Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Learning Geometric Complexes for 3D Shape Classification

ARTICLE INFO

Article history:
Received September 26, 2024

Keywords: Skeletonization, Simplicial Complex, Alpha Complex, Topology, Point Cloud Classification

ABSTRACT

Geometry and topology are vital elements in discerning and describing the shape of an object. Geometric complexes constructed on the point cloud of a 3D object capture the geometry as well as topological features of the underlying shape space. Leveraging this aspect of geometric complexes, we present an attention-based dual stream graph neural network (DS-GNN) for 3D shape classification. In the first stream of DS-GNN, we introduce spiked skeleton complex (SSC) for learning the shape patterns through comprehensive feature integration of the point cloud's core structure. SSC is a novel and concise geometric complex comprising principal plane-based cluster centroids complemented with per-centroid spatial locality information. The second stream of DS-GNN consists of alpha complex which facilitates the learning of geometric patterns embedded in the object shapes via higher dimensional simplicial attention. To evaluate the model's response to different shape topologies, we perform a persistent homology-based object segregation that groups the objects based on the underlying topological space characteristics quantified through the second Betti number. Our experimental study on benchmark datasets such as ModelNet40 and ScanObjectNN shows the potential of the proposed GNN for the classification of 3D shapes with different topologies and offers an alternative to the current evaluation practices in this domain.

1. Introduction

In the domain of computer graphics and vision, the classification of three-dimensional (3D) shapes serves as a foundation for understanding the physical world and hence facilitates applications in robotics, augmented reality, and autonomous driving, among others. Amidst the various representations of 3D objects, point clouds are a versatile representation of 3D shapes due to their ability to capture intricate details with precision and efficiency. Moreover, the rapid advancement of 3D sensing technologies such as Light Detection and Ranging (LiDAR) scanning and depth cameras have brought point clouds to the forefront as a fundamental representation of spatial information making them indispensable in many real-world applications [1]. Point cloud classification deals with the process of identifying the overall properties of the point cloud to assign a common category or class to each point cloud. However, raw point clouds are often limited by their sparsity, irregularity, and noise which present inherent complexities in tasks such as object classification and scene understanding. Sophisticated approaches that leverage both geometric properties and topological relationships within the point cloud can help overcome such limitations. To be more specific, geometric features such as curvature, normal vectors, and surface descriptors provide information about the local shape, orientation, and distribution of points. Topological information (e.g., connectivity and spatial arrangements), on the other hand, aids in identifying meaningful patterns and relationships within the point cloud. By integrating these approaches, point cloud classification algorithms can robustly navigate through shortcomings of point cloud, ultimately improving the accuracy and robustness of point cloud processing tasks.

Many previous deep learning methods [1] have employed the processing of point clouds to learn rich information about the geometry of the underlying 3D shape. These works present

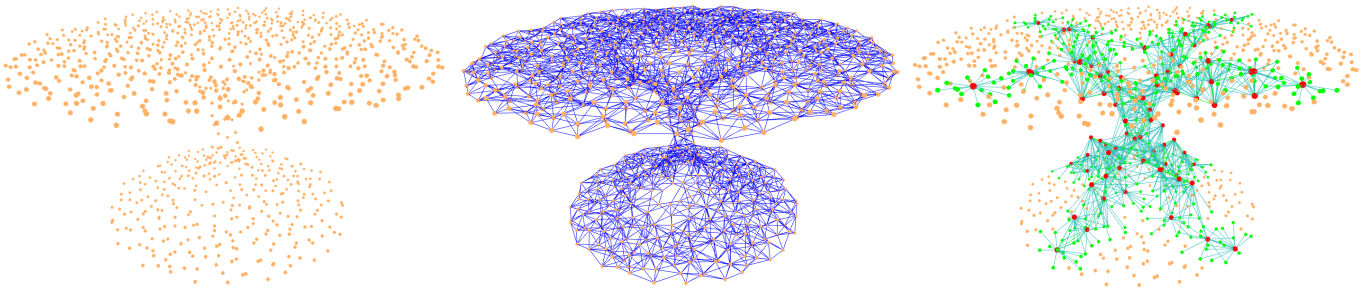

Figure 1: Geometric complexes used in our method. Note that orange shows the points in the original point cloud, blue shows the edges of the Alpha complex, and red and green show the nodes and the edges of the Spiked Skeleton complex of the point cloud, respectively.

a variety of deep learning models ranging from the neural networks that can directly operate on point clouds [2, 3], to the learning models that process a different representation of point clouds such as multi-view projection, voxel grid, or octree, or graphs [4, 5] for better representation of the 3D scene/object. As defined in terms of vertices and edges, graphs leverage the relative neighborhood information to provide additional knowledge of connectedness among points. Existing graph-based methods mostly rely on the geometric information of the point cloud which provides access to local properties of shapes, such as angles and distances between nearby points. However, topological representations such as geometrical complexes [6] provide a deeper perception of the underlying structure of space and hence capture global properties of spaces such as connectedness and compactness. Moreover, such representations aid in understanding the more subtle properties of the space that are often ignored by conventional geometric measurements. Therefore, the utilization of topology along with geometry for point cloud analysis can provide a more complete understanding of spatial structures.

Topological features are often adaptable to complex structures which assist in filtering out the noise or outliers. The structural features of the simplified representation of a 3D shape, often known as the skeleton of the shape, contribute to the discriminative power, efficiency, robustness, and contextual understanding of 3D shape. Skeletal nodes represent semantic abstractions of the object's structure. Thus, the connectivity of a skeleton can capture global context and long-range dependencies within the shape. This allows the deep learning network to learn the spatial relationships among different parts of the object, even when they are distant from each other in the point cloud representation. Therefore these features enhance the ability to accurately capture the underlying structural characteristics of the object. Furthermore, the proven ability of skeleton data to learn discriminative features for action recognition and other applications [7] could be beneficial for object classification. Therefore, in this work, we present a concise representation of a point cloud that filters out irrelevant details in the point cloud while preserving the local neighborhood knowledge.

However, the capacity of skeletons to capture subtle variations on the shape boundaries is limited and hence, may lead to sub par performance while dealing with the analysis of similar objects such as bench and table. To complement such limitations of the skeleton graph in shape analysis, we employ a geometric complex constructed over the Euclidean space covering the object shape. The majority of graph-based methods have used the relative neighborhood or k-nearest neighbor (kNN) algorithm to determine the neighborhood of each point in the point cloud. Such neighborhood selection is solely based on the distance between the two points and hence only contributes to the geometric knowledge of the 3D shape and often loses the topological context of the shape. The alpha complex representation of the point cloud complements the spiked skeleton complex structure through two and lower-dimensional simplices formed by the point cloud data and hence captures fine-grained details and subtle variations in shape that may be overlooked by skeletal representation alone. The higher-dimensional simplicial knowledge of point cloud captures the relationship among three points beyond just pairwise connections which implies a higher degree of local connectivity to provide insights into the geometric structure of the point cloud. Furthermore, alpha complex captures the relationship between different parts of the point cloud and preserves topology which improves the performance of the point cloud classification [8].

Leveraging the complementarity of skeleton graph- alpha complex pair in capturing the geometric and topological characteristics of shapes, we design a graph attention-based dual-stream network for 3D shape classification. The features extracted from graph representations are fused to gather insights into the connectedness of the object; an important aspect when dealing with densely packed or overlapping structures. To strengthen the non-local feature aggregation power of the skeleton complex, we attach spatial locality information to each skeleton node. This helps in generating a feature map representative of the whole shape generated through the skeleton node feature aggregation process. We analyze the potentials of the proposed dual-stream network under a new evaluation paradigm that segregates the benchmark data into categories based on persistent homology and rectilinearity. To summarize, the key contributions of this paper are quadruple;

- **Spiked Skeleton Complex (SSC):** Inspired by the topology-preserving ability of skeleton graphs, we propose a concise graph-based representation of point clouds called spiked skeleton complex that aids in non-local feature ag-

gregation for point cloud learning;

- **Dual-Stream GCN (DS-GCN):** We implement a graph attention-based dual network pipeline that extracts global features of the point cloud from its spiked skeleton complex and aggregates local features from high dimensional simplices in alpha complex representation and fuse the extracted features to perform 3D shape classification;

- **ModelNet40 Grouping:** Unlike previous studies that give little to no attention to the shape topologies in performance evaluation, we study the efficacy of our model under segregation of ModelNet40 (the main benchmark for classification) objects into zero Vs non-zero second Betti number (β_2) and rectilinear Vs freeform objects. To the best of our knowledge, no such classification of the Modelnet40 dataset has been undertaken yet.

The rest of the paper is organized as follows. We discuss the existing state-of-the-art on 3D shape classification in Section 2. Section 3 discusses the definition of SSC and other fundamental concepts. Section 4 describes the proposed network architecture. In Section 5, we present the quantitative and qualitative analysis of the proposed algorithm for the classification of the point cloud. Finally, Section 6 provides the conclusion and future directions.

