Deep Shape Representations for 3D Object Recognition

Hamed Ghodrati Asbfroushani

A Thesis
in
The Concordia Institute
for
Information Systems Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy (Information and Systems Engineering) at
Concordia University
Montreal, QC, Canada

October 2017

CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Hamed Ghodrati Asbfroushani

Entitled: Deep Shape Representations for 3D Object Recognition

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information and Systems Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________________ Chair
Dr. Rabin Raut

______________________________ External Examiner
Dr. Prabir Bhattacharya

______________________________ External Examiner to Program
Dr. Hassan Rivaz

______________________________ Internal Examiner
Dr. Jamal Bentahar

______________________________ Internal Examiner
Dr. Nizar Bouguila

______________________________ Thesis Supervisor
Dr. Abdessamad Ben Hamza

Approved by ______________________________
Dr. Chadi Assi, Graduate Program Director

October 23, 2017

____
Dr. Amir Asif, Dean, Faculty of Engineering and Computer Science

Abstract

Deep Shape Representations for 3D Object Recognition

Hamed Ghodrati Asbfroushani, Ph.D.

Concordia University, 2017

Deep learning is a rapidly growing discipline that models high-level features in data as multilayered neural networks. The recent trend toward deep neural networks has been driven, in large part, by a combination of affordable computing hardware, open source software, and the availability of pre-trained networks on large-scale datasets.

In this thesis, we propose deep learning approaches to 3D shape recognition using a multi-level feature learning paradigm. We start by comprehensively reviewing recent shape descriptors, including hand-crafted descriptors that are mostly developed in the spectral geometry setting and also the ones obtained via learning-based methods. Then, we introduce novel multi-level feature learning approaches using spectral graph wavelets, bag-of-features and deep learning. Low-level features are first extracted from a 3D shape using spectral graph wavelets. Mid-level features are then generated via the bag-of-features model by employing locality-constrained linear coding as a feature coding method, in conjunction with the biharmonic distance and intrinsic spatial pyramid matching in a bid to effectively measure the spatial relationship between each pair of the bag-of-feature descriptors.

For the task of 3D shape retrieval, high-level shape features are learned via a deep auto-encoder on mid-level features. Then, we compare the deep learned descriptor of a query shape to the descriptors of all shapes in the dataset using a dissimilarity measure for 3D shape retrieval. For the task of 3D shape classification, mid-level features are represented as 2D images in order to be fed into a pre-trained convolutional neural network to learn high-level features from the penultimate fully-connected layer of the network. Finally, a multiclass support vector machine classifier is trained on these deep learned descriptors, and the classification accuracy is subsequently computed. The proposed 3D shape retrieval and classification approaches are evaluated on three standard 3D shape benchmarks through extensive experiments, and the results show compelling superiority of our approaches over state-of-the-art methods.

Acknowledgements

First and foremost, I would like to express my sincerest gratitude and appreciation to my supervisor, Prof. Ben Hamza, for providing me with the opportunity to work in the area of Deep Learning, for his invaluable guidance and mentorship, and for his encouragement and endless support throughout all levels of my research.

I am very grateful to Concordia University and NSERC, Canada for providing me with financial support through my supervisor's research grants. Without such a support, this thesis would not have been possible.

Finally, I would like to express my love and appreciation to my parents and rest of my family and thank them for their consistent encouragement and care during my doctoral study in Canada.

Table of Contents

CHAPTER 1

Introduction

In this chapter, we present the framework and motivation behind this work, followed by the problem statement, objectives of the study, literature review and thesis contributions.

1.1 Framework and Motivation

The availability of low-cost 3D digitization and acquisition devices, coupled with recent advancements in consumer electronics and computation power, have led to an abundant increase of 3D shape repositories that are easily accessible on-line. The continued growth of these large databases has sparked the need to organize, search and retrieve the most relevant collections. The main challenge in 3D shape analysis is to compute an invariant shape descriptor that captures well the geometric and topological properties of a shape. Hence, this sheer volume of 3D objects publicly available has led to the burgeoning design of a plethora of shape descriptors in the computer vision, graphics and medical imaging literature. These compact descriptors have been the driving force behind the development of efficient algorithms for nonrigid 3D shape retrieval and classification, achieving state-of-the-art performance on the latest benchmarks contests [1–4].

In recent years, spectral geometric methods have been successfully applied to 3D shape retrieval and classification, achieving state-of-the-art performance [5–13]. Most of these approaches are based on the spectral analysis of the Laplace-Beltrami operator (LBO) [14–16], and usually represent a shape by a spectral signature, which is a concise and compact shape descriptor aimed at facilitating the classification and retrieval tasks.

As a branch of the broader discipline of machine learning, deep learning has become a pervasive and wide reaching technology, growing at a breathtaking rate and underlying many modern applications, including internet search, healthcare, marketing, cyber-security, and speech recognition [17]. The success of deep neural networks has been greatly accelerated by using graphics processing units (GPUs), which have become the platform of choice for training large, complex learning systems [3, 18–20]. The most popular deep learning models that have been successfully applied to image data include deep convolutional neural networks, deep auto-encoders, deep belief networks and deep Boltzmann machines [21–33]. Applying such models directly to 3D shapes, particularly to mesh data, is however not straightforward. Fortunately, these technical challenges are not insurmountable, and have been recently tackled using volumetric and view-based deep learning approaches [18–20, 34]. Volumetric deep learning models encode a 3D shape as a 3D tensor of real or binary numbers, while view-based methods encode a 3D shape as a collection of its rendered views on 2D images. The key challenge with volumetric representations is how to deal with the additional computational complexity resulting from the voxelization resolution of 3D shapes. A major drawback of view-based methods is their sensitivity to consistent model orientations, resulting in lower performance [4].

Alternatively, there is another type of 3D deep learning models that rely on extracting discriminative features from 3D shapes in an effort to design a 2D global shape descriptor, which can be used as an input to the deep neural network. In this thesis, we adopt such a strategy in a bid to obtain 3D deep shape descriptors which later on are used for shape retrieval and classification. More specifically, we introduce several multi-level feature learning approaches using spectral graph wavelets, bag-of-features and deep learning models. In particular, we use SGWS as a local descriptor due to its ability to capture different details provided at different levels from low to high frequencies. We also use locality-constrained linear coding (LLC) as a feature coding scheme in the BoF model due largely to the lower quantization error of LLC as well as its codewords locality properly. In addition, we employ the biharmonic distance together with intrinsic spatial pyramid matching (ISPM) to effectively measure the spatial relationship between the LLC codes. Unlike the geodesic distance which is not globally shape-aware, the biharmonic distance is shape-aware, isometry invariant, computationally efficient, robust to various shape deformations, and possesses good discriminative capabilities [12, 35].

1.2 Problem Statement

In this study, we introduce high-level shape descriptors in order to deal with 3D object retrieval and classification problems. Nonrigid shape retrieval and classification are among the most challenging problems in 3D shape analysis.

1.2.1 Shape Retrieval

Given a database of 3D shapes, the objective of 3D shape retrieval is to find a set of shapes that are relevant to a query shape. The retrieval accuracy is usually evaluated by computing a pairwise dissimilarity measure between 3D shapes in the dataset. A good retrieval algorithm should result in few dissimilar shapes. A commonly used dissimilarity measure for content-based retrieval is the ℓ_1-distance, also known as Manhattan or city-block metric, which quantifies the difference between each pair of 3D shapes. The ranked list for each query shape is a set of other shapes in the dataset ranked from best to worst according to their computed distance from the query shape. In order to assess the retrieval performance several standard evaluation metrics are usually used including the precision-recall curve, nearest neighbor (NN), first-tier (FT), second-tier (ST), E-measure (E), discounted cumulative gain (DCG), and mean average precision (mAP). The definition of these evaluation measures are provided in Subsection 1.5.5.

1.2.2 Shape Classification

Shape classification is all about labeling shapes in a dataset and organizing them into a known number of classes so they can be found quickly and efficiently, and the goal is to assign new shapes to one of these classes. In supervised learning tasks, the available data $\mathcal{Z}$ for classification is usually split into two disjoint subsets: the training set $\mathcal{Z}_{\text{train}}$ for learning, and the test set $\mathcal{Z}_{\text{test}}$ for testing. The training and test sets are usually selected by randomly sampling a set of training instances from $\mathcal{Z}$ for learning and using the rest of instances for testing. The performance of a classifier is then assessed by applying it to test data with known target values and comparing the predicted values with the known values.

1.3 Objectives

In this thesis, we propose multi-level feature learning approaches using spectral graph wavelets, bag-of-features and deep learning models. The objective is to obtain high discriminative 3D shape descriptors in order to outperform the state-of-the-art methods that are used for either 3D shape retrieval and classification or both. Most of existing approaches are failed when it comes to dealing with recent challenging benchmarks. However, the proposed approaches show better performance in dealing with challenging datasets comparing to the state-of-the-art methods. The key factor that contributes to the success of our 3D shape descriptors is the benefit from deep learning which is used in the last stage of our feature learning frameworks to extract the most discriminative features. More specifically, we use deep auto-encoder to extract high-level features that are used to design a deep shape-aware descriptor on which retrieval test is performed. We also introduces a deep convolutional shape-aware (Deep-CSA) learning framework for 3D shape classification using a pre-trained convolutional neural network. The aim is to beat the approaches based on hand-crafted descriptors and those obtained by shallow models.

1.4 Literature Review

In recent years, numerous schemes have been proposed to construct shape descriptors in an effort to capture the more discriminative geometric information in a 3D shape [5–13]. Inspired by the success of some descriptors in image retrieval, some works introduced 3D shape descriptor such as SIFT-based [36], Mesh-HOG [37], covariance descriptor [38]. Nevertheless, the overwhelming majority of these works use spectral descriptors, which represent a shape using a concise and compact signature. A comprehensive overview on the available spectral descriptors can be found in [39, 40]. These shape representations may be broadly categorized into local and global descriptors. Local descriptors are defined on each point of the shape. Examples of local descriptors include the global point signature [5], heat kernel signature (HKS) [6], scale-invariant heat kernel signature (SIHKS) [7], wave kernel signature (WKS) [8], improved wave kernel signature (IWKS) [41], and spectral graph wavelet signature (SGWS) [9].

Global descriptors, on the other hand, are defined on the entire shape. One of the simplest global descriptors is Shape-DNA [42], which is defined as a truncated sequence of the LBO eigenvalues arranged in increasing order of magnitude. Gao *et al.* [43] introduced compact Shape-DNA (cShape-DNA) as a variant of Shape-DNA, which is an isometry-invariant signature obtained by applying the discrete Fourier transform to the area-normalized eigenvalues of the LBO. Chaudhari *et al.* [11] introduced a new version of the GPS signature by setting the LBO eigenfunctions to unity. Ye *et al.* [12] proposed a global descriptor for nonrigid shape retrieval using a reduced biharmonic distance matrix. A graph-theoretic approach has been introduced in [44] for 3D shape classification using graph regularized sparse coding together with the biharmonic distance map. Unlike the above methods, SD-GDM [45] proposed to compute a singular value decomposition as spectrum of the geodesic distance matrix. However, compared to other spectral descriptors developed based upon the eigensystem (eigenvalues and/or eigenfunctions) of LBO as spectrum, SD-GDM relies on all-pairs geodesic distances, which are computationally prohibitive to obtain even with the latest advances in fast geodesic distance computation [46].

On the other hand, the bag-of-features (BoF) model, which has shown significant levels of performance in text and image retrieval, is also commonly used to construct global descriptors by aggregating the local ones. In its simplest form, the BoF model quantizes each local descriptor to its nearest cluster center using K-means clustering and then encodes each shape as a histogram over cluster centers by counting the number of assignments per cluster. These cluster centers form a codebook whose elements are often referred to as codewords. Although the BoF paradigm has been shown to provide significant levels of performance, it does not, however, take into consideration the spatial relations between features, which may have an adverse effect not only on its descriptive ability but also on its discriminative power. To sidestep this issue, various solutions have been proposed including the spatially-sensitive bags-of-features (SS-BoF) [47], supervised learning of BoF shape descriptors using sparse coding [48], and geodesic-aware bags-of-features (GA-BoF) [49]. The SS-BoF, which is defined in terms of the heat kernel, can be represented by a square matrix whose elements represent the frequency of appearance of nearby codewords in the vocabulary. Similarly, the GA-BoF matrix is obtained by replacing the heat kernel in the SS-BoF with a geodesic exponential kernel.

Although the geodesic distance has proven to be effective in geometry processing due in large part to its isometry invariance property, it suffers, however, from several practical issues compared to the (squared) biharmonic distance [35]. While the geodesic distance is sensitive to topological noise and not globally shape-aware, the biharmonic distance is not only robust to noise and small topological changes, but also globally shape-aware and smooth. Our work builds upon the BoF framework to design a discriminative, shape-aware representation for 3D object classification and retrieval using the biharmonic distance in conjunction with deep neural networks [50].

Another issue with BoF model is that the codebook is usually constructed in an unsupervised manner using clustering, agnostic to the last step of the process which involves in pooling of the local descriptors into a BoF. To tackle this issue, Litman *et al.* [48] proposed to replace clustering with a dictionary (codebook) learning approach coupled with sparse coding as a feature coding method. As a result, their learned BoFs have obtained in a supervised manner, being aware of feature pooling which is the last stage of BoF paradigm.

The recent trend in 3D shape analysis is to use deep learning models to learn high-level features of 3D shapes. Deep learning, which involves training neural networks on lots of data and then having them make predictions about new data, has been making big waves over the past several years due largely to its great success in computer vision, natural language processing and speech understanding. Deep learning models have recently been applied to 3D shape analysis to learn high-level features from 3D shapes. Wu *et al.* [18] proposed a deep learning framework for volumetric shapes via a convolutional deep belief network by representing a 3D shape as a probabilistic distribution of binary variables on a 3D voxel grid. Brock *et al.* [51] proposed a voxel-based approach to 3D object classification using variational autoencoders and deep convolutional neural networks, achieving improved classification performance on the ModelNet benchmark. Sedaghat *et al.* [52] showed that forcing the convolutional neural network to produce the correct orientation during training yields improved classification accuracy. The key challenge with volumetric representations is how to deal with the additional computational complexity resulting from the voxelization resolution of 3D shapes. Zhu *et al.* [20] introduced a a view-based technique by projecting 3D shapes into 2D images and then using an auto-encoder for feature learning. Su *et al.* [34] presented a convolutional neural network architecture that combines information from multiple views of a 3D shape into a single and compact shape descriptor. Qi *et al.* [19] proposed a multiresolution filtering strategy in order to improve the performance of multi-view convolutional neural networks on 3D shape classification. Kanezaki *et al.* [53] introduced RotationNet, a CNN-based framework that uses a set of multi-view images of a 3D object as input for 3D object classification and pose estimation. View-based methods tend to suffer from a relatively long running time due primarily to analyzing a large amount of redundant data provided by multi-view images. Also, a major drawback of view-based methods is their sensitivity to consistent model orientations, resulting in lower performance. Moreover, it is almost impossible in many real-world applications to set up multiple cameras in order to project all required views. Fang *et al.* [54] introduced a deep learning framework in which the heat kernel signature is fed to deep neural networks with target values in a bid to obtain a 3D deep shape descriptor that demonstrated good performance in 3D shape retrieval. Inspired by the Shape Google framework for 3D shape retrieval [47], Bu *et al.* [49] introduced a deep learning based approach (3D-DL) for 3D shape classification and retrieval. The 3D-DL framework uses a 2D global shape descriptor, which is represented by a full matrix defined in terms of the geodesic distance and eigenfunctions of the LBO. The geodesic distance, however, has some major limitations, the most serious of which are the sensitivity to topological noise and the lack of shape-awareness [35]. More recently, Bu *et al.* [55] presented a multi-modal feature learning approach to 3D shape recognition using CNNs and convolutional deep belief networks. This hybrid approach combines both view-based and geometry-based feature learning in an effort to learn a more discriminative shape descriptor by fusing different modalities. Bai *et al.* [56] introduced a real-time 3D shape search engine based on the projective images of 3D shapes. Xie *et al.* [57] proposed a multi-metric deep neural network for 3D shape retrieval by learning non-linear distance metrics from multiple types of shape features, and by enforcing the outputs of different features to be as complementary as possible via the Hilbert-Schmid independence criterion. Tabia *et al.* [58] proposed a 3D shape retrieval framework using queries of different modalities including 3D models, images and sketches. The different features extracted from different modalities are embedded into a common space using a CNN model. Chen *et al.* [59] introduced a multimodal learning approach to view-based 3D object classification that three modalities of image features including SIFT descriptor, Outline Fourier transform descriptor,and Zernike Moments descriptor are combined using a support vector machine. A comprehensive review of deep learning advances in 3D shape recognition can be found in [60].

1.5 Background

A 3D shape is usually modeled as a triangle mesh $\mathbb{M}$ whose vertices are sampled from a Riemannian manifold. A triangle mesh $\mathbb{M}$ may be defined as a graph $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ or $\mathbb{G} = (\mathcal{V}, \mathcal{T})$, where $\mathcal{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_m\}$ is the set of vertices, $\mathcal{E} = \{e_{ij}\}$ is the set of edges, and $\mathcal{T} = \{\mathbf{t}_1, \dots, \mathbf{t}_g\}$ is the set of triangles. Each edge $e_{ij} = [\mathbf{v}_i, \mathbf{v}_j]$ connects a pair of vertices $\{\mathbf{v}_i, \mathbf{v}_j\}$. Two distinct vertices $\mathbf{v}_i, \mathbf{v}_j \in \mathcal{V}$ are adjacent (denoted by $\mathbf{v}_i \sim \mathbf{v}_j$ or simply $i \sim j$) if they are connected by an edge, i.e. $e_{ij} \in \mathcal{E}$.

1.5.1 Laplace-Beltrami Operator

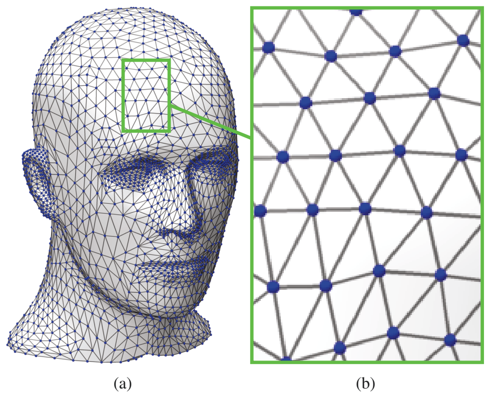
Given a compact Riemannian manifold $\mathbb{M}$, the space $L^2(\mathbb{M})$ of all smooth, square-integrable functions on $\mathbb{M}$ is a Hilbert space endowed with inner product $\langle f_1, f_2 \rangle = \int_{\mathbb{M}} f_1(\mathbf{x}) f_2(\mathbf{x})\, da(\mathbf{x})$, for all $f_1, f_2 \in L^2(\mathbb{M})$, where $da(x)$ (or simply dx) denotes the measure from the area element of a Riemannian metric on $\mathbb{M}$. Given a twice-differentiable, real-valued function $f : \mathbb{M} \to \mathbb{R}$, the Laplace-Beltrami operator (LBO) is defined as $\Delta_{\mathbb{M}} f = -\operatorname{div}(\nabla_{\mathbb{M}} f)$, where $\nabla_{\mathbb{M}} f$ is the intrinsic gradient vector field and div is the divergence operator [14]. The LBO is a linear, positive semi-definite operator acting on the space of real-valued functions defined on $\mathbb{M}$, and it is a generalization of the Laplace operator to non-Euclidean spaces.

Discretization. A real-valued function $f : \mathcal{V} \to \mathbb{R}$ defined on the mesh vertex set may be represented as an m-dimensional vector $\mathbf{f} = (f(i)) \in \mathbb{R}^m$, where the ith component $f(i)$ denotes the function value at the ith vertex in $\mathcal{V}$. Using a mixed finite element/finite volume method on triangle meshes [61], the value of $\Delta_{\mathbb{M}} f$ at a vertex $\mathbf{v}_i$ (or simply i) can be approximated using the cotangent weight scheme as follows:

$$\Delta_{\mathbb{M}} f(i) \approx \frac{1}{a_i} \sum_{j \sim i} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} \big(f(i) - f(j) \big), \tag{1.1}$$

where α_{ij} and β_{ij} are the angles $\angle(\mathbf{v}_i \mathbf{v}_{k_1} \mathbf{v}_j)$ and $\angle(\mathbf{v}_i \mathbf{v}_{k_2} \mathbf{v}_j)$ of two faces $\mathbf{t}^{\alpha} = \{\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_{k_1}\}$ and $\mathbf{t}^{\beta} = \{\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_{k_2}\}$ that are adjacent to the edge $[i, j]$, and a_i is the area of the Voronoi cell (shaded polygon) at vertex i, as shown in Figure 1.1. It should be noted that the cotangent weight scheme is numerically consistent and preserves several important properties of the continuous LBO, including symmetry and positive semi-definiteness.

Spectral analysis. The $m \times m$ matrix associated to the discrete approximation of the LBO is given by $\mathbf{L} = \mathbf{D}^{-1}\mathbf{E}$, where $\mathbf{D} = \operatorname{diag}(d_i)$ is a positive definite diagonal matrix (mass matrix), and $\mathbf{E} = \operatorname{diag}(\sum_{k \neq i} c_{ik}) - (c_{ij})$ is a sparse symmetric matrix (stiffness matrix). Each diagonal element



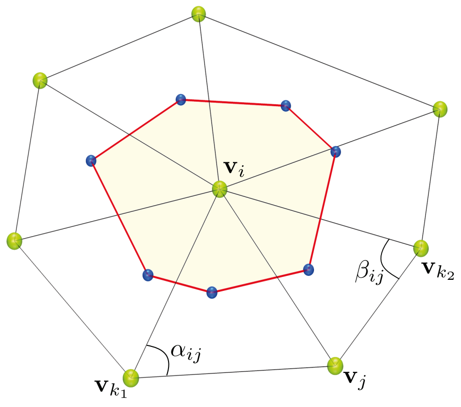


Figure 1.1: Triangular mesh representation (left); Cotangent scheme angles (right).

d_i is the area of the Voronoi cell at vertex i, and the weights c_{ij} are given by

$$c_{ij} = \begin{cases} \dfrac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} & \text{if } i \sim j \\ 0 & \text{o.w.} \end{cases} \tag{1.2}$$

where α_{ij} and β_{ij} are the opposite angles of two triangles that are adjacent to the edge $[i, j]$.

The eigenvalues and eigenvectors of $\mathbf{L}$ can be found by solving the generalized eigenvalue problem $\mathbf{E}\boldsymbol{\varphi}_\ell = \lambda_\ell \mathbf{D}\boldsymbol{\varphi}_\ell$ using for instance the Arnoldi method of ARPACK[1], where λ_ℓ are the eigenvalues and $\boldsymbol{\varphi}_\ell$ are the unknown associated eigenfunctions (i.e. eigenvectors which can be thought of as functions on the mesh vertices). We may sort the eigenvalues in ascending order as $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_m$ with associated orthonormal eigenfunctions $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \ldots, \boldsymbol{\varphi}_m$, where the orthogonality of the eigenfunctions is defined in terms of the $\mathbf{D}$-inner product, i.e.

$$\langle \boldsymbol{\varphi}_k, \boldsymbol{\varphi}_\ell \rangle_{\mathbf{D}} = \sum_{i=1}^{m} d_i \varphi_k(i) \varphi_\ell(i) = \delta_{k\ell}, \;\; \forall k, \ell = 1, \ldots, m. \tag{1.3}$$

We may rewrite the generalized eigenvalue problem in matrix form as $\mathbf{E\Phi} = \mathbf{D\Phi\Lambda}$, where $\mathbf{\Lambda}$ is an $m \times m$ diagonal matrix with the λ_ℓ on the diagonal, and $\mathbf{\Phi}$ is an $m \times m$ orthogonal matrix whose ℓ-th column is the unit-norm eigenvector $\boldsymbol{\varphi}_\ell$. It should be noted that since the first eigenvalue λ_1 is zero, its associated eigenvector $\boldsymbol{\varphi}_1$ is an m-dimensional constant vector given by

$$\boldsymbol{\varphi}_1 = \left(\frac{1}{\sqrt{a}}, \frac{1}{\sqrt{a}}, \ldots, \frac{1}{\sqrt{a}} \right)^{\mathsf{T}}, \tag{1.4}$$

where $a = \text{area}(\mathbb{M})$ is the total area of the mesh.

[1]ARPACK (ARnoldi PACKage) is a MATLAB library for computing the eigenvalues and eigenvectors of large matrices.

1.5.2 Spectral Shape Signatures

In recent years, a great deal of 3D shape descriptors has been proposed using the spectral analysis (based on eigensystem i.e. eigenvalues and/or eigenfunctions) of the LBO such as Shape-DNA [42], global point signature [5], heat kernel signature (HKS) [6], scale-invariant heat kernel signature (SIHKS) [7], wave kernel signature (WKS) [8]. It is important to point out that all of these 3D shape signatures are local descriptors, except Shape-DNA which is a global descriptor. What follows is a terse review on these spectral shape signatures.

Shape-DNA. It is one of the early proposed 3D shape signatures which is a normalized sequence of the first eigenvalues of the LBO. The simple representation (a vector of numbers) and scale invariance are the main advantages of Shape-DNA. Despite its simplicity, the shapeDNA yet has a comparable performance in 3D shape retrieval. However, the Shape-DNA cannot be used for local or partial shape analysis as it is a global descriptor. The Eigenvalue Descriptor(EVD) [62], on the other hand, is a sequence of the eigenvalues of the geodesic distance matrix. Both Shape-DNA and EVD can be normalized by the second eigenvalue.

Global Point Signature. The global point signature (GPS) [5] at a surface point is a vector of scaled eigenfunctions of the LBO. The GPS is a global feature in the sense that it cannot be used for partial shape matching. It is defined in terms of the eigenvalues and eigenfunctions of $\Delta_{\mathbb{M}}$ as follows:

$$\mathrm{GPS}(x) = \left(\frac{\varphi_2(x)}{\sqrt{\lambda_2}}, \frac{\varphi_3(x)}{\sqrt{\lambda_3}}, \dots, \frac{\varphi_i(x)}{\sqrt{\lambda_i}}, \dots \right) \tag{1.5}$$

GPS is invariant under isometric deformations of the shape, but it suffers for the problem of eigenfunctions switching whenever the associated eigenvalues are close to each other.

Heat Kernel Signature. The heat kernel $\mathfrak{p}_t(x, y)$ is an essential solution to the heat equation [63] at point x at time t with initial distribution $u_0(x) = \delta(x - y)$ at point $y \in \mathbb{M}$, and it is defined in terms of the eigenvalues and eigenfunctions of $\Delta_{\mathbb{M}}$ as follows:

$$\mathfrak{p}_t(x, y) = \sum_{i=1}^{\infty} e^{-\lambda_i t} \varphi_i(x) \varphi_i(y) \tag{1.6}$$

Intuitively, $\mathfrak{p}_t(x, y)$ describes the amount of heat that is propagated or transferred from point x to point y in time t. In the same spirit, $\mathfrak{p}_t(x, x)$ describes the amount of heat that remains at point x after time t. For each point $x \in \mathbb{M}$, the Heat Kernel Signature (HKS) [6] is represented in the discrete temporal domain by a n-dimensional feature vector

$$\mathrm{HKS}(x) = (\mathfrak{p}_{t_1}(x, x), \mathfrak{p}_{t_2}(x, x), \dots, \mathfrak{p}_{t_n}(x, x)) \tag{1.7}$$

where $t_1, t_2, \dots, t_n$ are different time-scales.

Scale Invariant Heat Kernel Signature. Let $\mathbb{M}$ and $\mathbb{M}'$ be a shape and its uniformly scaled version by a factor of a, respectively. Denote by $\mathfrak{p}_{\alpha^\tau}(x,y)$ the heat kernel with scale logarithmically sampled using some basis α at each point x. Thus, the heat kernel of the scaled shape becomes $\mathfrak{p}'(\tau) = a^{-2}\mathfrak{p}(\tau + 2\log_\alpha a)$. In order to remove the dependence on the multiplicative constant a^{-2}, the logarithm of the signal is taken and then differentiated with respect to the scale variable [7]:

$$\begin{aligned} \frac{d}{d\tau}\log \mathfrak{p}'(\tau) &= \frac{d}{d\tau}(-2\log a + \log \mathfrak{p}(\tau + 2\log_\alpha a) \\ &= \frac{\frac{d}{d\tau}\mathfrak{p}(\tau + 2\log_\alpha a)}{\mathfrak{p}(\tau + 2\log_\alpha a)}. \end{aligned} \tag{1.8}$$

Let $\mathfrak{p}' = \frac{\frac{d}{d\tau}\mathfrak{p}(\tau)}{\mathfrak{p}(\tau)} = \frac{-\sum_{i\geq 0}\lambda_i \alpha^\tau \log \alpha e^{-\lambda_i\alpha^\tau}\varphi_i^2(x)}{-\sum_{i\geq 0} e^{-\lambda_i\alpha^\tau}\varphi_i^2(x)}$ then a new function $\tilde{\mathfrak{p}}$ which transforms $\tilde{\mathfrak{p}'}(\tau) = \tilde{\mathfrak{p}}(\tau + 2\log_\alpha a)$ as a result of scaling is obtained. The Fourier transform of $\tilde{\mathfrak{p}}$ and its absolute value are given by

$$\begin{aligned} F\left[\tilde{\mathfrak{p}'}\right](\omega) &= \tilde{H}'(\omega) = \tilde{H}(\omega)e^{-j\omega 2\log_\alpha a} \\ |\tilde{H}'(\omega)| &= |\tilde{H}(\omega)|. \end{aligned} \tag{1.9}$$

Thus, the Scale-Invariant Heat Kernel Signature (SIHKS) is defined as

$$\text{SIHKS}(x) = \left(|\tilde{H}(\omega_1)|, |\tilde{H}(\omega_2)|, \ldots, |\tilde{H}(\omega_n)|\right). \tag{1.10}$$

Wave Kernel Signature. The basic idea of the Wave Kernel Signature (WKS) [8] is to describe a point $x \in \mathbb{M}$ by the average probabilities of quantum particles of different energy levels to be measured in x. Assume a quantum particle with unknown position is on the surface. Then the wave function of the particle is the Schrödinger equation solution, which can expressed in the spectral domain as

$$\psi_E(x,t) = \sum_{k=1}^{\infty} e^{i\lambda_k t}\varphi_k(x) f_E(\lambda_k) \tag{1.11}$$

where E denotes the energy of the particle at time $t = 0$ and f_E its initial distribution.

Since $|\psi_E(x,t)|^2$ is the probability to measure the particle at a point x at time t, it follows that the average probability (over time) to measure a particle in x is given by

$$\mathtt{P}_E(x) = \lim_{T\to\infty}\frac{1}{T}\int_0^T |\psi_E(x,t)|^2 = \sum_{k=1}^{\infty}\varphi_k(x)^2 f_E(\lambda_k)^2 \tag{1.12}$$

Let $E_1, E_2, \ldots, E_n$ be n log-normal energy distributions. Then, each point x on the surface $\mathbb{M}$ is associated with a wave kernel signature, which can represented by a n-dimensional feature vector of average probabilities as follows:

$$\text{WKS}(x) = (\mathtt{P}_{e_1}(x), \mathtt{P}_{e_2}(x), \cdots, \mathtt{P}_{e_n}(x)) \tag{1.13}$$

where $e_i = \log E_i$ is the logarithmic energy scale. The WKS represents the average probability of measuring a quantum particle at a specific surface point. Unlike the HKS, the WKS separates influences of different frequencies, treating all frequencies equally. In other words, HKS uses low-pass filters, while WKS uses band-pass filters.

