across modalities [15, 55, 25, 27]. Recently, a variety of deep learning based approaches have been proposed for joint representation learning and matching [50, 58, 12, 54, 48, 7, 37]. In [50, 12, 53], discriminative representations for both sketches and 3D shapes were learned by two Siamese CNNs. In [58], the cross-domain neural networks with pyramid structures were presented to mitigate cross-domain divergences. [7] addressed the same issue by developing a Generative Adversarial Networks based deep adaptation model. In [37], 3D shape representations were learned by PointNet [38], which was jointly trained with a deep sketch network through semantic embedding. However, all the above works focused on improving the matching accuracy, ignoring the time costs and memory load.

Learning-based Hashing. Hashing/binary coding [22, 35, 17, 34, 29, 13, 24, 42, 9, 20, 8, 40, 6, 32] has been extensively studied recently, due to its promising performance in processing large-scale data. Among various approaches, cross-modality hashing [5, 23, 30, 19, 33], which learns semantics-correlated binary codes for two heterogeneous modalities, is the most relevant to our work. However, most of these methods focus on text or sketch-based image retrieval tasks. [16] discussed the semi-supervised hashing based 3D model retrieval, by applying the existing ITQ [17] method. Nevertheless, all of the above hashing approaches are not specifically designed for the sketch-based 3D shape retrieval, and thus neglect to explore the intrinsic relationships between hand-drawn sketches and 3D shapes.

3. Deep Sketch-Shape Hashing

As illustrated in Fig. 2, our DSSH is composed of two branches of networks, *i.e.*, the *Sketch Network (SN)* and the *Segmented Stochastic-viewing Shape Network (S^{3}N)*. Specifically, SN extracts high-dimensional features using convolutional layers, followed by several hash layers to further learn compact binary representations of sketches. On

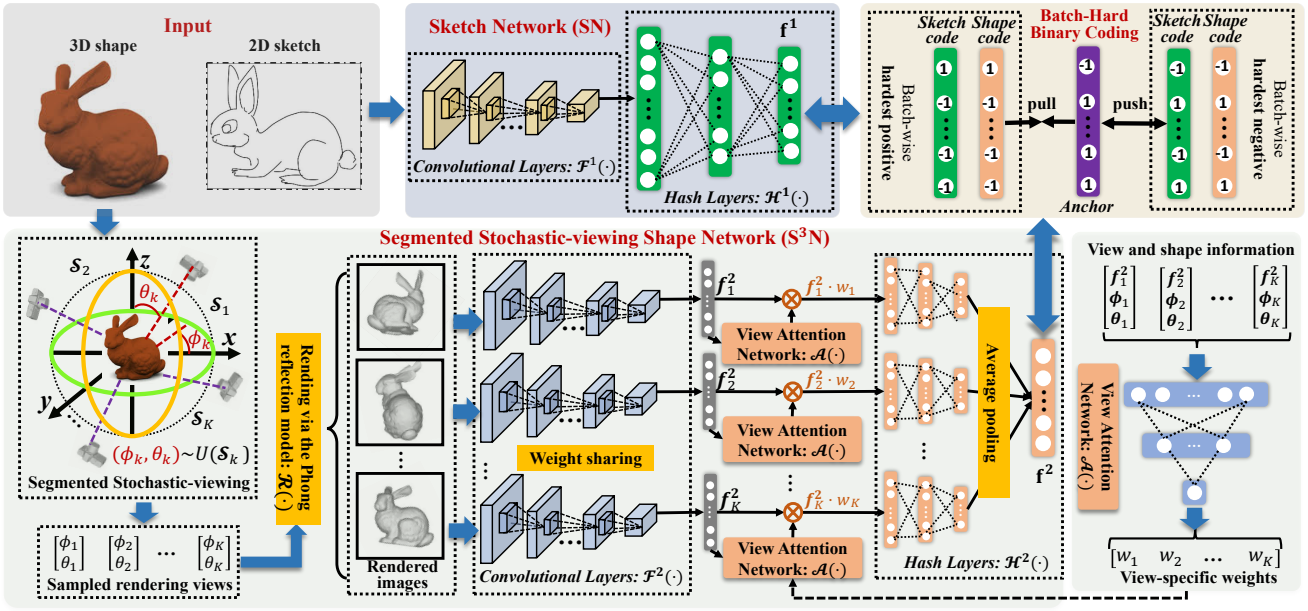


Figure 2. The overall framework of Deep Sketch-Shape Hashing (DSSH). DSSH consists of two branches, *i.e.*, Sketch Network (SN) and Segmented Stochastic-viewing Shape Network (S^3N). SN encodes sketches into compact representations via convolutional layers and hash layers. Besides the above layers, S^3N projects a 3D shape into a set of rendered 2D images with a Segmented Stochastic-viewing module. The view-specific weights are exploited by an intuitive View Attention Network. The weighted features are then aggregated into the final compact representations for the 3D shape via average pooling. To learn semantics-preserving binary codes, we propose a Batch-Hard Binary Coding scheme, which is jointly trained with SN and S^3N for the task of efficient sketch-based 3D shape retrieval.

the other hand, S^3N first obtains a set of K rendered 2D images using the stochastic sampling strategy, which are simultaneously fed into K weight-sharing convolutional layers. View attention networks are employed to learn view-specific weights. Finally, the weighted features are embedded into a low-dimensional subspace through hash layers and aggregated to obtain the final compact binary codes.

For convenience, we use the superscripts $^{\mathbf{1}}$ and $^{\mathbf{2}}$ to indicate the modalities of the 2D sketch and 3D shape, respectively. We denote the functions representing the convolutional layers of the 2D sketch and 3D shape as $\mathcal{F}^{\mathbf{1}}(\cdot)$ and $\mathcal{F}^{\mathbf{2}}(\cdot)$, respectively. Similarly, the hash layers for the 2D sketch and 3D shape are denoted by $\mathcal{H}^{\mathbf{1}}(\cdot)$ and $\mathcal{H}^{\mathbf{2}}(\cdot)$, respectively. As for the *Shape Network*, the additional 2D rendering operation on 3D shapes and the view attention networks are respectively denoted as $\mathcal{R}(\cdot)$ and $\mathcal{A}(\cdot)$. In the following, we will first introduce the 3D shape network and our binary coding scheme in detail. Then, the joint objective function and the optimization algorithm will be elaborated.

3.1. Segmented Stochastic-viewing Shape Network

To represent a 3D shape, we adopt the widely-used projection-based method, *i.e.*, rendering a 3D model into a set of 2D images. Specifically, a virtual camera viewpoint (or rendering view) is selected. The pixel value of the rendered image is determined by interpolating the reflected intensity of the polygon vertices of the 3D shape from the selected viewpoint, via the Phong reflection model [36]. A rendering view is determined by a 3-d vector (r, ϕ, θ). Here, r is the Euclidean distance between the viewpoint and the origin. $\phi \in [0, 2\pi)$ is the angle of the azimuth or the horizontal rotation, and $\theta \in [0, \pi)$ indicates the angle of the vertical elevation. Usually, r is set to a fixed value, which means the virtual camera views are actually located on a sphere $\boldsymbol{\mathcal{S}}$ with radius r. Without loss of generality, we omit the radius r for simplicity, since the rendering view only depends on $(\phi,\ \theta)$.