2. Related Work

2.1. Point Cloud Classification

Early deep learning models for point cloud classification are designed to operate either on the projection of point clouds on 2D image space or the 3D volumetric grid (voxel/octree) generated from the point clouds [1]. These methods utilize the existing advances in the convolutional neural network (CNN) for feature learning for image classification. However, in addition to high memory consumption and increased computational complexity, there is a significant loss of contextual knowledge of the 3D shape due to the complete transformation of the input into lower-dimensional data or in the quantized voxels. To overcome the limitations of these methods, point-based methods [2, 3] are introduced that directly operate on raw point clouds and efficiently utilize the characteristics of the raw point cloud while reducing the computational complexity of the deep learning network. The pioneering work PointNet [2] uses multilayer perceptron (MLP) to extract features of each point individually and employ a symmetric function to achieve permutation invariance for regular and unordered point clouds. In the follow-up work [3] the authors extract layer-by-layer local features along with the global features obtained through PointNet with the help of a sampling layer, grouping layer, and multiple Pointnet layers. These methods have been used in many following works which have built upon these architectures to obtain global and local feature embeddings for point cloud learning [9]. Few methods have applied convolution operation on point data by generating a weighted sum over the subset of the point set [1, 9]. Most of these methods provide a flexible and computationally efficient way to perform point cloud processing. However, these approaches only consider the spatial information of the point data without any knowledge of spatial relationships such as direction or distance between the neighboring points which is quite essential for understanding 3D objects of varying shapes and sizes.

2.2. Graph-based Methods

Graphs are unique representations of the point data and provide underlying relationships among data points in the point cloud. Therefore, to leverage these expressive capabilities of graph structures, graph neural networks (GNNs) have been explored to learn the global as well as local geometric features of point clouds. Graph convolution network (GCN) is a variant of CNN that is optimized for the graph data. Several methods apply convolution directly on the graphs and exploit the edge information to encode the underlying shape features [9]. These methods retain the properties of conventional convolution on the regular grids i.e. locality and weight sharing using the distance between the nodes as weights for weighted average convolution. In the pioneering work [4] on graph convolution, authors proposed an edge conditional convolutional network that constructs directed edges between all neighbors of each vertex and considers these distances as the edge weights to perform weighted average convolution. DGCNN [10] dynamically searches the local neighborhood of points using the nearest neighbor to construct a graph of the point cloud after each network layer and aggregates learned feature maps using channel-wise symmetric aggregation. Several later works have adopted DGCNN [10] in their network to leverage the convolution on graph edges [1]. GridGCN [11] presents a fast and scalable solution for point cloud classification by improving the spatial coverage in grid space using coverage-aware network query learned through grid context aggregation.

Few graph-based networks [12, 13, 14, 15] apply spectral filtering for the convolution on the graphs. RGCNN [12] learns the complete graph of the point cloud using the graph Laplacian matrix updated in each layer. AGCN [13] operates on the graph using a learnable distance metric to define the similarity among the vertices of the graph. These networks use full graphs and hence are computationally expensive. On the contrary, PointGCN [14] and LocalSpecGCN [15] use k-nearest neighbors to construct local graphs and apply convolutional filters in the graph spectral domain. These models provide solutions to utilize the connectivity knowledge of graphs of the point cloud however they are often sensitive to noise/outliers and computationally intensive for large point clouds.

To leverage the added connectivity information in graphs, graph attention has been used in many further works. The concept of attention mechanism [16] on GNNs was introduced as graph attention networks (GATs) [17] which involves a self-attention-based convolution-style neural network that deals with the different-sized neighborhoods for assigning different importance of the neighboring points to the center point. A recent deep learning model FatNet [18] combines PointNet [2] and DGCNN [10] to capture global and local embeddings using an attention mechanism.

While Global features are obtained using point embeddings,

local features rely on the edges. Among many other works that followed the concept of attention, transformer networks [19, 20] efficiently deal with the order invariance of raw point clouds without forming the graph of the point cloud. However, they are limited to the self-attention mechanism and cannot utilize other semantic knowledge of the 3D shape for the attention operation which could be obtained through graph-structured data. The aforementioned works provide a way to focus on the relatively important local features to capture the local geometry of the 3D shape. The major benefit of the attention mechanism is that these networks consider the most relevant parts of the input for decision-making while allowing it to deal with variable-sized inputs. Therefore, these networks could be beneficial for learning about the local neighborhood through the importance of neighboring.

2.3. Skeletons for Point Cloud Learning

3D skeleton extraction methods in point cloud processing aim to identify and extract the essential structural features enabling reasoning about shape geometry and topology compared to traditional representations. These methods typically utilize algorithms such as voxelization, medial axis transformation, or distance-based approaches to represent the underlying structure [21]. These skeletonization methods have been used for many point cloud processing tasks, however, they require complex heuristics to extract relevant features and make accurate predictions for point cloud classification. A recent GCN-based model in [5] constructs a Reeb graph of the point cloud which presents the high-order skeletal representation of 3D shape to provide local contextual knowledge of the underlying shape. Reeb graph is a concise topological representation of a 3D space obtained by corrugating the connected components of level sets to points and helps in extracting the global context of 3D shapes. This work shows that skeleton representations provide an efficient alternative to the graph formulation for point cloud classification.

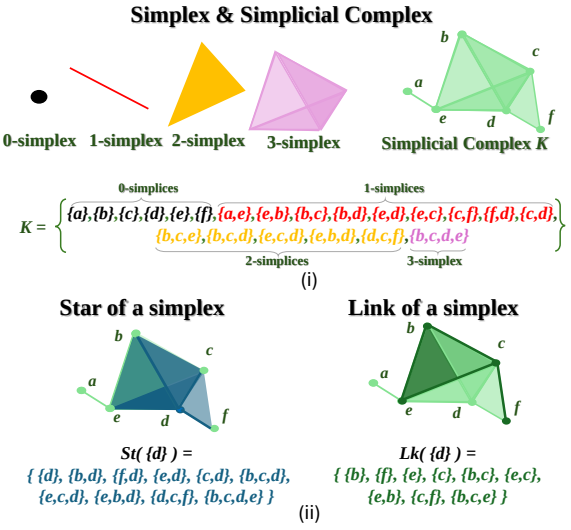

Figure 2: (i) Different simplices and a simplicial complex K, (ii) Example of the star (left) and link (right) of simplex $\{d\}$ in K.

3. Preliminaries

In this section, we formally define *spiked skeleton complex* and review other necessary fundamentals (e.g., alpha complex) required to explain the deep learning model for point cloud classification. To learn the fundamentals and more advanced details on geometric complexes, we refer the readers to [6].

3.1. Spiked Skeleton Complex

Let a finite point set in Euclidean space, denoted as $X \in \mathbb{R}^d$ be a topological space. For a non-negative k, a k-simplex $\sigma \subset \mathbb{R}^d$ is defined as the convex hull of $k+1$ affinely independent points in Euclidean space $\mathbb{R}^d$. A k-simplex consists of $k+1$ vertices. A non-empty set of simplices (σ) is defined as an abstract simplicial complex K if all the non-empty subsets $\sigma' \subseteq \sigma$ of every $\sigma \in K$ are also in K. Figure 2(i) shows a simplicial complex with six 0-simplices (vertices), nine 1-simplices (edges), four 2-simplices (triangles), and one 3-simplex (tetrahedron). *Star* $St(\sigma)$ of a simplex $\sigma \in K$ is the set of all the simplices in K that have the simplex σ as a face. *Link* $Lk(\sigma)$ of simplex $\sigma \in K$ is the set of all simplices in the closed star of σ which are disjoint from σ. Figure 2(ii) shows an example of star and link of a 0-simplex $\{d\}$ in the simplicial complex K shown in Figure 2(i). Star and link in the simplicial complex K are highlighted in dark blue and dark green respectively.

Skeleton Nodes (V_a). A point cloud $P \in \mathbb{R}^3$ consisting of N points can be described as a simplicial complex containing only 0-simplices, i.e., the vertex set $P = \{p_0, p_1, ..., p_{N-1}\}$. Let there be two projection functions $f_x : \mathbb{R}^3 \rightarrow \mathbb{R}$ that maps $p \in P$ onto first and second principal axes of P. Note that the subscript $x = \{1, 2\}$ denotes the first and second principal axes. Consider two functions $g_x : \mathbb{R} \rightarrow \{B_x^1, B_x^2, ..., B_x^m\}$ that maps the projected points $f_x(p)$ in both the principal axes onto m equal bins individually. Each bin in both projection axes is determined by $B_x^i = [a_x^i, b_x^i)$ for $1 \leq i \leq m, i \in \mathbb{N}$ where $a_x^i = min(f_x(p)) + (i-1)W$ and $b_x^i = min(f_x(p)) + iW$ with width $W = \frac{max(f_x(p)) - min(f_x(p))}{m}$ (please refer to Figure 3). Next we define a mapping $h_x : \mathbb{R}^3 \rightarrow \{B_x^1, B_x^2, ..., B_x^m\}$ between the points in the original point cloud P to one of the m bins based on the value of $f_x(p)$. All the points that map to a single bin B_x^i will constitute a cluster of P. Let function $c_x : \mathbb{R}^3 \rightarrow \{l_x^1, l_x^2, ..., l_x^m\}$

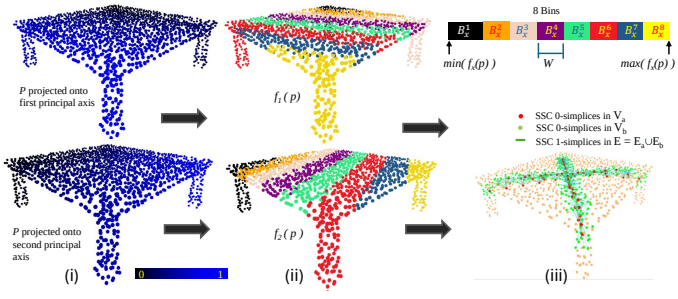

Figure 3: Stages of SSC construction: (i) Each point is assigned with the function f for first (top row) and second (bottom row) PCA eigenfunctions, (ii) B bins in the projected plane are identified for each SSC components, and (iii) SSC nodes and edges corresponding to each bin are generated which consist of the neighborhood edges for each SSC node.