1.5.3 Deep Auto-Encoders

An auto-encoder is a neural network that learns to reproduce its input as its output. It is an unsupervised learning algorithm that learns features from unlabeled data using backpropagation via stochastic gradient descent, and has typically an input layer representing the original data, one hidden layer and an output layer. An auto-encoder is comprised of an encoder and a decoder, as shown in Figure 1.2. The encoder, denoted by $f_{\boldsymbol{\theta}}$, maps an input vector $\mathbf{x} \in \mathbb{R}^d$ to a hidden representation (referred to as code, activations or features) $\mathbf{a} \in \mathbb{R}^r$ via a deterministic mapping

$$\mathbf{a} = f_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}), \tag{1.14}$$

parameterized by $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}\}$, where $\mathbf{W} \in \mathbb{R}^{r \times d}$ and $\mathbf{b} \in \mathbb{R}^d$ are the encoder weight matrix and bias vector, and σ is a nonlinear element-wise activation function such as the logistic sigmoid or hyperbolic tangent. The decoder, denoted by $g_{\boldsymbol{\theta}'}$, maps back the hidden representation $\mathbf{h}$ to a reconstruction $\hat{\mathbf{x}}$ of the original input $\mathbf{x}$ via a reverse mapping

$$\hat{\mathbf{x}} = g_{\boldsymbol{\theta}'}(\mathbf{a}) = \sigma(\mathbf{W}'\mathbf{a} + \mathbf{b}'), \tag{1.15}$$

parameterized by $\boldsymbol{\theta}' = \{\mathbf{W}', \mathbf{b}'\}$, where $\mathbf{W}' \in \mathbb{R}^{d \times r}$ and $\mathbf{b}' \in \mathbb{R}^d$ are the decoder weight matrix and bias vector, respectively. The encoding and decoding weight matrices $\mathbf{W}$ and $\mathbf{W}'$ are usually constrained to be of the form $\mathbf{W}' = \mathbf{W}^{\mathsf{T}}$, which are referred to as tied weights. Assuming the tied weights case for simplicity, the parameters $\{\mathbf{W}, \mathbf{b}, \mathbf{b}'\}$ of the network are often optimized by minimizing the following squared error cost function

$$\mathcal{L}(\mathbf{W}, \mathbf{b}, \mathbf{b}') = \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2, \tag{1.16}$$

where N is the number of samples in the training set, $\mathbf{x}_i$ is the ith input sample and $\hat{\mathbf{x}}_i$ is its reconstruction. To penalize large weight coefficients in an effort to avoid over-fitting the training data, the following objective function is minimized instead

$$\mathcal{L}(\mathbf{W}, \mathbf{b}, \mathbf{b}') = \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 + \frac{\lambda}{2}\|\mathbf{W}\|_F^2, \tag{1.17}$$

where λ is a regularization parameter that determines the relative importance of the sum-of-squares error term and the weight decay term. This parameter should typically be quite small. Note that the features learned from the training data are encapsulated in $\mathbf{W}$.

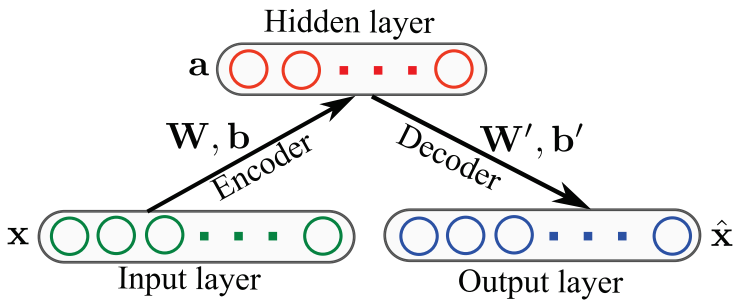


Figure 1.2: Auto-encoder architecture.

An auto-encoder with multiple hidden layers is referred to as a stacked or deep auto-encoder. A stacked auto-encoder is a deep neural network consisting of multiple layers of stacked encoders from several auto-encoders. This stacked network is pre-trained layer by layer in a unsupervised fashion, where the output from the encoder of the first auto-encoder is the input of the second auto-encoder, the output from the encoder of the second auto-encoder is the input to the third auto-encoder, and so on. In other words, the hidden layer of the ℓ-th auto-encoder acts as an input layer to the $(\ell+1)$-th auto-encoder. More formally, the encoding and decoding stages of an L-layer deep auto-encoder having parameters $\mathbf{\Theta} = \{\mathbf{\Theta}_\ell : \ell = 1, 2, ..., L\}$, with $\mathbf{\Theta}_\ell = \{\mathbf{W}_\ell, \mathbf{W}'_\ell, \mathbf{b}_\ell, \mathbf{b}'_\ell\}$, can be formulated as follows:

$$\begin{aligned} \mathbf{a}_\ell &= \sigma(\mathbf{W}_\ell \mathbf{a}_{\ell-1} + \mathbf{b}_\ell) \\ \hat{\mathbf{a}}_{\ell-1} &= \sigma(\mathbf{W}'_\ell \mathbf{a}_\ell + \mathbf{b}'_\ell), \end{aligned} \tag{1.18}$$

where $\mathbf{W}_\ell$ and $\mathbf{b}_\ell$ (resp. $\mathbf{W}'_\ell$ and $\mathbf{b}'_\ell$) are the encoder (resp. decoder) weight matrix and bias vector of the ℓ-th auto-encoder, $\mathbf{a}_0 = \mathbf{x}$ and $\hat{\mathbf{a}}_0 = \hat{\mathbf{x}}$. After pre-training, the entire stacked auto-encoder can be trained using backpropagation to fine-tune all the parameters of the network.

1.5.4 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a deep architecture inspired by the way humans process visual information [21]. It makes use of feedforward artificial neural networks in which individual neurons are tiled in such a way that they respond to overlapping regions in the visual field. CNNs are comprised of multiple layers that can be categorized into three types: convolutional, subsampling and fully-connected. A convolutional layer consists of a rectangular grid of neurons, and applies a set of filters that process small local parts of the input where these filters are replicated along the whole input space. Each neuron takes inputs from a rectangular section of the previous layer; the weights for this rectangular section are the same for each neuron in the convolutional layer. Thus, the convolutional layer is just an image convolution of the previous layer, where the weights specify the convolution filter. A subsampling (pooling) layer takes small rectangular blocks from the convolutional layer and subsamples it with average or max pooling to produce a single output from that block. This adds translation invariance and tolerance to minor differences of positions of objects parts. Higher layers use more broad filters that work on lower resolution inputs to process more complex parts of the input. Similar to a feedforward neural network, a fully connected layer takes all neurons in the previous layer and connects them to each of its neurons. CNNs can be trained using standard backpropagation. The CNN architecture shown in Figure 1.3 is composed of 5 layers: two convolutional layers (C1 and C2), two subsampling layers (S1 and S2) and one fully connected layer. For classification tasks, an output layer is added after the fully connected layer.

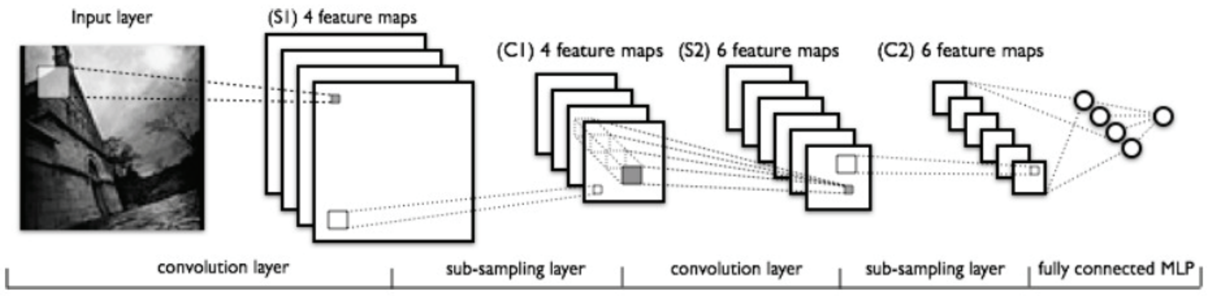


Figure 1.3: Basic architectures of a CNN.

More specifically, the input to a convolutional layer is an $m \times m \times r$ image where m is the height and width of the image and r is the number of channels, e.g. an RGB image has $r = 3$. The convolutional layer will have k filters (or kernels) of size $n \times n \times q$ where n is smaller than the dimension of the image and q can either be the same as the number of channels r or smaller and may vary for each kernel. The size of the filters gives rise to the locally connected structure which are each convolved with the image to produce k feature maps of size $m - n + 1$. Each map is then subsampled typically with average or max pooling over $p \times p$ contiguous regions, where p ranges between 2 for small images and is usually not more than 5 for larger inputs. Either before or after the subsampling layer an additive bias and a sigmoidal nonlinear activation function is applied to each feature map. After the convolutional layers there may be any number of fully connected layers. The densely connected layers are identical to the layers in a standard multilayer neural network.

1.5.5 Performance Evaluation Measures

In this section, we discuss in detail the measures that are commonly used to evaluate the performance of nonrigid 3D shape retrieval and classification. We first discuss the evaluation metrics for

3D shape retrieval which are precision-recall curve, nearest neighbor (NN), first-tier (FT), second-tier (ST), E-measure (E), discounted cumulative gain (DCG), and mean average precision (mAP).

Precision-Recall Graph. A precision-recall graph demonstrates the behavior of precision and recall in a ranked list of retrieved shapes. Assume the category that query shape belongs to has C members including query shape itself and we retrieve top K matches. Recall is the ratio of shapes in query's category that are retrieved among top K matches, while precision is the ratio of top K matches that belong to the query's category. The perfect retrieval results must give the highest precision (i.e. 100%) for all recall which may be illustrated by a horizonal line at the top of the plot (i.e. precision $= 1.0$). Hence, a precision-recall graph that is shifted upwards and to the right indicates superior performance.

Nearest Neighbor. The NN metric is the percentage of the closest matches that belong to the same category of query's, i.e. for each shape in the dataset, the second best result (obviously, the best result is a match with query itself) is verified wether it is a member of the same category that the query shape belongs to. The ideal score is definitely 100% and the higher score indicates the better results.

First-Tier and Second-Tier. The FT metric is the percentage of the shapes belong to the query's category that are retrieved in the top $C - 1$ matches, where query's category has C members. The recall for ST metric is twice as big as for ST metric, i.e. the percentage of the shapes belong to the query's category that are retrieved in the top $2(C - 1)$ matches. Obviously, the ideal score for both metrics are 100% and the higher values represents better results, while the higher score is more likely to appear for ST metric as the members of query's category have more chance to be retrieved among top matches.

E-measures. This metric is obtained when precision and recall are calculated for the first 32 matches in the ranked list (i.e. $K = 32$). The E-measure is defined as:

$$E = \frac{2}{\frac{1}{P} + \frac{1}{R}}, \tag{1.19}$$

where P and R are precision and recall, respectively. The maximum value for this metric is 1.0 (or equivalently 100% in terms of percentages) and the higher scores indicates the better results.

Discounted Cumulative Gain. This metric weighs relevant results on the top of ranked list more than the relevant results at the bottom of the ranked list. The intuition is that the query results of the first pages are more of interest to a user of a search engine than those of the later pages. This metric have scores ranging from 0% to 100% and the higher score indicates the better retrieval performance.

Mean Average Precision. The mAP metric is defined as:

$$mAP = \sum_{K} P(K)R(K), \tag{1.20}$$

where precision and recall are calculated for all values of K. Intuitively, mAP is considered the area below the precision-recall graph. A perfect retrieval algorithm has mAP $= 100\%$ and a higher value indicates better results.

Confusion Matrix. The performance of a classifier is usually evaluated via the confusion matrix, which displays the number of correct and incorrect predictions made by the classifier compared with the actual classifications in the test set. The confusion matrix shows how the predictions are made by the model. The rows correspond to the actual (true) class of the data (i.e., the labels in the data), while the columns correspond to the predicted class (i.e., predictions made by the model). When an instance is classified, it is the same as making a prediction that the instance is correctly classified. The elements of the confusion matrix for binary (two-class) classification problem are

- TP (true positives) is the number of positive instances correctly classified
- FP (false positives) is the number of negative instances incorrectly classified as positive
- FN (false negatives) is the number of positive instances incorrectly classified as negative
- TN (true negatives) is the number of negative instances correctly classified

The value of each element in the confusion matrix is the number of predictions made with the class corresponding to the column for instances (examples) with the correct value as represented by the row. Thus, the diagonal elements show the number of correct classifications made for each class, and the off-diagonal elements show the errors made.

Classification Accuracy. Another intuitively appealing measure is the classification accuracy, which is a summary statistic that can be easily computed from the confusion matrix as the total number of correctly classified instances (i.e. diagonal elements of confusion matrix) divided by the total number of test instances. Alternatively, the accuracy of a classification model on a test set may be defined as follows

$$\begin{aligned}\text{Accuracy} &= \frac{\text{Number of correct classifications}}{\text{Total number of test cases}} \\ &= \frac{|\mathbf{z} : \mathbf{z} \in \mathcal{Z}_{\text{test}} \wedge \hat{y}(\mathbf{z}) = y(\mathbf{z})|}{|\mathbf{z} : \mathbf{z} \in \mathcal{Z}_{\text{test}}|},\end{aligned} \tag{1.21}$$

where $y(\mathbf{z})$ is the actual (true) label of $\mathbf{z}$, and $\hat{y}(\mathbf{z})$ is the label predicted by the classification algorithm. A correct classification means that the learned model predicts the same class as the original class of the test case.

1.6 Overview and Contributions

The organization of this thesis is as follows

- Chapter 1 begins with the basic concepts which we refer to throughout the thesis, gives our motivations and goals for this research, followed by the problem statement, the objective of this study, a literature review, and a brief discussion of background context to the development of our 3D shape analysis framework.

- In Chapter 2, we introduce a multi-level feature learning framework for 3D shape retrieval using using spectral graph wavelets, bag-of-features, and deep learning [64]. The proposed 3D shape retrieval approach is evaluated on three standard 3D shape datasets through extensive experiments, and the results show compelling superiority of our approach over state-of-the-art methods.

- In Chapter 3, we propose a deep learning approach to 3D shape retrieval using a multi-level feature learning paradigm [65]. Low-level features are first extracted from a 3D shape using spectral graph wavelets. Then, mid-level features are generated via the bag-of-features model by employing locality-constrained linear coding as a feature coding method, in conjunction with the biharmonic distance and intrinsic spatial pyramid matching in a bid to effectively measure the spatial relationship between each pair of the bag-of-feature descriptors. Finally, high-level shape features are learned by applying a deep auto-encoder on mid-level features. Extensive experiments on three standard 3D shape datasets demonstrate the much better performance of the proposed framework in comparison with state-of-the-art methods, and also a framework developed based on a shallow model.

- In Chapter 4, we present a deep learning approach to 3D shape classification using convolutional neural networks [66] using the bag-of-features model in conjunction with intrinsic spatial pyramid matching that leverages the spatial relationship between features. These 2D images are then fed into a pre-trained convolutional neural network to learn deep convolutional shape-aware descriptors from the penultimate fully-connected layer of the network. Finally, a multiclass support vector machine classifier is trained on the deep descriptors, and the classification accuracy is subsequently computed. The effectiveness of our approach is demonstrated on three standard 3D shape benchmarks, yielding higher classification accuracy rates compared to existing methods.

- Chapter 5 presents a summary of the contributions of this proposal, limitations, and outlines several directions for future research in this area of study.

CHAPTER 2

Deep Shape-Aware Descriptor for 3D Object Retrieval

Deep learning has become a pervasive and wide reaching technology, growing at a breathtaking rate and achieving remarkable results on a variety of fields, including computer vision, image and speech recognition, and natural language processing. In this chapter, we propose a deep learning approach for 3D shape retrieval using a multi-level feature learning methodology. We first extract low-level features or local descriptors from a 3D shape using spectral graph wavelets. Then, we construct mid-level features from these local descriptors via the bag-of-features paradigm by employing locality-constrained linear coding as a feature coding method, together with the biharmonic distance as a measure of the spatial relationship between each pair of bag-of-feature descriptors. Finally, high-level shape features are learned via a deep auto-encoder, resulting in a deep shape-aware descriptor that is compact, geometrically informative and efficient to compute. The proposed 3D shape retrieval approach is evaluated on SHREC-2014 and SHREC-2015 datasets through extensive experiments, and the results show compelling superiority of our approach over state-of-the-art methods.

2.1 Introduction

In recent years, spectral geometry has been key in the development of efficient algorithms for nonrigid 3D shape retrieval, achieving state-of-the-art performance on the latest shape retrieval contests [1, 2]. Most spectral-geometric methods make use of a shape signature or descriptor, which is a concise and compact representation of a shape, aimed at facilitating the retrieval task.

These shape representations may be categorized into local and global descriptors. Local descriptors (also known as point signatures) are usually defined on each point of the shape, while global descriptors are defined on the entire 3D shape. Examples of local descriptors include the global point signature (GPS) [5], heat kernel signature (HKS) [6], scale-invariant heat kernel signature (SI-HKS) [7], wave kernel signature (WKS) [8], and spectral graph wavelet signature (SGWS) [9]. On the other hand, many global descriptors can be constructed from point signatures by integrating over the entire shape. One of the simplest global descriptors is Shape-DNA [42], which is defined as a truncated sequence of the Laplace-Beltrami operator (LBO) eigenvalues arranged in increasing order of magnitude. Chaudhari *et al.* [11] presented a slightly modified version of the GPS signature by setting the LBO eigenfunctions to unity. Ye *et al.* [12] proposed a global descriptor for nonrigid shape retrieval using a reduced biharmonic distance matrix.

Another type of commonly-used global descriptors are constructed by aggregating the local descriptors using the bag-of-features (BoF) paradigm. In its simplest form, the BoF model quantizes each local descriptor to its nearest cluster center using K-means clustering and then encodes each shape as a histogram over cluster centers by counting the number of assignments per cluster. These cluster centers form a codebook whose elements are often referred to as codewords. Although the BoF paradigm has been shown to provide significant levels of performance, it does not, however, take into consideration the spatial relations between features, which may have an adverse effect not only on its descriptive ability but also on its discriminative power. To account for the spatial relations between features, Bronstein *et al.* [47] introduced a generalization of a bag of features, called spatially sensitive bags of features (SS-BoF). Litman *et al.* [48] proposed a supervised approach to learn BoF shape descriptors using sparse coding.

Deep learning models have been recently used in 3D shape analysis to learn high-level features of 3D shapes. The most popular deep learning models that have been successfully applied to image data include deep convolutional neural networks, deep auto-encoders, deep belief networks and deep Boltzmann machines [21–33]. Although a few studies [67, 68] proposed to apply deep models directly to 3D data, many frameworks first represent a 3D shape by a 2D image and then apply a deep architecture for feature learning. For this purpose, the more conventional way is to capture the object by a set of 2D images from different views. Zhu *et al.* [20] introduced a a view-based technique by projecting 3D shapes into 2D images and then using an auto-encoder for feature learning. A major drawback of view-based methods is their sensitivity to consistent model orientations, resulting in lower performance [3].

Another route to represent a 3D shape as a 2D image is to capture geometric and topological properties of the model and then demonstrate it as a 2D signal. These graphical informative representation are usually obtained by using global shape descriptors. For instance, Bu *et al.* [49] presented a deep learning framework (3D-DL) for 3D shape classification and retrieval. 3D-DL extracts high-level features by applying deep belief networks (DBNs) on 2D global descriptor obtained by the geodesic distance and eigenfunctions of the LBO. The main issue with geodesic distance lies in its sensitivity to topological noise not to mention, it often fails to capture the global properties of a shape compared to the (squared) biharmonic distance [35].

In this chapter, we adopt a similar strategy as [49] in the sense that we employ deep learning to 3D shape retrieval, but our approach differs in the way our deep shape descriptor is computed. More specifically, we introduce a multi-level feature learning approach using spectral graph wavelets, bag-of-features and deep auto-encoders. In particular, we use the spectral graph wavelet signature as a local descriptor due is its ability to capture different details provided at different levels from low to high frequencies. We also use locality-constrained linear coding (LLC) as a feature coding scheme in the BoF model due largely to the lower quantization error of LLC as well as its codewords locality properly. In addition, we employ the biharmonic distance to measure the spatial relationship between the LLC codes. Unlike the geodesic distance which is not globally shape-aware, the biharmonic distance is shape-ware, isometry invariant, computationally efficient, robust to various shape deformations, and possesses good discriminative capabilities [12, 35]. The main contributions of this chapter may be summarized as follows:

1. We present low-level shape descriptors using spectral graph wavelets.
2. We construct mid-level features using the BoF model in which we employ LLC as a feature coding scheme. We then measure the spatial relationship between the LLC codes via the biharmonic distance in order to generate shape-aware bag-of-features.
3. We employ a deep auto-encoder to learn high-level features that are used to design a deep shape-aware descriptor for 3D shape retrieval tasks.

The rest of this chapter is structured as follows. In Section 2.2, we introduce a multi-level 3D shape feature learning framework using deep learning, and we discuss in detail its major components as well as its algorithmic steps. Section 2.3 presents the experimental results and Section 2.4 concludes the chapter.

2.2 Proposed Framework

In this section, we describe the main components and algorithmic steps of the proposed multi-level feature learning framework. The approach consists of three major components: low-level features, mid-level features and high-level features, as illustrated in Figure 2.1. In the low-level features construction, we use spectral graph wavelets to generate local descriptors for each 3D shape in the dataset. In the mid-level features step, we used the BoF model in conjunction with the biharmonic distance to construct shape-aware global descriptors. In the third step, high-level shape features are learned using deep auto-encoders.

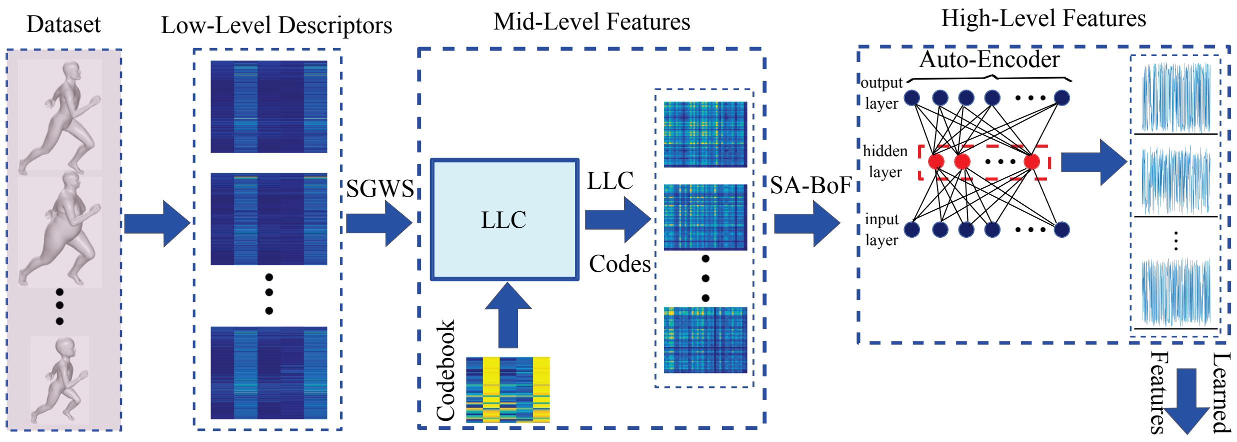


Figure 2.1: Main components of the proposed feature learning method: low-level features, mid-level features and high-level features.

2.2.1 Low-Level Features

Wavelets are useful in describing functions at different levels of resolution. Motivated by the effectiveness of the multiresolution SGWS in 3D shape retrieval [9], we propose an improved spectral graph wavelet signature by incorporating the vertex area into the signature. For a given resolution parameter R, the improved SGWS at vertex j is a p-dimensional vector defined as

$$\mathbf{s}_j = \{\mathbf{s}_Q(j) \mid Q = 1, \ldots, R\}, \tag{2.1}$$

where $\mathbf{s}_Q(j)$ is the shape signature at vertex j and resolution level Q, and is given by

$$\mathbf{s}_Q(j) = \{W_{\delta_j}(t_q, j) \mid q = 1, \ldots, Q\} \cup \{S_{\delta_j}(j)\}. \tag{2.2}$$

At each resolution level Q, the signature $\mathbf{s}_Q(j)$ at vertex j is an $(Q+1)$-dimensional vector consisting of spectral graph wavelet coefficients $W_{\delta_j}(t_q, j)$ given by

$$W_{\delta_j}(t_q, j) = \sum_{\ell=1}^{m} a_j^2 g(t_q \lambda_\ell) \varphi_\ell^2(j), \quad q = 1, \ldots, Q \tag{2.3}$$

and scaling function coefficients $S_{\delta_j}(j)$ given by

$$S_{\delta_j}(j) = \sum_{\ell=1}^{m} a_j^2 h(\lambda_\ell) \varphi_\ell^2(j), \tag{2.4}$$

where g and h are the spectral graph wavelet generating kernel and scaling function, respectively. The spectral graph wavelet generating kernel g acts as a band-pass filter, while h is used as a low-pass filter to encode the low-frequency content of a function defined on the mesh vertices [9]. The wavelet scales t_q ($t_q > t_{q+1}$) are selected to be logarithmically equispaced between maximum and minimum scales t_1 and t_Q, respectively. The dimension of $\mathbf{s}_j$ can be expressed in terms of the resolution parameter R as follows:

$$p = \frac{(R+1)(R+2)}{2} - 1. \tag{2.5}$$

For example, at resolution $R = 2$, the spectral graph wavelet signature $\mathbf{s}_j$ is a 5-dimensional vector consisting of five elements (four elements of spectral graph wavelet function coefficients and one element of scaling function coefficients).

For a p-dimensional signature $\mathbf{s}_i$, we define a $p \times m$ spectral graph wavelet signature matrix as $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_m)$, where $\mathbf{s}_i$ is the signature at vertex i and m is the number of mesh vertices. In our implementation, we used the cubic spline wavelet and the scaling functions given by

$$g(x) = \begin{cases} x^2 & \text{if } x < 1 \\ -5 + 11x - 6x^2 + x^3 & \text{if } 1 \le x \le 2 \\ 4x^{-2} & \text{if } x > 2 \end{cases} \tag{2.6}$$

and

$$h(x) = \gamma \exp\left(-\left(\frac{x}{0.6\lambda_{\min}}\right)^4\right), \tag{2.7}$$

where $\lambda_{\min} = \lambda_{\max}/20$ and γ is set such that $h(0)$ has the same value as the maximum value of g. The maximum and minimum scales are set to $t_1 = 2/\lambda_{\min}$ and $t_Q = 2/\lambda_{\max}$.

2.2.2 Mid-Level Features

In the second step of the proposed approach, we compute sparse codes for the local descriptors using the BoF model, which aggregates these descriptors in order to provide a simple representation that may be used to facilitate comparison between 3D shapes. We then propose new shape descriptors that are globally shape-ware, robust to topological noise and practical to compute. These shape-aware descriptors are defined in terms of the biharmonic distance and the sparse codes.

Bag-of-Features Model

The BoF model consists of four main steps: feature extraction and description, codebook design, feature coding and feature pooling. We model a 3D shape as a triangle mesh $\mathbb{M}$ with m vertices.

Feature extraction and description. In the BoF paradigm, a 3D shape $\mathbb{M}$ is represented as a collection of m local descriptors of the same dimension p, where the order of different feature vectors is of no importance. Local descriptors may be classified into two main categories: dense and sparse. Dense descriptors are computed at each vertex of the mesh, while sparse descriptors are computed by identifying a set of salient points using a feature detection algorithm. In our approach, we represent the shape $\mathbb{M}$ by a $p \times m$ matrix $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_m)$ of spectral graph wavelet signatures, where each p-dimensional feature vector $\mathbf{s}_i$ is a dense, local descriptor that encodes the local structure around the i-th vertex of the mesh.

Codebook design. We construct a codebook (also called vocabulary or dictionary) offline by applying the K-means algorithm to a representative collection of local descriptors. To this end, we used the idea of intrinsic spatial partition [69] to select representative descriptors in a way that ensures each partition of a shape participates in the codebook design procedure. We may represent the codebook by a $p \times k$ vocabulary matrix $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_k)$ of p-dimensional vectors $\mathbf{v}_i$ called codewords (also known as basis vectors or atoms), which are the centroids of the clusters.

Feature coding. Given a codebook, each local descriptor $\mathbf{s}_i$ may be mapped to a codeword in the vocabulary space using feature coding techniques such hard-assignment, soft-assignment, sparse coding and locality-constrained linear coding [70], to name just a few. While sparse coding has shown promising results as a feature coding method in the BoF model [48], it uses, however, sparsity constraint and has no priorities for the closer codewords to each local descriptor over the further ones. Locality-constrained linear coding (LLC), on the other hand, employs locality constraint to enforce codebook locality instead of sparsity. As a result, LLC yields smaller coefficients for codewords farther away from $\mathbf{s}_i$. More precisely, the LLC code $\mathbf{u}_i$ is obtained by solving the following regularized least-squares problem

$$\mathbf{u}_i = \arg\min_{\mathbf{1}^{\intercal}\mathbf{u}_i = 1} \|\mathbf{s}_i - \mathbf{V}\mathbf{u}_i\|_2^2 + \lambda \|\mathbf{d}_i \odot \mathbf{u}_i\|_2^2, \tag{2.8}$$

where $\odot$ denotes the element-wise multiplication, $\mathbf{d}_i = \exp(\text{dist}(\mathbf{s}_i, \mathbf{V})/\delta)$ measures the similarity between the i-th descriptor and all the codewords with dist$(\mathbf{s}_i, \mathbf{V}) = (\|\mathbf{s}_i - \mathbf{v}_1\|_2, \ldots, \|\mathbf{x}_i - \mathbf{v}_k\|_2)$, and δ is a parameter to adjust the weight decay speed for the locality adaptor.

It should be noted that the LLC code is not sparse in the sense of ℓ_0-norm, but it is sparse in the sense that the codes have only a few elements with significant values. In practice, an approximated LLC is employed for fast encoding by removing the regularization term (i.e. locality constraint) from (2.8) and instead using the r nearest neighbors of $\mathbf{s}_i$ as a set of codewords [70], thereby reducing the computational complexity from $\mathcal{O}(k^2)$ to $\mathcal{O}(k + r^2)$, where k is the number of codewords in the vocabulary and $r \ll k$.

Hence, each p-dimensional local descriptor $\mathbf{s}_i$ is encoded by a k-dimensional LLC code $\mathbf{u}_i$, resulting in a $k \times m$ matrix $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_m)$ which we refer to as the LLC codes matrix.

Feature pooling. Each spectral graph wavelet signature is mapped to a certain codeword through the clustering process and the shape is then represented by the histogram $\mathbf{h}$ of the codewords, which is a k-dimensional vector given by

$$\mathbf{h} = \mathbf{U}\mathbf{1}_m = (h_r)_{r=1,\ldots,k} \tag{2.9}$$

where $h_r = \sum_{i=1}^{m} u_{ri}$. That is, the histogram consists of the column-sums of the cluster assignment matrix $\mathbf{U}$. Other feature pooling methods include average- and max-pooling. In general, a feature vector is given by $\mathbf{h} = \mathbf{P}(\mathbf{U})$, where $\mathbf{P}$ is a predefined pooling function that aggregates the information of different codewords into a single feature vector.

Shape-Aware Bag-of-Features

A major drawback of the BoF model is that it only considers the distribution of the codewords and disregards all information about the spatial relations between features, and hence the descriptive ability and discriminative power of the BoF paradigm may be negatively impacted. To circumvent this limitation, various solutions have been recently proposed including the spatially sensitive bags of features (SS-BoF) [47] and geodesic-aware bags of features (GA-BoF) [49]. The SS-BoF, which is defined in terms of the heat kernel, can be represented by a square matrix whose elements represent the frequency of appearance of nearby codewords in the vocabulary. Similarly, the GA-BoF matrix is obtained by replacing the heat kernel in the SS-BoF with a geodesic exponential kernel. Although the geodesic distance has proven to be effective in tackling nonrigid 3D shape matching and retrieval [71, 72] due in large part to its isometry invariance property, it suffers, however, from several disadvantages compared to the (squared) biharmonic distance [35]. While the geodesic distance is not smooth, sensitive to topological noise and not globally shape-aware, the biharmonic distance is not only robust to noise and small topological changes, but also globally shape-aware and smooth. As shown in Figure 2.2, the level sets of the biharmonic distance are much smoother than those of the geodesic distance. Notice that the source point is displayed as a small green sphere, located in the vicinity of the mouth of the 3D face model. Both distances are computed from the source point to all the remaining points of the 3D face model.

In addition to its isometry invariance, the biharmonic distance is practical to compute, and strikes a balance between nearly geodesic distances for small distances and global shape-awareness for large distances. Inspired by these nice properties, we define a shape-ware descriptor of a 3D shape as a $k \times k$ matrix $\mathbf{F}$ given by

$$\mathbf{F} = \mathbf{U}\mathbf{K}\mathbf{U}^{\intercal}, \tag{2.10}$$

where $\mathbf{U}$ is a $k \times m$ matrix of LLC codes, and $\mathbf{K} = (\kappa_{ij})$ is an $m \times m$ biharmonic distance kernel matrix whose elements are defined in terms of the eigenvalues and eigenfunctions of the LBO as

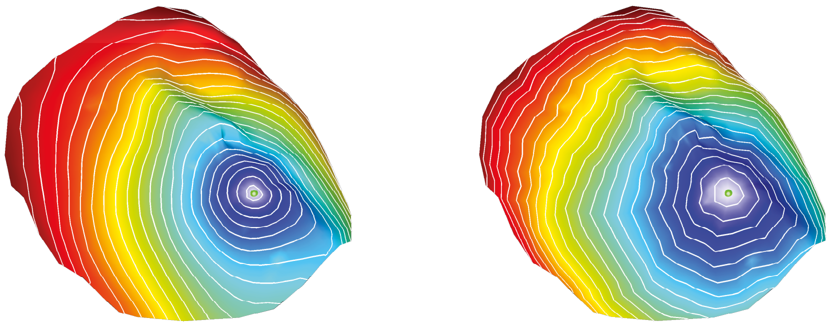


Figure 2.2: A 3D face model color-coded by the biharmonic (left) and geodesic distances (right). Darker blue regions indicate smaller distances, while darker red regions indicate larger distances. Level sets (isocontours) are displayed as white lines at equally spaced intervals of distance.

follows:

$$\kappa_{ij} = \sum_{\ell=1}^{m} \frac{1}{\lambda_\ell^2} (\boldsymbol{\varphi}_\ell(i) - \boldsymbol{\varphi}_\ell(j))^2. \tag{2.11}$$

We refer to $\mathbf{F}$ as a shape-aware bag-of-features (SA-BoF) matrix, which indicates the occurrence distribution of the codewords and the spatial relationships between them. Hence, for each 3D shape, the mid-level features are represented by a $k \times k$ matrix $\mathbf{F}$ containing global descriptors.