Different from existing methods that manually select multiple views in the horizontal plane, we develop a stochastic sampling method to obtain arbitrary rendering views, namely *Segmented Stochastic-viewing*. In particular, we divide $\boldsymbol{\mathcal{S}}$ into K segments $\{\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{S}}_2, \cdots, \boldsymbol{\mathcal{S}}_K\}$ with equal spatial coverage. For instance, if $K = 4$, we can split $\boldsymbol{\mathcal{S}}$ into the following four segments:

$$\begin{cases} \boldsymbol{\mathcal{S}}_1 = \{(\phi,\ \theta) \mid \phi \in [0, \pi);\ \theta \in [0, \pi/2)\}; \\ \boldsymbol{\mathcal{S}}_2 = \{(\phi,\ \theta) \mid \phi \in [\pi, 2\pi);\ \theta \in [0, \pi/2)\}; \\ \boldsymbol{\mathcal{S}}_3 = \{(\phi,\ \theta) \mid \phi \in [0, \pi);\ \theta \in [\pi/2, \pi)\}; \\ \boldsymbol{\mathcal{S}}_4 = \{(\phi,\ \theta) \mid \phi \in [\pi, 2\pi);\ \theta \in [\pi/2, \pi)\}. \end{cases} \tag{1}$$

After segmenting the sphere, we select one random rendering view (ϕ_k, θ_k) from each segment $\boldsymbol{\mathcal{S}}_k$, and finally obtain K views $\{(\phi_k, \theta_k)\}_{k=1}^{K}$. Thereafter, a given shape $\boldsymbol{M}$ can be represented by an image set with stochastic render-

ing views as follows:

$$\mathbf{I}_{\boldsymbol{M}} = \{\mathcal{R}(\boldsymbol{M}; \phi_k, \theta_k) | (\phi_k, \theta_k) \sim U(\boldsymbol{\mathcal{S}}_k)\}_{k=1}^{K}, \quad (2)$$

where $\mathcal{R}(\boldsymbol{M}; \phi_k, \theta_k)$ indicates the rendering operation on $\boldsymbol{M}$ based on the viewpoint (ϕ_k, θ_k), and $U(\boldsymbol{\mathcal{S}}_k)$ is the uniform distribution on $\boldsymbol{\mathcal{S}}_k$.

In practice, we employ a batch-wise training process so that each 3D shape can be selected as training data for multiple times. Supposing that $\boldsymbol{M}$ is selected $T_{\boldsymbol{M}}$ times, a sequence of rendering views $\mathbf{V}_{\boldsymbol{M}} = \left\{\{(\phi_k(t), \theta_k(t))\}_{k=1}^{K}\right\}_{t=1}^{T_{\boldsymbol{M}}}$ will be generated using our sampling strategy, where $(\phi_k(t), \theta_k(t))$ is the sampled rendering view from the k-th spatial segment $\boldsymbol{\mathcal{S}}_k$ during the t-th sampling. Correspondingly, $\boldsymbol{M}$ is modeled by a sequence of image snippets $\mathbf{I}_{\boldsymbol{M}} = \{\mathbf{I}_{\boldsymbol{M}}(t)\}_{t=1}^{T_{\boldsymbol{M}}}$, where $\mathbf{I}_{\boldsymbol{M}}(t)$ is generated based on the sampled views $\{(\phi_k(t), \theta_k(t))\}_{k=1}^{K}$ during the t-th sampling. Thereafter, S^{3}N learns the representation of a 3D shape $\boldsymbol{M}$ from the sequence $\{\mathbf{I}_{\boldsymbol{M}}(t)\}_{t=1}^{T_{\boldsymbol{M}}}$.

We have the following intuitive observations w.r.t. S^{3}N:

1) When the number of spatial segments is small (*e.g.*, K=4), a 3D shape $\boldsymbol{M}$ is modeled by a small image set, leading to high computational efficiency.

2) We choose non-overlapping spatial segments $\boldsymbol{\mathcal{S}}_1, \cdots, \boldsymbol{\mathcal{S}}_K$ that cover the entire sphere $\boldsymbol{\mathcal{S}}$ jointly. By sampling in a stochastic way, the selected views $\{(\phi_k, \theta_k)\}_{k=1}^{K}$ can capture non-redundant and complementary characteristics of a 3D shape, making the rendered image set always informative for sketch-based 3D shape retrieval.

3) When $T_{\boldsymbol{M}} \to +\infty$, $\cup_{t=1}^{T_{\boldsymbol{M}}}\{(\phi_k(t), \theta_k(t))\}_{k=1}^{K} \to \boldsymbol{\mathcal{S}}$. In other words, the sampled rendering views can cover the whole space $\boldsymbol{\mathcal{S}}$. Therefore, the proposed sampling strategy has the ability of capturing all 2D viewings of $\boldsymbol{M}$, which is beneficial for learning discriminative and comprehensive representations of 3D shapes.

After obtaining the rendering views via the segmented stochastic sampling, we model $\boldsymbol{M}$ by S^{3}N as follows: $S^3N(\boldsymbol{M}) = \mathcal{H}^{\mathbf{2}}\left(\mathcal{A}(\mathcal{F}^{\mathbf{2}}(I_{\phi_1,\theta_1})), \cdots, \mathcal{A}(\mathcal{F}^{\mathbf{2}}(I_{\phi_K,\theta_K}))\right)$, where $I_{\phi_k,\theta_k} = \mathcal{R}(\boldsymbol{M}; \phi_k, \theta_k)$. Here, $\mathcal{F}^{\mathbf{2}}(\cdot)$ performs convolutional operations on the rendered images and generates a high-dimensional real-valued feature vector $\boldsymbol{f}_k^{\mathbf{2}} = \mathcal{F}^{\mathbf{2}}(I_{\phi_k,\theta_k})$ for the k-th view.

View Attention Network. To fully exploit complementary information across different views, we propose a view attention network $\mathcal{A}(\cdot)$ to capture view-specific weights of the features $\{\boldsymbol{f}_k^{\mathbf{2}}\}_{k=1}^{K}$ from K rendering views. For computational convenience, the scalar ϕ_k is encoded into a 360-d one-hot vector $\boldsymbol{\phi}_k \in \mathbb{R}^{360}$, of which the i-th element is set to 1 if $(i-1) \times \pi/180 \leq \boldsymbol{\phi}_k < i \times \pi/180$, and 0 otherwise. In a similar manner, θ_k is encoded into a 180-d vector. Considering realistic 3D shapes are usually not aligned, the weights of rendering views vary for different 3D shapes. To address this problem, $\mathcal{A}(\cdot)$ also takes the feature $\boldsymbol{f}_k^{\mathbf{2}} = \mathcal{F}^{\mathbf{2}}(I_{\phi_k,\theta_k}) \in \mathbb{R}^{d^{\mathbf{2}}}$ as the input to learn shape-dependant weights. As a result, the concatenated vector $\boldsymbol{a}_k = [\boldsymbol{f}_k^{\mathbf{2}}; \boldsymbol{\phi}_k; \boldsymbol{\theta}_k]^T$ is fed into $\mathcal{A}(\cdot)$, which then outputs a weight $w_k = \mathcal{A}(\boldsymbol{a}_k) \in (0, 1)$ for $\boldsymbol{f}_k^{\mathbf{2}}$.

By using $\mathcal{A}(\cdot)$, we can obtain a set of weighted features $\{w_k \cdot \boldsymbol{f}_k^{\mathbf{2}}\}_{k=1}^{K}$. The hash function $\mathcal{H}^{\mathbf{2}}$ w.r.t. 3D shapes further embeds the weighted features into low-dimensional ones, which are subsequently aggregated as one feature vector $\mathbf{f}^{\mathbf{2}} = \mathcal{H}^{\mathbf{2}}(\{w_k \cdot \boldsymbol{f}_k^{\mathbf{2}}\}_{k=1}^{K})$ via average pooling.