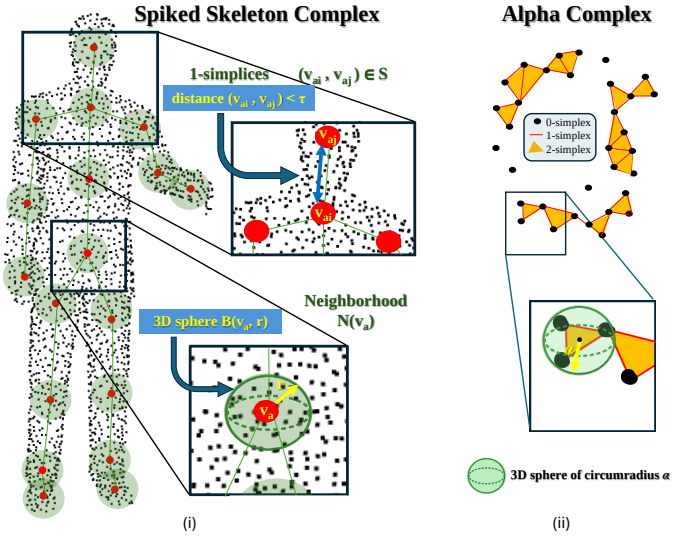

Figure 4: An overview of simplices in SSC and alpha complex. (i) In SSC, E_a consists the 1-simplices formed by the pairs of 0-simplices in V_a closer than τ. Per centroid spatial locality of SSC S is visualized in the neighborhood $N(v_a)$ for each $v_a \in S$. (ii) In alpha complex, spheres of radius α centered at circumcenter of three 0-simplices decides the underlying space of the simplicial complex.

computes the centroid of each cluster in P where l_x^i denotes the centroid of i^{th} cluster in P. We utilize the mappings f, g, h, and c to define a composite function $F : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, i.e., $F(p) = c_x(h_x^{-1}(g_x(f_x(p))))$ that maps points of the point cloud to a set V_a comprising the centroids $\{l_x^1, l_x^2, ..., l_x^m\}$ for $1 \leq i \leq m$, of all the clusters due to each principal axis in Equation 1:

$$V_a = \{l_x^i \in F(p), \forall p \in P, x \in \{1, 2\}, 1 \leq i \leq m, i \in \mathbb{N}\} \quad (1)$$

Skeleton Edges (E_a). The union of all the the pairs of vertices $(v_i, v_j) \in V_a$ that satisfy the following criteria, i.e., $\min_{v_i, v_j \in V_a} \|v_i - v_j\|_2 < \tau$.

Local Neighbor Nodes (V_b). Let $B(c, r)$ be a sphere centered at c with radius $r \geq 0$ as shown in Figure 4(i). We define V_b as the set of all points in P which lie in the neighborhoods of all the vertices in V_a as given in Equation 2.

$$V_b = \{p \in P \mid p \in B(v_a, r), \forall v_a \in V_a\} \quad (2)$$

Local Neighbor Edges (E_b). Local neighbor edges of a vertex $v \in V_a$ consist of all the edges that connect v with the points lying in its neighborhood. The set E_b is defined as follows (Equation 3).

$$E_b = \{(v, p) \mid v \in V_a, p \in P \cap B(v, r)\} \quad (3)$$

Armed with these preliminaries, let us now define *spiked skeleton complex (SSC)* of P (Definition 3.1). For the notations such as V_a, W, $f_x(p)$ etc., please refer Figure 3 and Figure 4(i).

Definition 3.1 (Spiked Skeleton Complex)**.** Given a point set $P \in \mathbb{R}^3$, the spiked skeleton complex S is a one-dimensional simplicial complex (V, E) of P having vertex set $V = V_a \cup V_b$ and the edge set $E = E_a \cup E_b$.

As SSC is defined for the point set projected on the principal axes which is equivariant under rotations, reflection, translation, and scaling. So the following two lemmas (Lemmas 3.1-3.2) which provide key insights into SSC are immediate.

Lemma 3.1. Given a topological space P equipped with an action of a group G of rigid transformations, denoted by $\phi : G \times P \rightarrow P$, where G is the group of rotations, reflections, and translations in $\mathbb{R}^3$, the spiked skeleton complex $S(P)$ is equivariant under the action of the group G.

Lemma 3.2. Given a topological space P equipped with an action of a group G, denoted by $\phi : G \times P \rightarrow P$, where G is the group of scaling transformations in $\mathbb{R}^3$, the spiked skeleton complex $S(P)$ is equivariant under the action of the group G considering the neighbourhood radius is adjusted accordingly to maintain the relative scale.

Next we reflect on a few interesting topological attributes of SSC. The star of all 1-simplices σ_1 in SSC are singleton set, i.e. $\text{St}(\sigma_1) = \{\sigma_1, \ \forall \ \sigma_1 \in S\}$. Similarly, the link of all 1-simplices σ_1 in SSC are empty set, i.e. $\text{Lk}(\sigma_1) = \{\emptyset, \ \forall \ \sigma_1 \in S\}$. Further, for a given $m, n, k \in \mathbb{Z}_{\geq 0}$ where m is the number of clusters in the point cloud of size n and k is the number of neighbors of each $v_a \in V_a$, the size of the set of 0-simplices V in SSC is $m(1 + k)$. Thus, the parameters m and k can be used to control the size of the SSC of each point cloud.

The *Betti number* β_k is a mathematical quantity that measures the number of *k-dimensional holes* in a topological space. It is computed based on the connectivity within its simplicial complexes. A chain is a linear combination of simplices in a simplicial complex and the boundary of a simplex is the sum of its faces in a simplicial complex with the orientation. A k-dimensional cycle is the chain for which the boundary is zero. Thus, a k-dimensional hole refers to a k-dimensional cycle that is not a boundary of a $(k + 1)$-dimensional chain. We observe that for a point set $P \in \mathbb{R}^3$, SSC of P lies in the underlying Euclidean space of P i.e. $S(P) \in \mathbb{R}^3$ for non-zero second-betti number β_2 objects and thus acts as a volume skeleton graph for point clouds with void(s).

3.2. Alpha Complex

Delaunay Complex $Del(P)$ of $P \in \mathbb{R}^d$ is a special simplicial complex with vertices in P where vertices of each simplex $\sigma \in Del(P)$ are in P and lie at the boundary of an empty open d-Ball. Thus, the convex hull of P coincides with the underlying space $|Del(P)|$ of Delaunay complex. Alpha complex D_p^α is a subcomplex of the Delaunay complex $Del(P)$ that is derived by filtering the larger simplices against a real-valued parameter $\alpha \geq 0$.

Definition 3.2 (Alpha Complex)**.** Given a point set $P \in \mathbb{R}^3$, alpha complex is the set of all the simplices $\sigma \in Del(P)$ whose radius r of the circumscribing ball is $0 \leq r \leq \alpha$.

All 0-simplices are part of alpha complex. As shown in Figure 4(ii), a 2-simplex is considered in the alpha complex if all three vertices lie inside or on the circumcircle of radius α. Further, a 1-simplex is part of alpha complex only if it is a face of 2-simplex and the distance between both vertices is $\leq \alpha$.