2.2.3 High-Level Features

In the third step of our framework, more discriminative 3D shape descriptors are extracted using high-level features learned by performing a deep auto-encoder on the mid-level features. Unlike images, a 3D mesh cannot be fed directly into a deep learning model. To tackle this issue, we use the $k \times k$ SA-BoF matrix (viewed as an image) $\mathbf{F}$ or more precisely the k^2-dimensional vector $\mathbf{x}$ as an input to the deep auto-encoder, where $\mathbf{x}$ is obtained by stacking the columns of $\mathbf{F}$ one underneath the other. The high-level features are then extracted from the output of the last hidden layer of the deep auto-encoder, resulting in an r_L-dimensional high-level feature vector $\mathbf{a}_L$, which we refer to as a deep SA-BoF descriptor, where r_L is the total number of neurons in the last hidden layer, as illustrated in Figure 2.3.

2.2.4 Proposed Algorithm

The goal of 3D shape retrieval is to search and extract the most relevant shapes to a query object from a dataset of 3D shapes. The retrieval accuracy is usually evaluated by computing a dissimilarity measure between pairs of 3D shapes in the dataset. A good retrieval algorithm should result in few dissimilar shapes. A commonly used dissimilarity measure for content-based retrieval is the ℓ_1-distance, which quantifies the difference between each pair of 3D shapes.

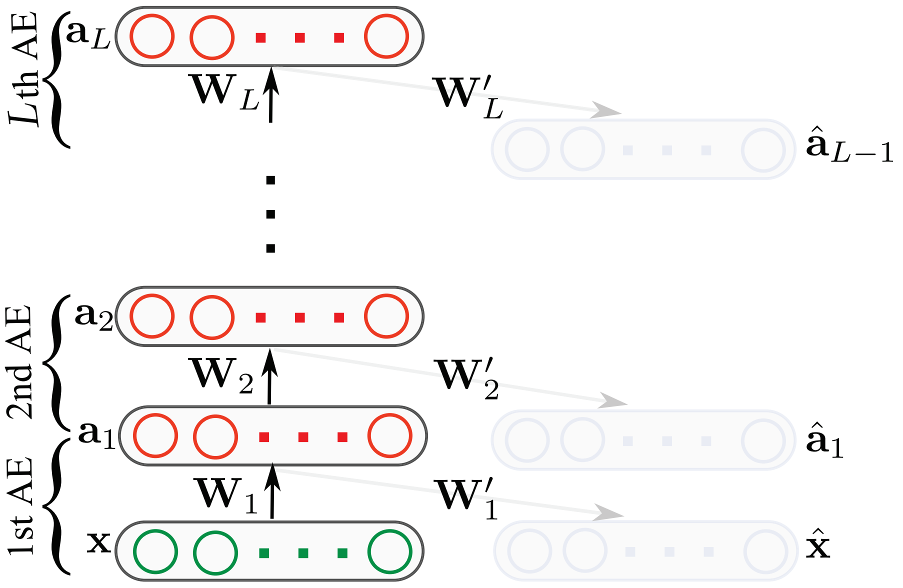


Figure 2.3: Deep auto-encoder architecture. The hidden layer of the 1st auto-encoder (AE) is trained to reconstruct the input data. Then, the hidden layer of the 2nd AE is trained to reconstruct the hidden layer of the 1st AE, and so on.

As stated previously, our learning framework consists of three main components. In the first step, we represent each 3D shape in the dataset by a spectral graph wavelet signature matrix, which is a feature matrix whose columns are the local shape descriptors. More specifically, let $\mathcal{D}$ be a dataset of n shapes modeled by triangle meshes $\mathbb{M}_1, \dots, \mathbb{M}_n$. We represent each mesh $\mathbb{M}_i$ by a $p \times m$ spectral graph wavelet signature matrix $\mathbf{S}_i$, where m is the number of mesh vertices. The spectral graph wavelet signatures are then encoded via LLC, resulting in a $k \times m$ matrix $\mathbf{U}_i$ whose columns are the k-dimensional LLC codes. In the second step, the $k \times k$ SA-BoF matrix $\mathbf{F}_i$ is computed using the LLC codes matrix and the biharmonic distance kernel matrix, followed by reshaping $\mathbf{F}_i$ into a k^2-dimensional SA-BoF vector $\mathbf{x}_i$. In the third step, the SA-BoF vectors $\mathbf{x}_i$ of all n shapes in the dataset are arranged into a $k^2 \times n$ data matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ on which a deep auto-encoder is performed, resulting in an $r_L \times n$ matrix $\mathbf{A} = (\mathbf{a}_L^{(1)}, \dots, \mathbf{a}_L^{(n)})$ whose columns are deep shape-ware global descriptors, where r_L is the total number of units in the last hidden layer of the network. Finally, we compare a query shape to all shapes in the dataset using ℓ_1-distance to measure the dissimilarity between each pair for 3D shape retrieval. We summarize our multi-level 3D shape descriptor approach in Algorithm 1.

2.3 Experiments

To evaluate the efficacy and performance of our method on 3D shape shape retrieval tasks, we conducted several experiments and comparisons.

Algorithm 1 Deep Shape-Aware Framework

```
Input: Dataset $\mathcal{D} = \{\mathbb{M}_1, \dots, \mathbb{M}_n\}$ of 3D shapes and a query.
1: for $i = 1$ to $n$ do
2:     Compute the $p \times m$ SGWS matrix $\mathbf{S}_i$ for each shape $\mathbb{M}_i$
3:     Compute the $k \times m$ LLC codes matrix $\mathbf{U}_i$
4:     Compute the $k \times k$ SA-BoF matrix $\mathbf{F}_i$, and reshape it into a $k^2$-dimensional vector $\mathbf{x}_i$
5: end for
6: Arrange all the $n$ SA-BoF vectors into a $k^2 \times n$ data matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$
7: Apply deep auto-encoder on $\mathbf{X}$ to find the $r_L \times n$ deep SA-BoF matrix $\mathbf{A} = (\mathbf{a}_L^{(1)}, \dots, \mathbf{a}_L^{(n)})$
8: Compute the $\ell_1$-distance between the deep SA-BoF vector of the query and all deep SA-BoF
   vectors in the dataset, and find the closest shape(s).
Output: Retrieved set of most relevant shapes to the query.
```

Datasets. We tested the proposed algorithm on two standard and publicly available 3D shape benchmarks: SHREC 2014 and SHREC 2015. Sample shapes from these benchmarks are shown in Figure 2.4. The SHREC-2014 benchmark contains two datasets: real and synthetic human models. The real SHREC-2014 dataset is made up of ‘real’ data, obtained by scanning real human participants [5], and it consists of 400 shapes, made up of 40 human subjects in 10 different poses. Half the human subjects are male, and half female. The poses of each subject are built by using a data-driven deformation technique, which can produce realistic deformations of articulated meshes. The synthetic SHREC-2014 dataset was built using DAZ Studio and consists of 300 human models (adults and children) subdivided into 15 classes of 20 members each. Objects are considered as part of the same class if they share the same body shape.

The SHREC-2015 benchmark is a dataset of 3D shapes consisting of 1200 watertight mesh models from 50 classes [2], where each class contains 24 objects with distinct postures.

Implementation details. All the experiments were performed on a desktop computer with a CPU Core i5 processor running at 3.4 GHz and 16 GB RAM, and the algorithms were implemented in MATLAB R2016a (version 9.0). We use the first 201 eigenvalues and eigenvectors of the LBO to compute the low-level descriptors and the biharmonic distance. The resolution level of SGWS is set to $R = 2$, that is each local descriptor is of length 5 (i.e. $p = 5$). The number of vertices varies from shape to shape, but it is set to approximately 1000 and 2200 for the shapes in the SHREC-2015 and SHREC-2014 datasets, respectively. For the mid-level features, a codebook of size 5×48 (i.e. setting $k = 48$) is constructed using a representative collection containing 10 local descriptors from each shape, and the LLC codes are computed using 5 nearest neighbors, yielding a SA-BoF matrix of size 48×48 for each shape in the dataset. Then, the mid-level features of all shapes are reshaped into 48^2-dimensional vectors, resulting in a SA-BoF data matrix $\mathbf{X}$ of size $48^2 \times n$, where n is the total number of 3D shapes in the dataset. To compute high-level features, we use a deep

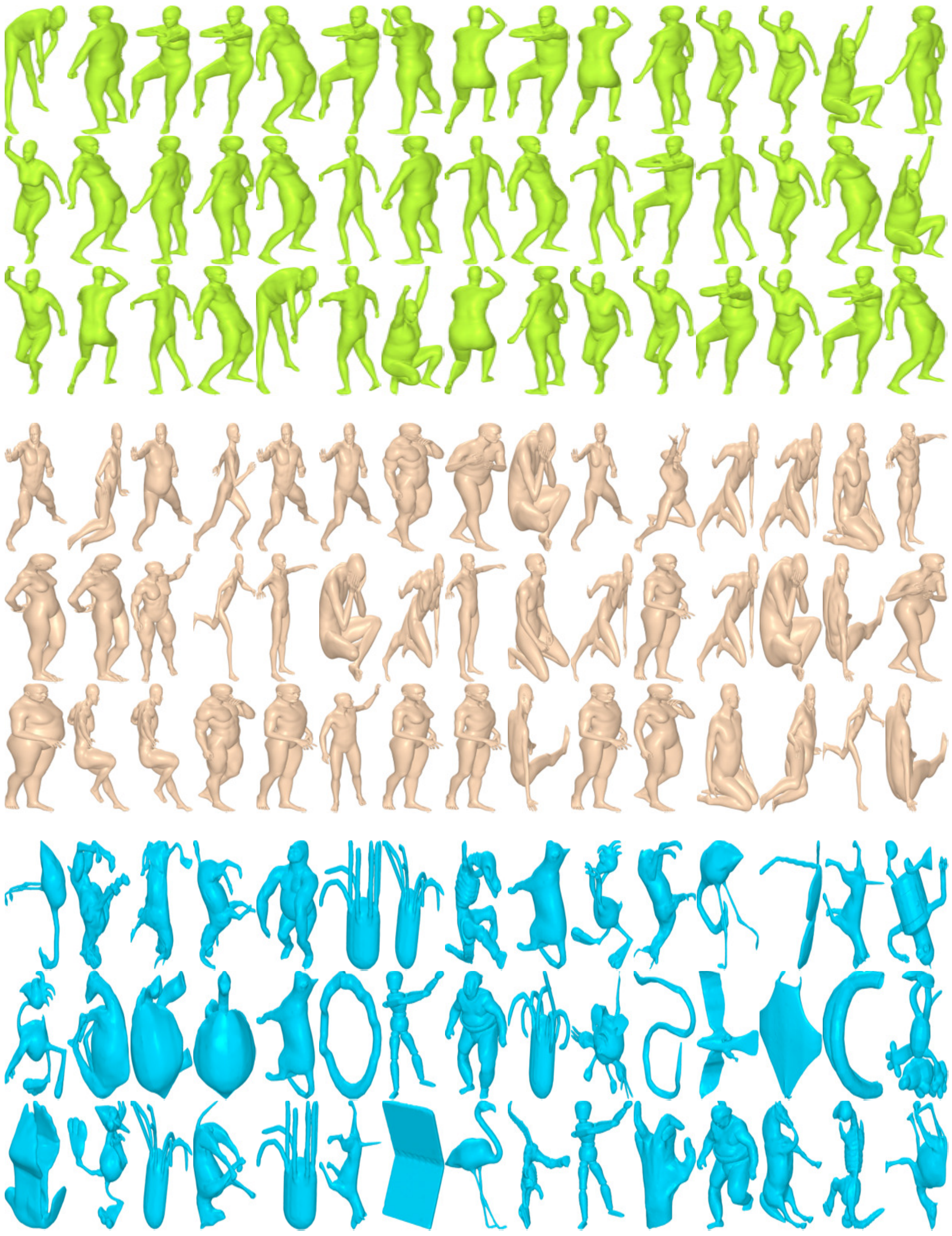


Figure 2.4: Sample shapes from real SHREC 2014 (top), synthetic SHREC 2014 (middle), and SHREC 2015 (bottom).

auto-encoder consisting of an input layer of size 48^2, a hidden layer of size $r_L = 300$, and an output layer of size equal to the number of classes in each 3D shape dataset. This yields a deep SA-BoF matrix $\mathbf{A}$ of length $300 \times n$. It is important to note that the dimension of mid-level features was reduced from $48^2 = 2304$ to 300, which indicates the compactness of the deep SA-BoF descriptor. The regularization parameter in the objective function of the deep auto-encoder is set to $\lambda = 0.001$.

Baseline methods. We compare the effectiveness of the proposed framework with several state-of-the-art methods, including histograms of area projection transform (HAPT) [73], heat kernel signature based on time serial (HKS-TS) [1, 2], spectral graph wavelet signature (SGWS) [9], Euclidean distance based canonical forms (EDBCF) [74], supervised dictionary learning (SupDL-train) [48], reduced biharmonic distance matrix (R-BiHDM) [12], and high-level feature learning using deep belief networks (3D-DL) [49]. We also compare our deep SA-BoF with SA-BoF to show the advantage of the deep network in improving the retrieval performance. It should be noted that SA-BoF is a special case of deep SA-BoF, with the designed network having only two layers and an identity activation function (i.e. $\sigma(\mathbf{x}) = \mathbf{x}$). For all baseline methods, we use the default parameters when available.

2.3.1 Results

We evaluate the retrieval performance of the proposed approach in comparison with existing methods using several standard evaluation metrics, including the precision-recall curve, nearest neighbor (NN), first-tier (FT), second-tier (ST), E-measure (E), discounted cumulative gain (DCG), and mean average precision (mAP). The formal definitions of these metrics can be found in [75].

Results on SHREC 2015. For this dataset of 1200 shapes, we first compute the SA-BoF data matrix $\mathbf{X}$, which is of size $48^2 \times 1200$. Training the auto-encoder on the training dataset yields learned features that form a deep SA-BoF data matrix $\mathbf{A}$ of size 300×1200. Then, a distance matrix of size 1200×1200 is constructed by computing the ℓ_1-distance between each pair of the 300-dimensional deep feature vectors. Finally, a retrieval test on this distance matrix is conducted and the scores for the evaluation metrics are computed. Table 3.7 shows the retrieval results of deep SA-BoF and several baseline methods. As can be seen in the table, deep SA-BoF outperforms all baseline methods on almost all the evaluation metrics, except HAPT [73] which achieves a slightly higher NN value. Moreover, the performance gap between deep SA-BoF and HAPT is significant in terms of the other evaluation metrics, indicating that the proposed approach performs significantly better than the competitors. Note that using the deep auto-encoder to extract high-level features improves the retrieval performance of SA-BoF by 12% and 5.1% in terms of mAP and DCG, respectively. Overall, deep SA-BoF is consistently the best, delivering robust retrieval performance.

We also use precision-recall graphs to evaluate the retrieval performance of the proposed approach in comparison with the baseline methods. A precision-recall graph is an informative graph that illustrates the tradeoff between precision as a function of recall, and it shows the retrieval performance at each point in the ranking. If, for instance, the $(\tau + 1)$-th shape retrieved is relevant, then both precision and recall increase. However, if it is irrelevant then recall is the same as for the top τ shapes, but precision decreases. Hence, a precision-recall graph that is shifted upwards and to the right indicates superior performance. Figure 3.7 compares the proposed framework with several baseline methods using precision-recall curves on the SHREC-2015 benchmark. As can be seen, deep SA-BoF performs significantly better than the competitors. It is important to point out that for fair comparison with SI-HKS and WKS, which also are local descriptors, we generated their corresponding global descriptors using our mid-level feature extraction strategy based on LLC as feature coding method and the biharmonic distance as the kernel for feature pooling.

Table 2.1: Performance comparison results on the SHREC-2015 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
HAPT [73]	**99.8**	96.6	98.2	81.5	99.2	-
HKS-TS [2]	6.5	6.4	12.4	7.4	39.1	-
SGWS [9]	97.3	76.0	81.4	66.0	91.9	-
EDBCF [74]	97.8	79.1	88.4	70.8	94.3	-
SA-BoF	96.1	80.1	89.3	71.3	94.5	82.2
Deep SA-BoF	99.7	**98.3**	**99.2**	**82.7**	**99.6**	**94.2**

Results on SHREC 2014. For the real SHREC-2014 dataset, the SA-BoF data matrix $\mathbf{X}$ is of size $48^2 \times 400$ and the deep SA-BoF data matrix $\mathbf{A}$ is of size 300×400. Hence, the resulting distance matrix is of size 400×400. For the synthetic SHREC-2014 dataset, the SA-BoF data matrix $\mathbf{X}$ is of size $48^2 \times 300$, the deep SA-BoF data matrix $\mathbf{A}$ is of size 300×300, and the distance matrix is of size of 300×300. Tables 3.9 and 3.10 compare the retrieval results of SA-BoF and deep SA-BoF with baseline methods. As reported in Table 3.9, this difference on a noisy data like the SHREC-2014 human real dataset grows by 54.3% in mAP and 38.2% in DCG which is a significant improvement. From Table 3.10, we see that deep SA-BoF improves SA-BoF on the synthetic SHREC-2014 dataset by 20.6% and 8.8% in terms of mAP and DCG, respectively. This clearly indicates the importance of using high-level features in further improving the retrieval results. Moreover, Table 3.9 indicates the proposed method improves the original SGWS significantly e.g. it increases DCG from 48.8% to 93.2%, and mAP from 25.8% to 88.3%. SupDLtrain, which is a supervised learning method, yields slightly better ST and E scores, while deep SA-BoF significantly outperforms SupDLtrain in terms of the other scores on the real SHREC-2014

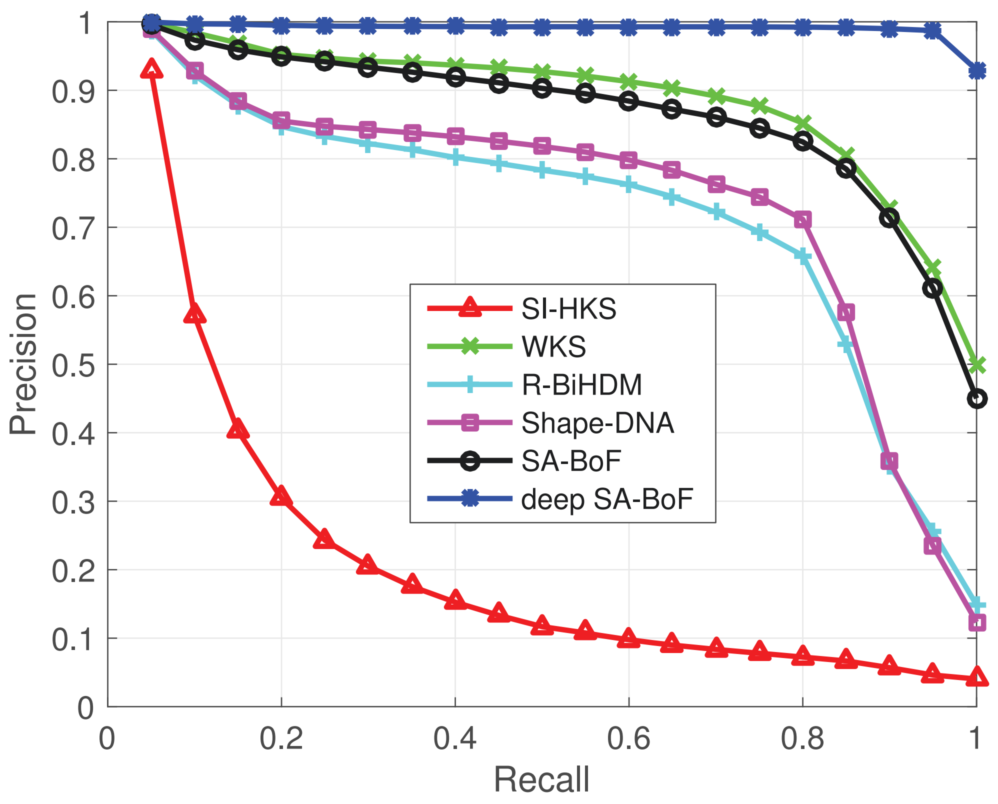


Figure 2.5: Precision-recall graphs comparing the performance of the proposed method with other state-of-the-art approaches on SHREC 2015.

dataset. 3DDL, which is a deep learning based approach, achieves a mediocre retrieval performance on the real SHREC-2014 dataset. Moreover, deep SA-BoF performs the best among all the baseline methods on the synthetic SHREC-2014 benchmark.

Figure 3.8 shows the performance comparison of the proposed method with various baseline methods using the precision-recall graphs on the real and synthetic SHREC-2014 datasets. As can be seen in the figure, the precision-recall graphs indicate the superiority of the proposed method. Note that even for full recall, the precision is still higher than 0.7. Interestingly, Shape-DNA, which is the simplest spectral descriptor, outperforms SI-HKS except on the real SHREC-2014 dataset. In addition, WKS achieves better performance that SI-HKS, providing further evidence that WKS outperforms HKS as reported in [8]. Moreover, SA-BoF yields better retrieval results than WKS and SI-HKS on SHREC 2014, which strengthens our view that SGWS has a more discriminative ability than WKS and SI-HKS. This is largely attributed to the fact that SGWS captures geometric information at multiple scales.

As can be seen in Tables 3.9 and 3.10, deep SA-BoF performs 24.6% and 17.6% better than HAPT in terms of mAP on the real and synthetic SHREC-2014 datasets, respectively. However, HAPT performs practically at par with deep SA-BoF on SHREC 2015 in terms of almost all the

Table 2.2: Performance comparison results on the real SHREC-2014 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
HAPT [73]	84.5	53.4	68.1	35.5	79.5	63.7
HKS-TS [2]	24.5	25.9	46.1	31.4	54.8	-
SGWS [9]	31.3	20.6	32.3	19.2	48.8	25.8
EDBCF [74]	1.0	1.2	4.0	4.3	27.9	-
SupDltrain [48]	79.3	72.7	**91.4**	**43.2**	89.1	79.1
R-BiHDM [12]	68.5	54.1	74.2	38.7	78.1	64.0
3D-DL [49]	22.5	19.3	37.4	26.2	50.4	-
SA-BoF	33.0	26.6	43.9	27.2	55.0	34.0
Deep SA-BoF	**92.8**	**81.8**	91.2	42.7	**93.2**	**88.3**

Table 2.3: Performance comparison results on the synthetic SHREC-2014 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
HAPT [73]	97.0	73.3	92.7	65.5	93.6	81.7
HKS-TS [2]	46.7	47.6	74.3	50.4	72.9	-
SGWS [9]	99.3	83.2	97.1	70.6	97.1	90.2
EDBCF [74]	11.3	18.2	33.3	21.7	50.7	-
SupDltrain [48]	96.0	88.7	99.1	72.1	97.5	95.4
R-BiHDM [12]	79.3	57.2	76.0	53.3	83.6	64.2
3D-DL [49]	92.3	76.0	91.1	64.1	92.1	-
SA-BoF	91.0	70.8	91.7	65.5	90.7	78.7
Deep SA-BoF	**99.3**	**98.4**	**99.3**	**73.9**	**99.5**	**99.3**

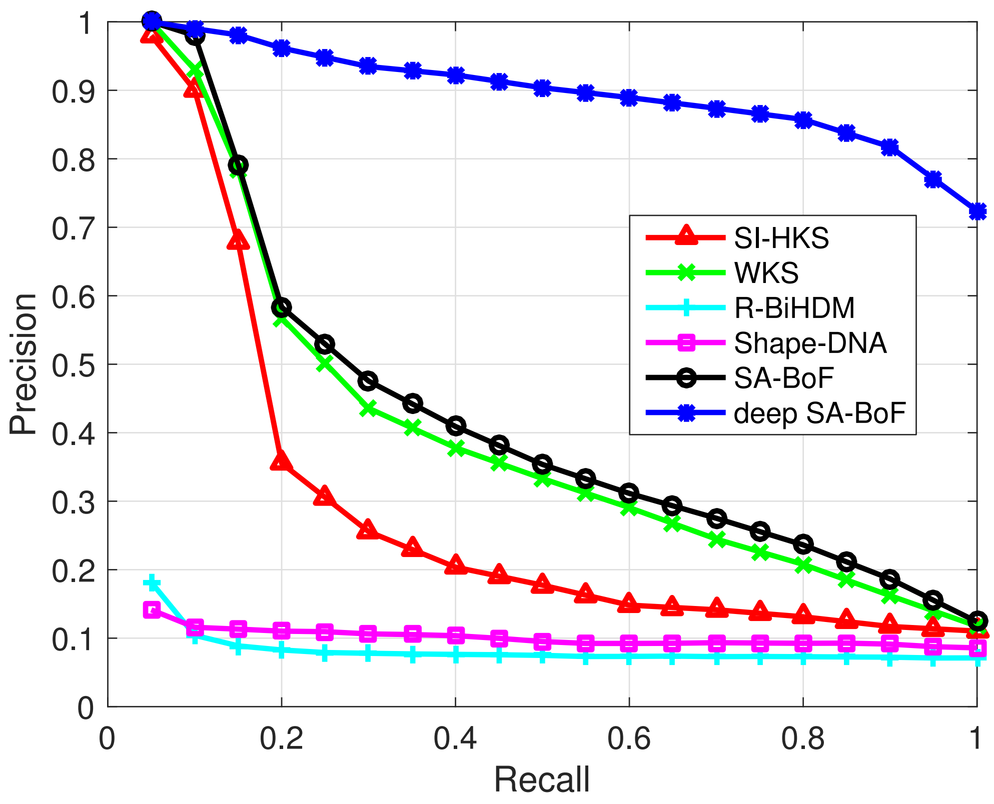
evaluation measures, as shown in Table 3.7. This good performance of HAPT on SHREC 2015 may be due in large part to two key points. First, nearly half of all shapes in SHREC 2015 come from SHREC 2011, in which HAPT was originally tested with a varying degree of success. Second, the difference between the various categories in SHREC 2015 is quite noticeable compared to the ones in SHREC 2014. In fact, even human observers may not easily distinguish between some categories in SHREC 2014, particularly with the real SHREC-2014 benchmark.

The accuracy of the retrieval results using deep SA-BoF is further illustrated in Figures 3.10 and 3.11. Two queries (male and female) from the real SHREC-2014 dataset are featured in the top-most row of these figures, followed by the top five retrieved shapes. The first query is the male model number 1 (M1) as shown in Figure 3.10, while the second query is a female model number 9 (F9) as depicted in Figure 3.11. We compared out results to several baseline methods, including SI-HKS, WKS, R-BiHDM and Shape-DNA. As can be seen in Figure 3.10, deep SA-BoF was able to correctly retrieve all the relevant shapes from the query's class (i.e. same shape in different poses), while the other methods failed more than once in retrieving the relevant shapes.

Similarly, we can see in Figure 3.11 that our approach outperforms all baseline methods. This better performance is largely attributed to the fact that deep learning models are able to extract/build better features than shallow models.

2.4 Conclusion

In this chapter, we presented a multi-level feature learning framework for 3D shape retrieval using deep learning. First, low-level local descriptors were obtained using spectral graph wavelets. Then, mid-level features were extracted via the bag-of-features model by aggregating local descriptors into global ones. We used locality-constrained linear coding as a feature coding method and measured the spatial relationships between codewords using the biharmonic distance in a bid to generate shape-aware bag-of-features as mid-level features. Finally, high-level features were learned using a deep auto-encoder. The proposed approach achieves significantly better performance than state-of-the-art methods.



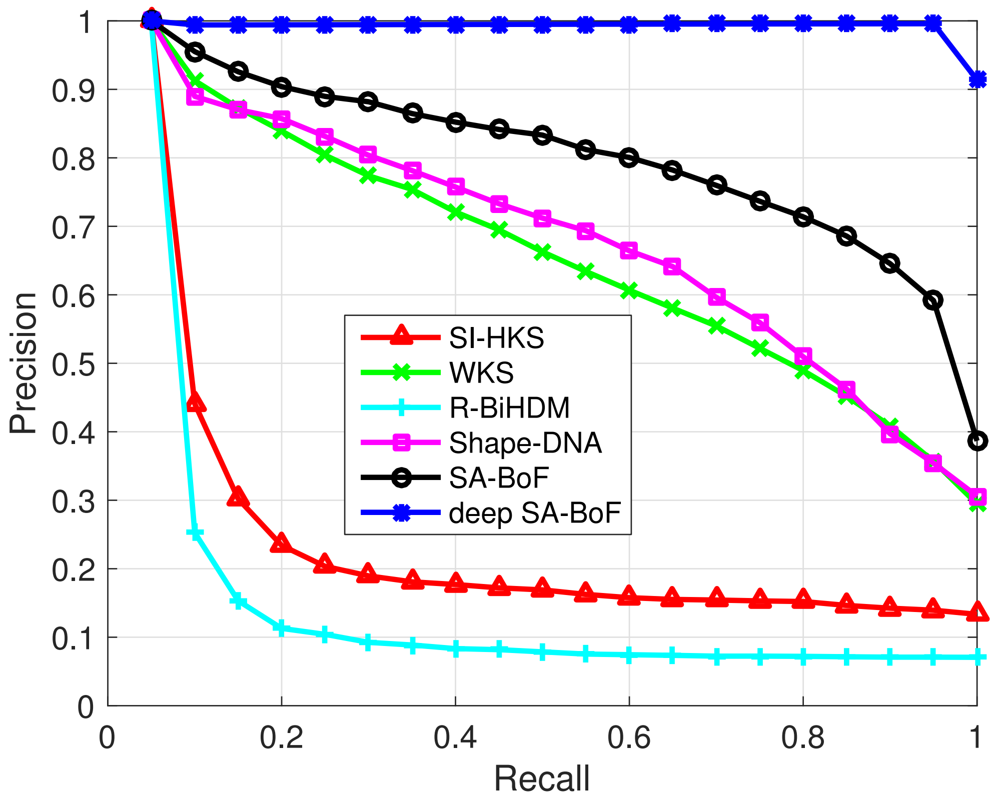
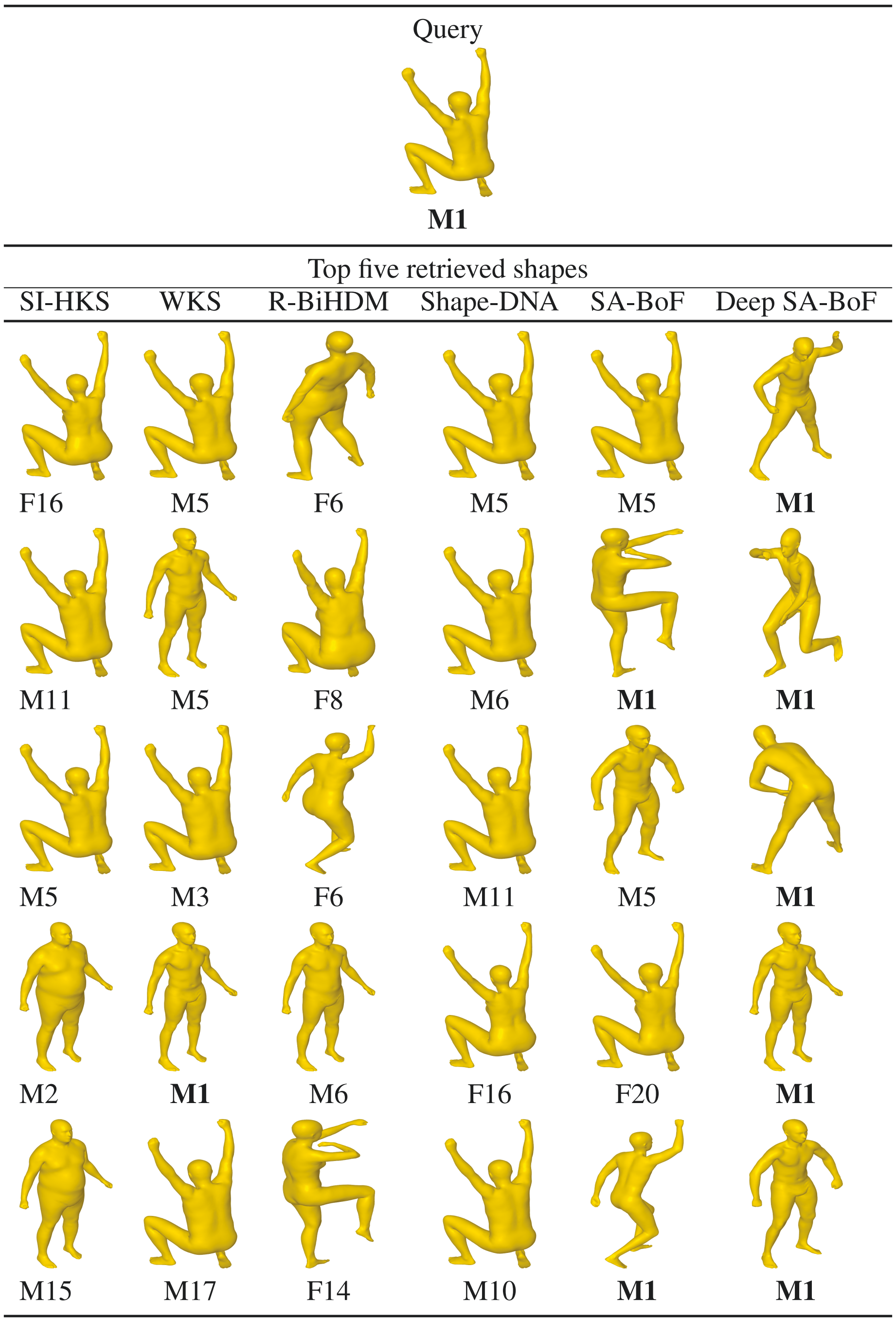


Figure 2.6: Precision-recall graphs comparing the performance of the proposed method with other state-of-the-art approaches on the real SHREC-2014 (up) and the synthetic SHREC-2014 datasets (down).



Query					
		M1			

Top five retrieved shapes

SI-HKS	WKS	R-BiHDM	Shape-DNA	SA-BoF	Deep SA-BoF
F16	M5	F6	M5	M5	**M1**
M11	M5	F8	M6	**M1**	**M1**
M5	M3	F6	M11	M5	**M1**
M2	**M1**	M6	F16	F20	**M1**
M15	M17	F14	M10	**M1**	**M1**

Figure 2.7: Top five retrieved shapes (ranked top-to-bottom) using SI-HKS, WKS, R-BiHDM, Shape-DNA, SA-BoF, and deep SA-BoF. The query shape is the male number 1 (M1) from the real SHREC-2014 dataset. Boldface numbers indicate the correctly retrieved shapes. M# (resp. F#) denotes the male (resp. female) model in class #.

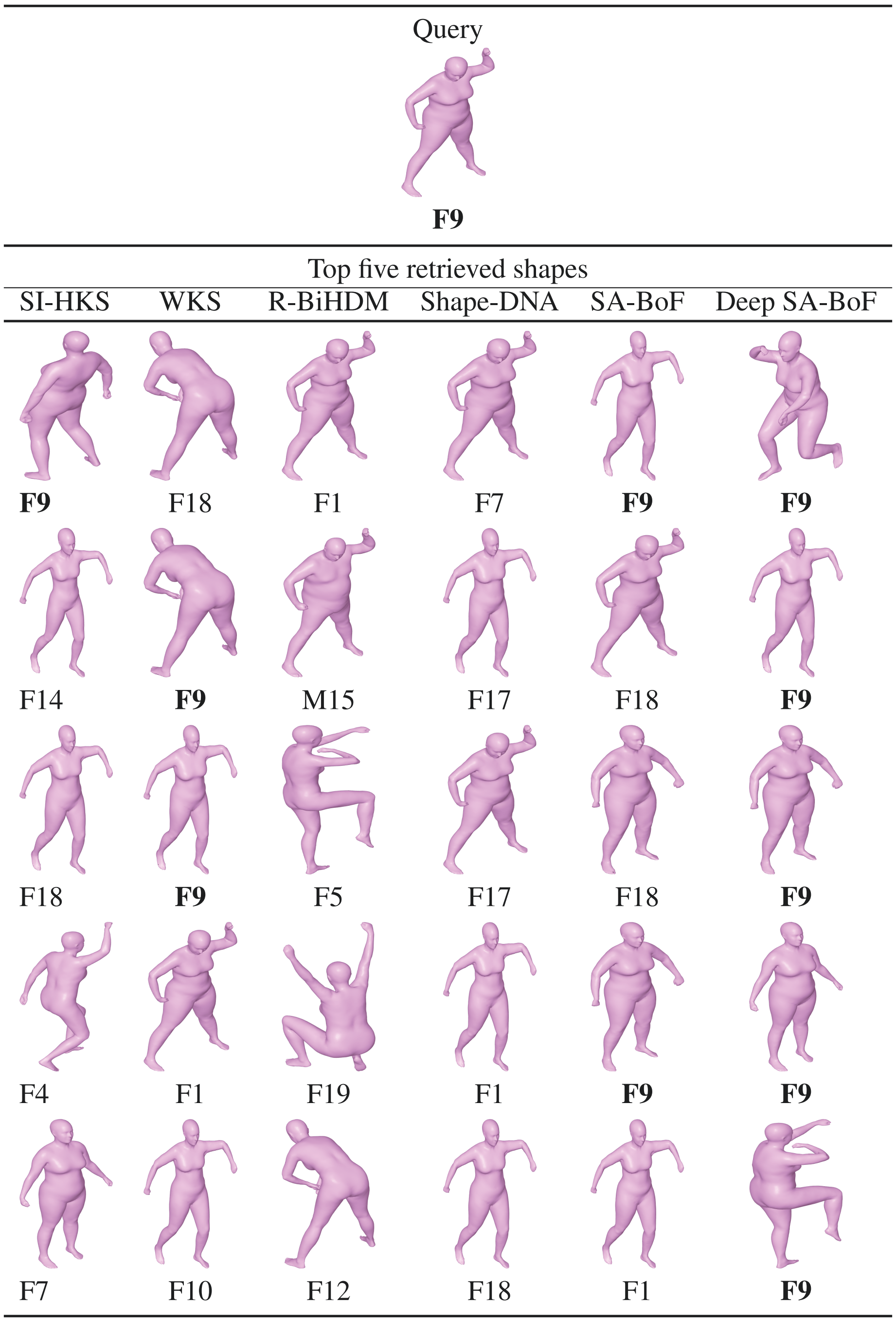

Query
F9
Top five retrieved shapes
SI-HKS WKS R-BiHDM Shape-DNA SA-BoF Deep SA-BoF
F9 F18 F1 F7 **F9** **F9**
F14 **F9** M15 F17 F18 **F9**
F18 **F9** F5 F17 F18 **F9**
F4 F1 F19 F1 **F9** **F9**
F7 F10 F12 F18 F1 **F9**

Figure 2.8: Top five retrieved shapes (ranked top-to-bottom) using SI-HKS, WKS, R-BiHDM, Shape-DNA, SA-BoF, and deep SA-BoF. The query shape is the female number 9 (F9) from the real SHREC-2014 dataset. Boldface numbers indicate the correctly retrieved shapes. M# (resp. F#) denotes the male (resp. female) model in class #.

CHAPTER 3

Intrinsic Spatial Pyramid Matching for 3D Shape Retrieval

The soaring popularity of deep learning in a wide variety of fields ranging from computer vision and speech recognition to self-driving vehicles has sparked a flurry of research interest from both academia and industry. In this chapter, we propose a deep learning approach to 3D shape retrieval using a multi-level feature learning paradigm. Low-level features are first extracted from a 3D shape using spectral graph wavelets. Then, mid-level features are generated via the bag-of-features model by employing locality-constrained linear coding as a feature coding method, in conjunction with the biharmonic distance and intrinsic spatial pyramid matching in a bid to effectively measure the spatial relationship between each pair of the bag-of-feature descriptors. Finally, high-level shape features are learned by applying a deep auto-encoder on mid-level features. Extensive experiments on SHREC-2014 and SHREC-2015 datasets demonstrate the much better performance of the proposed framework in comparison with state-of-the-art methods.

3.1 Introduction

Deep learning has recently gained increasing popularity due largely to its competitive results in many tasks most notably for machine learning, computer vision, and speech recognition [76]. In spite of improvements in hand-crafted descriptors and shallow representations, deep learning frameworks [4, 18–20] often beat these conventional methods by a large margin. Deep learning models have recently been applied to 3D shape analysis to learn high-level features from 3D shapes. Fang *et al.* [54] introduced a deep learning framework in which the heat kernel signature is fed to deep neural networks with target values in a bid to obtain a 3D deep shape descriptor that demonstrated good performance in 3D shape retrieval. Inspired by the Shape Google framework for 3D shape retrieval [47], Bu *et al.* [49] introduced a deep learning based approach (3D-DL) for 3D shape classification and retrieval. The 3D-DL framework uses a 2D global shape descriptor, which is represented by a full matrix defined in terms of the geodesic distance and eigenfunctions of the LBO. A major drawback of the geodesic distance is its sensitivity to topological noise as well as its inability to capture the global features of a shape compared to the (squared) biharmonic distance [35].

In this chapter, we propose a multi-level feature learning approach using spectral graph wavelets, bag-of-features and deep auto-encoders. In particular, we use SGWS as a local descriptor due to its ability to capture different details provided at different levels from low to high frequencies. We also use locality-constrained linear coding (LLC) as a feature coding scheme in the BoF model due largely to the lower quantization error of LLC as well as its codewords locality properly. In addition, we employ the biharmonic distance together with intrinsic spatial pyramid matching (ISPM) to effectively measure the spatial relationship between the LLC codes. Unlike the geodesic distance which is not globally shape-aware, the biharmonic distance is shape-aware, isometry invariant, computationally efficient, robust to various shape deformations, and possesses good discriminative capabilities [12, 35]. Our contributions are as follows:

1. We extract low-level features from 3D shapes using spectral graph wavelets.
2. We construct mid-level features using the BoF model in which we employ LLC as a feature coding scheme. We then measure the spatial relationship between the LLC codes via the biharmonic distance together with ISPM in order to generate shape-aware bag-of-features.
3. We apply a deep auto-encoder to learn high-level features that are used to design a deep shape-aware descriptor.

The remainder of this chapter is organized as follows. In Section 3.2, we introduce a multi-level 3D shape feature learning framework using deep learning, and we discuss in detail its major components as well as it algorithmic steps. Section 3.3 presents the experimental results and Section 3.4 concludes the chapter.

3.2 Method

In this section, we describe the main components and algorithmic steps of the proposed multi-level feature learning framework. The approach consists of three major components: low-level features, mid-level features and high-level features, as illustrated in Figure 4.1. In the low-level features construction, we use spectral graph wavelets to generate local descriptors for each 3D shape in the dataset. In the mid-level features step, we used the BoF model in conjunction with the biharmonic distance and intrinsic spatial pyramid matching to construct shape-aware global descriptors. In the third step, high-level shape features are learned using deep auto-encoders.

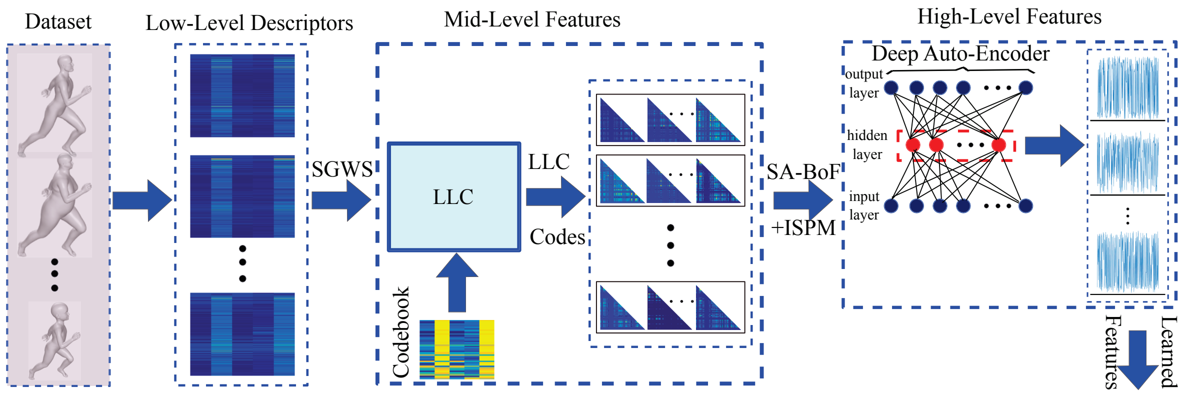


Figure 3.1: Main components of the proposed feature learning method: low-level features, mid-level features and high-level features.

3.2.1 Global Descriptors

A major drawback of the BoF model is that it only considers the distribution of the codewords and disregards all information about the spatial relations between features, and hence the descriptive ability and discriminative power of the BoF paradigm may be negatively impacted. To circumvent this limitation, two major classes of approaches have been recently proposed.

The first class includes approaches define a global descriptor of a 3D shape as a $k \times k$ matrix $\mathbf{F}$ given by

$$\mathbf{F} = \mathbf{U}\mathbf{K}\mathbf{U}^{\mathsf{T}}, \tag{3.1}$$

where $\mathbf{U}$ is a $k \times m$ matrix of sparse codes, and $\mathbf{K} = (\kappa_{ij})$ is a spatial relationship measurements matrix, such as the heat kernel in the spatially sensitive bags of features (SS-BoF) approach [47] or the geodesic exponential kernel in the geodesic-aware bags of features (GA-BoF) framework [49]. Although the geodesic distance has proven to be effective in tackling nonrigid 3D shape matching and retrieval [71, 72] due in large part to its isometry invariance property, it suffers, however, from several disadvantages compared to the (squared) biharmonic distance [35]. While the geodesic distance is not smooth, sensitive to topological noise and not globally shape-aware, the biharmonic distance is not only robust to noise and small topological changes, but also globally shape-aware and smooth.

In addition to its isometry invariance, the biharmonic distance is practical to compute, and strikes a balance between nearly geodesic distances for small distances and global shape-awareness for large distances. Besides, employing biharmonic distance as a spatial relationship measurements matrix avoids the parameter tuning which is necessary for heat kernel (time scale) and geodesic exponential kernel (kernel width). Inspired by these nice properties, we define a shape-aware descriptor of a 3D shape as a $k \times k$ matrix $\mathbf{F}$ by replacing $\mathbf{K}$ in (3.1) with an $m \times m$ biharmonic distance matrix whose elements are defined in terms of the eigenvalues and eigenfunctions of the LBO as follows:

$$\kappa_{ij} = \sum_{\ell=1}^{m} \frac{1}{\lambda_\ell^2} (\boldsymbol{\varphi}_\ell(i) - \boldsymbol{\varphi}_\ell(j))^2. \tag{3.2}$$

In this case, we refer to $\mathbf{F}$ as a shape-aware bag-of-features (SA-BoF) matrix, which indicates the occurrence distribution of the codewords and the spatial relationships between them. Hence, for each 3D shape, the mid-level features are represented by a $k \times k$ matrix $\mathbf{F}$ containing global descriptors.

On the other hand, the second class of approaches includes the intrinsic spatial pyramid matching (ISPM) method [69], which considers the distribution of local descriptors in different spatial patches by the intrinsic spatial partitions. Motivated by the invariance properties of the second eigenfunction $\boldsymbol{\varphi}_2$ of the LBO, Li *et al.* [69] proposed to use the level sets (isocontours) of $\boldsymbol{\varphi}_2$ as cuts to partition a surface. Examples of the level curves of $\boldsymbol{\varphi}_2$ are shown in Figure 3.2. Instead of representing the whole shape by the codeword model without considering spatial layout of local descriptors, each shape cut is represented by isocontours at resolution $\mathtt{s}$ according to its description $\mathbf{H}$ which is the concatenation of $\mathtt{s}$ sub-histograms:

$$\mathbf{H} = [\mathbf{h}^1, \mathbf{h}^2, \ldots, \mathbf{h}^i, \ldots, \mathbf{h}^{\mathtt{s}}], \tag{3.3}$$

where $\mathbf{h}^i$ is the sub-histogram ordered in the ith position according to the intrinsic spatial partition from one end to the other. Note that the isocontours sequence may begin from either end even for the shapes from the same categories. This difference in isocontours sequence is shown in Figure 3.2. For example, the heads of the first and third man in the first row are colored blue but for the second and fourth one are colored red, whose orders are exactly the opposite. In order to make sure that the semantic correspondent parts are considered in the comparison, an order-insensitive strategy comparison method is used. First, a new histogram $\mathbf{T}$ is defined by making the order of the sub-histogram inverted in $\mathbf{H}$ as follows:

$$\mathbf{T} = [\mathbf{h}^{\mathtt{s}}, \mathbf{h}^{\mathtt{s}-1}, \ldots, \mathbf{h}^i, \ldots, \mathbf{h}^1]. \tag{3.4}$$

Then, the difference between two shapes $\mathbb{M}_a$ and $\mathbb{M}_b$ is measured using the dissimilarity given by

$$\mathcal{B}^{\mathtt{s}}(\mathbb{M}_a, \mathbb{M}_b) = \min(\mathcal{A}^{\mathtt{s}}(H_{\mathbb{M}_a}, H_{\mathbb{M}_b}), \mathcal{A}^{\mathtt{s}}(H_{\mathbb{M}_a}, T_{\mathbb{M}_b})), \tag{3.5}$$

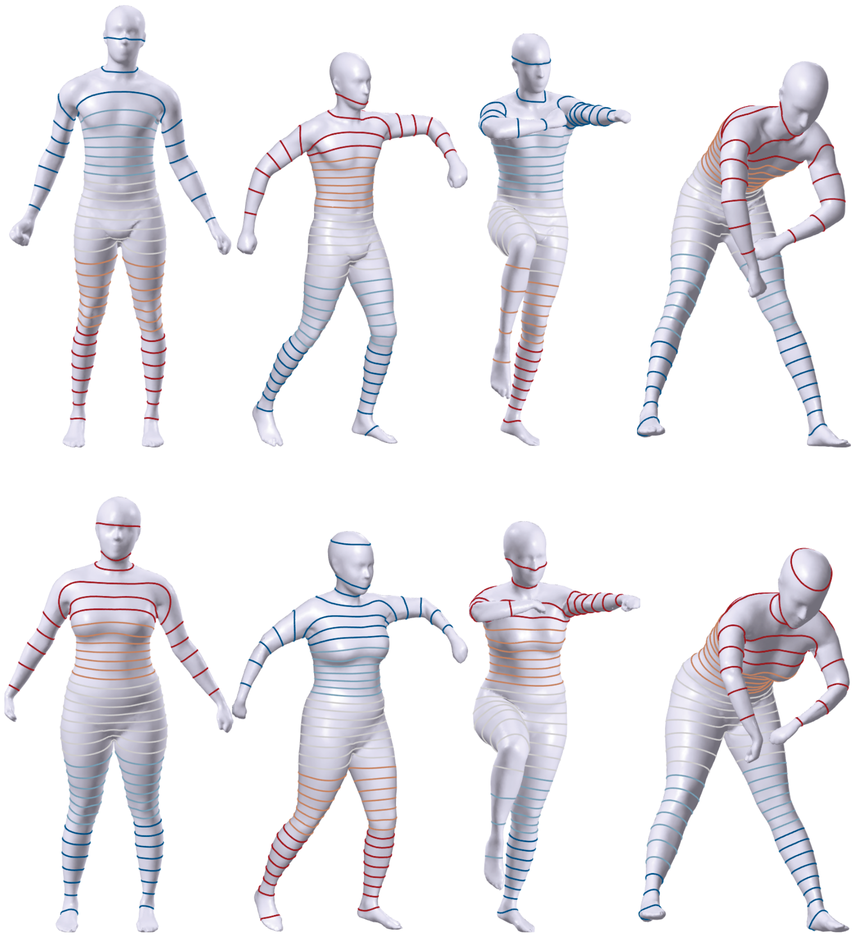


Figure 3.2: Level curves of the second eigenfunction of the LBO. The isocontours sequence may begin from either end even for the shapes from the same categories. The top two rows are some samples from the SHREC-2014 dataset and the bottom two rows from the SHREC-2015 dataset.

where $H_{\mathbb{M}_a}$ and $H_{\mathbb{M}_b}$ denote the histograms of $\mathbb{M}_a$ and $\mathbb{M}_b$, respectively. In other words, there are two possible matching schemes between two shapes based on their isocontours sequences, head-to-head and head-to-end. The schemes with the minimum cost to be better matched are considered. For each scheme, the dissimilarity measure $\mathcal{A}^{\mathtt{s}}(\cdot,\cdot)$ is defined as

$$\mathcal{A}^{\mathtt{s}}(H_{\mathbb{M}_a}, H_{\mathbb{M}_b}) = \sum_{i=1}^{\mathtt{s}} \sum_{j=1}^{\mathtt{k}} \Psi(h^i_{\mathbb{M}_a}(j), h^i_{\mathbb{M}_b}(j)), \tag{3.6}$$

where $\Psi(\cdot,\cdot)$ can be any histogram comparison metric. The spatial pyramid divides an image into a multi-level pyramid of increasingly fine subregions and computes a codebook descriptor for each subregion. A sequence of histograms at resolutions $\{\mathtt{s} = 2^c, c = 0, \ldots, C\}$ is constructed such that the surface at level c has 2^c patches, for a total of $2^C - 1$ patches. Thus, the dissimilarity between the histograms of $\mathbb{M}_r$ and $\mathbb{M}_s$ is given by

$$\begin{aligned}\mathcal{D}^C(\mathbb{M}_r, \mathbb{M}_s) = \mathcal{B}^C(\mathbb{M}_r, \mathbb{M}_s)\\ &+ \sum_{c=0}^{C-1} \frac{1}{2^{C-c}}(\mathcal{B}^c(\mathbb{M}_r, \mathbb{M}_s) - \mathcal{B}^{c+1}(\mathbb{M}_r, \mathbb{M}_s))\\ &= \frac{1}{2^C}\mathcal{B}^0(\mathbb{M}_r, \mathbb{M}_s) + \sum_{c=1}^{C} \frac{1}{2^{C-c+1}}\mathcal{B}^c(\mathbb{M}_r, \mathbb{M}_s).\end{aligned}$$

The weight associated with each level is set to $1/2^{C-c}$, which is inversely proportional to the cell width at that level. Intuitively, the matches found in larger cells are penalized because they involve increasingly dissimilar features.

In order to measure the spatial relations between codewords, we propose instead of extracting bag-of-features for each intrinsic spatial partition, and a $k \times k$ SA-BoF matrix $\mathbf{F}$ can be extracted for each shape patch. It is worth noting that the matrix $\mathbf{F}$ is always symmetric, as depicted shown in Figure 3.3. Therefore, it suffices to use either the upper or lower triangular part of $\mathbf{F}$. Consequently, the $k \times k$ global descriptor matrix (viewed as an image) $\mathbf{F}$ can be compactly represented by a q-dimensional feature vector $\mathbf{f}$, where $\mathsf{q} = k(k+1)/2$.

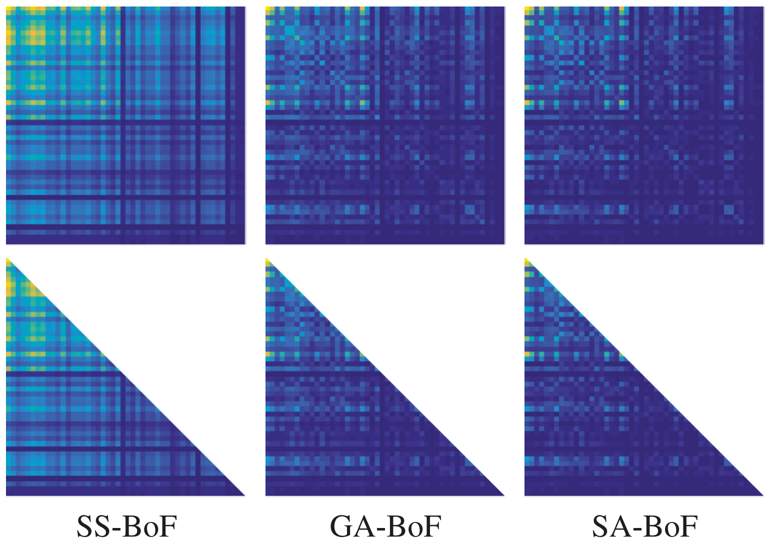


Figure 3.3: The symmetry of spatially sensitive (left), geodesic-aware (middle), and biharmonic-aware (right) bag-of-features matrices. These matrices are shown in the top row, whereas their lower triangular parts are shown in the bottom row.

Hence, for any shape cut by isocontours of φ_2 at resolution s, we obtain s feature vectors, each of which is q-dimensional, as shown in Figure 3.4. We need to concatenate these s feature vectors to form mid-level feature vectors, but the fact that the isocontours sequence may start from either end can be problematic. Li *et al.* [69] defined two histograms and computed both possible matching schemes via (3.5). However, we cannot have two sets of features since mid-level features are going to be fed into a deep auto-encoder. Alternatively, we concatenate these feature vectors for each shape according to the histogram $\mathbf{H} = [\mathbf{f}^1, \mathbf{f}^2, \ldots, \mathbf{f}^i, \ldots, \mathbf{f}^{\mathrm{s}}]$ model if in most cases the minimum dissimilarity between the shape and other shapes obtained by this histogram model. Otherwise, we use the histogram $\mathbf{T} = [\mathbf{f}^{\mathrm{s}}, \mathbf{f}^{\mathrm{s}-1}, \ldots, \mathbf{f}^i, \ldots, \mathbf{f}^1]$.

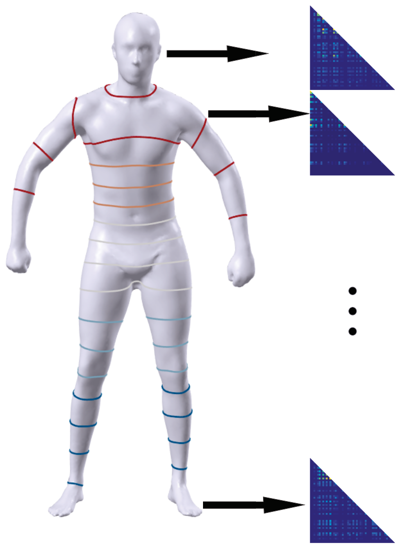


Figure 3.4: Combining intrinsic spatial pyramid matching (ISPM) and shape-aware bag-of-features (SA-BoFs) are extracted for each intrinsic spatial partition.

3.2.2 High-Level Features

In the third step of our framework, more discriminative 3D shape descriptors are extracted using high-level features learned by performing a deep auto-encoder on the mid-level features. Unlike images, a 3D mesh cannot be fed directly into a deep learning model. To tackle this issue, we use κ-dimensional vector $\mathbf{x}$ as an input to the deep auto-encoder, where $\mathbf{x}$ is obtained by concatenating all SA-BoF vectors extracted from each shape patch that we refer to as SA-BoF+ISPM vector, and $\kappa = \mathtt{sq}$. The high-level features are then extracted from the output of the last hidden layer of the deep auto-encoder, resulting in an r_L-dimensional high-level feature vector $\mathbf{a}_L$, which we refer to as a deep learned shape descriptor (DLSD), where r_L is the total number of neurons in the last hidden layer.

3.2.3 Algorithm

The goal of 3D shape retrieval is to search and extract the most relevant shapes to a query object from a dataset of 3D shapes. The retrieval accuracy is usually evaluated by computing a dissimilarity measure between pairs of 3D shapes in the dataset. A commonly used dissimilarity measure for content-based retrieval is the ℓ_1-distance, which quantifies the difference between each pair of 3D shapes.

As stated previously, our learning framework consists of three main components. In the first step, we represent each 3D shape in the dataset by a spectral graph wavelet signature matrix, which is a feature matrix whose columns are the local shape descriptors. More specifically, let $\mathcal{D}$ be a dataset of n shapes modeled by triangle meshes $\mathbb{M}_1, \ldots, \mathbb{M}_n$. We represent each mesh $\mathbb{M}_i$ by a $p \times m$ spectral graph wavelet signature matrix $\mathbf{S}_i$, where m is the number of mesh vertices. The spectral graph wavelet signatures are then encoded via LLC, resulting in a $k \times m$ matrix $\mathbf{U}_i$ whose columns are the k-dimensional LLC codes. In the second step, we cut each shape into $\mathtt{s}$ intrinsic spatial partitions using the level sets (i.e. isocontours) of the second eigenfunction of LBO. The $k \times k$ SA-BoF matrix $\mathbf{F}_{ij}$, where $j = 1, ..., \mathtt{s}$, is computed for j-th intrinsic spatial partition using the LLC codes matrix and the biharmonic distance kernel matrix, followed by shortening $\mathbf{F}_{ij}$ into a $\mathtt{q}$-dimensional SA-BoF vector $\mathbf{f}_{ij}$. Then, these $\mathtt{s}$ vectors are concatenated together to form SA-BoF+ISPM vector $\mathbf{x}_i$. In the third step, SA-BoF+ISPM vectors $\mathbf{x}_i$ of all n shapes in the dataset are arranged into a $\kappa \times n$ data matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ on which a deep auto-encoder is performed, resulting in an $r_L \times n$ matrix $\mathbf{A} = (\mathbf{a}_L^{(1)}, \ldots, \mathbf{a}_L^{(n)})$ whose columns are deep learned shape descriptors, where r_L is the total number of units in the last hidden layer of the network, and $\kappa = \mathtt{sq}$. Finally, we compare a query shape to all shapes in the dataset using ℓ_1-distance to measure the dissimilarity between each pair for 3D shape retrieval. We summarize our multi-level 3D shape descriptor approach in Algorithm 3.

3.3 Experiments

To evaluate the efficacy and performance of our method on 3D shape retrieval tasks, we conducted several experiments and comparisons.

Datasets. We tested the proposed algorithm on two standard and publicly available 3D shape benchmarks: SHREC 2014 and SHREC 2015. The SHREC-2014 benchmark contains two datasets: real and synthetic human models. The real SHREC-2014 dataset is made up of 'real' data, obtained by scanning real human participants [5], and it consists of 400 shapes, made up of 40 human subjects in 10 different poses. Half the human subjects are male, and half female. The poses of each subject are built by using a data-driven deformation technique, which can produce realistic deformations of articulated meshes. The synthetic SHREC-2014 dataset was built using DAZ Studio and consists of 300 human models (adults and children) subdivided into 15 classes of 20 members each. Objects are considered as part of the same class if they share the same body shape.

Algorithm 2 Deep Learned Shape Descriptor (DLSD)

```
Input: Dataset $\mathcal{D} = \{\mathbb{M}_1, \dots, \mathbb{M}_n\}$ of 3D shapes and a query, and resolution parameter $\mathtt{s}$.
 1: for $i = 1$ to $n$ do
 2:     Compute the $p \times m$ SGWS matrix $\mathbf{S}_i$ for each shape $\mathbb{M}_i$
 3:     Compute the $k \times m$ LLC codes matrix $\mathbf{U}_i$
 4:     Cut each shape into $\mathtt{P}$ intrinsic spatial partitions
 5:     for $j = 1$ to $\mathtt{s}$ do
 6:         Compute the $\mathtt{q}$-dimensional SA-BoF vector $\mathbf{f}_{ij}$ for each shape partition $j$, where $\mathtt{q} = k(k+1)/2$.
 7:     end for
 8:     Concatenate all SA-BoF vectors $\mathbf{f}_{ij}$, and reshape them into a $\kappa$-dimensional SA-BoF+ISPM vector $\mathbf{x}_i$, where $\kappa = \mathtt{sq}$.
 9: end for
10: Arrange all the $n$ SA-BoF+ISPM vectors into a $\kappa \times n$ data matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$
11: Apply deep auto-encoder on $\mathbf{X}$ to find the $r_L \times n$ deep learned shape descriptor (DLSD) $\mathbf{A} = (\mathbf{a}_L^{(1)}, \dots, \mathbf{a}_L^{(n)})$
12: Compute the $\ell_1$-distance between the DLSD vector of the query and all DLSD vectors in the dataset, and find the closest shape(s).
Output: Retrieved set of most relevant shapes to the query.
```

The SHREC-2015 benchmark is a dataset of 3D shapes consisting of 1200 watertight mesh models from 50 classes [2], where each class contains 24 objects with distinct postures.

Implementation Details. All the experiments were performed on a desktop computer with a CPU Core i5 processor running at 3.4 GHz and 16 GB RAM, and the algorithms were implemented in MATLAB R2016a (version 9.0). We use the first 201 eigenvalues and eigenvectors of the LBO to compute the low-level descriptors and the biharmonic distance. The resolution level of SGWS is set to $R = 2$, that is each local descriptor is of length 5 (i.e. $p = 5$). The number of vertices varies from shape to shape, but it is set to approximately 1000 and 2200 for the shapes in the SHREC-2015 and SHREC-2014 datasets, respectively. For the mid-level features, a codebook of size 5×48 (i.e. setting $k = 48$) is constructed using a representative collection containing 10 local descriptors from each shape, and the LLC codes are computed using 5 nearest neighbors. Then, the mid-level features of all shapes form a data matrix $\mathbf{X}$ of size $\kappa \times n$, where $\kappa = \mathtt{sq}$ with $\mathtt{s} = 2$, $\mathtt{q} = k(k+1)/2 = 1176$, and n is the total number of 3D shapes in the dataset. To compute high-level features, we use a deep auto-encoder consisting of an input layer of size equal to $\kappa = 2352$, two hidden layers of sizes $r_{L-1} = 1000$ and $r_L = 300$, and an output layer of size equal to the number of classes in each 3D shape dataset. This yields a DLSD matrix $\mathbf{A}$ of length $300 \times n$. It is important to note that the dimension of mid-level features was reduced from $\kappa = 2352$ to 300, which indicates the compactness of the DLSD. The regularization parameter in the objective function of the deep auto-encoder is set to $\lambda = 0.001$.

Baseline Methods. We compare the effectiveness of the proposed framework with several state-of-the-art methods, including histograms of area projection transform (HAPT) [73], HKS-TS [1,2], spectral graph wavelet signature (SGWS) [9], Euclidean distance based canonical forms (ED-BCF) [74], supervised dictionary learning (supDLtrain) [48], reduced biharmonic distance matrix (R-BiHDM) [12], and high-level feature learning using deep belief networks (3D-DL) [49]. We also compare our DLSD approach using mid-level features as well as high-level features learned by applying an auto-encoder with only one hidden layer as a shallow model (SLSD) to show the advantage of the deep network in improving the retrieval performance. For all baseline methods, we use the default parameters when available.

3.3.1 Results

We evaluate the retrieval performance of the proposed approach in comparison with existing methods using several standard evaluation metrics, including the precision-recall curve, nearest neighbor (NN), first-tier (FT), second-tier (ST), E-measure (E), discounted cumulative gain (DCG), and mean average precision (mAP).

Retrieval Performance on Mid-Level Features

We compare the retrieval performance of SA-BoF with SS-BoF and GA-BoF in order to evaluate the improvement by using biharmonic distance as the spatial relationship measurement matrix instead of heat kernel or geodesic exponential kernel. We also compare the retrieval performance of SA-BoF+ISPM with the other three mid-level features to evaluate the improvement of combining SA-BoF and ISPM.

Results on SHREC 2015. For this dataset of 1200 shapes, we first compute the SS-BoF, GA-BoF, and SA-BoF, data matrices, which all are of size 1176×1200. Then, a distance matrix of size 1200×1200 is constructed by computing the ℓ_1-distance between each pair of the 1176-dimensional mid-level feature vectors. Finally, a retrieval test on this distance matrix is conducted and the scores for the evaluation metrics are computed. It is important to point out that for fair comparison with SS-BoF and GA-BoF, we generated their corresponding global descriptors using our low-level feature extraction strategy based on SGWS and our mid-level feature extraction strategy based on LLC as feature coding method and we only replace the biharmonic distance as the spatial relationship measurement matrix by heat kernel and geodesic exponential kernel, respectively. We also compute the SA-BoF+ISPM for different values of the resolution parameter $\mathtt{s}$, and the best results are obtained when $\mathtt{s} = 2$, yielding a data matrix of size 2352×1200. Nevertheless, the resulting distance matrix for these mid-level feature is still of size 1200×1200. Table 3.1 shows the retrieval results for all mid-level features. Aa can be seen, SA-BoF+ISPM outperforms the other methods. For example, the mAP and DCG for SS-BoF, GA-BoF, SA-BoF, and SA-BoF+ISPM are 81.2% and 94.0%, 81.2% and 93.9%, 82.1% and 94.4%, and 84.2% and 95.4%, respectively. This table also indicates that SA-BoF outperforms SS-BoF and GA-BoF.

Table 3.1: Performance comparison results of ISPM and single partition mid-level features on the SHREC-2015 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
SS-BoF	96.2	79.2	88.3	70.5	94.0	81.2
GA-BoF	96.5	79.3	87.7	70.3	93.9	81.2
SA-BoF	96.0	80.1	89.3	71.3	94.4	82.1
SA-BoF+ISPM	**97.4**	**82.8**	**90.6**	**73.1**	**95.4**	**84.2**

We also use precision-recall graphs to evaluate the retrieval performance for different mid-level features. A precision-recall graph is an informative graph that illustrates the tradeoff between precision as a function of recall, and it shows the retrieval performance at each point in the ranking. If, for instance, the $(\tau + 1)$-th shape retrieved is relevant, then both precision and recall increase. However, if it is irrelevant then recall is the same as for the top τ shapes, but precision decreases. Hence, a precision-recall graph that is shifted upwards and to the right indicates superior performance. Figure 3.5 compares precision-recall curves of all mid-level features. As can be seen, SA-BoF+ISPM outperforms the other methods significantly as there is a big performance gap between them. However, the performance gap between SA-BoF and SS-BoF or GA-BoF do not seem significant. In order to assess if the retrieval performance is improved significantly or not by replacing the heat kernel and geodesic exponential kernel with the biharmonic distance, we run some statistical tests on NDCG values.

Paired-sample t-test. Denote X and Y two retrieval algorithms, where X is a new algorithm and Y is a baseline. Given n queries, the evaluation scores (e.g. NDCG values) generated by the algorithms X and Y may be represented as n-dimensional vectors $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$, where x_i and y_i are the scores of X and Y for query i. In order to show that algorithm X significantly outperforms algorithm Y, a hypothesis test (also referred to as statistical test or significance test) is usually conducted using as sample data the differences $\delta_i = x_i - y_i$ between the matched pairs of scores for each query. This is an upper-tailed hypothesis test, where the null hypothesis H_0 is that there is no significant difference in performance between X and Y (i.e. the mean of two paired samples are almost equal), and the alternative hypothesis is that X performs significantly better than Y. After setting up the hypotheses, we choose the level of significance α, which is the probability of making the mistake of rejecting H_0 when it is true. In most of the cases,

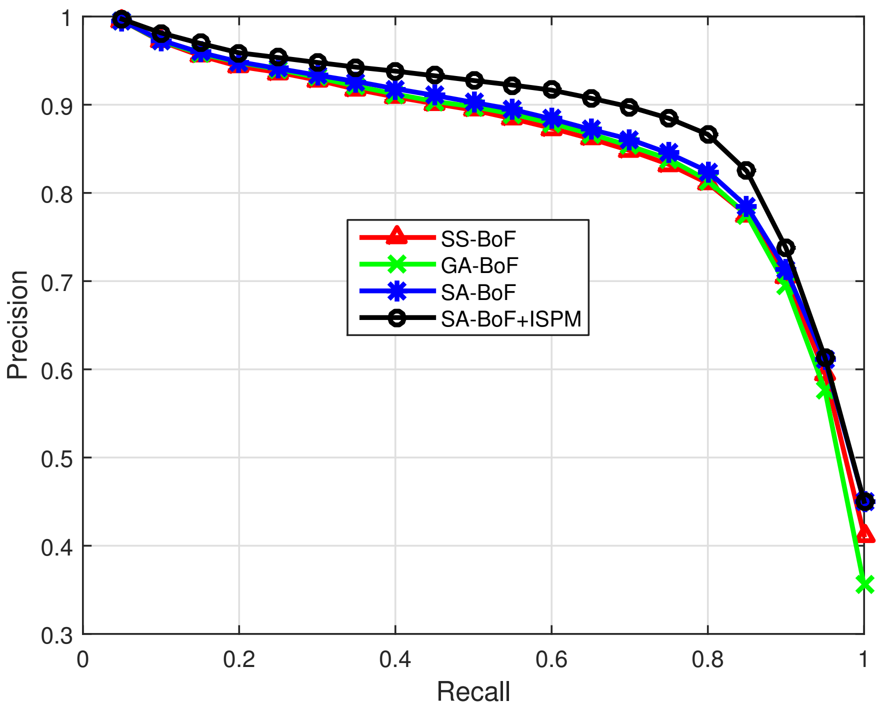


Figure 3.5: Precision-recall curves comparing the performance of ISPM and single partition mid-level features on SHREC 2015.

significance level is 5%. Assuming n is large, we use the paired sample t-test

$$t = \frac{\bar{\delta}}{s/\sqrt{n}}, \tag{3.7}$$

where $\bar{\delta}$ and s are the sample mean and sample standard deviation, respectively, from all the n differences between paired scores. This test statistic follows a t-distribution with $n-1$ degrees of freedom. The value of the test statistic is then used to compute the p-value, which is the probability of observing the given sample result under the assumption that the null hypothesis is true. If the p-value is less than α, then we reject the null hypothesis.

Two-way ANOVA. In the case of several baseline methods, the performance of a new algorithm can be analyzed using a two-way ANOVA with data consisting of observations y_{ij}, which correspond to the score of method j for query i, where $i = 1, \ldots, n$ and $j = 1, \ldots, m$. Each observation can be modeled as

$$y_{ij} = \mu + \tau_i + \beta_j + \varepsilon_{ij}, \tag{3.8}$$

where μ is the overall mean effect (true mean performance), τ_i is the effect of the ith query, β_j is the effect of the jth retrieval algorithm, and ε_{ij} is Gaussian random error with zero mean and variance σ^2. In order to decide whether there is a significant mean difference between retrieval

Table 3.2: Significance comparison results of ISPM and single partition mid-level features in terms of DCG on the SHREC-2015 dataset.

	Significance Tests		
Comparing Methods	p-value (t-test)	f_0 (two-way ANOVA)	Significance
SA-BoF vs. GA-BoF	0.0057	7.66	high
SA-BoF vs. SS-BoF	2.42×10^{-6}	22.44	high
GA-BoF vs. SS-BoF	0.8316	0.05	low
SA-BoF+ISPM vs. SA-BoF	4.82×10^{-5}	16.64	high
SA-BoF+ISPM vs. GA-BoF	1.62×10^{-7}	28.43	high
SA-BoF+ISPM vs. SS-BoF	1.97×10^{-8}	31.96	high

algorithms, we compute the value of the F-test statistic given by

$$f_0 = \frac{MS_B}{MS_E}, \tag{3.9}$$

where MS_B denotes the mean squares of the factor B (i.e. methods), and MS_E denotes the mean square error. Then, the value of the test statistic is compared to $f_{\alpha,m-1,(m-1)(n-1)}$, which is the percentage point of the F-distribution with $m-1$ and $(m-1)(n-1)$ degrees of freedom.

Table 3.2 shows the results of these statistical tests on mid-level features. Since we compare each pair of methods (i.e. $m = 2$), we have $f_{\alpha,m-1,(m-1)(n-1)} = f_{\alpha,1,n-1}$. If $f_0 > f_{\alpha,1,n-1}$, then we reject the null hypothesis, i.e. there is a significant mean difference. The SHREC-2015 dataset consists of $n = 1200$ shapes; so for $\alpha = 0.05$, we have $f_{\alpha,1,n-1} = f_{0.05,1,1199} = 3.85$. As can be seen, SA-BoF improved GA-BoF and SS-BoF significantly, e.g. p-value $= 0.0057 < 0.05$ and $f_0 = 7.66 > 3.85$ for the pair of SA-BoF and GA-BoF, and p-value $= 2.42 \times 10^{-6} < 0.05$ and $f_0 = 22.44 > 3.85$ for the pair of SA-BoF and SS-BoF. However, these results show that GA-BoF was not able to outperform SS-BoF significantly, i.e. p-value $= 0.8316 \not< 0.05$ and $f_0 = 7.66 \not> 3.85$. Furthermore, these tests indicate that SA-BoF+ISPM improves the retrieval performance of other mid-level features significantly. For instance, $f_0 = 16.64$ for the pair of SA-BoF+ISPM and SA-BoF, $f_0 = 28.43$ for the pair of SA-BoF+ISPM and GA-BoF, and $f_0 = 31.96$ for the pair of SA-BoF+ISPM and SS-BoF, all are larger than 3.85. Note that these results also confirm that the performance gap between SA-BoF+ISPM and the other methods is bigger than the one between SA-BoF and the others, e.g. $f_0 = 28.43$ for the pair of SA-BoF+ISPM and GA-BoF, while $f_0 = 7.66$ for the pair of SA-BoF and GA-BoF.

Results on SHREC 2014. Following the setting of the previous experiment for the real SHREC-2014 dataset of 400 shapes, the SS-BoF, GA-BoF, and SA-BoF data matrices are all of size 1176×400. Hence, the resulting distance matrices is of size 400×400. We also compute the SA-

BoF+ISPM for different values of the resolution parameter $\mathtt{s}$, and the best results are obtained when $\mathtt{s} = 2$, resulting in a data matrix of size 2352×400 and a distance matrix of size 400×400. Table 3.3 shows the retrieval results for all mid-level features. As can be seen, SA-BoF+ISPM outperforms the other methods. For example, the mAP and DCG for SS-BoF, GA-BoF, SA-BoF, and SA-BoF+ISPM are 30.6% and 52.4%, 32.8% and 54.1%, 34.0% and 55.0%, and 43.4% and 62.6%, respectively. This table also indicates that SA-BoF outperforms SS-BoF and GA-BoF. Figure 3.6 compares precision-recall curves of all these mid-level features.

Table 3.3: Performance comparison results of ISPM and single partition mid-level features on the real SHREC-2014 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
SS-BoF	26.0	24.3	39.8	24.5	52.4	30.6
GA-BoF	31.8	25.6	42.5	26.2	54.1	32.8
SA-BoF	33.0	26.6	43.9	27.2	55.0	34.0
SA-BoF+ISPM	**48.0**	**36.1**	**54.6**	**31.0**	**62.6**	**43.4**

We can see the performance improvement by ISPM is the most significant compared to the results for the other datasets. For example, NN for SA-BoF+ISPM is improved by 15%, which is the highest improvement by ISPM among all evaluation metrics. The statistical tests provided in Table 3.4 show that the gap between SA-BoF+ISPM and the other mid-level features is large. For instance, $f_0 = 161.94$ for the pair of SA-BoF+ISPM and SA-BoF, $f_0 = 182.33$ for the pair of SA-BoF+ISPM and GA-BoF, and $f_0 = 220.95$ for the pair of SA-BoF+ISPM and SS-BoF all are larger than $f_{0.05,1,399} = 3.86$. These tests also confirm that SA-BoF improved GA-BoF and SS-BoF significantly, e.g. $f_0 = 11.63 > 3.86$ for the pair of SA-BoF and GA-BoF, and $f_0 = 34.24 > 3.86$ for the pair of SA-BoF and SS-BoF. Therefore, ISPM can come in handy, particularly when dealing with challenging datasets such as the real SHREC-2014 benchmark.

For the synthetic SHREC-2014 dataset of 300 shapes, the SS-BoF, GA-BoF and SA-BoF data matrices are all of size 1176×300. Hence, the resulting distance matrices are all of size 300×300. We also compute the SA-BoF+ISPM for different values of the resolution parameter $\mathtt{s}$, and the best results are obtained when $\mathtt{s} = 2$, yielding a data matrix of size 2352×300 and a distance matrix of size 300×300. Table 3.5 shows the retrieval results for all mid-level features. As can be seen, SA-BoF+ISPM outperforms the other methods. For example, the mAP and DCG for SS-BoF, GA-BoF, SA-BoF, and SA-BoF+ISPM are 77.1% and 90.6%, 78.4% and 90.5%, 80.4% and 91.7%, and 80.7% and 91.9%, respectively. This table also indicates that SA-BoF outperforms SS-BoF and GA-BoF.

Table 3.4: Significance comparison results of ISPM and single partition mid-level features in terms of DCG on the real SHREC-2014 dataset.

	Significance Tests		
Comparing Methods	p-value (t-test)	f_0 (two-way ANOVA)	Significance
SA-BoF vs. GA-BoF	0.7×10^{-3}	11.63	high
SA-BoF vs. SS-BoF	1.01×10^{-8}	34.24	high
GA-BoF vs. SS-BoF	1.52×10^{-6}	23.83	high
SA-BoF+ISPM vs. SA-BoF	2.26×10^{-31}	161.94	high
SA-BoF+ISPM vs. GA-BoF	1.75×10^{-34}	182.33	high
SA-BoF+ISPM vs. SS-BoF	4.38×10^{-40}	220.95	high

Table 3.5: Performance comparison results of ISPM and single partition mid-level features on the synthetic SHREC-2014 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
SS-BoF	89.0	68.5	90.5	64.0	90.6	77.1
GA-BoF	90.3	70.6	91.6	65.4	90.5	78.4
SA-BoF	**91.3**	72.4	92.5	66.0	91.7	80.4
SA-BoF+ISPM	91.0	**72.6**	**92.7**	**66.1**	**91.9**	**80.7**

Figure 3.6 compares precision-recall curves of all these mid-level features. Although it can be seen that SA-BoF+ISPM outperforms GA-BoF and SS-BoF significantly; it is not the case for SA-BoF. The statistical tests provided in Table 3.6 indicate that there is no significant mean difference between the retrieval performance of SA-BoF+ISPM and SA-BoF as p-value $= 0.4441 \nless 0.05$ and $f_0 = 0.59 \ngtr f_{0.05,1,399} = 3.86$. This is due in part to the fact that the biharmonic distance can measure the spatial relations quite well enough as the 3D shapes are more visible in this dataset. Nevertheless, these statistical tests show that SA-BoF+ISPM outperforms GA-BoF and SS-BoF significantly, e.g. $f_0 = 27.48 > 3.86$ for the pair of SA-BoF+ISPM and GA-BoF, and $f_0 = 15.14 > 3.86$ for the pair of SA-BoF+ISPM and SS-BoF. Moreover, Table 3.6 shows that SA-BoF improved GA-BoF and SS-BoF significantly, e.g. $f_0 = 38.92 > 3.86$ for the pair of SA-BoF and GA-BoF, and $f_0 = 16.71 > 3.86$ for the pair of SA-BoF and SS-BoF.

In view of the superiority of SA-BoF+ISPM for all datasets, we choose it as the final mid-level features data matrix $\mathbf{X}$ to feed into a deep auto-encoder in order to extract the high-level features.

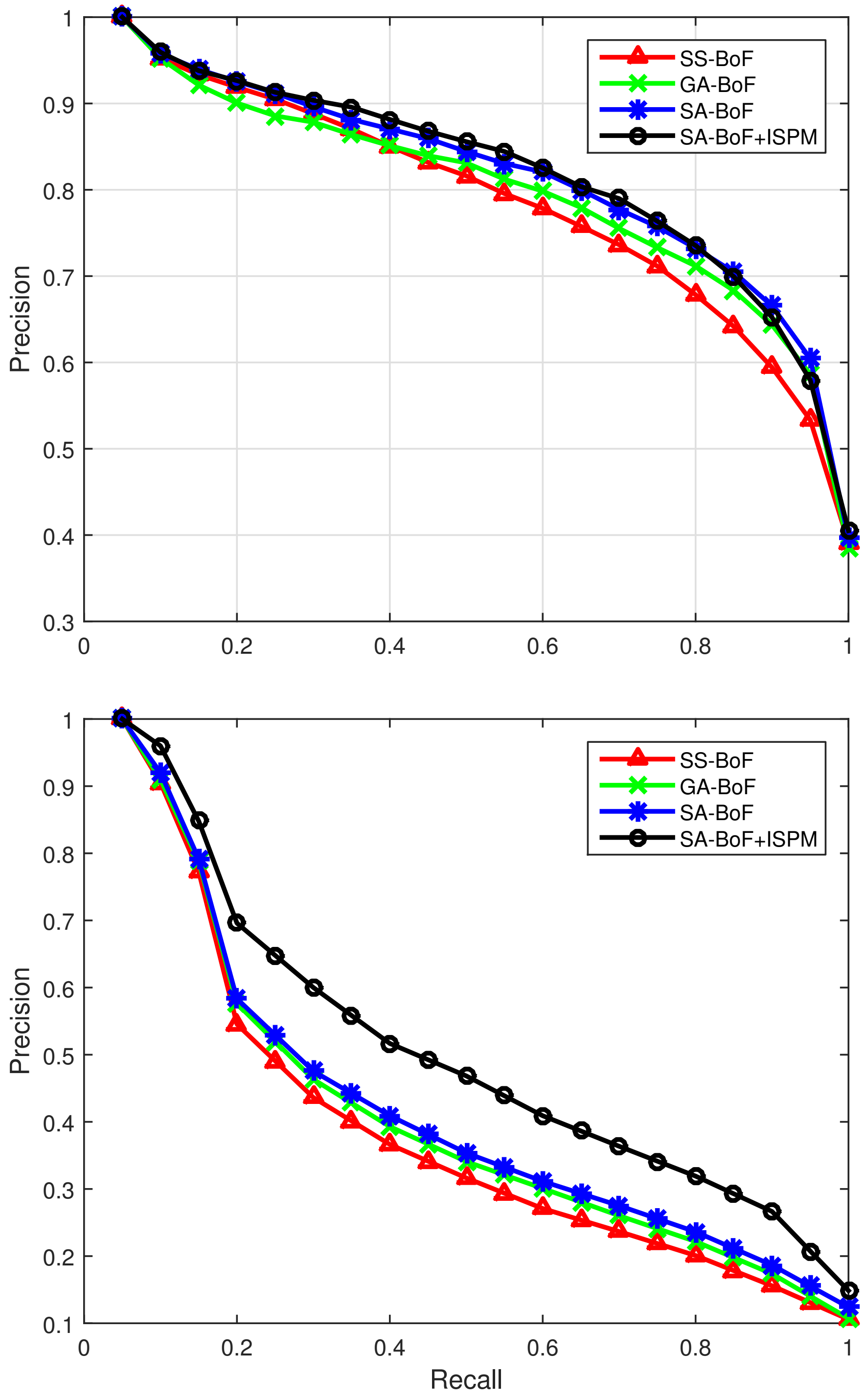


Figure 3.6: Precision-recall curves comparing the performance of ISPM and single partition mid-level features on the synthetic SHREC-2014 (up) and the real SHREC-2014 datasets (down).

Retrieval Performance on High-Level Features

In this subsection, we evaluate the performance of the high-level features learned by performing a deep auto-encoder on SA-BoF+ISPM that showed the highest performance on the retrieval tests

Table 3.6: Significance comparison results of ISPM and single partition mid-level features in terms of DCG on the synthetic SHREC-2014 dataset.

	Significance Tests		
Comparing Methods	p-value (t-test)	f_0 (two-way ANOVA)	Significance
SA-BoF vs. GA-BoF	1.51×10^{-9}	38.92	high
SA-BoF vs. SS-BoF	5.61×10^{-5}	16.71	high
GA-BoF vs. SS-BoF	0.8373	0.04	low
SA-BoF+ISPM vs. SA-BoF	0.4441	0.59	low
SA-BoF+ISPM vs. GA-BoF	3.00×10^{-7}	27.48	high
SA-BoF+ISPM vs. SS-BoF	1.00×10^{-4}	15.14	high

conducted in the previous subsection.

Results on SHREC 2015. For this dataset, the mid-level features data matrix $\mathbf{X}$ is SA-BoF+ISPM, which is of size 2352×1200. Training the deep auto-encoder on the training dataset yields learned features that form a deep learned shape descriptor (DLSD) data matrix $\mathbf{A}$ of size 300×1200. Hence, the resulting distance matrix is of size 1200×1200. Table 3.7 shows the retrieval results of DLSD and several baseline methods. We also trained a shallow architecture using the auto-encoder with only one hidden layer of size 300 that yields learned features, forming a data matrix called shallow learned shape descriptor (SLSD) of size 300×1200. We include the retrieval results for SLSD in order to evaluate the performance improvement by deep learning. As can be seen in the table, DLSD outperforms all baseline methods on almost all the evaluation metrics, except HAPT [73] which achieves a slightly higher NN value. Moreover, the performance gap between DLSD and HAPT is significant in terms of the other evaluation metrics, indicating that the proposed approach performs significantly better than the competitors.

Note that using the deep auto-encoder to extract high-level features improves the retrieval performance of SA-BoF+ISPM by 10% and 4% in terms of mAP and DCG, respectively. DLSD outperforms the shallow model SLSD, strongly suggesting that deep models can improve the retrieval results. Overall, DLSD is consistently the best, delivering robust retrieval performance.

Figure 3.7 compares the proposed framework with several baseline methods using precision-recall curves on the SHREC-2015 benchmark. As can be seen, DLSD performs significantly better than the competitors. It is important to point out that for fair comparison with SI-HKS and WKS, which also are local descriptors, we generated their corresponding global descriptors using our mid-level feature extraction strategy based on LLC as feature coding method and the biharmonic distance as the kernel for feature pooling combined with ISPM for $\mathtt{s} = 2$. As illustrated in the figure, the precision-recall graphs indicate the superiority of the proposed method. Inter-

Table 3.7: Performance comparison results of the proposed method on the SHREC-2015 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
HAPT [73]	**99.8**	96.6	98.2	81.5	99.2	-
HKS-TS [2]	6.5	6.4	12.4	7.4	39.1	-
SGWS [9]	97.3	76.0	81.4	66.0	91.9	-
EDBCF [74]	97.8	79.1	88.4	70.8	94.3	-
SA-BoF+ISPM	97.4	82.8	90.6	73.1	95.4	84.2
SLSD	99.3	98.0	99.0	82.4	**99.4**	93.9
DLSD	99.3	**98.6**	**99.3**	**82.9**	**99.4**	**94.2**

estingly, Shape-DNA, which is the simplest spectral descriptor, outperforms SI-HKS. In addition, SA-BoF+ISPM yields better retrieval results than WKS and SI-HKS.

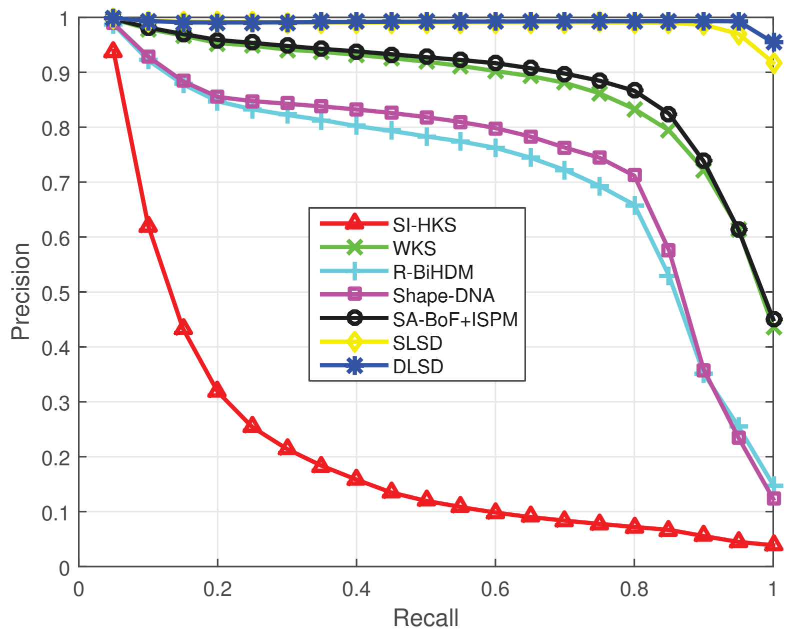


Figure 3.7: Precision-recall curves comparing the performance of the proposed method with other state-of-the-art approaches on SHREC 2015.

In order to show that deep learning helps improve the retrieval performance significantly, we run several statistical tests, and the results are listed in Table 3.8. As can be seen, both high-level features DLSD and SLSD improved the retrieval performance significantly compared to SA-BoF+ISPM, e.g. $f_0 = 246.58 > 3.85$ for the pair of DLSD and SA-BoF+ISPM, and $f_0 = 247.28 > 3.85$ for the pair of SLSD and SA-BoF+ISPM. However, these tests demonstrate

Table 3.8: Significance comparison results of the proposed method in terms of DCG on the SHREC-2015 dataset.

	Significance Tests		
Comparing Methods	p-value (t-test)	f_0 (two-way ANOVA)	Significance
SA-BoF+ISPM vs. DLSD	1.12×10^{-50}	246.58	high
SA-BoF+ISPM vs. SLSD	8.42×10^{-51}	247.28	high
DLSD vs. SLSD	0.14	2.19	low

that high-level features extracted by a deep model (DLSD) do not outperform the ones extracted by a shallow model (SLSD) significantly as p-value $= 0.14 \nless 0.05$ and $f_0 = 2.19 \ngtr 3.85$. Therefore, DLSD outperforms all baseline methods , with the exception of SLSD.

Results on SHREC 2014. For the real SHREC-2014 dataset, the mid-level features data matrix $\mathbf{X}$ is SA-BoF+ISPM , which is of size 2352×400 and the DLSD data matrix $\mathbf{A}$ is of size 300×400. Hence, the resulting distance matrices are both of size 400×400. For the synthetic SHREC-2014 dataset, the SA-BoF+ISPM data matrix $\mathbf{X}$ of size 2352×300 is the mid-level features data matrix $\mathbf{X}$, the DLSD data matrix $\mathbf{A}$ is of size 300×300, and the resulting distance matrices are both of size of 300×300. We also extract SLSD by training an auto-encoder with only one hidden layer of size 300. Tables 3.9 and 3.10 compare the retrieval results of SA-BoF+ISPM, SLSD and DLSD with baseline methods.

Table 3.9: Performance comparison results of the proposed method on the real SHREC-2014 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
HAPT [73]	84.5	53.4	68.1	35.5	79.5	63.7
HKS-TS [2]	24.5	25.9	46.1	31.4	54.8	-
SGWS [9]	31.3	20.6	32.3	19.2	48.8	25.8
EDBCF [74]	1.0	1.2	4.0	4.3	27.9	-
supDLtrain [48]	79.3	72.7	**91.4**	**43.2**	89.1	79.1
R-BiHDM [12]	68.5	54.1	74.2	38.7	78.1	64.0
3D-DL [49]	22.5	19.3	37.4	26.2	50.4	-
SA-BoF+ISPM	48.0	36.1	54.6	31.0	62.6	43.4
SLSD	84.0	69.6	81.2	38.5	86.7	77.7
DLSD	**86.0**	**78.6**	89.6	41.3	**91.1**	**86.5**

As reported in Table 3.9, the performance gap between DLSD and SA-BoF+ISPM on the chal-

Table 3.10: Performance comparison results of the proposed method on the synthetic SHREC-2014 dataset. Boldface numbers indicate the best retrieval performance.

	Retrieval Evaluation Measures (%)					
Method	NN	FT	ST	E	DCG	mAP
HAPT [73]	97.0	73.3	92.7	65.5	93.6	81.7
HKS-TS [2]	46.7	47.6	74.3	50.4	72.9	-
SGWS [9]	99.3	83.2	97.1	70.6	97.1	90.2
EDBCF [74]	11.3	18.2	33.3	21.7	50.7	-
supDLtrain [48]	96.0	88.7	99.1	72.1	97.5	95.4
R-BiHDM [12]	79.3	57.2	76.0	53.3	83.6	64.2
3D-DL [49]	92.3	76.0	91.1	64.1	92.1	-
SA-BoF+ISPM	91.0	72.6	92.7	66.1	91.9	80.7
SLSD	97.0	93.0	98.8	72.4	98.2	96.5
DLSD	**99.7**	**98.0**	**99.8**	**74.0**	**99.5**	**98.9**

lenging, real SHREC-2014 benchmark grows by 43.1% in mAP and 28.5% in DCG, which is a significant improvement. Likewise, SLSD is improved by 8.8% in mAP and 4.4% in DCG by DLSD, indicating that deep learning models can further improve the retrieval results compared to shallow ones. Moreover, Table 3.9 also indicates the proposed method improves the original SGWS significantly e.g. it increases DCG from 48.8% to 91.1%, and mAP from 25.8% to 86.5%. The supDLtrain, which is a supervised learning method, yields slightly better ST and E scores, while DLSD significantly outperforms supDLtrain in terms of the other scores on the real SHREC-2014 dataset. 3DDL, which is a deep learning based approach, achieves a mediocre retrieval performance on the real SHREC-2014 dataset.

The statistical tests provided in Table 3.11 indicate that DLSD outperforms SA-BoF+ISPM and SLSD significantly, e.g. $f_0 = 781.3 > 3.85$ for the pair of DLSD and SA-BoF+ISPM, and $f_0 = 73 > 3.85$ for the pair of DLSD and SLSD. As can be seen, the most significant retrieval performance improvement by DLSD is occurred on this dataset, clearly indicating the importance of using high-level features in further improving the retrieval results in dealing with a challenging benchmark such as real SHREC 2014. From Table 3.10, we see that DLSD improves SA-BoF+ISPM on the synthetic SHREC-2014 dataset by 18.2% and 7.6% in terms of mAP and DCG, respectively. As the models in the synthetic dataset are more visible, the performance gap between SLSD and DLSD is not as big as the one for the real dataset. However, the high-level features learned by a deep architecture still achieve better results than the ones learned by a shallow model. The statistical tests provided in Table 3.12 show that DLSD outperforms SA-BoF+ISPM and SLSD significantly, e.g. $f_0 = 205.45 > 3.86$ for the pair of DLSD and SA-BoF+ISPM, and

$f_0 = 15.62 > 3.86$ for the pair of DLSD and SLSD. Moreover, DLSD performs the best among all the baseline methods on the synthetic SHREC-2014 benchmark.

Figure 3.8 shows the performance comparison of the proposed method with various baseline methods using the precision-recall graphs on the real and synthetic SHREC-2014 datasets. As can be seen in the figure, the precision-recall graphs indicate the superiority of the proposed method. Note that even for full recall, the precision is still higher than 0.68. Shape-DNA outperforms SI-HKS on the synthetic SHREC-2014 dataset. In addition, WKS achieves better performance than SI-HKS on all three datasets, providing further evidence that WKS outperforms HKS as reported in [8]. Moreover, SA-BoF+ISPM yields better retrieval results than WKS and SI-HKS on SHREC 2014, which strengthens our view that SGWS has a more discriminative ability than WKS and SI-HKS. This is largely attributed to the fact that SGWS captures geometric information at multiple scales. SLSD also outperforms all baseline methods and SA-BoF+ISPM, which is a powerful testimony of higher discrimination power of high-level features learned even by a shallow model. As illustrated in the figure, DLSD beats SLSD, which indicates the advantages of using a deep learning approach.

Table 3.11: Significance comparison results of the proposed method in terms of DCG on the real SHREC-2014 dataset.

	Significance Tests		
Comparing Methods	p-value (t-test)	f_0 (two-way ANOVA)	Significance
SA-BoF+ISPM vs. DLSD	4.84×10^{-96}	781.3	high
SA-BoF+ISPM vs. SLSD	2.57×10^{-99}	827.23	high
DLSD vs. SLSD	2.78×10^{-16}	73	high

The high-level features learned by our deep learning approach can be visualized using the t-Distributed Stochastic Neighbor Embedding (t-SNE) [77], which is a dimensionality reduction technique that is particularly well-suited for embedding high-dimensional data into a space of

Table 3.12: Significance comparison results of the proposed method in terms of DCG on the synthetic SHREC-2014 dataset.

	Significance Tests		
Comparing Methods	p-value (t-test)	f_0 (two-way ANOVA)	Significance
SA-BoF+ISPM vs. DLSD	7.92×10^{-36}	205.45	high
SA-BoF+ISPM vs. SLSD	4.10×10^{-31}	170.29	high
DLSD vs. SLSD	9.66×10^{-5}	15.62	high

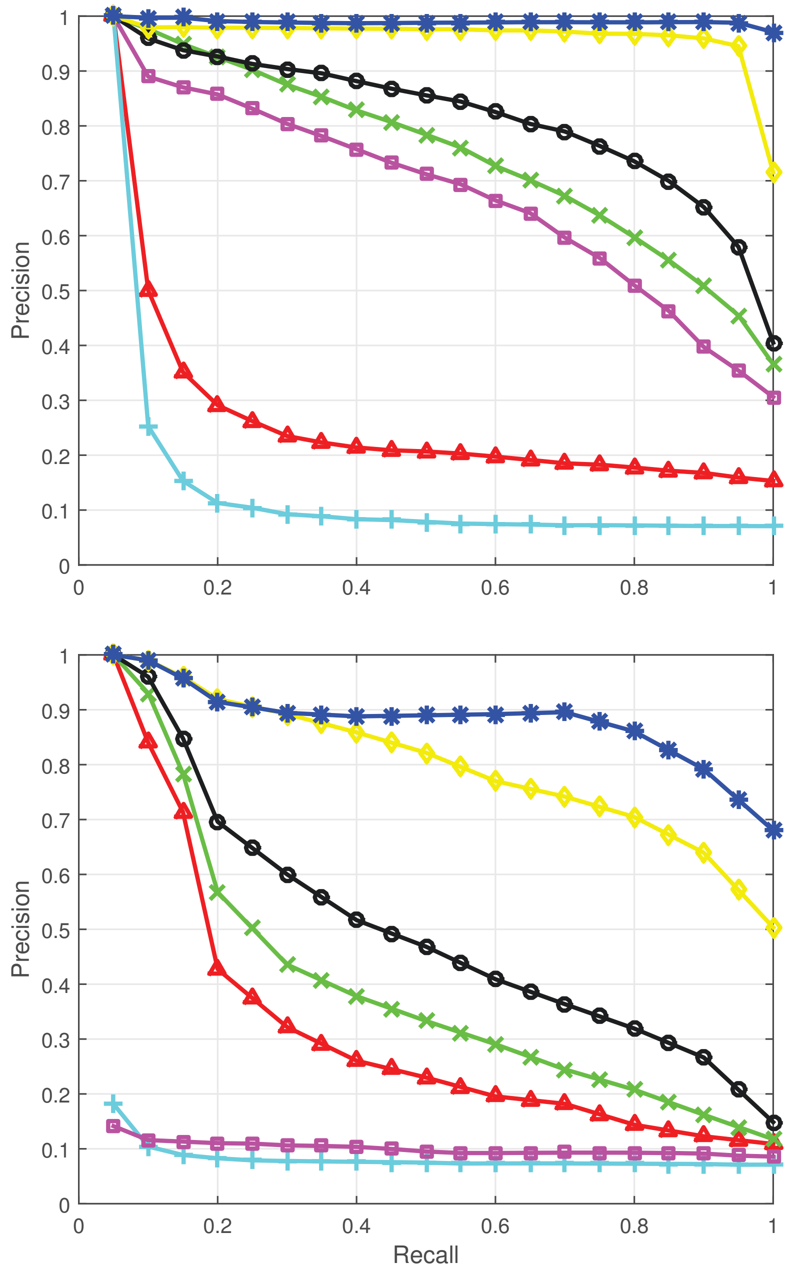


Figure 3.8: Precision-recall curves comparing the performance of the proposed method with other state-of-the-art approaches on the synthetic SHREC-2014 (up) and the real SHREC-2014 datasets (down).

two or three dimensions. The two-dimensional plots (i.e. t-SNE embeddings) in Figure 4.8 were generated by applying the t-SNE algorithm to DLSD and SA-BoF+ISPM for all datasets. The plots in Figure 4.8 (d)-(f) show that the two-dimensional embeddings corresponding to DLSD are more separable than the ones corresponding to SA-BoF+ISPM, as illustrated in Figure 4.8 (a)-(c).

The accuracy of the retrieval results using DLSD is further illustrated in Figures 3.10 and 3.11.

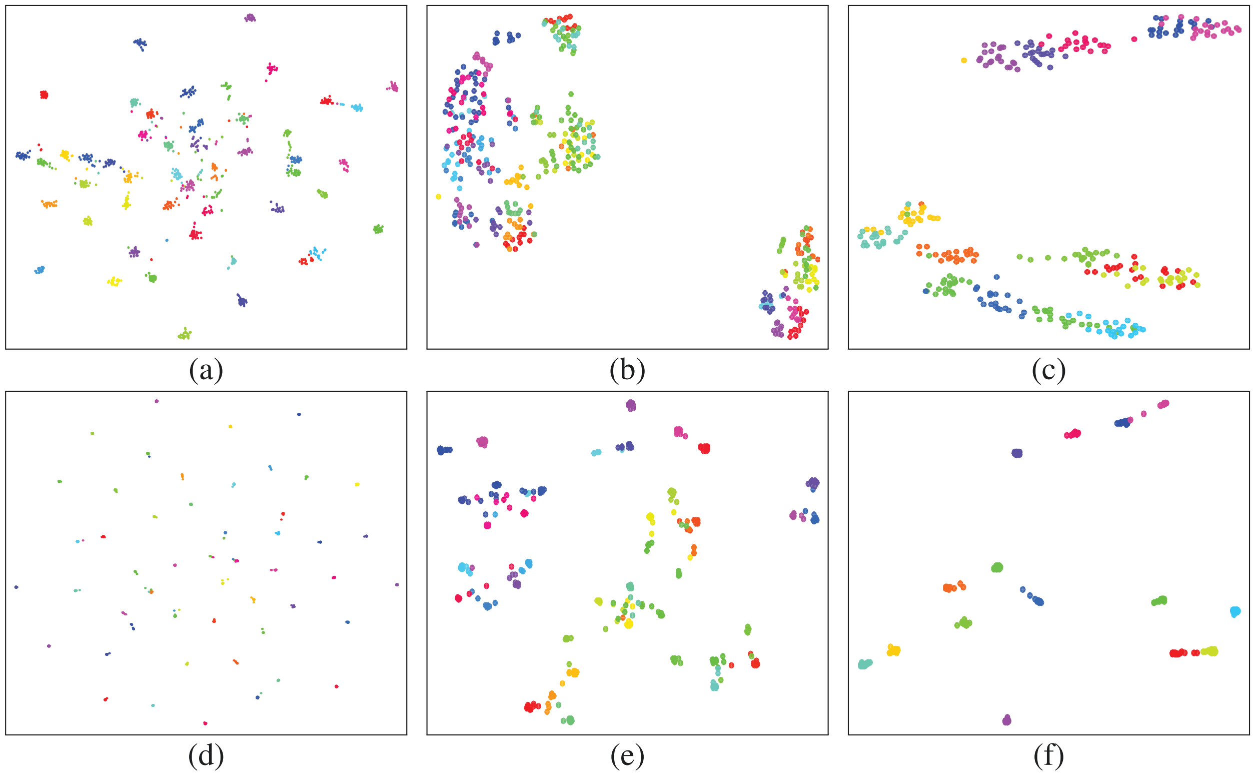


Figure 3.9: Two-dimensional t-SNE feature visualization of SA-BoF+ISPM (top) and DLSD (bottom) for the SHREC-2015 (left), synthetic SHREC-2014 (middle) and real SHREC-2014 datasets (right).

Two queries (male and female) from the real SHREC-2014 dataset are featured in the top-most row of these figures, followed by the top five retrieved shapes. The first query is the male model from class 17 (M17) as shown in Figure 3.10, while the second query is a female model from class 17 (F17) as depicted in Figure 3.11. We compared out results to several baseline methods, including SI-HKS, WKS, R-BiHDM and Shape-DNA. As can be seen in Figure 3.10, DLSD was able to correctly retrieve all the relevant shapes from the query's class (i.e. same shape in different poses), while the other methods failed more than once in retrieving the relevant shapes.

Similarly, we can see in Figure 3.11 that our approach outperforms all baseline methods. This better performance is largely attributed to the fact that deep learning models are able to extract/build better features than shallow models.

3.3.2 Runtime Analysis

In this subsection, we report the runtime performance comparison between our proposed approach and the baseline methods both in the training and testing phases. The runtime for computing low- and mid-level features depends in large part on the number of mesh vertices. The low-level local

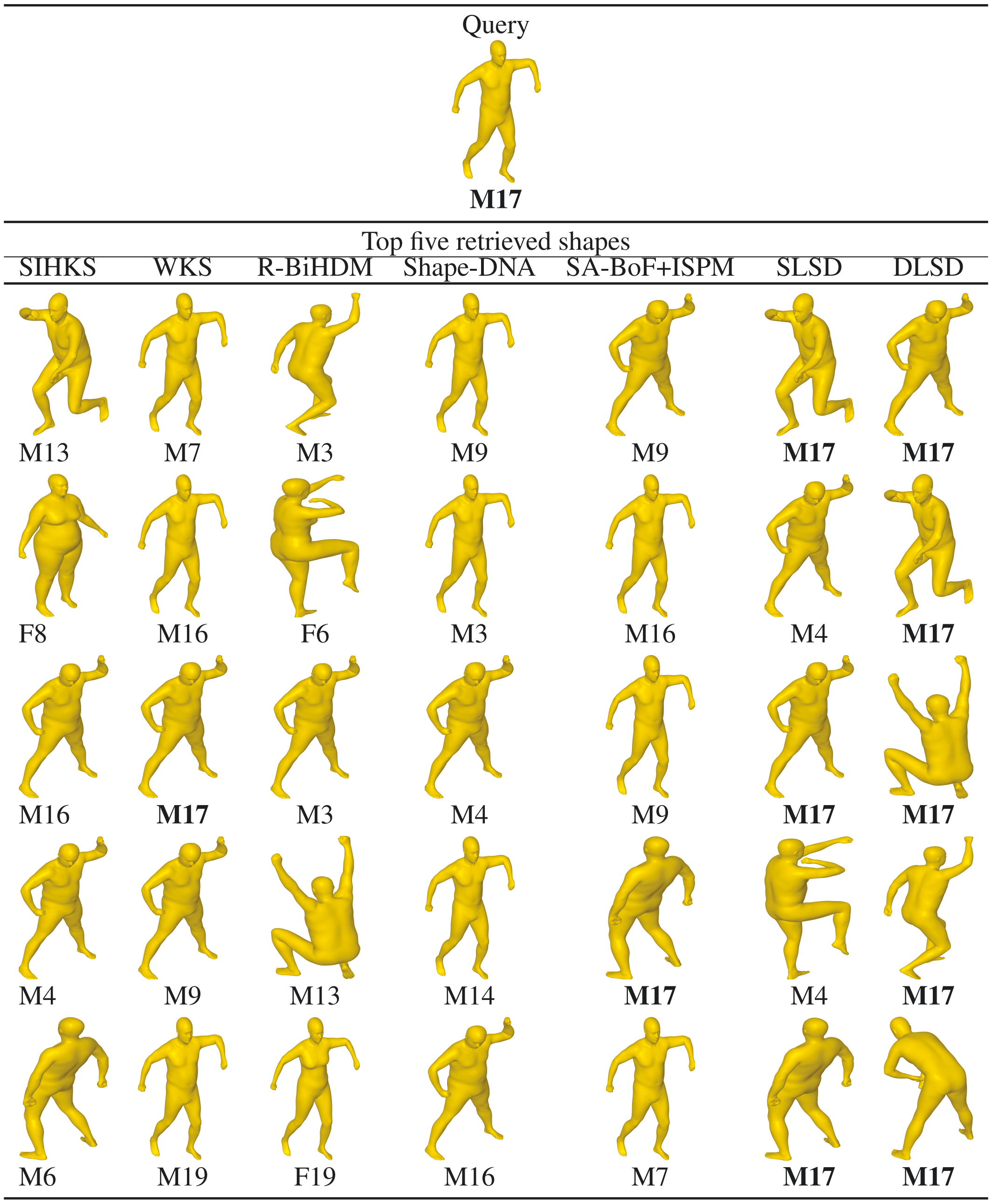

Query
M17
Top five retrieved shapes
SIHKS | WKS | R-BiHDM | Shape-DNA | SA-BoF+ISPM | SLSD | DLSD
M13 | M7 | M3 | M9 | M9 | **M17** | **M17**
F8 | M16 | F6 | M3 | M16 | M4 | **M17**
M16 | **M17** | M3 | M4 | M9 | **M17** | **M17**
M4 | M9 | M13 | M14 | **M17** | M4 | **M17**
M6 | M19 | F19 | M16 | M7 | **M17** | **M17**

Figure 3.10: Top five retrieved shapes (ranked top-to-bottom) using SIHKS, WKS, R-BiHDM, Shape-DNA, SA-BoF2, SLSD, and DLSD. The query shape is from the male class 17 (M17) from the real SHREC-2014 dataset. Boldface numbers indicate the correctly retrieved shapes. M# (resp. F#) denotes the male (resp. female) model in class #.

descriptors obtained via spectral graph wavelets take 0.133, 0.19 and 0.2 seconds on average to be computed for each shape in the SHREC-2015, real SHREC-2014 and synthetic SHREC-2014 datasets, respectively. In the same vein, the mid-level features take 4.32, 5.26 and 5.38 seconds on average to be computed for each shape in the SHREC-2015, real SHREC-2014 and synthetic

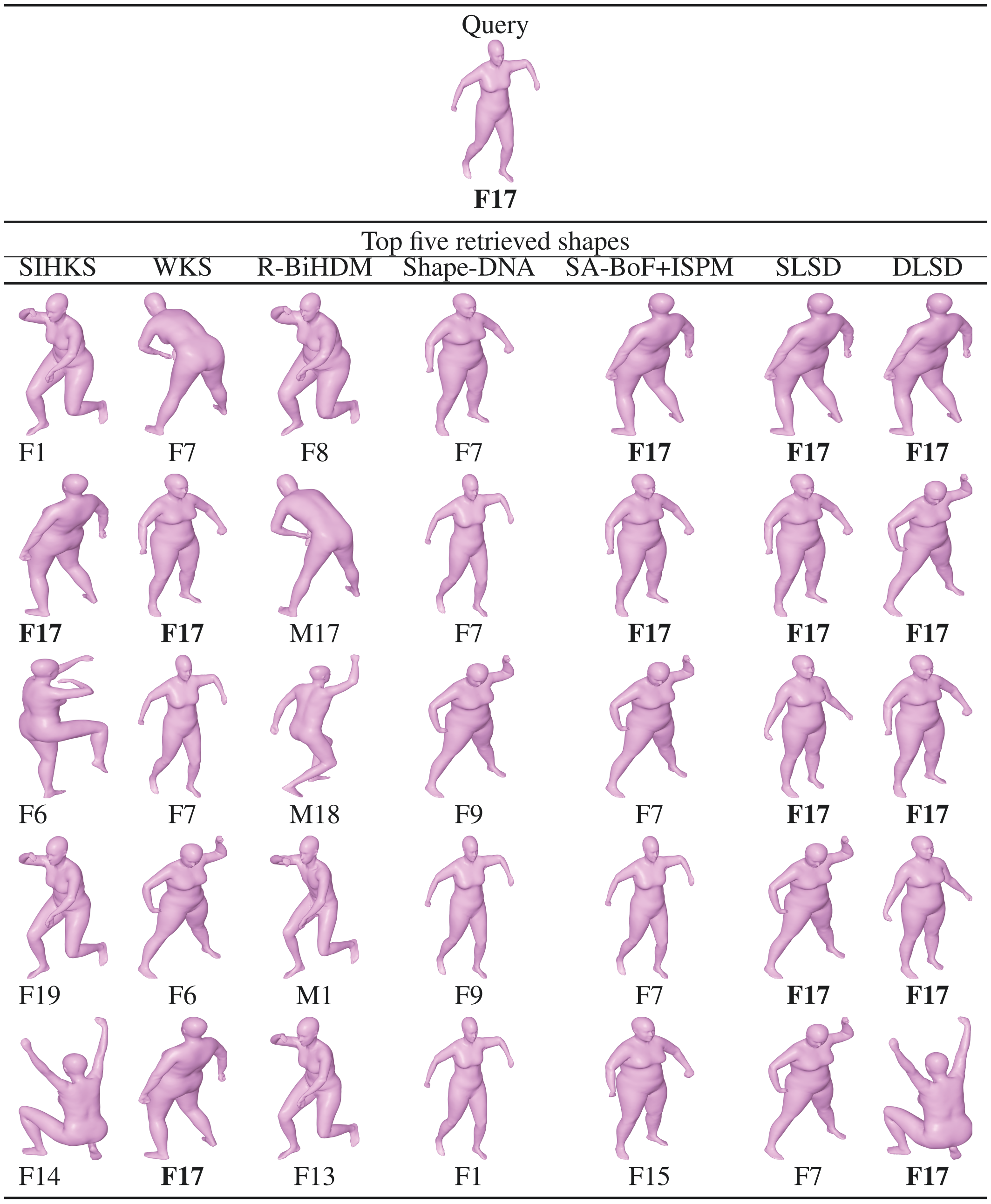


Figure 3.11: Top five retrieved shapes (ranked top-to-bottom) using SIHKS, WKS, R-BiHDM, Shape-DNA, SA-BoF2, SLSD, and DLSD. The query shape is from the female class 17 (F17) from the real SHREC-2014 dataset. Boldface numbers indicate the correctly retrieved shapes. M# (resp. F#) denotes the male (resp. female) model in class #.

SHREC-2014 datasets, respectively. Table 3.13 shows the runtime comparison between the proposed algorithm and the baseline methods SS-BoF, GA-BoF and SA-BoF. As can been seen in the table, the runtime results indicate that the use of the geodesic exponential kernel as a spatial relationship matrix yields the highest computational burden. It can also be seen that SA-BoF+ISPM outperforms SA-BoF and the other kernels, while it slightly increases the runtime. Since the same number of local descriptors is used to construct the codebook related to each dataset, the runtimes for the codebooks are quite similar: 51, 55 and 51 seconds for the SHREC-2015, real SHREC-2014 and synthetic SHREC-2014 datasets, respectively.

On the other hand, the training process takes between 9 and 10 hours for a deep model, and between 2 and 3 hours for a shallow one. The test process, including distance measure, retrieval and all scores' computations, takes 42.80, 7.56 and 4.88 seconds for the SHREC-2015, real SHREC-2014 and synthetic SHREC-2014 datasets, respectively. As can be seen in Table 3.13, the runtime of performing the retrieval test on SHREC 2015 is higher than SHREC 2014 because of the large size of the former dataset, resulting in longer computation time for the distance matrix as well as the retrieval process.

Table 3.13: Runtime performance comparison.

	Runtime		
	SHREC 2015	Real SHREC 2014	Synthetic SHREC 2014
SGWS	0.133 s	0.19 s	0.2 s
Codebook design	51 s	55 s	51 s
SA-BoF+ISPM	4.32 s	5.26 s	5.38 s
SA-BoF	4.28 s	5.22 s	5.22 s
GA-BoF	4.51 s	6.16 s	6.64 s
SS-BoF	4.33 s	4.89 s	4.86 s
Testing	42.80 s	7.56 s	4.88 s
Deep training	9-10 h		
Shallow training	2-3 h		

3.4 Conclusion

In this chapter, we proposed a multi-level feature learning paradigm for 3D shape retrieval using deep learning in conjunction with intrinsic spatial pyramid matching. First, low-level local descriptors were obtained using spectral graph wavelets. Then, mid-level features were extracted via the bag-of-features model by aggregating local descriptors into global ones. We used locality-constrained linear coding as a feature coding method and measured the spatial relationships between codewords using the biharmonic distance combined by intrinsic spatial pyramid matching in a bid to generate shape-aware bag-of-features as mid-level features. Finally, high-level features were learned using a deep auto-encoder. The proposed approach achieves significantly better performance than state-of-the-art methods.

CHAPTER 4

Convolutional Shape-Aware Representation for 3D Object Classification

Deep learning has recently emerged as one of the most popular and powerful paradigms for learning tasks. In this chapter, we present a deep learning approach to 3D shape classification using convolutional neural networks. The proposed framework takes a multi-stage approach that first represents each 3D shape in the dataset as a 2D image using the bag-of-features model in conjunction with intrinsic spatial pyramid matching that leverages the spatial relationship between features. These 2D images are then fed into a pre-trained convolutional neural network to learn deep convolutional shape-aware descriptors from the penultimate fully-connected layer of the network. Finally, a multiclass support vector machine classifier is trained on the deep descriptors, and the classification accuracy is subsequently computed. The effectiveness of our approach is demonstrated on three standard 3D shape benchmarks, yielding higher classification accuracy rates compared to existing methods.

4.1 Introduction

The 3D shape classification problem is of paramount importance in many computer vision, geometry processing, and computer graphics applications [19, 34, 50, 76, 78, 79]. Shape classification is all about labeling shapes in a dataset and organizing them into a known number of classes so they can be found quickly and efficiently, and the goal is to assign new shapes to one of these classes. The growing interest in 3D shape classification is partly driven by the availability of large-scale 3D shape benchmarks, and also by the emergence of powerful deep learning algorithms [78].

Deep learning has emerged in recent years as a very powerful way to hierarchically find abstract patterns using large amounts of training data [76]. It has proved especially valuable for speech recognition, computer vision, natural language processing, and geometry processing [18–20, 80]. The tremendous success of deep neural networks in image classification, for instance, is largely attributed to open source software, inexpensive computing hardware, and the availability of large-scale datasets. In particular, convolutional neural networks (CNNs) have achieved impressive classification accuracy on the challenging ImageNet dataset [30, 81, 82]. However, it is not straightforward to apply CNNs directly to 3D shapes, particularly mesh data that are usually modeled as graphs. To mitigate this practical difficulty, several deep learning architectures have been recently proposed to learn higher level representations of shapes by first representing a 3D shape as a 2D image and then applying a deep learning model for feature learning. Kanezaki *et al.* [53] introduced RotationNet, a CNN-based framework that uses a set of multi-view images of a 3D object as input for 3D object classification and pose estimation. View-based methods tend to suffer from a relatively long running time due primarily to analyzing a large amount of redundant data provided by multi-view images. Also, a major drawback of view-based methods is their sensitivity to consistent model orientations, resulting in lower performance. Bu *et al.* [49] proposed a deep belief network approach for 3D shape recognition using a shape descriptor represented by a 2D image defined in terms of the geodesic distance and eigenfunctions of the LBO. The geodesic distance, however, has some major limitations, the most serious of which are the sensitivity to topological noise and the lack of shape-awareness [35]. More recently, Bu *et al.* [55] presented a multi-modal feature learning approach to 3D shape recognition using CNNs and convolutional deep belief networks. This hybrid approach combines both view-based and geometry-based feature learning in an effort to learn a more discriminative shape descriptor by fusing different modalities.

This chapter introduces a deep convolutional shape-aware (Deep-CSA) learning framework for 3D shape classification using a pre-trained convolutional neural network. The proposed approach is a multi-level feature learning paradigm consisting of three major steps. First, we represent each 3D shape in a dataset by a spectral graph wavelet signature, which is a local descriptor that has been shown to effectively capture low and high frequency details of the shape [9]. Locality-constrained linear coding (LLC) [83] is then employed as a feature coding method in the BoF model. In contrast to other feature encoding techniques, LLC has a lower quantization error and enjoys a nice locality properly of codewords. To take into consideration the occurrence distribution and spatial relationship between the codewords, we use the biharmonic distance together with intrinsic spatial pyramid matching to leverage the structural information in the LLC codes, resulting in mid-level features that are shape-aware. The biharmonic distance strikes a balance between nearly geodesic distances for small distances and global shape-awareness for large distances. Next, the mid-level features are reshaped into a color image, which in turn is fed into a pre-trained CNN model in order to learn high-level feature descriptors from the penultimate fully-connected layer of the network. This CNN model was trained on the challenging, large-scale ImageNet benchmark, and yields high-quality features that facilitate transferability to other learning tasks. Finally, a multiclass support vector machine classifier is trained on the learned descriptors, and the performance of our approach is subsequently assessed using several evaluation metrics, including the confusion matrix, average classification accuracy and standard deviation.

The remainder of this chapter is organized as follows. In Section 4.2, we introduce a deep learning approach to 3D shape classification using the bag-of-features paradigm in conjunction with intrinsic spatial pyramid matching that leverages the spatial relationship between features. We also discuss in detail the major components of our multi-stage approach, and summarize its main algorithmic steps. In Section 4.3, we present experimental results to demonstrate the competitive performance of our approach on several 3D shape benchmarks. Finally, Section 4.4 concludes the chapter.

4.2 Method

In this section, we describe the main components and algorithmic steps of the proposed multi-stage feature learning framework for 3D shape classification. The flowchart of the learning process is illustrated in Figure 4.1.

A 3D shape is usually modeled as a triangle mesh $\mathbb{M}$ whose vertices are sampled from a Riemannian manifold. A triangle mesh $\mathbb{M}$ may be defined as a graph $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ or $\mathbb{G} = (\mathcal{V}, \mathcal{T})$, where $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ is the set of vertices, $\mathcal{E} = \{e_{ij}\}$ is the set of edges, and $\mathcal{T}$ is the set of triangles. Each edge $e_{ij} = [\mathbf{v}_i, \mathbf{v}_j]$ connects a pair of vertices $\{\mathbf{v}_i, \mathbf{v}_j\}$ (or simply $\{i, j\}$).

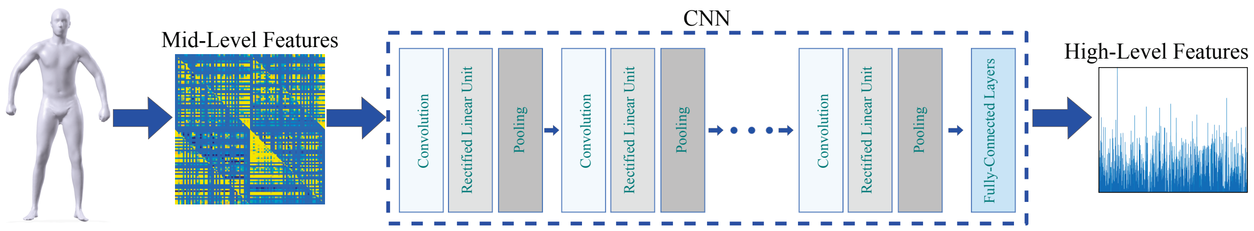


Figure 4.1: Main components of the proposed feature learning framework.

4.2.1 Convolutional Shape-Aware Features

In the third step of our framework, more discriminative 3D shape descriptors are extracted using high-level features learned with a pre-trained CNN model on mid-level features. A CNN is a deep architecture that makes use of feedforward artificial neural networks in which individual neurons are tiled in such a way that they respond to overlapping regions in the visual field [21].

A CNN is designed specifically for 2D image recognition, and is typically comprised of multiple layers that can be broadly categorized into three main types: convolutional, pooling, and fully-connected, as shown in Figure 4.1. A convolutional layer consists of a rectangular grid of neurons, and applies a set of filters that process small local parts of the input where these filters are replicated along the whole input space. A convolutional layer is usually followed by a non-linear layer or activation function such as rectified linear unit (ReLU). A pooling layer takes small rectangular blocks from the convolutional layer and subsamples it with average or max pooling to produce a single output from that block. Similar to a feedforward neural network, a fully-connected layer takes all neurons in the previous layer and connects them to each of its neurons. It should be noted that all weights in all layers of a CNN are learned through training via backpropagation.

Unlike images, it is not straightforward to apply a deep learning architecture directly to a 3D mesh. To circumvent this practical limitation, we reshape the κ-dimensional SA-BoF-ISPM vector into an image of a specific size so that it can used as an input to a pre-trained CNN model. The high-level features are then extracted from the penultimate fully-connected layer (i.e. the layer preceding the softmax classification layer) with n_L neurons, resulting in an n_L-dimensional high-level feature vector, which we refer to as a deep convolutional shape-aware (Deep-CSA) descriptor. In other words, the Deep-CSA descriptor is the high-level representation of a 3D shape in our deep learning framework.

4.2.2 Algorithm

Shape classification is a supervised learning method that assigns shapes in a dataset to target classes, and the objective is to classify new shapes into one of the given classes via a machine learning model whose parameters were optimized using a training set of data.

Our proposed algorithm consists of three main steps. In the first step, we represent each 3D shape in a dataset by a 2D image of mid-level features that are shape-aware. More specifically, let $\mathcal{D}$ be a dataset of n shapes modeled by triangle meshes $\mathbb{M}_1, \dots, \mathbb{M}_n$. Each mesh $\mathbb{M}_i$ in the dataset is first represented by a $p \times m$ spectral graph wavelet signature matrix $\mathbf{S}_i$, where m is the number of mesh vertices. The spectral graph wavelet signatures are then encoded via LLC, resulting in a $k \times m$ matrix $\mathbf{U}_i$ whose columns are the k-dimensional LLC codes. To capture the spatial relations between LLC codes, we cut each shape into r intrinsic spatial partitions using the level sets of the second eigenfunction of LBO, yielding a set of q-dimensional SA-BoF vectors $\mathbf{f}_{ij}$, where $j = 1, ..., r$ and $q = k(k+1)/2$. These r vectors are then concatenated to form the SA-BoF-ISPM vector $\mathbf{x}_i$ of length $\kappa = qr$.

For simplicity, we set the resolution parameter to $r = 2$, i.e. the SA-BoF-ISPM vector $\mathbf{x}_i$ becomes $k(k+1)$-dimensional. Hence, we reshape the SA-BoF-ISPM vector into a $k \times (k+1)$ matrix, replicate it three times, and resize it into an RGB image of size $227 \times 227 \times 3$ so that it can be used as an input to the pre-trained VGG-F model [84]. VGG-F is a CNN model comprising of an input layer (the input must be an image of size $227 \times 227 \times 3$) and eight learnable layers, including five convolutional layers, each followed by a rectified linear unit (ReLU) and a max pooling layer, and then three fully-connected layers (fc1, fc2 and fc3), where fc3 is the softmax classification layer. VGG-F was trained on the challenging ImageNet dataset, which has 1.2 million training images categorized into 1000 classes. Moreover, the VGG-F model has successfully demonstrated to yield high-quality generic features that not only produce state-of-the-art results on image classification, but are also transferable to other learning tasks, and even to other modalities. So after discarding the softmax classification layer, the pre-trained VGG-F model can be used as a feature extractor for new datasets.

In the second step, we apply VGG-F on these n color images and fine-tune the parameters of the network in an effort to extract high-level features from the penultimate fully-connected layer (i.e. fc2 layer consisting of n_L neurons) of the network, resulting in an $n_L \times n$ matrix $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_n)$ whose columns are the Deep-CSA feature vectors (i.e. high-level feature vectors), each of which is n_L-dimensional. In the VGG-F model, the dimensions of the Deep-CSA descriptor is $n_L = 4096$.

Finally, a one-vs-all multiclass linear SVM classifier is performed on Deep-CSA descriptors to find the best hyperplane that separates all data points of one class from those of the other classes. Given a training data of the form $\mathcal{Z}_{\text{train}} = \{(\mathbf{z}_i, y_i)\}$, where $\mathbf{z}_i \in \mathbb{R}^{n_L}$ is the ith example (i.e. Deep-CSA descriptor) and $y_i \in \{1, \ldots, C\}$ is its ith class label, we aim at finding a learning model that contains the optimized parameters from the SVM algorithm. Then, the trained SVM model is applied to a test data $\mathcal{Z}_{\text{test}}$, resulting in predicted labels $\hat{y}_i$ of new data. These predicted labels are subsequently compared to the labels of the test data to evaluate the classification accuracy of the learning model.

The main algorithmic steps of our approach are summarized in Algorithm 3.

4.3 Experiments

In this section, extensive experiments are conducted to evaluate the performance of the proposed 3D shape classification approach on three standard benchmarks.

Datasets: We tested and analyzed the proposed algorithm on three standard and publicly available 3D shape benchmarks: SHREC-2015, real SHREC-2014 and synthetic SHREC-2014. The SHREC-2015 benchmark is a dataset of 3D shapes consisting of 1200 watertight mesh models from 50 classes [2], where each class contains 24 objects with distinct postures.

Algorithm 3 Deep-CSA Classifier

Input: Dataset $\mathcal{D} = \{\mathbb{M}_1, \dots, \mathbb{M}_n\}$ of 3D shapes and number k of codebook bases.
Output: Vector $\hat{\mathbf{y}}$ containing predicted class labels.

```
1: for $i = 1$ to $n$ do
2:     Cut each shape into two intrinsic spatial partitions
3:     for $j = 1$ to 2 do
4:         Compute the $q$-dimensional SA-BoF vector $\mathbf{f}_{ij}$ for each shape partition $j$, where $q = k(k+1)/2$.
5:     end for
6:     Concatenate the SA-BoF vectors $\mathbf{f}_{ij}$, and reshape them into a $\kappa$-dimensional SA-BoF-ISPM vector $\mathbf{x}_i$, where $\kappa = 2q = k(k+1)$.
7:     Reshape the SA-BoF-ISPM vector $\mathbf{x}_i$ into a $k \times (k+1)$ image, replicate it three times and then resize it as a color image of size $227 \times 227 \times 3$
8:     Apply the pre-trained VGG-F model on the color image to find the $n_L$-dimensional Deep-CSA vector $\mathbf{z}_i$, where $n_L$ is the number of neurons in the penultimate fully-connected layer of the network.
9: end for
10: Arrange all the $n$ Deep-CSA vectors into a $n_L \times n$ data matrix $\mathbf{Z} = (\mathbf{z}_1, \dots, \mathbf{z}_n)$
11: Split $\mathbf{Z}$ into training and test sets.
12: Train a multi-class SVM classifier on the training set, and then find the predicted class labels for the test set.
```

The real SHREC-2014 dataset is made up of ‘real’ data, obtained by scanning real human participants [5], and it consists of 400 shapes, made up of 40 human subjects in 10 different poses. Half of the human subjects are male, and half are female. The poses of each subject are built using a data-driven deformation technique, which can produce realistic deformations of articulated meshes. The synthetic SHREC-2014 dataset was built using DAZ Studio and consists of 300 human models (adults and children) subdivided into 15 classes of 20 members each. Objects are considered to belong to the same class if they share the same body shape.

Implementation details: All the experiments were conducted on a desktop computer with a CPU Core i5 processor running at 3.4 GHz and 16 GB RAM, and the algorithms were implemented in MATLAB. We used the first 201 eigenvalues and eigenvectors of the LBO to compute the low-level descriptors (i.e. spectral graph wavelet signatures) and the biharmonic distance. The number of vertices varies from shape to shape, but it is set to approximately 1000 and 2200 for the shapes in SHREC-2015 and SHREC-2014, respectively. We set the number of codebook bases to $k = 48$, resulting in a SA-BoF-ISPM vector of length $\kappa = k(k+1) = 2352$. We then reshaped the SA-BoF-ISPM vector into a 48×49 matrix, duplicated it three times, and resized it into an RGB image of size $227 \times 227 \times 3$. Next, we applied the pre-trained VGG-F model to these color images, and we extracted high-level features from the penultimate fully-connected layer with $n_L = 4096$ neurons, yielding a $4096 \times n$ matrix $\mathbf{Z}$ whose columns are the Deep-CSA descriptors, where n is the number of shapes in the dataset.

Baseline methods: We compare the effectiveness of the proposed framework with several state-of-the-art methods, including Shape-DNA [42], reduced biharmonic distance matrix (R-BiHDM) [12], graph biharmonic distance map (GraphBDM) [44], scale-invariant heat kernel signature (SIHKS) [7], and wave kernel signature (WKS) [8]. We also compare our Deep-CSA approach to the shallow SA-BoF model with and without ISPM.

4.3.1 Results

The classification performance of our method is assessed by conducting a comprehensive comparison with several baseline methods using various evaluation metrics.

SHREC-2015 results: For this dataset of 1200 shapes, the resulting Deep-CSA data matrix $\mathbf{Z}$ is of size 4096×1200. We randomly selected 30% of shapes in this dataset to hold out for the test set, and the remaining shapes for training. That is, the test data consists of 360 shapes. A one-vs-all multiclass SVM is first trained on the Deep-CSA descriptors of the training data to learn the model (i.e. classifier), which is subsequently used on the test data with known target values in order to predict the class labels. Figure 4.2 displays the confusion matrix for SHREC-2015 on the test data. This 50×50 confusion matrix shows how the predictions are made by the model. Its rows correspond to the actual (true) class of the data (i.e. the labels in the data), while its columns correspond to the predicted class (i.e. predictions made by the model). The value of each element in the confusion matrix is the number of predictions made with the class corresponding to the column for instances with the correct value as represented by the row. Thus, the diagonal elements show the number of correct classifications made for each class, and the off-diagonal elements show the errors made. As shown in Figure 4.2, the Deep-CSA approach was able to accurately classify all shapes in the test data, except for five shapes that were misclassified by a false positive rate of 14%. For example, shape number 5 was misclassified as shape number 22, and this is largely due to the fact that these shapes are highly similar to each other. Such a good performance strongly suggests that Deep-CSA captures well the discriminative features of the shapes.

We compared the proposed Deep-CSA method to Shape-DNA, R-BiHDM, GraphBDM, SIHKS, WKS, and also to the shallow SA-BoF and SA-BoF-ISPM models. It is important to point out that for fair comparison with SI-HKS and WKS, which also are local descriptors, we generated their corresponding global descriptors using our mid-level feature extraction strategy based on LLC as feature coding method and the biharmonic distance as the kernel for feature pooling. Following the common practice in classification tasks, we repeated the experimental process 10 times with different randomly selected training and test data in an effort to obtain reliable results, and the ac-

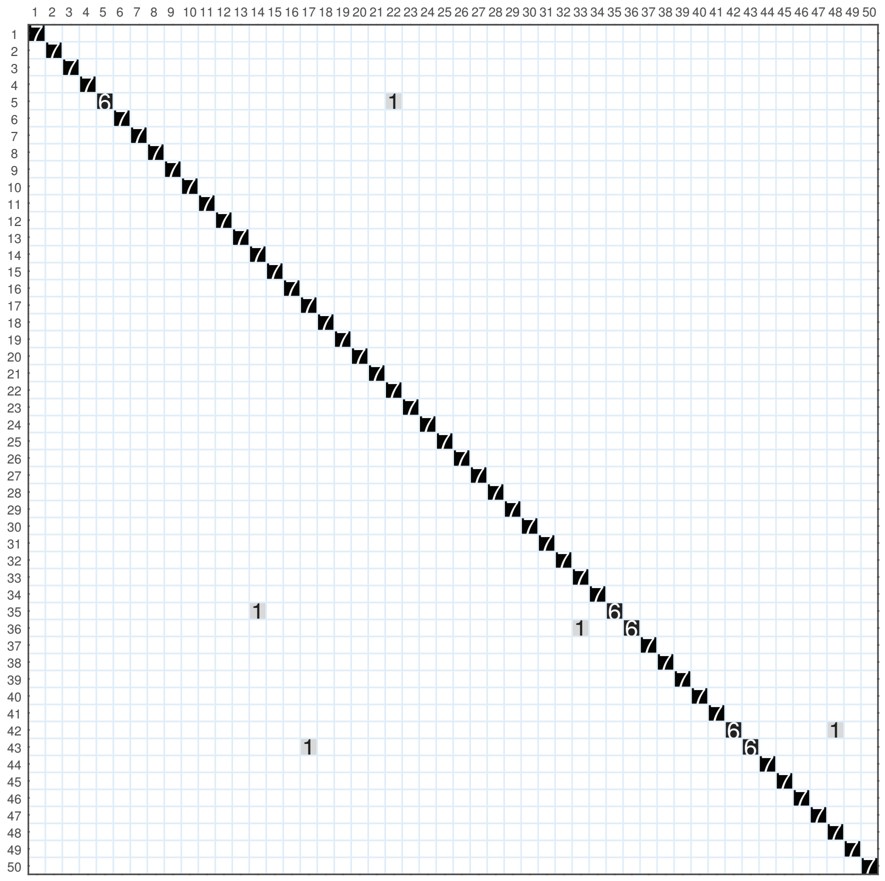


Figure 4.2: Confusion matrix for Deep-CSA on SHREC-2015.

Table 4.1: Classification accuracy results on SHREC-2015, real SHREC-2014 and synthetic SHREC-2014. Boldface numbers indicate the best classification performance.

	Average Accuracy (%)		
Method	SHREC-2015	Real SHREC-2014	Synthetic SHREC-2014
Shape-DNA [42]	70.42 ± 2.28	7.42 ± 1.59	56.78 ± 7.51
R-BiHDM [12]	74.40 ± 2.07	9.50 ± 2.73	56.11 ± 7.36
GraphBDM [44]	82.26 ± 2.29	18.75 ± 5.61	60.11 ± 3.90
SIHKS [7]	41.97 ± 1.73	4.17 ± 3.49	3.89 ± 3.28
WKS [8]	88.77 ± 0.36	16.75 ± 1.78	78.00 ± 3.58
SA-BoF	85.43 ± 1.79	38.00 ± 2.61	88.67 ± 3.04
SA-BoF-ISPM	91.91 ± 1.81	50.58 ± 3.62	88.89 ± 1.66
Deep-CSA	**97.74 ± 0.59**	**83.50 ± 1.51**	**97.00 ± 1.39**

curacy for each run was recorded. The classification accuracy results are summarized in Table 4.1, which shows the average (and standard deviation) results of the baseline methods and the proposed framework. As can be seen, Deep-CSA outperforms all other methods including the shallow SA-BoF-ISPM model. The average classification accuracy of Deep-CSA on the SHREC-2015 dataset is 97.74%. The performance improvement is 5.83% over to the shallow SA-BoF-ISPM model, which strongly suggests that deep models can substantially improve the classification results.

The superior performance of the Deep-CSA over the baselines may be attributed in large part to the discriminative power of the high-level features learned by a deep model, coupled with the robust shape representation provided by the shape-aware mid-level features. It is worth noting that SA-BoF-ISPM outperforms all baseline methods, and also surpasses SA-BoF by 6.48%, indicating the performance improvement by using ISPM in the mid-level feature extraction. This is also confirmed by the error bars for Deep-CSA and baseline methods, as shown in Figure 4.3. As can be seen, Deep-CSA achieves the highest accuracy rate, indicating its superiority over the baseline methods. Moreover, it can be seen in Figure 4.3 and Table 4.1 that the standard deviations are much smaller than the accuracy improvements, indicating that Deep-CSA is robust to random selection of training and test data. Overall, Deep-CSA is consistently the best, delivering robust classification results.

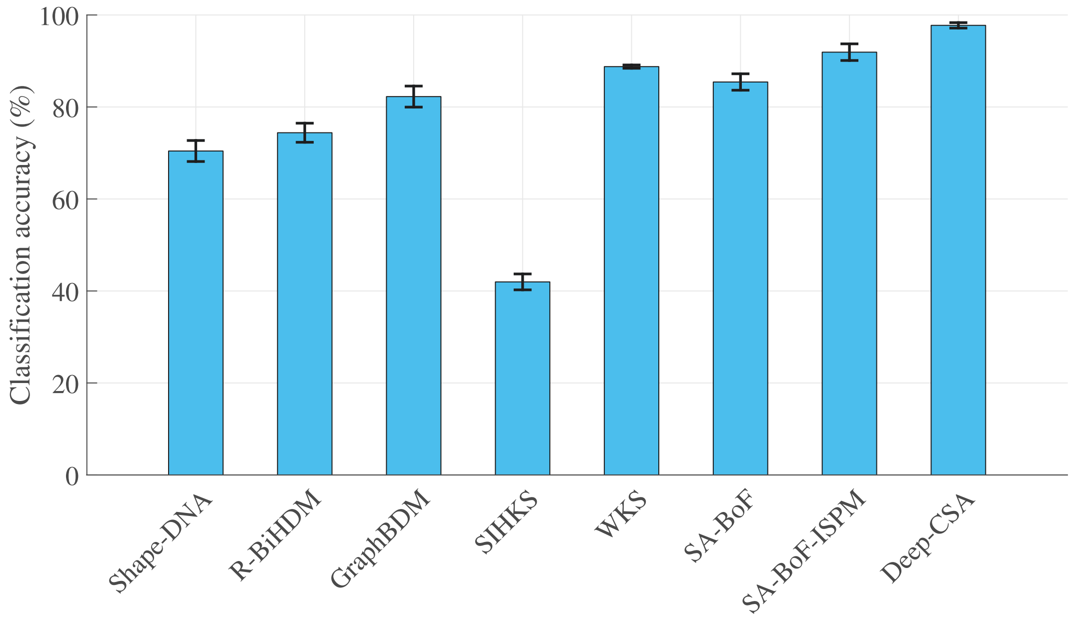


Figure 4.3: Classification accuracy rates with error bars for Deep-CSA and baseline methods on SHREC-2015.

Real SHREC-2014 results: For the real SHREC-2014 dataset of 400 shapes, we obtain a Deep-CSA data matrix $\mathbf{Z}$ of size 4096×400. We randomly selected 30% of shapes in this dataset to hold out for the test set, and the remaining shapes for training. That is, the test data consists of 120 shapes. First, we train a one-vs-all multiclass SVM on the training data to learn the classification model. Then, we use the resulting, trained model on the test data to predict the class labels. Figure 4.4 displays the 40×40 confusion matrix for real SHREC-2014 on the test data. Although the real SHREC-2014 dataset is a challenging benchmark, Deep-CSA was still able to accurately classify all sha

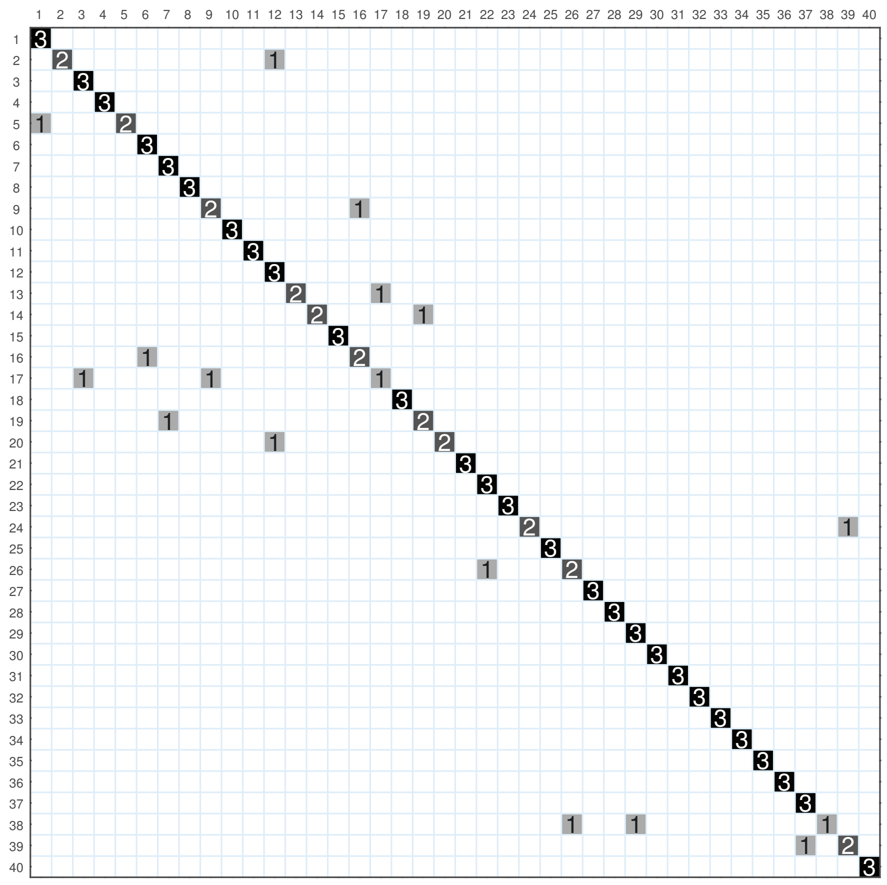


Figure 4.4: Confusion matrix for Deep-CSA on real SHREC-2014.

Synthetic SHREC-2014 results: For the synthetic SHREC-2014 dataset of 300 shapes, the resulting Deep-CSA data matrix is of size 4096×300. We randomly selected 30% of shapes in this dataset to hold out for the test set, and the remaining shapes for training. That is, the test data consists of 90 shapes. Figure 4.6 displays the 15×15 confusion matrix for synthetic SHREC-2014 on the test data. As can be seen, Deep-CSA was able to accurately classify all shapes in the test data, except for two shapes.

We repeated the experimental process 10 times with different randomly selected training and test data in an effort to obtain reliable results, and the accuracy for each run was recorded. The average accuracy results for the real and synthetic SHREC-2014 datasets are reported in Table 4.1. As can be seen, Deep-CSA significantly outperforms the baseline methods with large performance

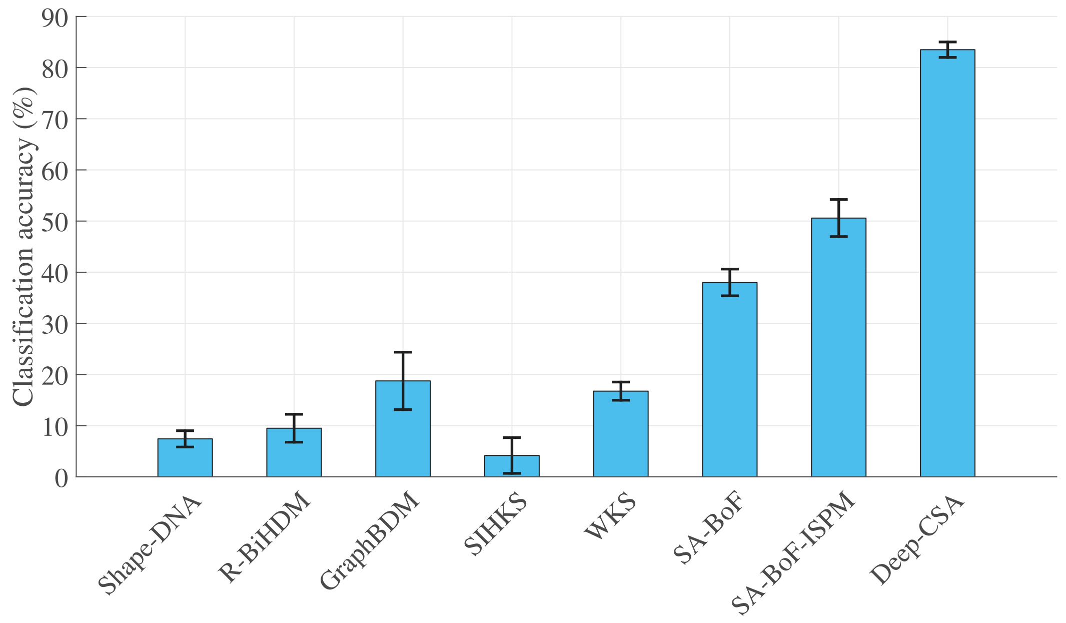


Figure 4.5: Classification accuracy rates with error bars for Deep-CSA and baseline methods on real SHREC-2014.

margins on the challenging, real SHREC-2014 benchmark. In part, this is due to the fact that the pre-trained VGG-F model was trained on the large-scale ImageNet dataset, yielding robust features that can be transferred to other learning tasks. The average classification accuracy of Deep-CSA on the real SHREC-2014 dataset is 83.50%. The performance improvement is 32.92% compared to the shallow SA-BoF-ISPM model, which strongly indicate the power of deep learning when dealing with challenging datasets such as the real SHREC-2014 benchmark. Furthermore, SA-BoF-ISPM outperforms all the baseline methods, and surpasses the shallow SA-BoF model by a comfortable margin of 12.58%, showing that the performance gap between SA-BoF and SA-BoF-ISPM may get even larger when dealing with challenging datasets. This is also confirmed by the error bar plots for Deep-CSA and the baseline methods, as shown in Figure 4.5. As can be seen, Deep-CSA gives the highest accuracy rate, which further proves its superiority over existing methods. Deep-CSA also outperforms the baseline methods on the synthetic SHREC-2014 dataset, yielding an average classification accuracy rate of 97%. The performance improvement is 8.11% compared to the shallow SA-BoF-ISPM model, which again demonstrates that deep learning models can substantially improve the classification results. However, SA-BoF-ISPM did not significantly outperform SA-BoF for this dataset, which indicates that the biharmonic distance can measure the spatial relations well enough for the 3D shapes in this dataset. Moreover, SA-BoF yields better classification results than WKS and SI-HKS on SHREC 2014, further strengthening our claim that SGWS has a more discriminative ability than WKS and SI-HKS thanks primarily

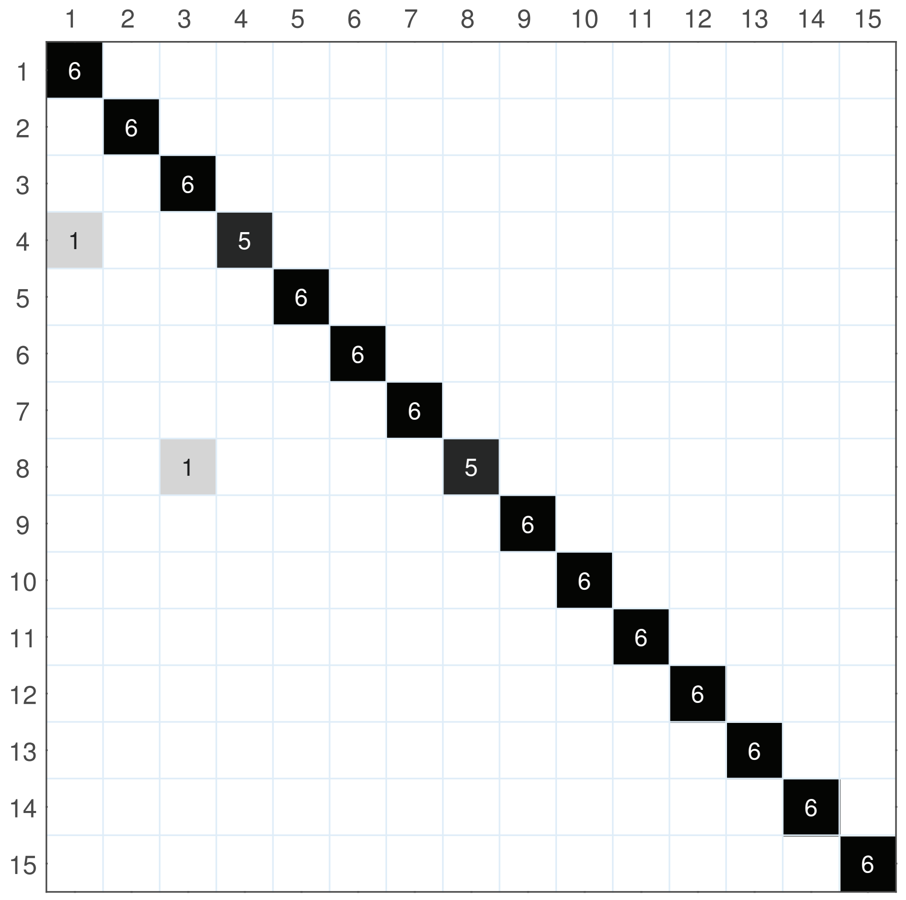


Figure 4.6: Confusion matrix for Deep-CSA on synthetic SHREC-2014.

to the fact that SGWS captures geometric information at multiple scales. This better performance is also evidenced by the error bar plots in Figure 4.7, where Deep-CSA achieves the best result, indicating its superiority over the baseline methods.

Feature visualization: The high-level features learned by our deep learning approach can be visualized using the t-Distributed Stochastic Neighbor Embedding (t-SNE) [77], which is a dimensionality reduction technique that is particularly well-suited for embedding high-dimensional data into a space of two or three dimensions. The two-dimensional plots (i.e. t-SNE embeddings) in Figure 4.8 were generated by running the t-SNE algorithm on Deep-CSA and SA-BoF-ISPM features for the SHREC-2015, real SHREC-2014 and synthetic SHREC-2014 datasets. The plots in Figure 4.8 show that the two-dimensional embeddings corresponding to Deep-CSA are more separable than the ones corresponding to the shallow SA-BoF-ISPM model. Moreover, the Deep-CSA features show very good clustering of classes. This suggests Deep-CSA is a good feature descriptor for 3D object recognition tasks.

Discussion: It is important to point out that we also tested our Deep-CSA approach with a soft-

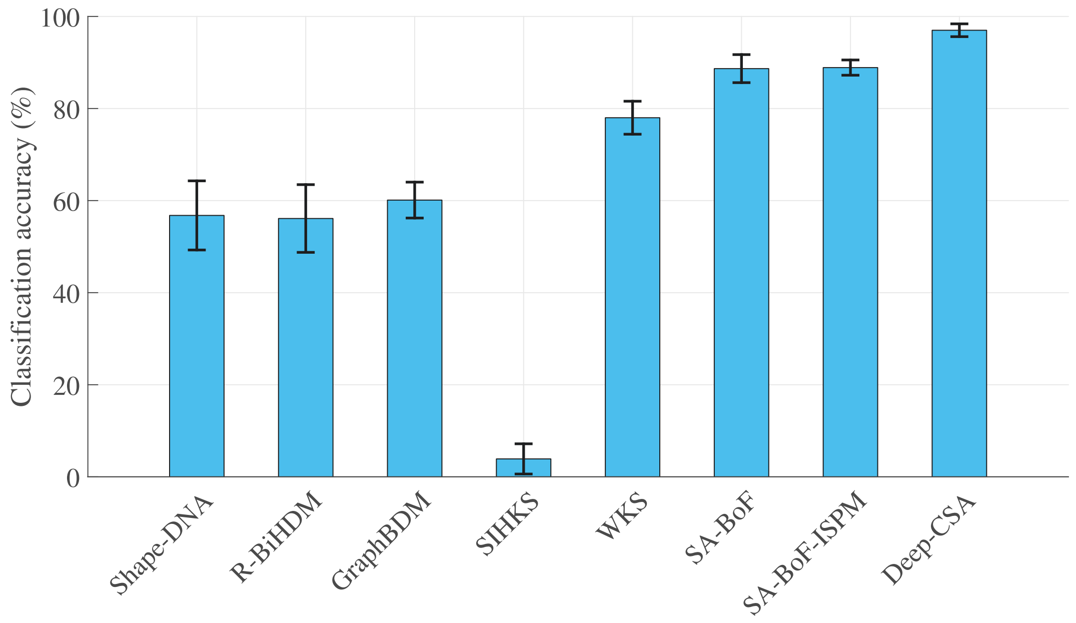


Figure 4.7: Classification accuracy rates with error bars for Deep-CSA and baseline methods on synthetic SHREC-2014.

max classifier instead of SVM, but the accuracy results were slightly lower, averaging 96.48% for SHREC-2015, 82.75% for the real SHREC-2014, and 94.67% for the synthetic SHREC-2014. With the use of the SVM classifier, the pre-trained VGG-F model is employed only as a feature extractor, without any fine-tuning of the network.

As argued, the real SHREC-2014 dataset is challenging, and even human observers may not easily distinguish between some categories. Figures 4.9 and 4.10 depict visual comparisons between Deep-CSA descriptors for sample shapes from real SHREC-2014. Figure 4.9 shows the Deep-CSA descriptors for a male model with different poses, while Figure 4.9 displays the Deep-CSA descriptors for a female model. As can be seen, Deep-CSA descriptors vary slightly within each category for different poses. However, there is a notable difference between the Deep-CSA descriptors associated to the same poses for different human models. This further justifies the use of hierarchical feature learning to learn robust features that can significantly improve the classification accuracy results.

4.4 Conclusion

In this chapter, we proposed a deep convolutional shape-aware framework for 3D shape classification using convolutional neural networks to hierarchically learn robust feature representations. We first represented each 3D shape in the dataset by a spectral graph wavelet signature, which

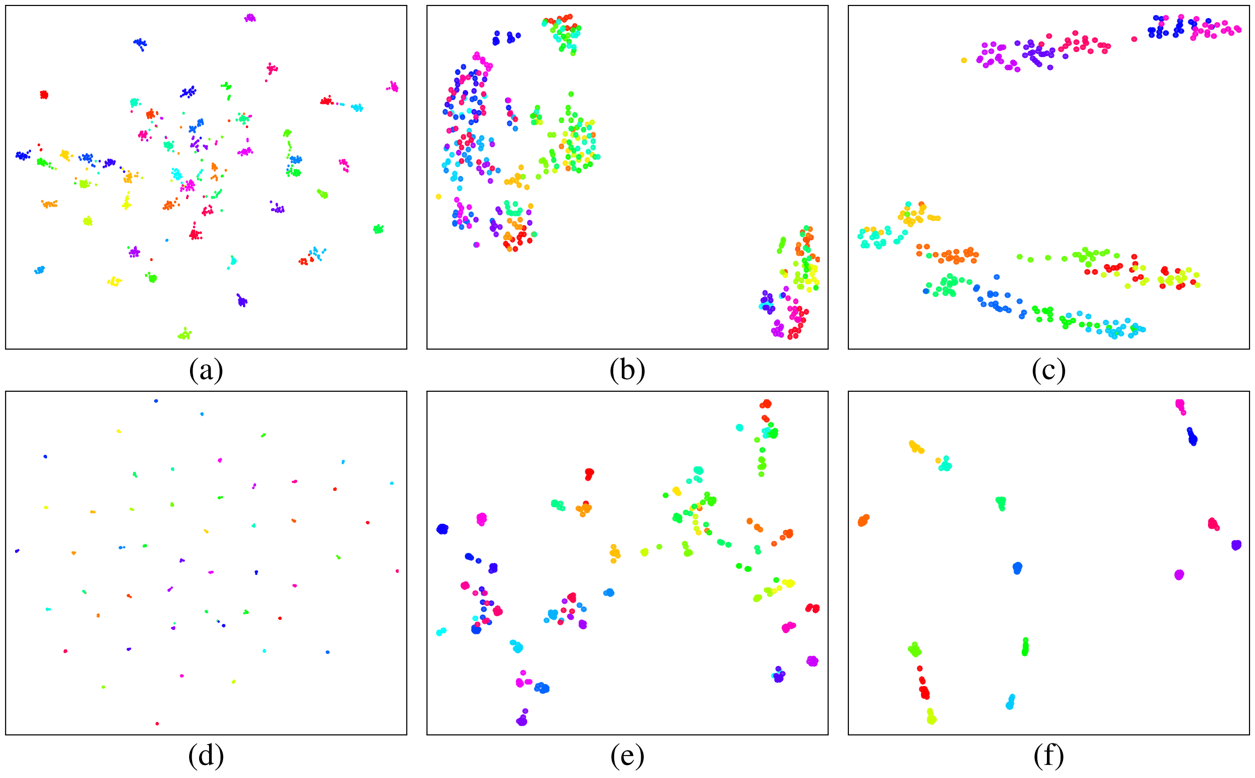


Figure 4.8: Two-dimensional t-SNE feature visualization of SA-BoF+ISPM (left column) and Deep-CSA (right column) for SHREC-2015 (top row), real SHREC-2014 (middle row), and synthetic SHREC-2014 (bottom row).

has been demonstrated to effectively capture low and high frequency details of the shape. We then extracted mid-level features via the bag-of-features paradigm. More specifically, we used locality-constrained linear coding as a feature coding method and measured the spatial relationships between codewords via the biharmonic distance combined with intrinsic spatial pyramid matching in a bid to generate shape-aware mid-level features. We employed a pre-trained deep learning model to extract high-level feature representations from the penultimate fully-connected layer of the network, resulting in Deep-CSA descriptors.

To assess the performance of our framework, we trained a multiclass support vector classifier on the deep learned shape descriptors, and we showed via rigorous experimental evaluations on several datasets that the proposed Deep-CSA approach outperforms existing methods by a relatively large margin, particularly on the challenging real SHREC-2014 benchmark. We also demonstrated through quantitative and qualitative comparisons with shallow models that using deep learning significantly improves the classification accuracy rates.

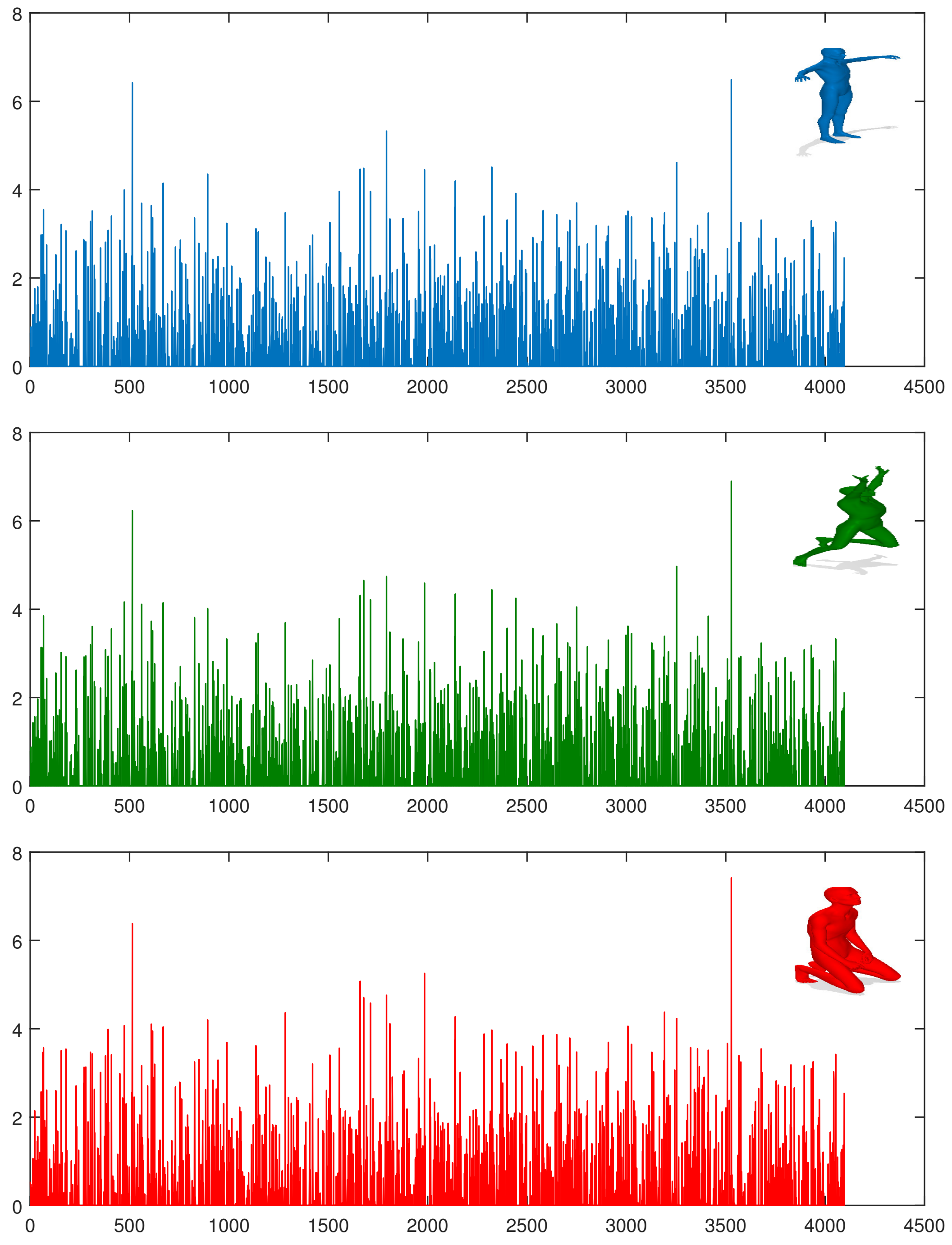


Figure 4.9: Deep-CSA descriptors learned by our approach for a male model with different poses.

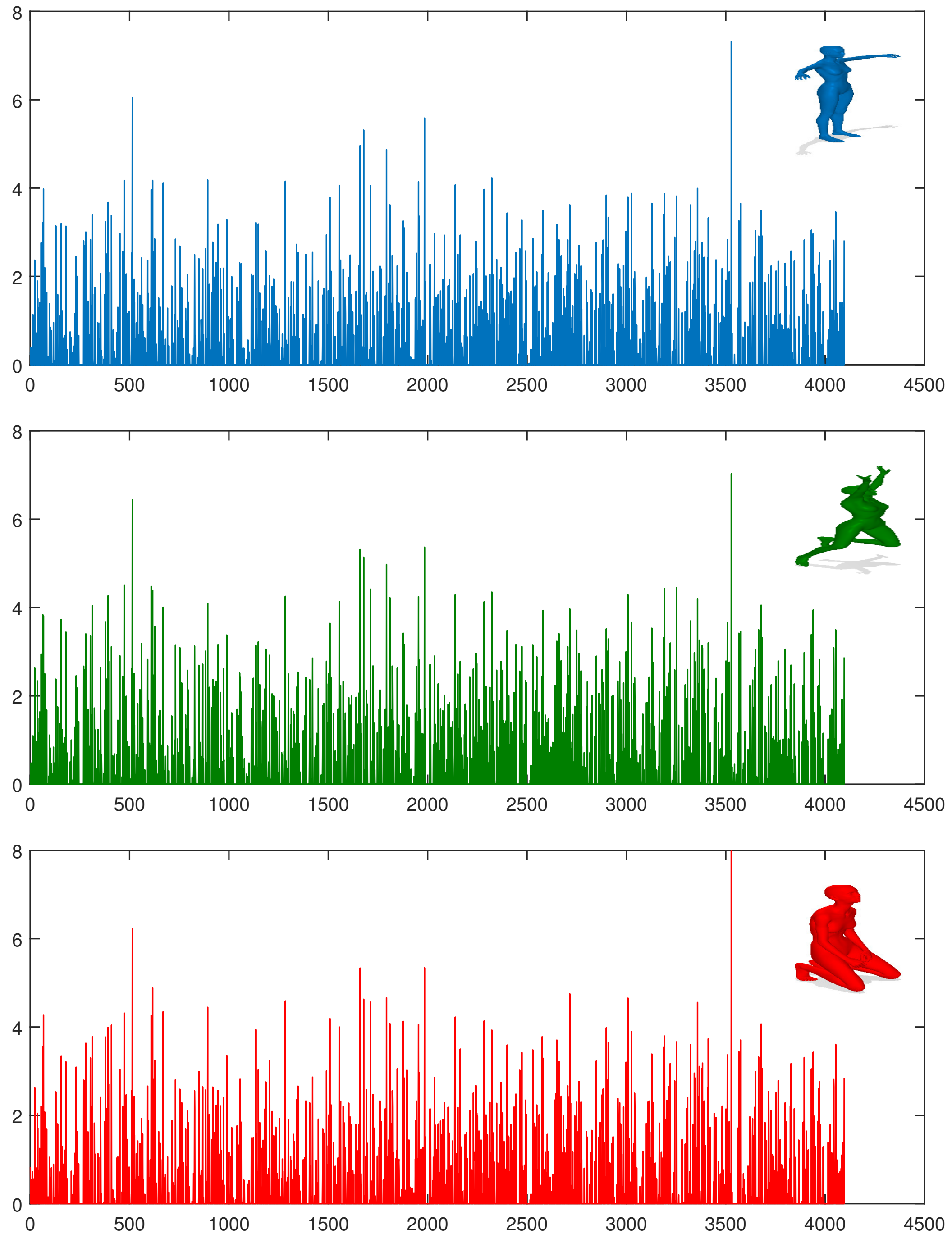


Figure 4.10: Deep-CSA descriptors learned by our approach for a female model with different poses.

CHAPTER 5

Conclusions and Future Work

This thesis has presented two high-level shape descriptors namely deep SA-BoF and DLSD for the task of 3D shape retrieval, and one high-level shape descriptor called Deep-CSA in order to classify 3D shapes. We have demonstrated through extensive experiments the much better performance of the proposed methods in comparison with existing techniques in the literature. In Section 5.1, the contributions made in each of the previous chapters and the concluding results drawn from the associated research work are presented. Suggestions for future research directions related to this thesis are also provided in Section 5.3.

5.1 Contributions of the Thesis

5.1.1 Shape-Aware Descriptor for 3D Object Retrieval

In Chapter 2, we presented a multi-level feature learning framework for 3D shape retrieval using deep learning. First, low-level local descriptors were obtained using spectral graph wavelets. Then, mid-level features were extracted via the bag-of-features model by aggregating local descriptors into global ones. We used locality-constrained linear coding as a feature coding method and measured the spatial relationships between codewords using the biharmonic distance in a bid to generate shape-aware bag-of-features as mid-level features. Finally, high-level features were learned using a deep auto-encoder. The proposed approach achieves significantly better performance than state-of-the-art methods.

5.1.2 Intrinsic Spatial Pyramid Matching for 3D Shape Retrieval

In Chapter 3, we proposed a multi-level feature learning paradigm for 3D shape retrieval using deep learning in conjunction with intrinsic spatial pyramid matching. Low-level local descriptors were again obtained using spectral graph wavelets. Mid-level features were also extracted via the bag-of-features model in which we employ the biharmonic distance together with intrinsic spatial pyramid matching to even better measure the spatial relationship between the locality-constrained linear codes. Finally, high-level features were learned using a deep auto-encoder. The proposed approach achieves significantly better performance than state-of-the-art methods.

5.1.3 Convolutional Shape-Aware Representation for 3D Object Classification

In Chapter 4, we proposed a deep convolutional shape-aware framework for 3D shape classification using convolutional neural networks to hierarchically learn robust feature representations. The proposed framework took a multi-stage approach that first represents each 3D shape in the dataset as a 2D image using the mid-level feature we extracted in Chapter 3 called SA-BoF-ISPM. These 2D images were then fed into a pre-trained convolutional neural network to extract high-level feature representations from the penultimate fully-connected layer of the network, resulting in Deep-CSA descriptors. To assess the performance of our framework, we trained a multiclass support vector classifier on the deep learned shape descriptors, and we showed via rigorous experimental evaluations on several datasets that the proposed Deep-CSA approach outperforms existing methods for 3D shape classification by a relatively large margin

5.2 Limitations

A key advantage of the proposed deep shape descriptors in this thesis is their ability to exploit discriminative information by learning several hierarchical nonlinear mappings, resulting in improved retrieval and classification performance. While deep learning models encode features more efficiently than shallow models, they are, however, prone to over-fitting due largely to the added layers of abstraction. In addition, the features learned by deep learning approaches are often not easily interpretable as is the case with most neural networks. This lack of insight may be considered one the main disadvantages of our deep shape descriptors compared to the traditional 3D shape retrieval and classification methods. Another limitation of our 3D shape descriptors presented in Chapters 2 and 3 is the computation time for learning the parameters of the model, albeit the use of graphics processing units (GPUs) can help ward off this issue. Furthermore, we need to compute the distances between all pairs of mid-level features concatenated based on both possible isocontours sequences (in Chapters 3 and 4) in order to find the right one that increases the computational burden.

5.3 Future Work

Several interesting research directions, motivated by this thesis, are discussed below:

5.3.1 Apply Deep Learning Directly to 3D shapes

Unlike images, a 3D mesh cannot be fed directly into a deep learning model. To tackle this issue, we used multi-level feature learning framework to be able to extract features learned by deep learning. Alternatively, some works proposed view-based deep learning approaches which represent a 3D shape using a set of 2D multi-view images as input for a deep model. View-based methods tend to suffer from a relatively long running time due primarily to analyzing a large amount of redundant data provided by multi-view images. Our multi-level feature learning approaches also increase the computational burden by representing a 3D shape using mid-level features or a 2D geometric informative image on which a deep model can be applied. But is there any way to avoid all these preprocessing by applying deep architectures directly to the 3D shapes? Fortunately, some studies have proposed to apply deep models directly to graph-structured data [67, 85, 86] in very recently. Inspired by this breakthrough, a future plan may be to develop a framework in order to directly apply such deep architecture to 3D mesh models.

5.3.2 Pre-trained CNN Models on 3D Shapes

Although we are facing a dramatic substantial increases in the availability of 3D shape data thanks to the recent advances in 3D scanning techniques, image domain still dominates the visual world. This ubiquity can be best characterized when it comes to pre-trained deep models where all of them have been trained using images as input and not 3D shapes. These models are usually trained using tens of thousands of images in order to include all kind of patterns and textures during training so that these pre-trained models can be used as a feature extractor later on without spending time and effort on training. To the best of our knowledge, there is no pre-trained models trained using 3D shapes as input at the present time. We, therefore, adopted a transfer-learning strategy in Chapter 4 to extract high-level features by applying a pre-trained CNN model trained using natural images on mid-level features extracted from 3D shapes which were represented as 2D informative graphical data. 3D shapes, however, are not very rich in terms of high-frequency details which are presented via different patterns and textures in real scene images and if there is anything, very often, those details are buried in noises. As a result, when the mid-level features which hold the geometric information of a 3D shape are represented as a 2D image do not show the typical patterns and textures of natural images. In spite of the superiority of our Deep-CSA approach for 3D shape classification over state-of-the-art methods, we think this approach may be improved by applying a pre-trained deep model trained on 3D shapes. This seems to come true especially with recently introduced ShapeNetCore subset of ShapeNet [4] which contains about 51,300 3D models. Another future direction, hence, is to obtain pre-trained models which are trained using immense number of 3D shapes as input and then use them as feature extractors. As we suggested in previous section that deep learning may be directly applied to 3D models, these pre-trained models, therefore, may be also trained directly on 3D shapes without representing them as 2D images.

5.3.3 3D Shape Clustering

Unlike classification in which objects are assigned to predefined classes, clustering is different in the sense that the number(and labels) of clusters or the cluster structure are not known in advance. The core goal of 3D shape clustering is to organize a dataset of 3D shapes into homogeneous subgroups or clusters in an unsupervised manner using a pre-defined similarity of dissimilarity measure. These clusters are formed in such a way that objects in the same cluster are very similar, while objects in different clusters are very dissimilar. In this thesis, we introduced several high-level 3D shape descriptors on which either shape retrieval or classification has been tested, yielding the much better performance compared to existing methods. Nevertheless, these 3D shape descriptors can be used for other 3D shape analysis applications such as 3D shape clustering. The K-means algorithm is arguably one of the most popular and effective clustering methods. In a nutshell, K-means assigns each data point to the cluster having the nearest centroid. One future work, therefore, is to apply the K-means algorithm on the deep SA-BoF, DLSD, and Deep-CSA shape descriptors to assess their performance on 3D shape clustering.

5.3.4 Exploring New Deep Learning Models

Deep learning is a leading tool employed to address numerous problems in machine learning. The reason that deep learning has been successfully applied to many fields, providing significant improvements is its growing advancements. Many improved deep models have been introduced in very recent years. For instance, Variational Auto-Encoder [87] is a stochastic variational inference and learning algorithm. It uses stochastic gradient variational Bayes (SGVB) estimator to efficiently learn the model parameters from large-scale datasets without involving in costly iterative inference schemes. He *et al.* [88] proposed a modified deep convolutional neural network by introducing a new pooling strategy called spatial pyramid pooling to be able to choose the size of the input image. Diffusion-convolutional neural network (DCNN) [85] uses a diffusion process rather than the standard convolution operation by scanning a square of parameters across the input. DCNN has been claimed to improve the accuracy, flexibility, and the speed. Replacing the deep models used in this thesis by the new models can be another future research plan in order to improve the performance of 3D shape retrieval and classification.

5.3.5 From Image Processing to Geometry Processing

Generally speaking, this thesis provides a bridge to borrow ideas from image processing for geometry processing including wavelet framework for local shape descriptors' design, bag-of-features paradigm, the intrinsic global coordinate system, a pre-trained CNN model trained on images, and shape classification and retrieval themselves. Abstractly, it generalizes methods in the Euclidean space to the weighted graph space, resulting in a fruitful way to understand 3D shapes by extending sophisticated methods in image domain via these tools. One future research direction, therefore, could be to explore other tools to link these two fields such as covariance-based descriptors and multimodal learning where both images and 3D shapes are used in learning process.

References

[1] D. Pickup, X. Sun, P. Rosin, R. Martin, Z. Cheng, Z. Lian, M. Aono, A. Ben Hamza, A. Bronstein, M. Bronstein, S. Bu, U. Castellani, S. Cheng, V. Garro, A. Giachetti, A. Godil, J. Han, H. Johan, L. Lai, B. Li, C. Li, H. Li, R. Litman, X. Liu, Z. Liu, Y. Lu, A. Tatsuma, and J. Ye, "SHREC'14 track: Shape retrieval of non-rigid 3D human models," in *Proc. Eurographics Workshop on 3D Object Retrieval*, pp. 1–10, 2014.

[2] J. Z. Z. Lian, S. Choi, H. ElNaghy, J. El-Sana, T. Furuya, A. Giachetti, R. G. L. Isaia, L. Lai, C. Li, H. Li, F. Limberger, R. Martin, R. Nakanishi, A. N. L. Nonato, R. Ohbuchi, K. Pevzner, D. Pickup, P. Rosin, A. Sharf, L. Sun, X. Sun, S. Tari, G. Unal, and R. Wilson, "SHREC'15 track: Non-rigid 3D shape retrieval," in *Proc. Eurographics Workshop on 3D Object Retrieval*, pp. 1–14, 2015.

[3] M. Savva, F. Yu, H. Su, M. Aono, B. Chen, D. Cohen-Or, W. Deng, H. Su, S. Bai, X. Bai, J. H. N. Fish, E. Kalogerakis, E. Learned-Miller, Y. Li, M. Liao, S. Maji, Y. Wang, N. Zhang, and Z. Zhou, "SHREC'16 track: Large-scale 3D shape retrieval from ShapeNet Core55," in *Proc. Eurographics Workshop on 3D Object Retrieval*, 2016.

[4] M. Savva, F. Yu, H. Su, A. Kanezaki, T. Furuya, R. Ohbuchi, Z. Zhou, R. Yu, S. Bai, X. Bai, M. Aono, A. Tatsuma, S. Thermos, A. Axenopoulos, G. T. Papadopoulos, P. Daras, X. Deng, Z. Lian, B. Li, H. Johan, and Y. L. S. Mk, "SHREC'17 track: Large-scale 3D shape retrieval from ShapeNet Core55," in *Proc. Eurographics Workshop on 3D Object Retrieval*, 2017.

[5] R. Rustamov, "Laplace-Beltrami eigenfunctions for deformation invariant shape representation," in *Proc. Symp. Geometry Processing*, pp. 225–233, 2007.

[6] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," *Computer Graphics Forum*, vol. 28, no. 5, pp. 1383–1392, 2009.

[7] M. Bronstein and I. Kokkinos, "Scale-invariant heat kernel signatures for non-rigid shape recognition," in *Proc. CVPR*, pp. 1704–1711, 2010.

[8] M. Aubry, U. Schlickewei, and D. Cremers, "The wave kernel signature: A quantum mechanical approach to shape analysis," in *Proc. Computational Methods for the Innovative Design of Electrical Devices*, pp. 1626–1633, 2011.

[9] C. Li and A. Ben Hamza, "A multiresolution descriptor for deformable 3D shape retrieval," *The Visual Computer*, vol. 29, pp. 513–524, 2013.

[10] R. Guler, S. Tari, and G. Unal, "Screened poisson hyperfields for shape coding," *SIAM Journal on Imaging Sciences*, vol. 7, no. 4, pp. 2558–2590, 2014.

[11] A. Chaudhari, R. Leahy, B. Wise, N. Lane, R. Badawi, and A. Joshi, "Global point signature for shape analysis of carpal bones," *Physics in Medicine and Biology*, vol. 59, pp. 961–973, 2014.

[12] J. Ye and Y. Yu, "A fast modal space transform for robust nonrigid shape retrieval," *The Visual Computer*, vol. 32, no. 5, pp. 553–568, 2015.

[13] W. Mohamed and A. Ben Hamza, "Deformable 3D shape retrieval using a spectral geometric descriptor," *Applied Intelligence*, vol. 45, no. 2, pp. 2213–229, 2016.

[14] S. Rosenberg, *The Laplacian on a Riemannian Manifold*. Cambridge University Press, 1997.

[15] A. Bronstein, M. Bronstein, and R. Kimmel, *Numerical Geometry of Non-rigid Shapes*. Springer, 2008.

[16] H. Krim and A. Ben Hamza, *Geometric methods in signal and image analysis*. Cambridge University Press, 2015.

[17] K. Noda, Y. Yamaguchi, K. Nakadai, H. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Applied Intelligence*, vol. 42, no. 4, pp. 722–737, 2015.

[18] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. CVPR*, pp. 1912–1920, 2015.

[19] C. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. CVPR*, 2016.

[20] Z. Zhu, X. Wang, S. Bai, C. Yao, and X. Bai, "Deep learning representation using autoencoder for 3D shape retrieval," *Neurocomputing*, 2016.

[21] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[22] D. Ciresan, U. Meier, J. Masci, L. Gambardella, and J. Schmidhuber, “Flexible, high performance convolutional neural networks for image classification,” in *Proc. IJAC*, pp. 1237–1242, 2011.

[23] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “OverFeat: Integrated recognition, localization and detection using convolutional networks,” in *Proc. ICLR*, 2014.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proc. CVPR*, 2015.

[25] H. Lee, R. Grosse, R. Ranganath, and A. Ng, “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” in *Proc. ICML*, pp. 609–616, 2009.

[26] J. Masci, U. Meier, D. Ciresan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” in *Proc. Int. Conf. Artificial Neural Networks*, p. 52Ű59, 2011.

[27] G. Hinton, S. Osindero, and Y. Teh, “A fast learning algorithm for deep belief nets,” *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[28] G. Hinton, “A practical guide to training restricted Boltzmann machines,” in *Neural Networks: Tricks of the Trade*, pp. 599–619, 2012.

[29] J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in *NIPS*, pp. 350–358, 2012.

[30] A. Krizhevsky, I. Sutskever, and G. Hinton, “ImageNet classification with deep convolutional neural networks,” in *NIPS*, pp. 1097–1105, 2012.

[31] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in *Proc. CVPR*, pp. 3128–3137, 2015.

[32] J. Zhang, S. Shan, M. Kan, and X. Chen, “Coarse-to-fine autoencoder networks (CFAN) for real-time face alignment,” in *Proc. ECCV*, pp. 1–16, 2014.

[33] S. Eslami, N. Heess, C. Williams, and J. Winn, “The shape Boltzmann machine: A strong model of object shape,” *Int. Journal of Computer Vision*, vol. 107, no. 2, pp. 155–176, 2014.

[34] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, “Multi-view convolutional neural networks for 3D shape recognition,” in *Proc. ICCV*, pp. 945–953, 2015.

[35] Y. Lipman, R. Rustamov, and T. Funkhouser, “Biharmonic distance,” *ACM Trans. Graphics*, vol. 29, no. 3, pp. 1–11, 2010.

[36] T. Darom and Y. keller, “Scale-invariant features for 3D mesh models,” *IEEE Trans. Image Processing*, vol. 21, no. 5, pp. 2758–2769, 2012.

[37] A. Zaharescu, E. Boyer, K. Varanasi, and R. Horaud, “Surface feature detection and description with applications to mesh matching,” in *Proc. CVPR*, pp. 373–380, 2009.

[38] H. Tabia, H. Laga, D. Picard, and P. Gosselin, “Covariance descriptors for 3D shape matching and retrieval,” in *Proc. CVPR*, pp. 4185–4192, 2014.

[39] Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavoué, H. Nguyen, R. Ohbuchi, Y. Ohkita, Y. Ohishi, F. Porikli, M. Reuter, I. Sipiran, D. Smeets, P. Suetens, H. Tabia, and D. Vandermeulen, “A comparison of methods for non-rigid 3D shape retrieval,” *Pattern Recognition*, vol. 46, no. 1, pp. 449–461, 2013.

[40] C. Li and A. Ben Hamza, “Spatially aggregating spectral descriptors for nonrigid 3D shape retrieval: A comparative survey,” *Multimedia Systems*, vol. 20, no. 3, pp. 253–281, 2014.

[41] F. Limberger and R. Wilson, “Feature encoding of spectral signatures for 3D non-rigid shape retrieval,” in *Proc. BMVC*, 2015.

[42] M. Reuter, F. Wolter, and N. Peinecke, “Laplace-Beltrami spectra as ‘Shape-DNA’ of surfaces and solids,” *Computer-Aided Design*, vol. 38, no. 4, pp. 342–366, 2006.

[43] Z. Gao, Z. Yu, and X. Pang, “A compact shape descriptor for triangular surface meshes,” *Computer-Aided Design*, vol. 53, pp. 62–69, 2014.

[44] A. Ben Hamza, “A graph-theoretic approach to 3D shape classification,” *Neurocomputing*, vol. 211, pp. 11–21, 2016.

[45] D. Smeets, T. Fabry, J. Hermans, D. Vandermeulen, and P. Suetens, “Isometric deformation modelling for object recognition,” in *Proc. CAIP*, pp. 757–765, 2009.

[46] X. Ying, X. Wang, and Y. He, “Saddle vertex graph (svg): a novel solution to the discrete geodesic problem,” *ACM Trans. Graphics*, vol. 32, no. 6, 2013.

[47] A. Bronstein, M. Bronstein, L. Guibas, and M. Ovsjanikov, “Shape Google: Geometric words and expressions for invariant shape retrieval,” *ACM Trans. Graphics*, vol. 30, no. 1, 2011.

[48] R. Litman, A. Bronstein, M. Bronstein, and U. Castellani, “Supervised learning of bag-of-features shape descriptors using sparse coding,” *Computer Graphics Forum*, vol. 33, no. 5, pp. 127–136, 2014.

[49] S. Bu, Z. Liu, J. Han, J. Wu, and R. Ji, “Learning high-level feature by deep belief networks for 3-D model retrieval and recognition,” *IEEE Trans. Multimedia*, vol. 24, no. 16, pp. 2154–2167, 2014.

[50] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015.

[51] A. Brock, T. Lim, J. Ritchie, and N. Weston, “Generative and discriminative voxel modeling with convolutional neural networks,” *arXiv:1608.04236*, 2016.

[52] N. Sedaghat, M. Zolfaghari, and T. Broxn, “Orientationboosted voxel nets for 3D object recognition,” *arXiv:1604.03351*, 2016.

[53] A. Kanezaki, Y. Matsushita, and Y. Nishida, “RotationNet: Joint learning of object classification and viewpoint estimation using unaligned 3D object dataset,” *arXiv:1603.06208*, 2016.

[54] Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu, and E. Wong, “3D deep shape descriptor,” in *Proc. CVPR*, pp. 2319–2328, 2015.

[55] S. Bu, L. Wanga, P. Hana, Z. Liu, and K. Li, “3-D shape recognition and retrieval based on multi-modality deep learning,” *Neurocomputing*, 2017.

[56] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. J. Latecki, “Gift: A real-time and scalable 3D shape search engine,” in *Proc. CVPR*, pp. 5023–5032, 2016.

[57] J. Xie, G. Dai, and Y. Fang, “Deep multi-metric learning for shape-based 3D model retrieval,” *IEEE Trans. Multimedia*, 2017.

[58] H. Tabia and H. Laga, “Learning shape retrieval from different modalities,” *Neurocomputing*, 2017.

[59] F. Chen, R. Ji, and L. Cao, “Multimodal learning for view-based 3D object classification,” *Neurocomputing*, vol. 195, pp. 23–29, 2016.

[60] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, “Deep learning advances in computer vision with 3D datadata: A survey,” *ACM Computing Surveys*, vol. 50, no. 2, 2017.

[61] M. Meyer, M. Desbrun, P. Schröder, and A. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," *Visualization and mathematics III*, vol. 3, no. 7, pp. 35–57, 2003.

[62] V. Jain and H. Zhang, "A spectral approach to shape-based retrieval of articulated 3D models," *Computer-Aided Design*, vol. 39, no. 5, pp. 398–407, 2007.

[63] E. Davies and Y. S. (Eds.), *Spectral Theory and Geometry*. Cambridge University Press, 1999.

[64] H. Ghodrati and A. Ben Hamza, "Deep shape-aware descriptor for nonrigid 3D object retrieval," *International Journal of Multimedia Information Retrieval*, vol. 5, no. 3, pp. 151–164, 2016.

[65] H. Ghodrati and A. Ben Hamza, "Nonrigid 3D shape retrieval using deep auto-encoders," *Applied Intelligence*, vol. 47, no. 1, pp. 44–61, 2017.

[66] H. Ghodrati, L. Luciano, and A. Ben Hamza, "Convolutional shape-aware representation for 3D object classification," *Submitted*, 2017.

[67] T. Kipf and M. WellingH, "Semi-supervised classification with graph convolutional networks," in *Under review at ICLR*, 2017.

[68] F. Monti, M. M. Bronstein, and X. Bresson, "Geometric matrix completion with recurrent multi-graph neural networks," *arXiv:1704.06803*, 2017.

[69] C. Li and A. Ben Hamza, "Intrinsic spatial pyramid matching for deformable 3D shape retrieval," *Int. Journal Multimedia Information Retrieval*, vol. 2, pp. 261–271, 2013.

[70] W. Dong, X. Li, D. Zhang, and G. Shi, "Sparsity-based image denoising via dictionary learning and structural clustering," in *Proc. CVPR*, pp. 3360–3367, 2010.

[71] A. Ben Hamza and H. Krim, "Geodesic matching of triangulated surfaces," *IEEE Trans. Image Processing*, vol. 15, no. 8, pp. 2249–2258, 2006.

[72] Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavoué, H. Nguyen, R. Ohbuchi, Y. Ohkita, Y. Ohishi, F. Porikli, M. Reuter, I. Sipiran, D. Smeets, P. Suetens, H. Tabia, and D. Vandermeulen, "SHREC'11 track: Shape retrieval on non-rigid 3D watertight meshes," in *Proc. Eurographics/ACM SIGGRAPH Symp. 3D Object Retrieval*, pp. 79–88, 2011.

[73] A. Giachetti and C. Lovato, “Radial symmetry detection and shape characterization with the multiscale area projection transform,” *Computer Graphics Forum*, vol. 31, no. 5, pp. 1669–1678, 2012.

[74] D. Pickup, X. Sun, P. Rosin, and R. Martin, “Geometry and context for semantic correspondences and functionality recognition in manmade 3D shapes,” *Pattern Recognition*, vol. 48, no. 8, pp. 2500–2512, 2015.

[75] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, “The Princeton shape benchmark,” in *Proc. SMI*, pp. 167–178, 2004.

[76] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[77] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[78] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. The MIT Press, 2016.

[79] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, “Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling,” in *NIPS*, 2016.

[80] M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, “Geometric deep learning: going beyond Euclidean data,” *arXiv:1611.08097*, 2016.

[81] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *ICLR*, 2015.

[82] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016.

[83] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, “Locality-constrained linear coding for image classification,” in *Proc. CVPR*, pp. 3360–3367, 2010.

[84] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, “Return of the devil in the details: Delving deep into convolutional nets,” in *Proc. BMVC*, 2014.

[85] J. Atwood and D. Towsley, “Diffusion-convolutional neural networks,” in *Proc. NIPS*, 2016.

[86] Y. Hechtlinger, P. Chakravarti, and J. Qin, “A generalization of convolutional neural networks to graph-structured data,” *arXiv:1704.08165*, 2017.

[87] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv:1312.6114*, 2013.

[88] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.