Note that, S^{3}N is *not sensitive* to the input order of $\{I_{\phi_k,\theta_k}\}_{k=1}^{K}$, since all K convolutional/hash layers (one for each segment) share weights, and the average pooling used for feature aggregation is order-invariant.

3.2. Learning Discriminative Binary Codes

As mentioned above, *S^{3}N* provides a framework to learn informative and discriminative representations of 3D shapes. In this subsection, we present the details about how to obtain the final discriminative binary representations.

In practice, mini batches are first constructed from the whole training data by following [7], which can mitigate the class imbalance problem of existing benchmarks. Specifically, we randomly select C classes and collect K 2D sketches and K 3D shapes for each class. We denote the selected $N = C \times K$ images of sketches and 3D shapes by $\mathbf{I}^{\mathbf{1}} = \left\{I_1^{\mathbf{1}}, I_2^{\mathbf{1}}, \cdots, I_N^{\mathbf{1}}\right\}$ and $\mathbf{M} = \{\boldsymbol{M}_1, \boldsymbol{M}_2, \cdots, \boldsymbol{M}_N\}$, respectively. Their corresponding class labels are denoted as $\mathbf{Y}^{\mathbf{1}} = \left\{y_1^{\mathbf{1}}, y_2^{\mathbf{1}}, \cdots, y_N^{\mathbf{1}}\right\}$ and $\mathbf{Y}^{\mathbf{2}} = \left\{y_1^{\mathbf{2}}, y_2^{\mathbf{2}}, \cdots, y_N^{\mathbf{2}}\right\}$.

By passing $\mathbf{I}^{\mathbf{1}}$ through the Sketch Network, we can extract the feature vectors for sketches: $\mathbf{F}^{\mathbf{1}} = [\mathbf{f}_1^{\mathbf{1}}, \cdots, \mathbf{f}_N^{\mathbf{1}}]^T \in \mathbb{R}^{N \times L}$. Here $\mathbf{f}_i^{\mathbf{1}} = \mathcal{H}^{\mathbf{1}}(\mathcal{F}^{\mathbf{1}}(I_i^{\mathbf{1}}))$, where $\mathcal{F}^{\mathbf{1}}$ and $\mathcal{H}^{\mathbf{1}}$ denote the functions of convolutional and hash layers with regard to sketches, respectively. Similarly, given a batch $\mathbf{M}$ of 3D shapes, we can extract the features $\mathbf{F}^{\mathbf{2}} = [\mathbf{f}_1^{\mathbf{2}}, \cdots, \mathbf{f}_N^{\mathbf{2}}]^T \in \mathbb{R}^{N \times L}$, where $\mathbf{f}_i^{\mathbf{2}} = \mathcal{H}^{\mathbf{2}}\left(\mathcal{A}\left[\mathcal{F}^{\mathbf{2}}(\mathcal{R}(\boldsymbol{M}_i))\right]\right)$.

We compute the binary representations $\mathbf{B}^{\mathbf{1}} = [\mathbf{b}_1^{\mathbf{1}}, \cdots, \mathbf{b}_N^{\mathbf{1}}]^T \in \{-1, 1\}^{N \times L}$ for sketches and $\mathbf{B}^{\mathbf{2}} = [\mathbf{b}_1^{\mathbf{2}}, \cdots, \mathbf{b}_N^{\mathbf{2}}]^T \in \{-1, 1\}^{N \times L}$ for 3D shapes as follows:

$$\mathbf{b}_i^m = \mathrm{sgn}(\mathbf{f}_i^m), \text{ for } m \in \{\mathbf{1}, \mathbf{2}\} \text{ and } i = 1, \cdots, N. \quad (3)$$

Here $\mathrm{sgn}(\cdot)$ indicates the element-wise sign function, which outputs 1 for non-negative values, and -1 otherwise.

3.2.1 Batch-Hard Binary Coding

In order to generate discriminative binary representations after the quantization step in Eq. (3), we propose *Batch-Hard Binary Coding (BHBC)* by incorporating semantic class-level information. Formally, given a binary code $\mathbf{b}_i^m$ of the i-th sample from the modality m, the 'hardest positive' code $\mathbf{b}_{i,p^*}^{\hat{m}}$ and the 'hardest negative' code $\mathbf{b}_{i,n^*}^{\hat{m}}$ within

the batch $\mathbf{B}^{\hat{m}}$ from the modality $\hat{m}$ ($m, \hat{m} \in \{\mathbf{1}, \mathbf{2}\}$) are exploited by

$$\begin{aligned} p^* &= \underset{p=1,\cdots,N;\ y_p^{\hat{m}}=y_i^m}{\operatorname{argmax}} d_h(\mathbf{b}_i^m, \mathbf{b}_p^{\hat{m}}), \\ n^* &= \underset{n=1,\cdots,N;\ y_p^{\hat{m}}\neq y_i^m}{\operatorname{argmin}} d_h(\mathbf{b}_i^m, \mathbf{b}_n^{\hat{m}}), \end{aligned} \tag{4}$$

where $d_h(\cdot)$ is the Hamming distance. Eq. (4) implies that $\mathbf{b}_{i,p^*}^{\hat{m}}$ has the maximal intra-class distance to $\mathbf{b}_i^m$, and $\mathbf{b}_{i,p^*}^{\hat{m}}$ has the minimal inter-class distance to $\mathbf{b}_i^m$.

We aim to learn the binary codes $\mathbf{B}^m$ by minimizing $d_h(\mathbf{b}_i^m, \mathbf{b}_{i,p^*}^{\hat{m}})$, whilst maximizing $d_h(\mathbf{b}_i^m, \mathbf{b}_{i,n^*}^{\hat{m}})$ within a large margin η_b. To this end, the loss function $\mathcal{L}_{BC}$ w.r.t. BHBC is defined as

$$\sum_{m,\hat{m},i} \left[\eta_b - d_h(\mathbf{b}_i^m, \mathbf{b}_{i,n^*}^{\hat{m}}) + d_h(\mathbf{b}_i^m, \mathbf{b}_{i,p^*}^{\hat{m}})\right]_+, \tag{5}$$

where $\eta_b > 0$ is the margin, and $[\cdot]_+ = \max(\cdot, 0)$.

It can be observed that minimizing Eq. (5) is equivalent to minimizing the following formula:

$$\sum_{m,\hat{m},i} \sigma_{i,\eta_b}^{m,\hat{m}} \left[d_h(\mathbf{b}_i^m, \mathbf{b}_{i,n^*}^{\hat{m}}) - d_h(\mathbf{b}_i^m, \mathbf{b}_{i,p^*}^{\hat{m}})\right],$$

where $\sigma_{i,\eta_b}^{m,\hat{m}} = 0$ if $d_h(\mathbf{b}_i^m, \mathbf{b}_{i,n^*}^{\hat{m}}) - d_h(\mathbf{b}_i^m, \mathbf{b}_{i,p^*}^{\hat{m}}) \geq \eta_b$, and 1 otherwise.