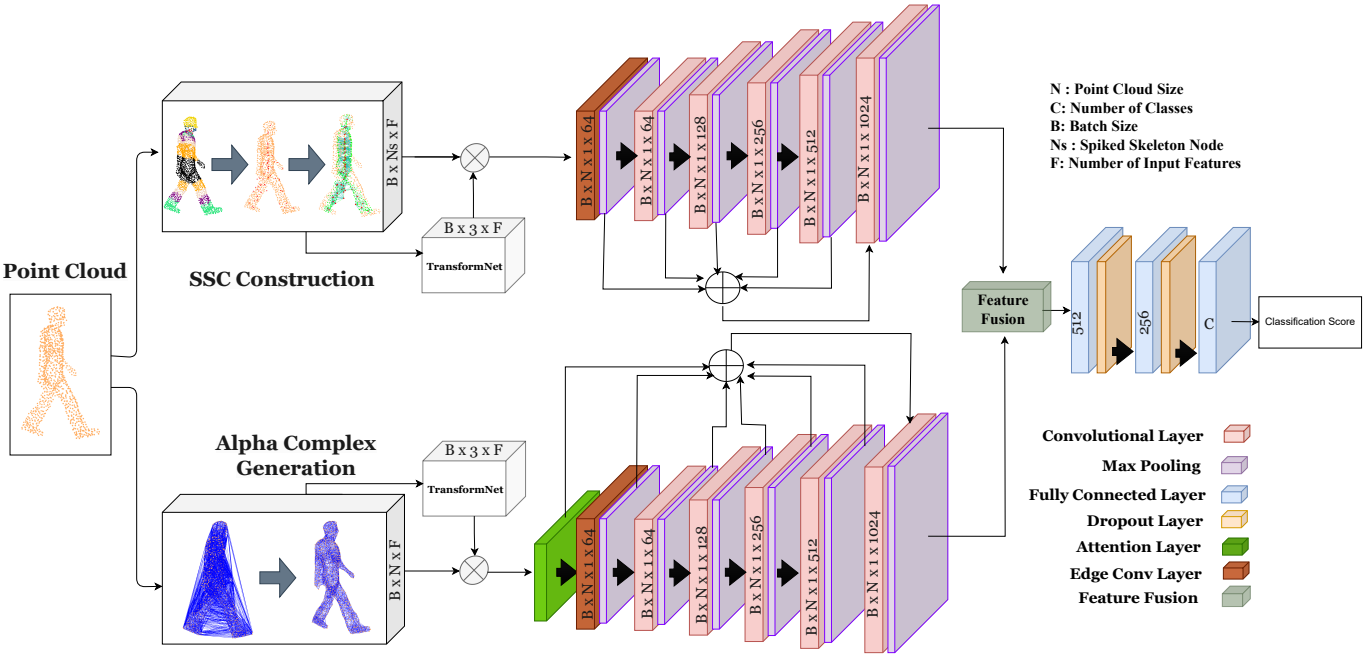

Figure 5: Architecture of proposed model consisting of two GNN streams. Each stream first constructs the SSC and the alpha complex representation of the point cloud resp. and feed the graph structures to GCN and GAT network resp.

4. Network Architecture

In this section, we discuss the network architecture including the graph construction, feature extraction and aggregation, and fusion technique. First, we construct normalized SSC (nSSC) and alpha complex of the input point cloud and then extract the feature maps of each representation using a dual-stream GNN. These feature maps are further fused to obtain the final feature map of the point cloud, which is finally fed to the classification backbone network to evaluate the classification score of each point cloud. Figure 5 shows the overall framework of the classification model.

4.1. Simplicial Complex Construction

Normalized SSC. To facilitate better learning on point cloud, we augment and normalize the SSC. First, we complement the original spatial information of points in the point cloud with a new set of features derived from the point cloud and include this additional information during principal component analysis (PCA) to evaluate the principal axes of point cloud. The additional attributes include curvature, linearity, planarity, scattering, omnivariance, eigentropy, and anisotropy as defined in [22]. As shown in Figure 3, the point cloud is projected to the first and second principal axes computed through PCA. Next, the projected point sets are divided into m equal one-dimensional bins, where each bin spans an equal interval along the projection axis. Further, all points p' projected on the principal axes in one bin are used to map the corresponding points p in the original point cloud to obtain points in one cluster. Hence, for m one-dimensional bins, we obtain m clusters in the original point cloud. After identifying all the clusters, a new vertex set V_a is generated which consists of the centroid of each cluster.

To define the set of edges E on vertex set v_a, we follow the edge generation strategy of [5]. For this, we search for such pairs of points in v_a where the distance between the pair is at maximum to τ. Once subsets v_a and E are generated, we normalize each of them using the graph normalization mechanism [5] to make the number of graph vertices consistent across different point clouds in the datasets. This generates a normalized SSC (nSSC) which consists of a normalized version of subset V_a and E. Finally, we select the neighborhood vertices of the vertex set of nSSC using the k-nearest neighbor within a ball of radius $r = 0.5$ to create the final nSSC of the point cloud.

Alpha Complex. SSC captures the core structure of the object, however, it might not present distinguishable graphs for the objects from structurally analogous classes. Therefore, we construct an alpha complex of each point cloud to analyze the high curvature areas in the point cloud. To generate the set of edges and triangles for alpha complex representation of the point cloud P, we follow the algorithm used in [23]. In this, first, we construct a Delaunay complex $Del(P)$ of P. Then the circumscribing spheres of radius α centered at each vertex in $Del(P)$ are generated. All the simplices that span outside of the union of all the spheres are removed such that the lower-dimensional simplices of each simplex must be present in the new set of simplices which constitutes an alpha complex of P. The real-valued parameter α helps control the level of detail captured in the 3D shape. Thus, incorporating alpha complexes alongside nSSC can enhance the robustness and discriminative power of the shape analysis algorithms in distinguishing structurally similar object classes.

4.2. Dual-stream Feature Extraction and Aggregation

We extract global features of the point cloud in dual streams where each one processes a separate well-defined simplicial

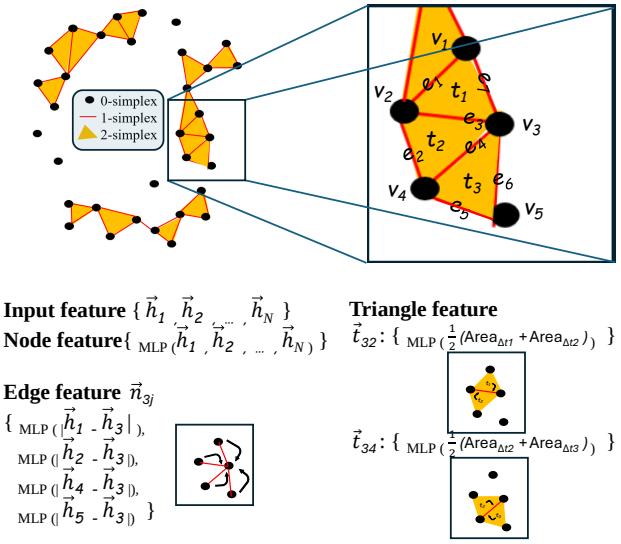

Figure 6: Example of simplicial feature extraction from an alpha complex consisting of node features, edge features for all the edges connected to node v_3, and triangle features for node pairs (v_3, v_2) and (v_3, v_4).

representation of the point cloud. The first global module uses a graph attention network to extract features from the alpha complex of the point data. We use the transform net [2] to include transformation invariance in the graph structure which is followed by edge convolution to extract global features of both graph representations individually. In each stream, feature maps are convolved using shared MLPs with output feature sizes sequentially doubled after each of the layers.

We use an additional attention layer in the alpha-complex module to obtain the most relevant neighboring features to capture local and global dependencies more effectively. To leverage the higher dimensional simplicial structure of alpha complex, we utilize the 0-simplices for node features, 1-simplices for edge features, and 2-simplices for triangle features. Thus, the input features consisting of (x, y, z) coordinates contribute to node features, the distance between the neighbors contributes to edge features, and areas of neighboring triangles contribute to triangle features. These features are further used to obtain self-attention, neighbor attention, and triangle attention.

A graph attentional layer attains adequate expressive power to transform the input features into higher-level features. As defined in [17], the input feature $h = \{\overrightarrow{h_1}, \overrightarrow{h_2}, ..., \overrightarrow{h_N}\}, \overrightarrow{h_i} \in R^F$ contains F features for each of N nodes which obtains output feature $h' = \{\overrightarrow{h'_1}, \overrightarrow{h'_2}, ..., \overrightarrow{h'_N}\}, \overrightarrow{h'_i} \in R^{F'}$ of cardinality F'. A shared linear transformation is applied to each node which is parametrized by weight matrices $W \in R^{F' \times F}$. To indicate the significance of neighboring node j's features to node i, the overall attention coefficient e_{ij} are defined using shared attention mechanisms $s : R^{F' \times F} \rightarrow R$ on the simplicial features $\overrightarrow{x_i}$, $\overrightarrow{n_{ij}}$, and $\overrightarrow{t_{ij}}$ of alpha complex respectively, as shown in Equation 4.

$$e_{ij} = s(\overrightarrow{x_i}, \overrightarrow{n_{ij}}, \overrightarrow{t_{ij}}), \quad \text{where,} \quad s : R^{F' \times F} \rightarrow R \tag{4}$$

As shown in Figure 6, simplicial features of the alpha complex are the resulting features of an MLP operating on node features x_i, edge features n_{ij}, and triangle features t_{ij}, individually. Node feature x_i is the input feature of node i. Similarly, the edge feature n_{ij} is the Manhattan distance between node pair i.e. $|\overrightarrow{h_j} - \overrightarrow{h_i}|$. Furthermore, the triangle feature t_{ij} is evaluated as the average of areas of all the triangles having nodes i and j as its two vertices. The edge features and triangle features learn the spatial relationship of a pair of nodes having a common edge and triangle and hence assign more weight to similar neighbors through the shared attention mechanism. Finally, to capture the intricate features from input data, the LeakyReLU nonlinearity is applied to the attention coefficient which is further normalized using the SoftMax function to make it comparable across different neighborhoods (Equation 5).

$$a_{ij} = \text{softmax}_j(\text{LeakyReLU}(e_{ij})) \tag{5}$$

Finally, with $*$ represents the element-wise multiplication, the output feature for i^{th} node can be formulated as Equation 6.

$$h'_i = \sum_{j \in N_i} a_{ij} * W\overrightarrow{h_j} \tag{6}$$

Once features are extracted in each stream, both feature maps are fused (Section 4.3) to ensure that the SSC and alpha-complex representations complement each other.

4.3. Feature Fusion

To fuse the features, we use the *linear addition fusion (LAF)* strategy. As shown in Figure 7, in LAF, we multiply each feature map with a scalar and then add them to get the final feature map. These scalar values are based on the contribution factor of each graph which is evaluated by the number of nodes of the corresponding graph concerning the total number of nodes in both the feature maps. This fusion strategy ensures the contribution of each feature map considering the size of each graph.

As an alternative to *LAF*, we evaluated our model's performance on two other techniques which are *concat* and *mAFF*. In *concat* fusion, both feature maps are simply concatenated and fed to the backbone layer. This fusion technique facilitates the backbone network to treat each feature map equally irrespective of the size of their original graph structures. Furthermore, in the third fusion technique *mAFF*, we proposed a modified version of the attentional feature fusion method [24]. As shown in Figure 7, *mAFF* uses a combination of two fusion methods MS-CAM and AFF defined in [24] such that it

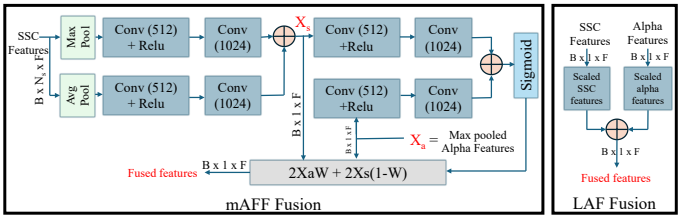

Figure 7: mAFF and LAF modules. In mAFF (left), SSC feature maps and alpha feature maps are fused using a modified version of attention-based feature fusion consisting of MS-CAM and AFF modules adapted from [24]. In LAF (right), both feature maps are scaled and then added together.

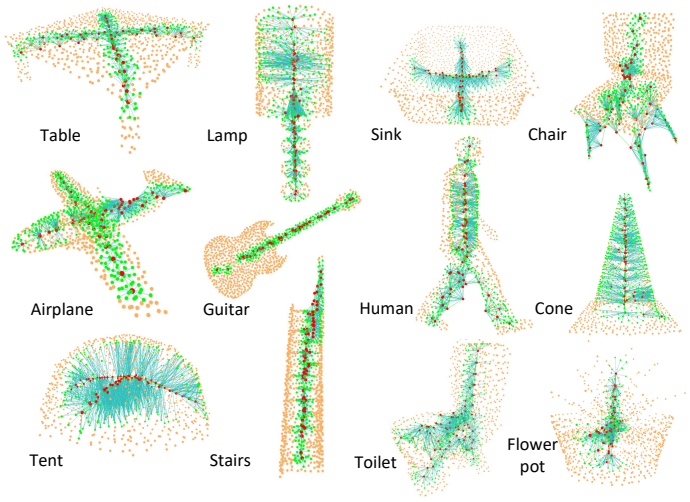

Figure 8: Spiked Skeleton Complexes of different shapes

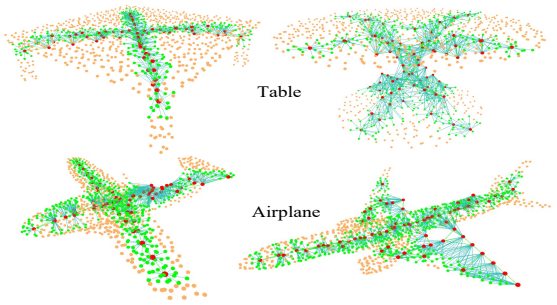

Figure 9: nSSC of different instances of object Table and airplane

considers the size inequality in the SSC and alpha complex representation. Therefore, in *mAFF*, we first use MS-CAM fusion (without applying sigmoid) on SSC features whose output is further fused with alpha complex features using AFF fusion. The key idea is to apply channel attention in multiple scales by varying the pooling size where point-wise convolution acts as the local channel context aggregator. The additional MS-CAM fusion on SSC features ensures the additional attention on the SSC feature map which is smaller in size comparatively to the alpha feature maps.

Finally, for generating the classification score for each point cloud we use the classification backbone network consisting of fully connected layers and dropout layers as shown in Figure 5.

4.4. Implementation

To address the multi-class point cloud classification we have used softmax cross-entropy (SCE) loss [10] where softmax ensures the sum of prediction scores across all classes to be 1 and cross-entropy helps in minimizing the difference between predicted and actual class distributions. These properties of SCE make it a common choice for multi-class classification tasks. For all the computations and training of the model, we use a Tesla V100 16GB GPU and 2 CPU cores machine.

5. Evaluation

5.1. Software and Datasets

The proposed architecture is developed in Python 3.7 and uses Tensorflow 1.13 for artificial neural network layer generation and training. Open3D library is used for point cloud processing and Sci-kit Learn and Numpy libraries are used for geometry processing and mathematical computations. For ModelNet40 data grouping experiment discussed in Section 5.5, we have used Python version GUDHI library [25] to construct the simplicial complex of the point cloud.

We train and evaluate our classification model on the ModelNet40 [26], ModelNet10 [26], and ScanObjectNN [27] datasets. ModelNet40 contains 12,311 meshed CAD models from 40 categories with 9843 training and 2468 testing models. ModelNet10 is a 10-class subset of ModelNet40 that contains 3991 training and 908 testing models. ScanObjectNN consists of 2902 real-world objects, categorized into 15 categories with 2309 training models and 581 testing models.

5.2. Performance Metrics

For quantitative analysis of the proposed classification predictive model, we use overall accuracy (OA) and mean class accuracy (mAcc). Both metrics measure the ability of the model to predict the correct classes of different objects in the dataset. The OA shows the percentage of correct class predictions against the total class predictions and mAcc shows the average percentage of correct class predictions per class.

5.3. Qualitative Analysis

For qualitative analysis, we present Figure 8 which shows the nSSC of different 3D objects. In this figure, the skeleton of most of the shapes (table, chair, airplane, etc.) has been perfectly captured. It also works for the concave geometry objects such as sink and toilet that have an interior cavity but lack a closed surface at the top. Figure 9 shows that this method works for different instances of the same class despite having variations in the object shape. For example, both tables have significant structural differences in terms of the number of legs and the shape of the tabletop. However, in the case of objects that have complex shapes (containing multiple disconnected ends) such as flower_pot, the nSSC does not span the whole shape and hence it has failed in capturing the 3D shape.

To study the feature extraction process of different layers for different objects, we have selected four objects having different topological and geometrical structures, i.e., chair and table representing rectilinear objects having zero betti number β_2 topology, and airplane and car, both are freeform objects having non-zero betti number β_2 topology. For these objects, Figure 10 shows the structure of feature spaces generated after the T-net layer, attention layer, and final convolutional layer of the alpha complex module along with nSSC skeletal nodes on the point cloud, feature spaces after the T-net layer, and final convolutional layer of SSC module of our architecture. These are visualized as the distance between the red point to the rest

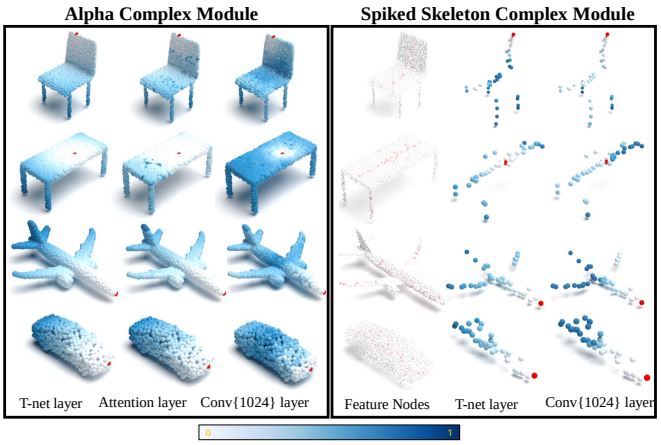

Figure 10: Feature spaces generated after different layers in the alpha complex module and spiked skeleton graph module of our classification architecture for chair, table, airplane, and car objects. The light blue color of the point shows a smaller distance from the red point.

of the points. These feature spaces describe the global transformation of the shape as it passes through the corresponding layer in each module of the classification network. As deeper layers have smaller distances in consecutive feature space representations of each object, this suggests that feature spaces become more discriminative and hence allow the classification network to better distinguish between different classes for different topology and geometry objects.

Table 1: Classification Results on ModelNet10 [26] and ModelNet40 [26] using 1024 points. OA and mACC in %. Results for PointNet++ [3] and DGCNN [10] on ModelNet10 dataset by training and testing using the standard experimental setup described in the original paper.

Methods	M10 (C=10)		M40 (C=40)	
	OA	mAcc	OA	mAcc
PointNet [2]	91.53	91.74	89.20	86.20
PointNet++ [3]	92.73	-	89.82	-
PointGCN [28]	91.91	91.57	89.51	86.05
DGCNN [10]	93.86	93.96	92.9	90.2
CurveNet [29]	96.3	-	**94.2**	-
Grid-GCN [11]	**97.5**	**97.4**	93.10	**91.3**
DNRG [5]	-	-	89.91	87.12
Our+*concat*	93.30	92.73	90.06	85.26
Our+*mAFF*	93.53	92.98	89.37	86.01
Our+*LAF*	94.42	94.22	90.38	85.49

5.4. Quantitative Performance

ModelNet40. We present the performance of the proposed method in terms of OA and mAcc in Table 1 on ModelNet10 and ModelNet40 datasets [26] along with the state-of-the-art results for an alpha value of 0.3 and the number of vertices in nSSC is 140. For LAF, the scaling factors are 0.12 and 0.88 for SSC features and alpha complex features respectively. For comparative analysis, we choose PointNet [2] and PointNet++ [3] as the standard reference to compare the performance of our method with the baseline architectures. Additionally, for comparison against deep learning models operating on the graph structure of the point cloud, we select DGCNN [10] and PointGCN [28]. At last, we consider the reeb graph-based method DNRG [5] to analogize with the topology-based classification model. Therefore, these methods cover a spectrum from basic point cloud processing to graph-based deep learning and topology-based point cloud classification model.

Our model performs fairly well on the ModelNet10 dataset and achieves OA of 94.42% and mAcc of 94.22%. On the ModelNet40 dataset, with OA of 90.38%, the proposed method performs better than pointNet [2], PointGCN [28] and reeb graph-based classification model DNRG [5]. A potential reason for the low accuracy of the network on the ModelNet40 dataset could be a poorly constructed nSSC for a few classes such as flower_pot having 30% class accuracy and cup having 43%, among the other few classes. Additionally, the point cloud instances of these classes differ drastically which results in lower class accuracies for these classes.

Table 2: Classification accuracies (in%) on ScanObjectNN [27] dataset.

Method	OA	mAcc
PointNet [2]	68.2	63.4
PointNet++ [3]	77.9	**75.4**
DGCNN [10]	**78.1**	73.6
Our+*concat*	71.53	65.23
Our+*mAFF*	71.70	64.73
Our+*LAF*	68.58	59.95

ScanObjectNN. The performance of the proposed method on the ScanObjectNN [27] dataset consisting of real-word object scans is shown in Table 2. Our model achieves 71.70% of overall accuracy with mAcc of 64.73% with the mAFF fusion module. A potential reason for the low accuracy of the network on this challenging dataset could be the outliers and noise in the real data scans which affects the clustering process during nSSC construction. Our method relies on the identification of a cluster of the points projected on the principal axes, however, such noise and outliers can introduce incorrect interpretations or connections between points that can potentially degrade the performance of PCA which is used to determine the principal axes of the point cloud.

5.5. Grouping of ModelNet40 Data

Zero Vs. non-zero β_2. For a simplicial complex, zeroth Betti number β_0 is the number of connected components, first betti number β_1 is the number of one-dimensional holes, and second betti number β_2 is the number of two-dimensional voids/cavities in a 3D shape. Therefore, the betti number provides an important insight into the topology of the 3D shape and can be used to differentiate between the objects based on their topology.

In this experiment, we split the 40-class ModelNet40 dataset into 2 smaller datasets based on the value of β_2 of the alpha complex of the point cloud. The first dataset, *positive-β_2 objects* consists of 18 classes having 4205 training point clouds and 1278 testing point clouds that have one or more voids inside the object surface. The second dataset, *zero-β_2 objects* contains the remaining 22 classes consisting of 5638 training point clouds and 1190 testing point clouds that do not have a void in their structures. Figure 11 shows a representative model from each class in zero-β_2 object dataset and positive-β_2 object dataset.

Table 3: Classification results on subsets of ModelNet40 [26] dataset based on different β_2 value of 3D shape. C denotes the number of classes in each dataset. OA and mACC in %.

Methods	Zero-β_2 objects (C=22)			Positive-β_2 objects (C=18)		
	#Params	OA	mAcc	#Params	OA	mAcc
PointNet [2]	813971	95.61	91.98	814999	90.54	84.74
DGCNN [10]	1831456	**96.70**	**95.26**	1830428	**92.38**	86.31
Our+*concat*	4749298	95.08	91.22	5273072	91.11	**86.52**
Our+*mAFF*	8961520	88.98	85.64	8961520	89.66	84.82
Our+*LAF*	4749298	95.10	91.48	4753924	90.87	85.57

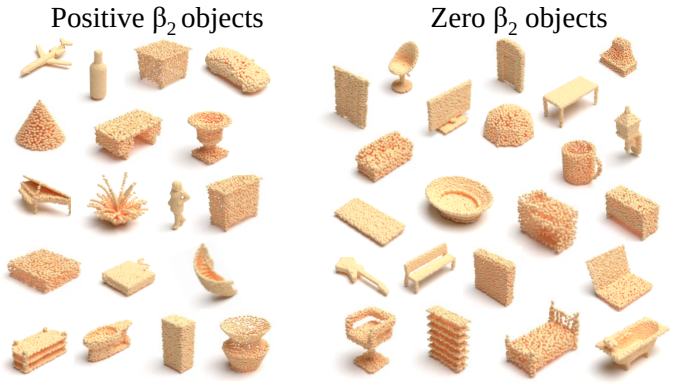

Figure 11: Visualization of the object of each class in the positive-β_2 object dataset and zero-β_2 object dataset.

The grouping of the Modelnet40 dataset into these two sub-datasets is based on the value of β_2 of an alpha complex of each class's object. We used the GUDHI library [25] to construct the alpha complex of the point cloud and evaluated β_2 of the complex within a specific range of α values that is based on the maximum and average distance of all the nearest neighbor pairs of the point cloud. This strategy ensures uniformity in the α value across the variably dense point clouds in the dataset. For *zero-β_2 objects*, β_2 of the alpha complex is zero, while *positive-β_2 objects* have a non-zero value of β_2 of the alpha complex.

We evaluated the classification score on each category of both datasets to understand the performance of our proposed method for objects having different values of second betti number β_2. As shown in Table 3, we found that with the concatenation fusion, our model achieves OA of 91.11% and mAcc of 86.52% for *positive-β_2* dataset. Whereas, our model achieves almost similar accuracies for concatenation and LAF fusion with the highest OA of 95.10% and 91.48% mAcc with LAF on *zero-β_2* dataset. These values show that our model captures the intrinsic features of the objects having the same topological variations. However, the model performs better for the *zero-β_2* dataset than the *positive-β_2* dataset. One possible reason for this could be the presence of more difficult classes such as flower_pot, night_stand, wardrobe, etc. in the *positive-β_2* dataset that have shown bad performance in case of training on the whole ModelNet40 dataset (Figure 12). Additionally, the *positive-β_2* dataset has less number of training samples than the *zero-β_2* dataset, which could also affect the overall learning on the objects with one or more voids.

Rectilinear Vs. Freeform. To observe the effect of the proposed method on the different sets of objects based on the planarity in their overall shape, we divided the 40-class ModelNet40 dataset into two separate datasets. This data segregation is done by inspecting the angles between the planes fitted on the point clouds of the object shapes. At first, we use Random Sample Consensus (RANSAC) [30] to detect planes in the point cloud which includes fitting planes iteratively to find the best fitting planes for each point cloud. Next, the angles between the normal vectors of each pair of detected planes are evaluated. Finally, a point cloud is declared as rectilinear if the angle between the adjacent planes is approximately 90^0 or a multiple of 90^0. To account for small deviations in planes due to noise or irregularities, A margin of 20^0 is used. Point clouds having any angle not following the criteria are declared as freeform. Object classes having more rectilinear objects than freeform objects are segregated as a rectilinear dataset and the rest of the object classes form a freeform dataset.

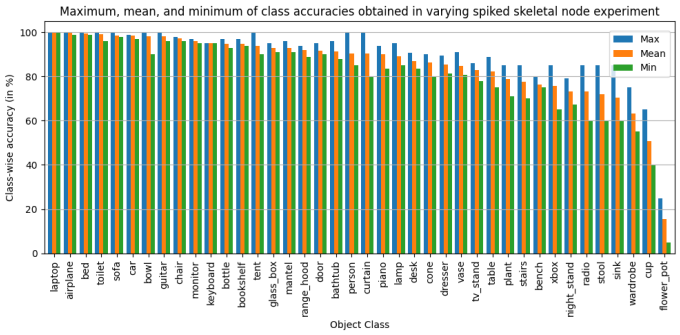

Figure 12: Statistical analysis of class-wise accuracies of Modelnet40 dataset for varying spiked skeletal node experiment.

Each of these datasets consists of 20 classes. The dataset of rectilinear shape objects consists of 5404 training point clouds and 1346 testing point clouds with 20 object classes and the dataset of freeform shape objects consists of 4436 training point clouds and 1122 testing point clouds with 20 object classes. As shown in Table 4, the classification results show that our model performs almost similar to DGCNN and with LAF fusion performs better in the rectilinear dataset and achieves OA of 93.11%. For the freeform dataset, our model with mAFF outperforms DGCNN and achieves an OA of 92.77%. The reason behind the better results of the rectilinear dataset than that of the freeform dataset is that in rectilinear objects, the major and minor axes of the surface are usually orthogonal to each other and hence it better aligns with PCA eigenvectors. However, this segregation is based on the higher number of objects of a class selected in either rectilinear or freeform category. Therefore there are few classes which has both types of object shapes

Table 4: Classification results on subsets of ModelNet40 [26] dataset based on rectilinear or freeform shape objects. C denotes the number of classes in each dataset. OA and mACC in %.

Methods	Rectilinear objects (C=20)			Freeform objects (C=20)		
	#Params	OA	mAcc	#Params	OA	mAcc
PointNet [2]	814485	91.07	87.32	814485	87.41	85.59
DGCNN [10]	1830942	**94.2**	**90.62**	1830942	92.32	89.85
Our+*concat*	5273072	92.04	86.41	5273072	91.79	89.63
Our+*mAFF*	8961520	92.67	86.85	8961520	**92.77**	89.89
Our+*LAF*	4748784	93.11	90.04	4748784	92.59	**90.59**

with close margins and hence the two datasets are subject to potential inaccuracies which results in lower accuracies.

Table 5: Effect of number of nSSC node on Classification of ModelNet40 [26] dataset. N_s denotes the number of nSSC node. All accuracies (OA_Tr , OA_Ts, and mAcc_Ts (denoted here as mA_Ts)) in %. Training time (Tr_tm) and Testing time (Ts_tm) in hours and memory consumed during training Tr_m and testing Ts_m is in GBs.

N_s	OA_Tr	Tr_tm	Tr_m	OA_Ts	mA_Ts	Ts_tm	Ts_m
40	89.57	11:25	13.91	89.53	86.31	1:29	2.77
50	89.57	11:46	14.03	89.57	85.35	1:30	2.80
60	89.53	11:19	13.98	89.45	85.45	1:31	2.83
70	89.12	11:42	14.30	89.08	84.50	1:35	2.89
80	89.49	12:07	14.44	89.41	85.33	1:40	3.69
90	89.65	12:51	14.41	89.73	85.54	1:52	2.94
100	90.10	13:27	14.92	90.06	86.57	2:05	3.75
110	89.57	13:58	15.71	89.57	85.12	2:08	3.31
120	89.61	14:49	16.46	89.57	85.06	2:14	3.53
130	89.85	15:43	15.33	89.81	85.87	2:23	3.78
140	**90.34**	17:09	16.04	**90.38**	85.49	2:39	4.03
150	89.45	19:01	16.88	89.45	85.47	2:52	4.21

Effects of Number of Spiked Skeletal Nodes. Table 5 shows the effect of the number of skeletal nodes on the classification scores. We conduct these experiments on the ModelNet40 dataset for an alpha value of 0.3 on our model with LAF fusion. The results show the OA during training, OA, and mAcc during testing accuracy along with the time and memory used during training and testing of the model. We observed that our model performs almost similarly for the different sizes of the constructed nSSC. This shows that the proposed method supports the idea of a concise representation of a point cloud using nSSC and the increase in the number of skeletal nodes does not affect the model's performance for classification very significantly. Figure 12 shows the statistics of the classwise accuracies of all 40 classes for this experiment. The statistic includes the maximum, average, and minimum class accuracies for each class when varying the nSSC node count from 40 to 140. It can be seen that 70% classes in the ModelNet40 dataset have achieved more than 80% of average accuracies. However, twelve classes including flower pot, cup, and wardrobe, achieved low class accuracies consistently throughout all the other experiments.

Table 6: Effect of value of alpha in alpha complex on Classification of ModelNet40 [26] dataset. All accuracies (OA_Tr , OA_Ts, and mAcc_Ts) in %. Training time (Tr_timme) and Testing time (Ts_tim) in hours.

Alpha	OA_Tr	Tr_tm	OA_Ts	mAcc_Ts	Ts_tm
0.1	89.69	24:16	89.61	85.40	3:59
0.2	89.29	18:45	89.37	85.56	3:04
0.3	**90.35**	17:09	**90.38**	85.49	2:39
0.4	89.65	15:49	89.65	85.49	2:30
0.5	89.73	15:29	89.73	85.83	2:26

Effects of Alpha value. Table 6 shows the effect of the value of alpha in the alpha complex on the classification scores. We conduct these experiments on the ModelNet40 dataset for 140 skeletal nodes (nSSC) in the SSC of each point cloud on our model with LAF fusion. The results show the OA during training, OA and mAcc during testing accuracy along with training and testing time. We observed that the performance of our model is almost similar for different alpha values for the alpha complex generation. This shows that the proposed method does not degrade the model performance for the low values of alpha and hence it is suitable for the sparse geometric complex representation. However for smaller alpha values, geometric complex construction taker longer duration which results in large computation time.

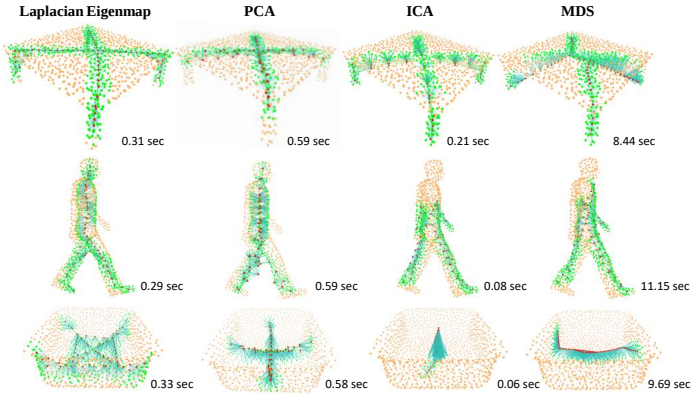

Figure 13: nSSC of a point cloud of Table, Person, and Sink classes for different mapping functions along with graph construction time.

5.6. Ablation study

Role of Mapping Functions. To understand the effects of different functions that map point cloud projected on principal axes P' to SSC S on the model's performance, we conducted an experiment. We selected four functions namely Laplacian eigenmaps from [31], Principal Component Analysis (PCA), Independent Component Analysis (ICA), and Multi-dimensional Scaling (MDS). Figure 13 shows the nSSCs constructed using these functions. We observe that PCA performs better than the other functions and is followed by the Laplacian eigenmaps function for most of the classes in the ModelNet40 dataset while ICA and MDS lack in capturing the skeletons of most of the objects in the dataset. PCA produces a better skeleton in moderate time in comparison to the other three functions.

Table 7: Effect of each stream on classification accuracy on ModelNet40 [26] dataset. OA and mACC in %.

Stream	#Params	OA	mAcc
No complex stream	2701938	85.43	82.77
Only SSC stream	2702130	87.86	83.81
Only alpha complex stream	2719691	89.45	85.19
SSC+ alpha complex	4753876	**90.38**	**85.49**

Single stream Vs. Dual stream. As shown in Table 7, the classification results of our model with LAF fusion with only SSC stream achieves OA of 86.16% while the dual-stream model reaches to OA of 90.38%. This verifies that alpha complex representation complements the SSC representation

for point cloud classification. For "No complex stream", we have used point cloud as input to the set of 5 layers arranged in the SSC stream directly and trained our model to evaluate the classification accuracy.

Table 8: Effect of higher dimensional simplicial features of the alpha complex on classification accuracy on ModelNet40 [26] dataset. OA and mACC in %.

Simplicial features	#Params	OA	mAcc
Only Node	4753630	88.68	84.49
Node + Edge	4753736	89.12	85.28
Node + Edge + Triangle	4753876	**90.38**	**85.49**

Role of Simplicial Features. Table 8 presents the classification results of our model with LAF fusion with both SSC and alpha stream in case of using different simplicial features in the alpha complex stream. Our model with all three simplicial features i.e. node feature, edge feature, triangle feature surpasses the overall accuracy of only node feature by 1.7% and node feature along with edge feature by 1.24%. This verifies that utilizing higher dimensional simplicial information of alpha complex representation benefits the classification of point clouds.

Table 9: Effect of network depth on Classification of ModelNet40 [26] dataset. L signifies the number of layers in each stream. All accuracies (OA_Tr, OA_Ts, and mAcc_Ts) in %. Training time (Tr_tm) and Testing time (Ts_tm) in hours.

L	OA_Tr	Tr_tm	OA_Ts	mAcc_Ts	Ts_tm	#Params
1	89.32	17:54	89.31	85.63	2:42	2429780
2	89.33	16:13	89.33	85.38	2:40	2569428
3	89.44	18:11	89.41	84.29	3:04	2848724
4	89.98	17:15	89.97	85.71	2:40	4614228
5	90.34	17:09	90.38	85.49	2:39	4753876
6	89.89	18:21	89.87	85.71	2:38	6329812
7	89.82	19:50	89.81	85.12	2:38	6986708
8	89.25	19:00	89.2	84.91	2:45	7282388

Network Depth. We selected the model depth through empirical experimentation. Table 9 presents the classification results of our model with LAF fusion for varying network depth in each stream. Our model with 5 layers in each stream presents a better choice considering a tradeoff of classification accuracy, training time, and number of model parameters.

Fusion Modules. We evaluated the performance of three distinct fusion techniques *concat*, *mAFF*, and *LAF* in our model on different datasets in Tables 1, 3, 4, and 2. We observed that all three techniques present comparable results. However, the LAF method performed better than the other two fusion techniques in most of the cases in Table 1, Table 3, and Table 4. Larger graphs typically allow for a denser representation of relationships among the data, and thus the increased connectivity leads to richer feature representations. On the other hand, nSSC consists of fewer nodes in comparison to the alpha complex. Therefore, we believe that the improved performance of LAF method comes from the fact that it is based on the contribution of each representation of the point cloud which is measured by the number of nodes in the individual representation.

In contrast to the linear fusion techniques such as concatenation or LAF which uses fixed weights to feature maps for fusion, attention-based fusion, such as the mAFF fusion technique assigns a learnable weight to each feature map for the fusion to address the feature inconsistency across different scales and hence facilitates the network to detect objects under extreme scale variation [24]. We observe that mAFF obtains classification accuracies comparable to the LAF and has better accuracy than other fusions on real-world object (Table 2).

Figure 14 (i) shows the model performance for shape classification of noisy point cloud data. For this experiment, the original point clouds in ModelNet40 dataset are introduced with the Gaussian noise with zero mean and standard deviation varying from 0 to 0.5. It can be seen that both linear fusion techniques show almost linear degradation in the classification accuracy with 3% and 4% respectively in overall classification accuracy when increasing the noise standard deviation from 0 to 0.5. However model with mAFF fusion presents an overall downward trend with a slight upward correction. This shows that attention-based fusion is better at handling the noise in the data in comparison to linear fusion techniques. In terms of model parameters, linear fusion model uses approximately half the parameters of the mAFF technique. This makes LAF a better choice for our model in terms of computational efficiency and design complexity.

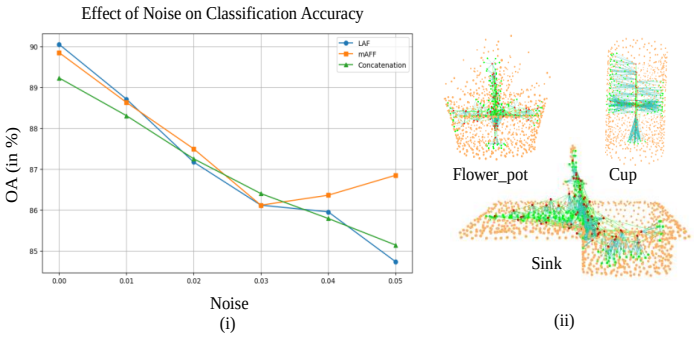

Figure 14: (i) Classification accuracy of the proposed model with three different fusion techniques for Gaussian noise with zero mean and standard deviation of 0 to 0.5. (ii) Failure cases of SSC of objects from flower_pot, sink, and cup class.

Limitations. In most cases, our model with LAF performs better than the other two fusion schemes. However, mAFF yields better results for noisy point clouds. Additionally, LAF loses its effectiveness when the node sizes of the SSC and alpha complex are equal or comparable. Therefore, a careful selection of the fusion scheme is essential based on the characteristics of the point cloud and its simplicial complex representations.

Furthermore, the class distribution in ModelNet40 is imbalanced and have high shape similarity in few classes such as flower_pot-plant class pair and cup-vase class pair. These are also a possible reason in the low classification accuracy of few classes. Additionally, as shown in Figure 14(ii), SSC sometimes fails to accurately represent the skeleton of the object. For example, SSC of flower_pot fails to capture the top portion of the shape having leaf-like shape which could be beneficial for this shape to differentiate it from other shapes. This results in lower classification accuracy for these classes.

6. Conclusion and Future Work

In this work, we propose a novel simplicial complex SSC that is a compact yet informative representation of a point cloud. Different from the existing works, we propose to use a concise representation of a point cloud for object classification that captures the core structure of the 3D shape. Additionally, the proposed simplicial complex constitutes spatial dependencies between points within the vicinity of the skeletal structure which enhances the discriminative power and robustness of the classification model in capturing the intricate relationships within 3D shape. Our dual-stream graph attention network utilizes SSC for non-local feature aggregation which is complemented by the local features aggregated using alpha complex. With this complementary feature fusion, our model leverages both global context and local details for classification. To verify the effectiveness of the proposed method, we grouped ModelNet40 dataset into two topologically different object categories which provides a structured framework for studying the topology in object representation and classification. Experiments show a significant improvement in classification accuracy when the proposed model was trained on topologically separated 3D shapes and hence, verify the effectiveness of our model in capturing the underlying shape of similar topology objects. We hope that the topological grouping of the ModelNet40 dataset constitutes an alternate evaluation paradigm for future models to foster a deeper understanding of the role of topology in object representation. Experimentation on rectilinear and freeform object classes presents the efficacy of our PCA-based model for rectilinear objects. Furthermore, in future research, to gain a comprehensive understanding of the underlying topology in a 3D shape, feature aggregation over higher-dimensional simplices such as tetrahedra presents a promising avenue for refinement.

References

[1] Guo, Y, Wang, H, Hu, Q, Liu, H, Liu, L, Bennamoun, M. Deep learning for 3d point clouds: A survey. IEEE transactions on pattern analysis and machine intelligence 2020;43(12):4338–4364.

[2] Qi, CR, Su, H, Mo, K, Guibas, LJ. Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 652–660.

[3] Qi, CR, Yi, L, Su, H, Guibas, LJ. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems 2017;30.

[4] Simonovsky, M, Komodakis, N. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 3693–3702.

[5] Wang, W, You, Y, Liu, W, Lu, C. Point cloud classification with deep normalized reeb graph convolution. Image and Vision Computing 2021;106:104092.

[6] Dey, TK, Wang, Y. Computational topology for data analysis. Cambridge University Press; 2022.

[7] Saha, PK, Borgefors, G, di Baja, GS. Skeletonization and its applications–a review. Skeletonization 2017;:3–42.

[8] de Surrel, T, Hensel, F, Carrière, M, Lacombe, T, Ike, Y, Kurihara, H, et al. Ripsnet: a general architecture for fast and robust estimation of the persistent homology of point clouds. In: Topological, Algebraic and Geometric Learning Workshops 2022. PMLR; 2022, p. 96–106.

[9] Zhang, H, Wang, C, Tian, S, Lu, B, Zhang, L, Ning, X, et al. Deep learning-based 3d point cloud classification: A systematic survey and outlook. Displays 2023;:102456.

[10] Wang, Y, Sun, Y, Liu, Z, Sarma, SE, Bronstein, MM, Solomon, JM. Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (tog) 2019;38(5):1–12.

[11] Xu, Q, Sun, X, Wu, CY, Wang, P, Neumann, U. Grid-gcn for fast and scalable point cloud learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, p. 5661–5670.

[12] Te, G, Hu, W, Zheng, A, Guo, Z. Rgcnn: Regularized graph cnn for point cloud segmentation. In: Proceedings of the 26th ACM international conference on Multimedia. 2018, p. 746–754.

[13] Li, R, Wang, S, Zhu, F, Huang, J. Adaptive graph convolutional neural networks. In: Proceedings of the AAAI conference on artificial intelligence; vol. 32. 2018,.

[14] Feng, Y, You, H, Zhang, Z, Ji, R, Gao, Y. Hypergraph neural networks. In: Proceedings of the AAAI conference on artificial intelligence; vol. 33. 2019, p. 3558–3565.

[15] Wang, C, Samari, B, Siddiqi, K. Local spectral graph convolution for point set feature learning. In: Proceedings of the European conference on computer vision (ECCV). 2018, p. 52–66.

[16] Vaswani, A, Shazeer, N, Parmar, N, Uszkoreit, J, Jones, L, Gomez, AN, et al. Attention is all you need. Advances in neural information processing systems 2017;30.

[17] Veličković, P, Cucurull, G, Casanova, A, Romero, A, Lio, P, Bengio, Y. Graph attention networks. arXiv preprint arXiv:171010903 2017;.

[18] Kaul, C, Pears, N, Manandhar, S. Fatnet: A feature-attentive network for 3d point cloud processing. In: 2020 25th International conference on pattern recognition (ICPR). IEEE; 2021, p. 7211–7218.

[19] Zhao, H, Jiang, L, Jia, J, Torr, PH, Koltun, V. Point transformer. In: Proceedings of the IEEE/CVF international conference on computer vision. 2021, p. 16259–16268.

[20] Yu, X, Tang, L, Rao, Y, Huang, T, Zhou, J, Lu, J. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 19313–19322.

[21] Tagliasacchi, A, Delame, T, Spagnuolo, M, Amenta, N, Telea, A. 3d skeletons: A state-of-the-art report. In: Computer Graphics Forum; vol. 35. Wiley Online Library; 2016, p. 573–597.

[22] Nurunnabi, A, Teferle, F, Li, J, Lindenbergh, R, Hunegnaw, A. An efficient deep learning approach for ground point filtering in aerial laser scanning point clouds. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 2021;43:31–38.

[23] Edelsbrunner, H, Mücke, EP. Three-dimensional alpha shapes. ACM Transactions On Graphics (TOG) 1994;13(1):43–72.

[24] Dai, Y, Gieseke, F, Oehmcke, S, Wu, Y, Barnard, K. