Deep Sketch-Shape Hashing with Segmented 3D Stochastic Viewing

Jiaxin Chen†, Jie Qin†, Li Liu†, Fan Zhu‡, Fuming Shen‡, Jin Xie§ and Ling Shao‡
†Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, UAE
‡University of Electronic Science and Technology of China, Chengdu, China
§Nanjing University of Science and Technology, Nanjing, China

chenjiaxin@gmail.com, qinjiebuua@gmail.com, liulil213@gmail.com, fanzhu1987@gmail.com
fumin.shen@gmail.com, csjxie@njust.edu.cn, ling.shao@ieee.org

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. First, 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 learned seamlessly in an end-to-end manner, and we develop an alternative iteration algorithm to address the joint optimization problem. Experimental results on three benchmarks clearly show that DSSH improves both the retrieval efficiency and the 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 [36, 10, 22, 45, 52, 8, 47, 43, 33, 5], 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 [43, 47, 5] or developing ranking/distance metrics robust to cross-modality variations [45, 11, 8, 23, 33]. 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 [40], 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 [45, 7]. 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 [7]. 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 Net-

1Though shapes can be aligned beforehand, 3D shape alignment is non-trivial and will induce considerable computational time additionally.
work ($S^3N$), to tackle the above challenge of 3D shape representation. $S^3N$ randomly samples rendering views from the sphere around a 3D shape. Concretely, it first divides the 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, $S^3N$ 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, $S^3N$ 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. In addition, 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 learning the above two modules and the Sketch Network (as shown in Fig. 2) in an end-to-end manner, 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 end-to-end 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., $S^3N$) is proposed for learning effective 3D representations, even when 3D shapes lack alignment. $S^3N$ represents a 3D shape as a set of 2D images rendered from segmented stochastic rendering views. Furthermore, $S^3N$ 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, $S^3N$, 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 [27, 2, 46, 1, 40, 35, 48, 7]. 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 [24, 22, 49, 50, 52, 23], or designing effective matching methods across modalities [11, 49, 21, 23]. Recently, a variety of deep learning based approaches have been proposed for joint representation learning and matching [45, 52, 8, 48, 43, 5, 33]. In [45, 8, 47], discriminative representations for both sketches and 3D shapes were learned by two Siamese CNNs. In [52], the cross-domain neural networks with pyramid structures were presented to mitigate cross-domain divergences. [5] addressed the same issue by developing a Generative Adversarial Networks based deep adaptation model. In [23], 3D shape representations were learned by PointNet [34], 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 efficiency issues, such as time costs and memory load.

Learning-based Hashing. Hashing/binar y coding [18, 31, 13, 30, 25, 9, 20, 37, 16, 4, 28] has been extensively studied recently, due to its promising performance in processing large-scale data. Among various approaches, cross-modality hashing [3, 19, 26, 15, 29], 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. [12] discussed the semi-supervised hashing based 3D model retrieval, by applying the existing ITQ [13] 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
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 (S³N). SN encodes sketches into compact representations via convolutional layers and hash layers. Besides the above layers, S³N 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 S³N for the task of efficient sketch-based 3D shape retrieval.

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 [32]. A rendering view is determined by a 3-d vector \((\phi, \theta)\). 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 \((r = 1.5\) in our work), which means the virtual camera views are actually located on a sphere \(S\) 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 \(S\) into \(K\) segments \(\{S_1, S_2, \cdots, S_K\}\) with equal spatial coverage. For instance, if \(K = 4\), we can split...
\[ S \] into the following four segments:
\[
\begin{align*}
S_1 & = \{ (\phi, \theta) | \phi \in [0, \pi); \theta \in [0, \pi/2) \}; \\
S_2 & = \{ (\phi, \theta) | \phi \in [\pi, 2\pi); \theta \in [0, \pi/2) \}; \\
S_3 & = \{ (\phi, \theta) | \phi \in [0, \pi); \theta \in [\pi/2, \pi) \}; \\
S_4 & = \{ (\phi, \theta) | \phi \in [\pi, 2\pi); \theta \in [\pi/2, \pi) \}.
\end{align*}
\]
After segmenting the sphere, we select one random rendering view \((\phi_k, \theta_k)\) from each segment \(S_k\), and finally obtain \(K\) views \(\{ (\phi_k, \theta_k) \}_{k=1}^{K}\). Thereafter, a given shape \(M\) can be represented by an image set with stochastic rendering views as follows:
\[
I_M = \{ \mathcal{R}(M; \phi_k, \theta_k) | (\phi_k, \theta_k) \sim U(S_k) \}_{k=1}^{K},
\]
where \(\mathcal{R}(M; \phi_k, \theta_k)\) indicates the rendering operation on \(M\) based on the viewpoint \((\phi_k, \theta_k)\), and \(U(S_k)\) is the uniform distribution on \(S_k\).

In practice, we employ a batch-wise training process so that each 3D shape is included multiple times. Supposing that \(M\) is selected \(T_M\) times, a sequence of rendering views \(V_M = \{ (\phi_k(t), \theta_k(t)) \}_{t=1}^{T_M} \) 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 \(S_k\) during the \(t\)-th sampling. Correspondingly, \(M\) is modeled by a sequence of image snippets \(I_M = \{ I_M(t) \}_{t=1}^{T_M}\), where \(I_M(t)\) is generated based on the sampled views \(\{ (\phi_k(t), \theta_k(t)) \}_{k=1}^{K}\) during the \(t\)-th sampling. Thereafter, \(S^3N\) learns the representation of a 3D shape \(M\) from the sequence \(\{ I_M(t) \}_{t=1}^{T_M}\).

We have the following intuitive observations w.r.t. \(S^3N\):
1) When the number of spatial segments is small (e.g., \(K=4\)), a 3D shape \(M\) is modeled by a small image set, leading to high computational efficiency.

2) We choose non-overlapping spatial segments \(S_1, \ldots, S_4\) that cover the entire sphere \(S\) 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 \(M\), making the rendered image set always informative for sketch-based 3D shape retrieval.

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

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

- **View Attention Network.** To fully exploit complementary information across different views, we propose a view attention network \(A(\cdot)\) to capture view-specific weights for the features \(\{ f_k^2 \}_{k=1}^{K}\) from \(K\) rendering views. For computational convenience, the scalar \(\phi_k\) is encoded into a \(360\)-d one-hot vector \(\phi_k \in \mathbb{R}^{360}\), of which the \(i\)-th element is set to 1 if \((i-1) \times \pi/180 \leq \phi_k < i \times \pi/180\), and 0 otherwise. In a similar manner, \(\theta_k\) 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, \(A(\cdot)\) also takes the feature weight \(w_k = F^2(I_{\phi_k, \theta_k}) \in \mathbb{R}^d\) as the input to learn shape-dependent weights. As a result, the concatenated vector \(a_k = [f_k^2; \phi_k; \theta_k]^T\) is fed into \(A(\cdot)\), which then outputs a weight \(w_k = A(a_k) \in (0, 1)\) for \(f_k^2\).

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

Note that, \(S^3N\) 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, \(S^3N\) 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 [5], 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 \(I_1 = \{ I_{11}, I_{12}, \ldots, I_{1K} \}\) and \(M = \{ M_1, M_2, \ldots, M_N \}\), respectively. Their corresponding class labels are denoted as \(Y^1 = \{ y_{11}, y_{12}, \ldots, y_{1K} \}\) and \(Y^2 = \{ y_{21}, y_{22}, \ldots, y_{2N} \}\).

By passing \(I^1\) through the Sketch Network, we can extract the feature vectors for sketches: \(F^1 = [f_{11}^1, \ldots, f_{1N}^1]^T \in \mathbb{R}^{N \times L}\). Here, \(f_{i}^1 = H^1(F^1(I_{i}))\), where \(F^1\) and \(H^1\) denote the functions of convolutional and hash layers with regard to sketches, respectively. Similarly, given a batch \(M\) of 3D shapes, we can extract the features \(F^2 = [f_{11}^2, \ldots, f_{N}^2]^T \in \mathbb{R}^{N \times L}\), where \(f_{i}^2 = H^2(F^2(R(M_i)))\).

We compute the binary representations \(B^1 = [b_{11}^1, \ldots, b_{1N}^1]^T \in \{-1, 1\}^{N \times L}\) for sketches and \(B^2 = [b_{11}^2, \ldots, b_{N}^2]^T \in \{-1, 1\}^{N \times L}\) for 3D shapes as follows:

\[
b_m^i = \text{sgn}(f_{i}^m), \quad \text{for } m \in \{1, 2\} \text{ and } i = 1, \ldots, N,
\]
where \( \text{sgn}(\cdot) \) 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 the binary code \( b^{m}_i \) of the \( i \)-th sample from the \( m \)-modality, the ‘hardest positive’ code \( b^{m,\ast}_{i,p} \) and the ‘hardest negative’ code \( b^{m,\ast}_{i,n} \) within the batch \( B^{\hat{m}} \) from the \( \hat{m} \)-th modality \((m, \hat{m} \in \{1, 2\})\) are exploited by

\[
p^* = \max_{p=1, \ldots, N; y^p_0 = y^m_0} d_h(b^m_i, b^m_p),
\]

\[
m^* = \arg \min_{n=1, \ldots, N; y^m_0 \neq y^m_n} d_h(b^m_i, b^m_n),
\]

(4)

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

We aim to learn the binary codes \( B^m \) by minimizing \( d_h(b^m_i, b^{m,\ast}_{i,p}) \), whilst maximizing \( d_h(b^m_i, b^{m,\ast}_{i,n}) \) within a large margin \( \eta_0 \). To this end, the loss function \( L_{BC} \) w.r.t. BHBC is defined as

\[
\sum_{m, \hat{m}, i} \left[ \eta \cdot d_h(b^m_i, b^{m,\ast}_{i,n}) - d_h(b^m_i, b^{m,\ast}_{i,p}) \right]_+,
\]

(5)

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

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

\[
\sum_{m, \hat{m}, i} \left[ \eta_0 \cdot d_h(b^m_i, b^{m,\ast}_{i,n}) - d_h(b^m_i, b^{m,\ast}_{i,p}) \right],
\]

where \( \eta_0 \) is the margin, and \([\cdot]_+ = \max(\cdot, 0)\).

Based on the above equation and the fact that \( d_h(b_i, b_j) = \frac{1}{2} \left\| b_i - b_j \right\|_2 \), we finally obtain

\[
L_{BC} := \sum_{m, \hat{m}, i} \sigma_{m, \hat{m}, i}^T (b^m_i - b^{m,\ast}_{i,n})^T.
\]

(6)

### 3.3. Joint Objective Function

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

\[
\min_{W, B^1, B^2} L_{DSSH} := L_D + \lambda_1 \cdot L_C + \lambda_2 \cdot L_Q + L_{BC},
\]

(7)

where \( W \) indicates the parameters of DSSH, \( L_D \) is the loss for learning discriminative features, \( L_C \) is the loss for enhancing semantic correlations between \( F^1 \) and \( F^2 \), \( L_Q \) is the binary quantization loss, \( L_{BC} \) is the loss function for BHBC, and \( \lambda_1, \lambda_2 > 0 \) are trade-off parameters. Since \( L_{BC} \) is introduced in the above subsection, we will present the remaining three loss functions in detail, respectively.

1) \( L_D \). We adopt a similar loss function as BHBC, i.e., batch-hard triplet loss \([13]\). Specifically, given the \( i \)-th sample \( f^m_i \) from the \( m \)-th modality, we first explore its ‘hardest positive’ samples \( f^m_{i,p} \) and ‘hardest negative’ samples \( f^m_{i,n} \), within the batch \( F^m \), from the \( \hat{m} \)-th modality, as follows:

\[
p^* = \arg \max_{p=1, \ldots, N; y^p_0 = y^m_0} d(f^m_i, f^m_p),
\]

\[
m^* = \arg \max_{n=1, \ldots, N; y^m_0 \neq y^m_n} d(f^m_i, f^m_n),
\]

(8)

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

From Eq. (8), we can see that \( f^m_{i,p} \) is the sample that has the maximal intra-class distance to \( f^m_i \), and \( f^m_{i,n} \) has the minimal inter-class distance to \( f^m_i \). \( L_D \) is then formulated as

\[
\sum_{m, \hat{m}, i=1, \ldots, N} \frac{1}{4N} \left( \eta - d(f^m_i, f^m_{i,n}) + d(f^m_i, f^m_{i,p}) \right),
\]

(9)

where \( \eta > 0 \) is the margin.

By minimizing \( L_D \), the maximal intra-class distance will decrease to a value smaller than the minimal inter-class distance within a margin \( \eta \). This indicates that DSSH can learn discriminative features based on \( L_D \).

2) \( L_C \). We first define the likelihood that \( f^1_i \) and \( f^2_j \) belong to the same class as \( \tilde{p}_{i,j} = \tilde{p}(y^1_i = y^2_j | f^1_i, f^2_j) = 1/(1 + e^{-(f^1_i - f^2_j)}) \), where \( f^1_i \) and \( f^2_j \) are the features of the \( i \)-th sketch and the \( j \)-th 3D shape, respectively. We compute \( L_C \) as the weighted cross-entropy between \( \tilde{p}(y^1_i = y^2_j | f^1_i, f^2_j) \) and the ground-truth probability \( p_{i,j} \) as follows:

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

(10)

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

3) \( L_Q \). We simply employ the quantization loss as

\[
L_Q = \frac{1}{2} \left\| F^1 - B^1 \right\|_F^2 + \left\| F^2 - B^2 \right\|_F^2,
\]

(11)

where \( \| \cdot \|_F \) indicates the matrix Frobenius norm.

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

In order to solve problem \(\{4\}\), we develop an optimization method based on alternative iteration. Specifically, we learn the binary codes \(\mathbf{B}^1\) and \(\mathbf{B}^2\) by fixing the parameters \(\mathbf{W}\) of the whole network. Subsequently, we update \(\mathbf{W}\) by diminishing \(\mathcal{L}_{\text{DSSH}}\) with fixed \(\mathbf{B}^1\) and \(\mathbf{B}^2\). We alternatively optimize \(\{4\}\) based on these two steps until convergence.

1) B-Step. When \(\mathbf{W}\) is fixed, \(\{4\}\) turns into

\[
\min_{\mathbf{v} \in \{-1, 1\}^n} \mathbf{v}^T \left[ \sum_{\bar{n} \in \{1, 2\}} \mathcal{L}_{i,n} \cdot (\hat{\mathbf{b}}_{i,n}^m - \mathbf{b}_{i,n}^m) \right] + \frac{\lambda_2}{2} \cdot \| \mathbf{f}^m - \mathbf{b}^m \|.
\]

Based on the fact that \(\mathbf{v}^T \mathbf{v} = L\), the loss of \(\{13\}\) is equivalent to

\[
\mathbf{v}^T \left[ \sum_{\bar{n} \in \{1, 2\}} \mathcal{L}_{i,n} \cdot (\hat{\mathbf{b}}_{i,n}^m - \mathbf{b}_{i,n}^m) \right] + \text{const}.
\]

It is clear that the closed-form solution to \(\{13\}\) is

\[
\hat{\mathbf{b}}_{i,n}^m = \text{sgn} \left( \sum_{\bar{n} \in \{1, 2\}} \mathcal{L}_{i,n} \cdot (\hat{\mathbf{b}}_{i,n}^m - \mathbf{b}_{i,n}^m) + \lambda_2 \cdot \mathbf{f}^m \right).
\]

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

\[
\min_{\mathbf{W}} \mathcal{L}_{\mathbf{W}} := \mathcal{L}_D + \lambda_1 \cdot \mathcal{L}_C + \lambda_2 \cdot \mathcal{L}_Q.
\]

In practice, we adopt the Adam stochastic gradient descent algorithm \(\{17\}\) 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 \(\{21\}\) collects human-drawn sketches and 3D shapes from the Princeton Shape Benchmark (PSB) \(\{33\}\) 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 \(\{24\}\) 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 \(\{33\}\) 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.

4.2. Implementation Details

For the convolutional layers \(\mathcal{F}^1\) and \(\mathcal{F}^2\), we adopt the Inception-ResNet-v2 \(\{41\}\) 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 \(\lambda_1\) and \(\lambda_2\) 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.

4.3. Evaluation Metrics

We utilize the following widely-adopted metrics \(\{22, 8, 47\}\) 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.

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 handcrafted methods \(\{11, 21, 39, 21, 44, 49\}\) and deep learning based ones \(\{45, 42, 8, 6, 47, 8, 47, 43, 5, 33\}\). 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 one. 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 DCM (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, $S^N$ 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 and 4. 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.

### Efficiency Analysis
As previously mentioned, by learning binary representations, our DSSH is much more efficient than 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 DCM (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, $S^N$ 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>NN</th>
<th>FT</th>
<th>ST</th>
<th>E</th>
<th>DCG</th>
<th>mAP</th>
<th>Query Time (sec.)</th>
<th>Memory (KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Siamese</td>
<td>0.118</td>
<td>0.076</td>
<td>0.132</td>
<td>0.073</td>
<td>0.400</td>
<td>0.067</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Semantic</td>
<td>0.840</td>
<td>0.634</td>
<td>0.745</td>
<td>0.526</td>
<td>0.848</td>
<td>0.676</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SHREC'13</th>
<th>SHREC'14</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN</td>
<td>FT</td>
</tr>
<tr>
<td>DSSH-16</td>
<td>0.810</td>
</tr>
<tr>
<td>DSSH-64</td>
<td>0.821</td>
</tr>
<tr>
<td>DSSH-256</td>
<td>0.835</td>
</tr>
<tr>
<td>DSSH-512</td>
<td>0.838</td>
</tr>
</tbody>
</table>

(*) indicates that the results are not reported, or the source codes as well as implementation details are not available.
Table 3. mAPs of hashing methods with various bit lengths.

<table>
<thead>
<tr>
<th>Methods</th>
<th>SHREC’13 64 bits</th>
<th>SHREC’13 128 bits</th>
<th>SHREC’13 256 bits</th>
<th>SHREC’14 64 bits</th>
<th>SHREC’14 128 bits</th>
<th>SHREC’14 256 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-View</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCMH [13]</td>
<td>0.672</td>
<td>0.715</td>
<td>0.728</td>
<td>0.658</td>
<td>0.695</td>
<td>0.711</td>
</tr>
<tr>
<td>SDHM [37]</td>
<td>0.498</td>
<td>0.547</td>
<td>0.589</td>
<td>0.476</td>
<td>0.524</td>
<td>0.541</td>
</tr>
<tr>
<td>SCM [51]</td>
<td>0.364</td>
<td>0.526</td>
<td>0.485</td>
<td>0.292</td>
<td>0.456</td>
<td>0.360</td>
</tr>
<tr>
<td>CVH [19]</td>
<td>0.544</td>
<td>0.351</td>
<td>0.150</td>
<td>0.346</td>
<td>0.497</td>
<td>0.277</td>
</tr>
<tr>
<td>Single-View</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCMH [13]</td>
<td>0.753</td>
<td>0.767</td>
<td>0.782</td>
<td>0.726</td>
<td>0.740</td>
<td>0.748</td>
</tr>
<tr>
<td>DCTQ [28]</td>
<td>0.741</td>
<td>0.735</td>
<td>0.773</td>
<td>0.713</td>
<td>0.737</td>
<td>0.742</td>
</tr>
<tr>
<td>DHCN [3]</td>
<td>0.719</td>
<td>0.723</td>
<td>0.731</td>
<td>0.669</td>
<td>0.687</td>
<td>0.695</td>
</tr>
<tr>
<td>COSDISH [15]</td>
<td>0.659</td>
<td>0.682</td>
<td>0.735</td>
<td>0.401</td>
<td>0.583</td>
<td>0.713</td>
</tr>
<tr>
<td>SDH [29]</td>
<td>0.383</td>
<td>0.510</td>
<td>0.646</td>
<td>0.479</td>
<td>0.569</td>
<td>0.615</td>
</tr>
<tr>
<td>DSSH (horizontal: Fig. 1 (b))</td>
<td>0.849</td>
<td>0.835</td>
<td>0.855</td>
<td>0.815</td>
<td>0.821</td>
<td>0.826</td>
</tr>
<tr>
<td>DSSH (stochastic: Fig. 1 (d))</td>
<td>0.849</td>
<td>0.835</td>
<td>0.855</td>
<td>0.815</td>
<td>0.821</td>
<td>0.826</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Methods</th>
<th>ID</th>
<th>16 bits</th>
<th>64 bits</th>
<th>256 bits</th>
<th>512 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSSH (horizontal)</td>
<td></td>
<td>0.676</td>
<td>0.734</td>
<td>0.742</td>
<td>0.749</td>
</tr>
<tr>
<td>DSSH (stochastic)</td>
<td></td>
<td>0.711</td>
<td>0.757</td>
<td>0.776</td>
<td>0.777</td>
</tr>
<tr>
<td>DSSH (with A)</td>
<td></td>
<td>0.759</td>
<td>0.792</td>
<td>0.803</td>
<td>0.806</td>
</tr>
</tbody>
</table>

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.

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 Segmentated Stochastic-viewing was specially developed to learn discriminative representations of 3D shapes. Moreover, a Batch-Ballard Binary Coding scheme was presented to learn semantics-preserving binary codes across modalities, which can diminish the binary quantization loss. The entire end-to-end framework was learned by an alternative iteration algorithm. Experimental results demonstrated that DSSH remarkably improved the retrieval accuracies of existing works, whilst significantly reducing the time costs and memory load for retrieval.
References