Based on the above equation and the fact that $d_h(\mathbf{b}_i, \mathbf{b}_j) = \frac{L - \mathbf{b}_i^T \cdot \mathbf{b}_j}{2}$, we finally obtain

$$\mathcal{L}_{BC} := \sum_{m,\hat{m},i} \sigma_{i,\eta_b}^{m,\hat{m}} \cdot (\mathbf{b}_i^m)^T \left[\mathbf{b}_{i,p^*}^{\hat{m}} - \mathbf{b}_{i,n^*}^{\hat{m}}\right]. \tag{6}$$

3.3. Joint Objective Function

Ideally, the features $\mathbf{F}^{\mathbf{1}}$ and $\mathbf{F}^{\mathbf{2}}$ learned by DSSH should be: 1) discriminative, 2) semantically correlative across modalities, and 3) robust to binary quantization. To this end, we develop the following joint loss function $\mathcal{L}_{DSSH}$:

$$\min_{\mathbf{W},\mathbf{B}^{\mathbf{1}},\mathbf{B}^{\mathbf{2}}} \mathcal{L}_{DSSH} := \mathcal{L}_D + \lambda_1 \cdot \mathcal{L}_C + \lambda_2 \cdot \mathcal{L}_Q + \mathcal{L}_{BC}, \tag{7}$$

where $\mathbf{W}$ indicates the parameters of DSSH, $\mathcal{L}_D$ is the loss for learning discriminative features, $\mathcal{L}_C$ is the loss for enhancing semantic correlations between $\mathbf{F}^{\mathbf{1}}$ and $\mathbf{F}^{\mathbf{2}}$, $\mathcal{L}_Q$ is the binary quantization loss, $\mathcal{L}_{BC}$ is the loss function for BHBC, and $\lambda_1, \lambda_2 > 0$ are trade-off parameters. Since $\mathcal{L}_{BC}$ is introduced in the above subsection, we will present the remaining three loss functions in detail, respectively.

1) $\mathcal{L}_D$. We adopt a similar loss function as BHBC, *i.e.*, batch-hard triplet loss [18]. Specifically, given the i-th sample $\mathbf{f}_i^m$ from the m-th modality, we first explore its '*hardest positive*' samples $\mathbf{f}_{i,p^*}^{\hat{m}}$ and '*hardest negative*' samples $\mathbf{f}_{i,n^*}^{\hat{m}}$, within the batch $\mathbf{F}^{\hat{m}}$, from the $\hat{m}$-th modality, as follows:

$$p^* = \underset{\substack{p=1,\cdots,N \\ y_p^{\hat{m}}=y_i^m}}{\operatorname{argmax}} d(\mathbf{f}_i^m, \mathbf{f}_p^{\hat{m}}), n^* = \underset{\substack{n=1,\cdots,N \\ y_p^{\hat{m}}\neq y_i^m}}{\operatorname{argmin}} d(\mathbf{f}_i^m, \mathbf{f}_n^{\hat{m}}), \tag{8}$$

where $d(\cdot)$ is the Euclidean distance.

From Eq. (8), we can see that $\mathbf{f}_{i,p^*}^{\hat{m}}$ is the sample that has the maximal intra-class distance to $\mathbf{f}_i^m$, and $\mathbf{f}_{i,n^*}^{\hat{m}}$ has the minimal inter-class distance to $\mathbf{f}_i^m$. $\mathcal{L}_D$ is then formulated as the following triplet hinge loss [51, 18]

$$\sum_{\substack{m,\hat{m}=\mathbf{1},\mathbf{2} \\ i=1,\cdots,N}} \frac{1}{4N} \left[\eta - d(\mathbf{f}_i^m, \mathbf{f}_{i,n^*}^{\hat{m}}) + d(\mathbf{f}_i^m, \mathbf{f}_{i,p^*}^{\hat{m}})\right]_+, \tag{9}$$

where $\eta > 0$ is the margin.

By minimizing $\mathcal{L}_D$, the maximal intra-class distance will decrease to a value smaller than the minimal inter-class distance within a margin η. This indicates that DSSH can learn discriminative features based on $\mathcal{L}_D$.

2) $\mathcal{L}_C$. We first define the likelihood that $\mathbf{f}_i^{\mathbf{1}}$ and $\mathbf{f}_j^{\mathbf{2}}$ belong to the same class as $\hat{p}_{i,j} = \hat{p}(y_i^{\mathbf{1}} = y_j^{\mathbf{2}} | \mathbf{f}_i^{\mathbf{1}}, \mathbf{f}_j^{\mathbf{2}}) = 1/\left(1 + e^{-\mathbf{f}_i^{\mathbf{1}T}\mathbf{f}_j^{\mathbf{2}}}\right)$, where $\mathbf{f}_i^{\mathbf{1}}$ and $\mathbf{f}_j^{\mathbf{2}}$ are the features of the i-th sketch and the j-th 3D shape, respectively. We compute $\mathcal{L}_C$ as the weighted cross-entropy loss between $\hat{p}(y_i^{\mathbf{1}} = y_j^{\mathbf{2}} | \mathbf{f}_i^{\mathbf{1}}, \mathbf{f}_j^{\mathbf{2}})$ and the ground-truth probability $p_{i,j}$ as follows:

$$-\sum_{i,j=1}^{N} [\frac{1}{N_p} p_{i,j} \log(\hat{p}_{i,j}) + \frac{1}{N_n}(1 - p_{i,j}) \log(1 - \hat{p}_{i,j})], \tag{10}$$

where $p_{i,j} = 1$ if $y_i^{\mathbf{1}} = y_j^{\mathbf{2}}$, and 0 otherwise. N_p is the number of pairs $(\mathbf{f}_i^{\mathbf{1}}, \mathbf{f}_j^{\mathbf{2}})$ from the same class, and N_n is the number of pairs $(\mathbf{f}_i^{\mathbf{1}}, \mathbf{f}_j^{\mathbf{2}})$ from different classes.

3) $\mathcal{L}_Q$. We simply employ the quantization loss as

$$\mathcal{L}_Q = \frac{1}{2}\left(\|\mathbf{F}^{\mathbf{1}} - \mathbf{B}^{\mathbf{1}}\|_F^2 + \|\mathbf{F}^{\mathbf{2}} - \mathbf{B}^{\mathbf{2}}\|_F^2\right), \tag{11}$$

where $\|\cdot\|_F$ indicates the matrix Frobenius norm.

By training DSSH with $\mathcal{L}_Q$, the learned features $\mathbf{F}^{\mathbf{1}}$ and $\mathbf{F}^{\mathbf{2}}$ can remain discriminative after the binary quantization operation in Eq. (3), if both $\mathbf{B}^{\mathbf{1}}$ and $\mathbf{B}^{\mathbf{2}}$ are semantics-preserving, which is guaranteed by the BHBC scheme.

3.3.1 Optimization

In order to solve problem (7), we develop an optimization method based on alternative iteration. Specifically, we learn the binary codes $\mathbf{B}^{\mathbf{1}}$ and $\mathbf{B}^{\mathbf{2}}$ by fixing the parameters $\mathbf{W}$ of the whole network. Subsequently, we update $\mathbf{W}$ by diminishing $\mathcal{L}_{DSSH}$ with fixed $\mathbf{B}^{\mathbf{1}}$ and $\mathbf{B}^{\mathbf{2}}$. We alternatively optimize (7) based on these two steps until convergence.

1) B-Step. When $\mathbf{W}$ is fixed, (7) turns into

$$\begin{aligned} &\min \mathcal{L}_{BC} + \tfrac{\lambda_2}{2} \cdot \left(\|\mathbf{F}^{\mathbf{1}} - \mathbf{B}^{\mathbf{1}}\|_F^2 + \|\mathbf{F}^{\mathbf{2}} - \mathbf{B}^{\mathbf{2}}\|_F^2\right) \\ &\text{s.t. } \mathbf{B}^{\mathbf{1}} \in \{-1, 1\}^{N \times L}, \mathbf{B}^{\mathbf{2}} \in \{-1, 1\}^{N \times L}. \end{aligned} \tag{12}$$

In general, (12) is an NP-hard problem. Inspired by [42], we propose to discretely learn $\mathbf{B}^m$ ($m \in \{\mathbf{1}, \mathbf{2}\}$)

by adopting an alternating optimization solution. Formally, we optimize $\mathbf{b}_i^m$ (*i.e.*, the binary code of the i-th sample from the m-th modality) by fixing the binary codes $\{\mathbf{b}_j^{\tilde{m}} | j = 1, \cdots, N; \tilde{m} \in \{\mathbf{1}, \mathbf{2}\}; j \neq i \text{ if } \tilde{m} = m\}$ of the remaining $2N$-1 samples. $\mathbf{b}_i^m$ can then be updated by optimizing the following problem:

$$\min_{\mathbf{v} \in \{-1,1\}^L} \mathbf{v}^T \left[\sum_{\hat{m} \in \{\mathbf{1},\mathbf{2}\}} \sigma_{i,\eta_b}^{m,\hat{m}} \cdot (\mathbf{b}_{i,p^*}^{\hat{m}} - \mathbf{b}_{i,n^*}^{\hat{m}}) \right] + \frac{\lambda_2}{2} \cdot \|\mathbf{f}_i^m - \mathbf{b}_i^m\|. \quad (13)$$

Based on the fact that $\mathbf{v}^T\mathbf{v} = L$, the loss of (13) is equivalent to $\mathbf{v}^T \left[\sum_{\hat{m} \in \{\mathbf{1},\mathbf{2}\}} \sigma_{i,\eta_b}^{m,\hat{m}} \cdot (\mathbf{b}_{i,p^*}^{\hat{m}} - \mathbf{b}_{i,n^*}^{\hat{m}}) - \lambda_2 \cdot \mathbf{f}_i^m \right] + const.$ It is clear that the closed-form solution to (13) is

$$\mathbf{b}_i^m = \text{sgn}\Big(\sum_{\hat{m} \in \{\mathbf{1},\mathbf{2}\}} \sigma_{i,\eta_b}^{m,\hat{m}} \cdot (\mathbf{b}_{i,n^*}^{\hat{m}} - \mathbf{b}_{i,p^*}^{\hat{m}}) + \lambda_2 \cdot \mathbf{f}_i^m \Big). \quad (14)$$

2) W-Step. If we fix the binary codes $\mathbf{B^1}$ and $\mathbf{B^2}$, the optimization over $\mathbf{W}$ is formulated as

$$\min_{\mathbf{W}} \mathcal{L}_{\mathbf{W}} := \mathcal{L}_D + \lambda_1 \cdot \mathcal{L}_C + \lambda_2 \cdot \mathcal{L}_Q. \quad (15)$$

In practice, we adopt the Adam stochastic gradient descent algorithm [21] to solve this problem.

4. Experimental Results and Analysis

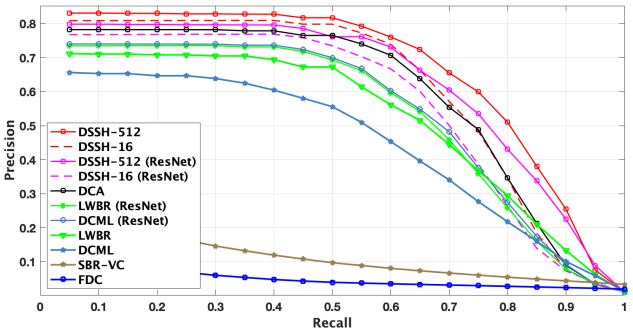
4.1. Datasets

We evaluate the proposed method on three benchmarks for sketch-based 3D shape retrieval, *i.e.*, SHREC'13, SHREC'14, and PART-SHREC'14.

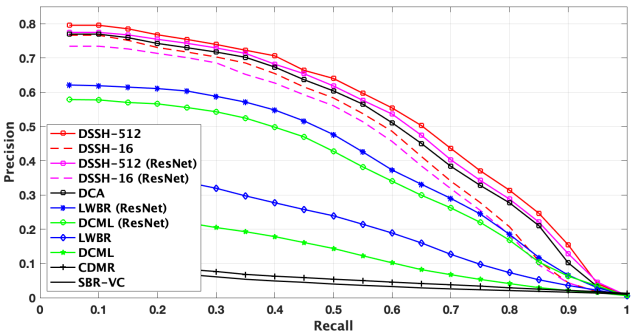
SHREC'13 [25] collects human-drawn sketches and 3D shapes from the Princeton Shape Benchmark (PSB) [43] that consists of 7,200 sketches and 1,258 shapes from 90 classes. There are a total of 80 sketches per class, 50 of which are selected for training and the rest for test. The numbers of 3D shapes are different for distinct classes, about 14 on average for each class.

SHREC'14 [28] contains 13,680 sketches and 8,987 3D shapes from 171 classes. There are 80 sketches, and around 53 3D shapes on average for each class. The sketches are split into 8,550 training data and 5,130 test data.

PART-SHREC'14 [37] is collected from SHREC'14 to overcome the shortcomings that all 3D shapes are used for both training and testing. By using this dataset, we can evaluate the performance of current models on retrieving unseen 3D shapes. Concretely, it selects 48 classes that have more than 50 shapes in SHREC'14, which thereafter result in 7,238 3D shapes and 3,840 sketches. The sketches are split into 50/30 training/test data, whilst 3D shapes are randomly split into a training set of 5,812 and a test set of 1,426.



(a) SHREC'13



(b) SHREC'14

Figure 3. Precision-Recall curves.

4.2. Implementation Details

For the convolutional layers $\mathcal{F}^1$ and $\mathcal{F}^2$, we adopt the Inception-ResNet-v2 [46] pretrained on ImageNet as the base network, by removing the last fully-connected layer. Both the hash layers $\mathcal{H}^1$ and $\mathcal{H}^2$ consist of three fully-connected layers as $[1536, 1024, 512, L]$. We utilize the '*ReLU*' activation functions for all layers, except that the last layer uses the '*Tanh*' activation function. The view attention network $\mathcal{A}$ contains two fully-connected layers as $[2076, 300, 1]$, where the last layer uses the '*Sigmoid*' activation function. The trade-off parameters λ_1 and λ_2 are selected by cross-validation on the training set, and are set to 1 and 0.001, respectively, for all datasets. Regarding the number of spatial segments, we empirically set $K = 4$ considering both computational efficiency and convergent speed[2].

4.3. Evaluation Metrics

We utilize the following widely-adopted metrics [26, 12, 53] for sketch-based 3D shape retrieval: nearest neighbor (**NN**), first tier (**FT**), second tier (**ST**), E-measure (**E**), discounted cumulated gain (**DCG**), and mean average precision (**mAP**). We also draw **precision-recall curves** for visually evaluating the retrieval performance.

[2]Sample code is available at `https://sites.google.com/site/chenjiaxinx/`

Table 1. Performance comparisons with the state-of-the-art methods on SHREC'13 and SHREC'14.

Method	SHREC'13								SHREC'14							
	NN	FT	ST	E	DCG	mAP	Query Time (sec.)	Memory (KB)	NN	FT	ST	E	DCG	mAP	Query Time (sec.)	Memory (KB)
CDMR [15]	0.279	0.203	0.296	0.166	0.458	0.250	-	-	0.109	0.057	0.089	0.041	0.328	0.054	-	-
SBR-VC [25]	0.164	0.097	0.149	0.085	0.348	0.114	-	-	0.095	0.050	0.081	0.037	0.319	0.050	-	-
SP [44]	0.017	0.016	0.031	0.018	0.240	0.026	-	-	-	-	-	-	-	-	-	-
FDC [25]	0.110	0.069	0.107	0.061	0.307	0.086	-	-	-	-	-	-	-	-	-	-
DB-VLAT [49]	-	-	-	-	-	-	-	-	0.160	0.115	0.170	0.079	0.376	0.131	-	-
CAT-DTW [55]	0.235	0.135	0.198	0.109	0.392	0.141	-	-	0.137	0.068	0.102	0.050	0.338	0.060	-	-
Siamese [50]	0.405	0.403	0.548	0.287	0.607	0.469	3.13×10^{-4}	3.1×10^{-1}	0.239	0.212	0.316	0.140	0.496	0.228	1.50×10^{-3}	2.2
KECNN [47]	0.320	0.319	0.397	0.236	0.489	-	-	-	-	-	-	-	-	-	-	-
DCML [12]	0.650	0.634	0.719	0.348	0.766	0.674	4.67×10^{-2}	2.0×10^{1}	0.272	0.275	0.345	0.171	0.498	0.286	3.95×10^{-1}	1.4×10^{2}
DCHML [10]	0.730	0.715	0.773	0.368	0.816	0.744	4.67×10^{-2}	2.0×10^{1}	0.403	0.329	0.394	0.201	0.544	0.336	3.95×10^{-1}	1.4×10^{2}
LWBR [53]	0.712	0.725	0.785	0.369	0.814	0.752	4.67×10^{-2}	2.0×10^{1}	0.403	0.378	0.455	0.236	0.581	0.401	3.95×10^{-1}	1.4×10^{2}
DCML (ResNet) [12]	0.740	0.752	0.797	0.365	0.829	0.774	2.52×10^{-2}	9.8	0.578	0.591	0.647	0.723	0.351	0.615	1.96×10^{-1}	7.0×10^{1}
LWBR (ResNet) [53]	0.735	0.745	0.784	0.359	0.825	0.767	2.52×10^{-2}	9.8	0.621	0.641	0.691	0.760	0.361	0.665	1.96×10^{-1}	7.0×10^{1}
Shape2Vec [48]	-	-	-	-	-	-	-	-	0.714	0.697	0.748	0.360	0.811	0.720	3.01×10^{-2}	1.7×10^{1}
DCA [7]	0.783	0.796	0.829	0.376	0.856	0.813	2.52×10^{-2}	6.1×10^{-1}	0.770	0.789	0.823	0.398	0.859	0.803	1.96×10^{-1}	4.4
Semantic [37]	0.823	0.828	0.860	0.403	0.884	0.843	-	-	**0.804**	0.749	0.813	0.395	0.870	0.780	-	-
DSSH-16 (ResNet)	0.768	0.777	0.797	0.371	0.841	0.793	$\mathbf{1.84\times10^{-6}}$	$\mathbf{2.4\times10^{-3}}$	0.735	0.726	0.771	0.377	0.830	0.742	$\mathbf{9.21\times10^{-6}}$	$\mathbf{1.7\times10^{-2}}$
DSSH-64 (ResNet)	0.798	0.808	0.844	0.391	0.869	0.826	7.10×10^{-6}	9.6×10^{-3}	0.758	0.758	0.792	0.385	0.841	0.769	3.76×10^{-5}	6.9×10^{-2}
DSSH-256 (ResNet)	0.804	0.817	0.863	0.402	0.876	0.834	2.83×10^{-5}	3.8×10^{-2}	0.771	0.777	0.822	0.400	0.862	0.792	1.45×10^{-4}	2.7×10^{-1}
DSSH-512 (ResNet)	0.799	0.814	0.860	0.404	0.873	0.831	5.00×10^{-5}	7.7×10^{-2}	0.775	0.788	0.831	0.404	0.870	0.806	2.61×10^{-4}	5.5×10^{-1}
DSSH-16	0.809	0.813	0.828	0.383	0.865	0.825	$\mathbf{1.84\times10^{-6}}$	$\mathbf{2.4\times10^{-3}}$	0.767	0.762	0.794	0.385	0.844	0.773	$\mathbf{9.21\times10^{-6}}$	$\mathbf{1.7\times10^{-2}}$
DSSH-64	0.823	0.835	0.867	0.403	0.885	0.849	7.10×10^{-6}	9.6×10^{-3}	0.788	0.799	0.839	0.407	0.875	0.815	3.76×10^{-5}	6.9×10^{-2}
DSSH-256	0.829	0.842	0.879	0.409	0.891	0.855	2.83×10^{-5}	3.8×10^{-2}	0.796	0.811	**0.851**	0.411	**0.881**	**0.826**	1.45×10^{-4}	2.7×10^{-1}
DSSH-512	**0.831**	**0.844**	**0.886**	**0.411**	**0.893**	**0.858**	5.00×10^{-5}	7.7×10^{-2}	0.796	**0.813**	**0.851**	**0.412**	**0.881**	**0.826**	2.61×10^{-4}	5.5×10^{-1}

('-' indicates that the results are not reported, or the source codes as well as implementation details are not available.)

4.4. Comparisons with the State-of-the-Art Methods for Sketch-Based 3D Shape Retrieval

We compare our DSSH with the state-of-the-art methods for sketch-based 3D shape retrieval, including the hand-crafted methods [15, 25, 44, 25, 49, 55] and deep learning based ones [50, 47, 12, 10, 53, 12, 53, 48, 7, 37]. For fair comparisons with deep learning based methods, we also report our performance by using ResNet-50 as the base network, denote by DSSH (ResNet). Since the bit length L affects both the retrieval efficiency and accuracy, we provide the results of DSSH using various bit lengths, denoted by DSSH-L, where L=16, 64, 256, and 512.

The comparison results are summarized in Tables 1 and 2. Generally, the performance of deep learning based methods is superior to hand-crafted ones. Due to the quantization loss, the accuracies of hashing methods are usually lower than non-hashing based ones. Despite this, our DSSH with 512 bits achieves higher performance than the best-performing non-hashing based models. Even with extremely short bits (*e.g.*, 16 bits), DSSH still performs competitively to existing works. Note that DSSH with ResNet-50 performs slightly worse than Inception-ResNet-v2. However, DSSH (ResNet) outperforms the deep models based on the same backbone, such as DCML (ResNet), LWBR (ResNet) and DCA. This is because: 1) DSSH designs an effective deep shape model to learn 3D representations by efficiently exploring its 2D projections. By segmented stochastic sampling, S^3N learns 3D features from a set of 2D images with more view variations than the compared projection-based methods, making the learned features more discriminative; 2) the Batch-Hard Binary Coding module mines the hardest samples, and learns semantics-preserving binary codes for both sketches and 3D shapes, which can significantly reduce the binary quantization loss.

We also show the precision-recall curves of DSSH with 16 and 512 bits in Figs. 3(a) and 3(b). As illustrated, the precision rates of DSSH-512 are higher than the compared approaches when the recall rate is less than 0.9, by either using ResNet-50 or Inception-ResNet-v2 as the backbone.

Table 2. Performance comparisons with the state-of-the-art methods on PART-SHREC'14.

Methods	NN	FT	ST	E	DCG	mAP
Siamese [50]	0.118	0.076	0.132	0.073	0.400	0.067
Semantic [37]	**0.840**	0.634	0.745	0.526	0.848	0.676
DSSH-16	0.810	0.748	0.796	0.594	0.866	0.759
DSSH-64	0.821	0.766	0.830	0.615	0.880	0.792
DSSH-256	0.835	**0.778**	0.846	0.619	0.886	0.803
DSSH-512	0.838	0.777	**0.848**	**0.624**	**0.888**	**0.806**

Efficiency Analysis. As previously mentioned, by learning binary representations, our DSSH is much more efficient than existing non-hashing based methods. To verify this, we report the average query time per sketch, by computing the similarities between one sketch and all 3D shapes (1,258 shapes on SHREC'13 and 8,987 shapes on SHREC'14). Moreover, we compare the memory load for storing all 3D shape data. All experiments are conducted on a PC with Intel Core CPU (2.6GHz) and 16GB RAM. As can be seen from Table 1, DSSH is remarkably more efficient than the compared methods. On both SHREC'13 and SHREC'14, DSSH is at least 100 times faster, whilst requiring much less memory load.

4.5. Comparisons with Hashing Methods

We further compare DSSH with the state-of-the-art hashing approaches, including 1) single view/modality hashing: SDH [42], COSDISH [20], deep model based DHN [59] and DCTQ [32], and 2) cross view/modality hashing: CVH [23], SCM [57], SePH [29], and deep cross-modal hashing DSMH [19]. For fair comparisons, we extract the 1,536-d vectors after the convolutional layers $\mathcal{F}$ of our model as the features for non-deep models. For deep models, they origi-

Table 3. mAPs of hashing methods with various bit lengths.

Methods		SHREC'13			SHREC'14		
		64 bits	128 bits	256 bits	64 bits	128 bits	256 bits
Cross-View	DCMH [19]	0.672	0.715	0.728	0.658	0.695	0.711
	SePH [30]	0.498	0.547	0.589	0.476	0.524	0.541
	SCM [57]	0.364	0.526	0.485	0.292	0.456	0.360
	CVH [23]	0.544	0.351	0.150	0.346	0.497	0.277
Single-View	DCTQ [32]	0.741	0.755	0.773	0.713	0.737	0.742
	DHN [59]	0.719	0.723	0.731	0.669	0.687	0.695
	COSDISH [20]	0.659	0.682	0.735	0.401	0.583	0.713
	SDH [42]	0.383	0.510	0.646	0.479	0.568	0.615
DSSH		**0.849**	**0.853**	**0.855**	**0.815**	**0.821**	**0.826**

Table 4. mAPs by using different view sampling strategies on PART-SHREC'14.

Methods	16 bits	64 bits	256 bits	512 bits
DSSH (horizontal w/o alignment: Fig. 1 (c))	0.676	0.734	0.742	0.749
DSSH (horizontal: Fig. 1 (b))	0.711	0.757	0.776	0.777
DSSH (stochastic: Fig. 1 (d))	**0.759**	**0.792**	**0.803**	**0.806**

Table 5. Effect of the view attention network on PART-SHREC'14.

Methods	NN	FT	ST	E	DCG	mAP
DSSH (w/o $\mathcal{A}$)	0.815	0.771	0.842	0.615	0.884	0.799
DSSH (with $\mathcal{A}$)	**0.838**	**0.777**	**0.848**	**0.624**	**0.888**	**0.806**

nally use relatively simple base networks such as AlexNet. We re-implement these methods and replace their base networks by Inception-ResNet-v2.

Table 3 shows the mAPs of various methods with different bit lengths. Clearly, DSSH significantly outperforms all non-deep hashing approaches. DSSH also achieves higher mAPs than other deep models, with at least 8% improvements over the second best ones on both datasets.

4.6. Ablation Study

Effect of the stochastic sampling strategy. To evaluate the effect of the proposed sampling strategy, we adopt two other sampling strategies for comparison, *i.e.*, **C1**: 12 rendering views are selected in the horizontal plane for the original aligned shape data (Fig. 1 (b)); **C2**: Each 3D shape is rotated by a random angle before selecting the 12 rendering views horizontally, which mimics the realistic scenario where shapes lack alignment (Fig. 1 (c)). From Table 4, we can see that the proposed stochastic sampling achieves the best performance. We also observe that the performance of DSSH with **C1** significantly drops when 3D shapes are rotated randomly, *i.e.*, without alignment.

Effect of the view attention network. As previously described, the view attention network $\mathcal{A}$ is employed to explore the view-specific weights. Table 5 demonstrates the results of DSSH with or without $\mathcal{A}$. It is clear that the attention network improves the performance of DSSH, especially w.r.t. the Nearest Neighbor (NN) matching accuracy.

Effect of the sampling times during test. As illustrated in Fig. 2, we generate K ($K = 4$) rendered images for a 3D shape during one sampling, based on which we learn one feature vector by DSSH. During test, we can sample t times and use the averaged feature vector as the final representation. As illustrated in Fig. 5, DSSH can achieve better performance with more sampling times, since more rendered images of a 3D shape provide more comprehensive information. Meanwhile, DSSH can already achieve competitive performance based on a single sampling (*i.e.*, using 4 rendered images). In all our experiments, we set t to 3 for fair comparisons, considering most projection-based approaches render a 3D shape to 12 2D images.

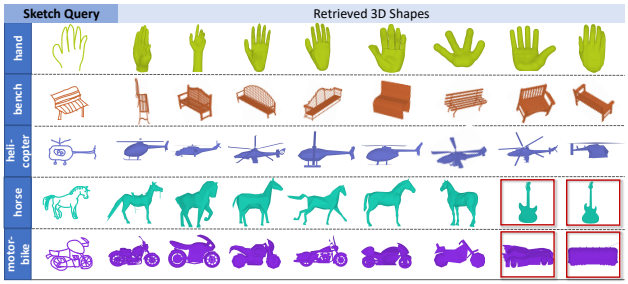


Figure 4. Top-ranked 3D shapes of some sketch queries by using DSSH (with 16 bits) on SHREC'13.

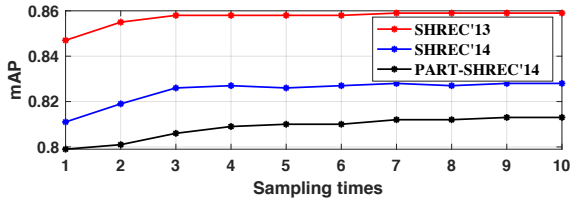


Figure 5. mAPs of DSSH with different sampling times for test.

Qualitative results. We also visually depict some retrieval results by using DSSH on SHREC'13 in Fig. 4. Failure cases are highlighted in the red rectangles.

5. Conclusion

In this paper, we have proposed a hashing based framework, namely Deep Sketch-Shape Hashing (DSSH), for efficient and accurate sketch-based 3D shape retrieval. A novel shape network with Segmented Stochastic-viewing was specially developed to learn discriminative representations of 3D shapes. Moreover, a Batch-Hard Binary Coding scheme was presented to learn semantics-preserving binary codes across modalities, which can diminish the binary quantization loss. Experimental results demonstrated that DSSH remarkably improved the retrieval accuracies of existing works, whilst significantly reducing the time costs and memory load for retrieval.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Project 61502081, Sichuan Science and Technology Program (No.2019YFG0003, 2018GZDZX0032), and Open Fund Project of Fujian Provincial Key Laboratory of Information Processing and Intelligent Control (Minjiang University) (No. MJUKF201708).

References

[1] S. Bai, X. Bai, Q. Tian, and L. J. Latecki. Regularized diffusion process on bidirectional context for object retrieval. *TPAMI*, 2019.

[2] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. Jan Latecki. Gift: A real-time and scalable 3d shape search engine. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5023–5032, 2016.

[3] S. Bai, Z. Zhou, J. Wang, X. Bai, L. Jan Latecki, and Q. Tian. Ensemble diffusion for retrieval. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 774–783, 2017.

[4] X. Bai, S. Bai, Z. Zhu, and L. J. Latecki. 3d shape matching via two layer coding. *IEEE transactions on pattern analysis and machine intelligence*, 37(12):2361–2373, 2015.

[5] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *CVPR*, 2010.

[6] Y. Cao, M. Long, B. Liu, J. Wang, and M. KLiss. Deep cauchy hashing for hamming space retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1229–1237, 2018.

[7] J. Chen and Y. Fang. Deep cross-modality adaptation via semantics preserving adversarial learning for sketch-based 3d shape retrieval. In *European Conference on Computer Vision*, pages 624–640, 2018.

[8] J. Chen, Y. Wang, J. Qin, L. Liu, and L. Shao. Fast person re-identification via cross-camera semantic binary transformation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.

[9] J. Chen, Y. Wang, and R. Wu. Person re-identification by distance metric learning to discrete hashing. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 789–793. IEEE, 2016.

[10] G. Dai, J. Xie, and Y. Fang. Deep correlated holistic metric learning for sketch-based 3d shape retrieval. *IEEE Transactions on Image Processing*, 2018.

[11] G. Dai, J. Xie, and Y. Fang. Siamese cnn-bilstm architecture for 3d shape representation learning. In *IJCAI*, pages 670–676, 2018.

[12] G. Dai, J. Xie, F. Zhu, and Y. Fang. Deep correlated metric learning for sketch-based 3d shape retrieval. In *AAAI*, pages 4002–4008, 2017.

[13] G. Ding, Y. Guo, and J. Zhou. Collective matrix factorization hashing for multimodal data. In *CVPR*, 2014.

[14] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa. Sketch-based shape retrieval. *ACM Trans. Graph.*, 31(4):31–1, 2012.

[15] T. Furuya and R. Ohbuchi. Ranking on cross-domain manifold for sketch-based 3d model retrieval. In *Cyberworlds (CW), 2013 International Conference on*, pages 274–281. IEEE, 2013.

[16] T. Furuya and R. Ohbuchi. Hashing cross-modal manifold for scalable sketch-based 3d model retrieval. In *3D Vision (3DV), 2014 2nd International Conference on*, volume 1, pages 543–550. IEEE, 2014.

[17] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, 2011.

[18] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[19] Q.-Y. Jiang and W.-J. Li. Deep cross-modal hashing. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 3270–3278. IEEE, 2017.

[20] W. C. Kang, W. J. Li, and Z. H. Zhou. Column sampling based discrete supervised hashing. In *AAAI*, 2016.

[21] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[22] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *Proc. Neural Inf. Process. Syst.*, 2009.

[23] S. Kumar and R. Udupa. Learning hash functions for cross-view similarity search. In *IJCAI proceedings-international joint conference on artificial intelligence*, volume 22, page 1360, 2011.

[24] H. Lai, Y. L. Y. Pan, and S. Yan. Simultaneous feature learning and hash coding with deep neural networks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015.

[25] B. Li, Y. Lu, A. Godil, T. Schreck, M. Aono, H. Johan, J. M. Saavedra, and S. Tashiro. *SHREC13 track: large scale sketch-based 3D shape retrieval*. 2013.

[26] 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.

[27] B. Li, Y. Lu, H. Johan, and R. Fares. Sketch-based 3d model retrieval utilizing adaptive view clustering and semantic information. *Multimedia Tools and Applications*, 76(24):26603–26631, 2017.

[28] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, H. Fu, T. Furuya, H. Johan, et al. Shrec14 track: Extended large scale sketch-based 3d shape retrieval. In *Eurographics workshop on 3D object retrieval*, volume 2014, 2014.

[29] G. Lin, C. Shen, Q. Shi, A. van den Hengel, and D. Suter. Fast supervised hashing with decision trees for high-dimensional data. In *CVPR*, 2014.

[30] Z. Lin, G. Ding, M. Hu, and J. Wang. Semantics-preserving hashing for cross-view retrievalg. In *CVPR*, 2015.

[31] R. Litman, A. Bronstein, M. Bronstein, and U. Castellani. Supervised learning of bag-of-features shape descriptors using sparse coding. In *Computer Graphics Forum*, volume 33, pages 127–136. Wiley Online Library, 2014.

[32] B. Liu, Y. Cao, M. Long, J. Wang, and J. Wang. Deep triplet quantization. *MM, ACM*, 2018.

[33] L. Liu, F. Shen, Y. Shen, X. Liu, and L. Shao. Deep sketch hashing: Fast free-hand sketch-based image retrieval. In *Proc. CVPR*, pages 2862–2871, 2017.

[34] W. Liu, J. Wang, R. Ji, Y. G. Jiang, and S. F. Chang. Supervised hashing with kernels. In *CVPR*, 2012.

[35] M. Norouzi and D. M. Blei. Minimal loss hashing for compact binary codes. In *ICML*, 2011.

[36] B. T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975.

[37] A. Qi, Y. Song, and T. Xiang. Semantic embedding for sketch-based 3d shape retrieval. In *British Machine Vision Conference*, 2018.

[38] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4, 2017.

[39] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016.

[40] J. Qin, L. Liu, L. Shao, F. Shen, B. Ni, J. Chen, and Y. Wang. Zero-shot action recognition with error-correcting output codes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2833–2842, 2017.

[41] J. M. Saavedra, B. Bustos, M. Scherer, and T. Schreck. Stela: sketch-based 3d model retrieval using a structure-based local approach. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, page 26. ACM, 2011.

[42] F. Shen, C. Shen, W. Liu, and H. T. Tao. Supervised discrete hashing. In *CVPR*, 2015.

[43] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The princeton shape benchmark. In *Shape modeling applications, 2004. Proceedings*, pages 167–178. IEEE, 2004.

[44] P. Sousa and M. J. Fonseca. Sketch-based retrieval of drawings using spatial proximity. *Journal of Visual Languages & Computing*, 21(2):69–80, 2010.

[45] H. Su, S. Mashaji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[46] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[47] H. Tabia and H. Laga. Learning shape retrieval from different modalities. *Neurocomputing*, 253:24–33, 2017.

[48] F. P. Tasse and N. Dodgson. Shape2vec: semantic-based descriptors for 3d shapes, sketches and images. *ACM Transactions on Graphics (TOG)*, 35(6):208, 2016.

[49] A. Tatsuma, H. Koyanagi, and M. Aono. A large-scale shape benchmark for 3d object retrieval: Toyohashi shape benchmark. In *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pages 1–10. IEEE, 2012.

[50] F. Wang, L. Kang, and Y. Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 1875–1883. IEEE, 2015.

[51] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[52] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[53] J. Xie, G. Dai, F. Zhu, and Y. Fang. Learning barycentric representations of 3d shapes for sketch-based 3d shape retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 3615–3623. IEEE, 2017.

[54] J. Xie, G. Dai, F. Zhu, E. K. Wong, and Y. Fang. Deepshape: deep-learned shape descriptor for 3d shape retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1335–1345, 2017.

[55] Z. Yasseen, A. Verroust-Blondet, and A. Nasri. View selection for sketch-based 3d model retrieval using visual part shape description. *The Visual Computer*, 33(5):565–583, 2017.

[56] G.-J. Yoon and S. M. Yoon. Sketch-based 3d object recognition from locally optimized sparse features. *Neurocomputing*, 267:556–563, 2017.

[57] D. Zhang and W. J. Li. Large-scale supervised multimodal hashing with semantic correlation maximization. In *AAAI*, 2014.

[58] F. Zhu, J. Xie, and Y. Fang. Learning cross-domain neural networks for sketch-based 3d shape retrieval. In *AAAI*, pages 3683–3689, 2016.

[59] H. Zhu, M. Long, J. Wang, and Y. Cao. Deep hashing network for efficient similarity retrieval. In *AAAI*, pages 2415–2421, 2016.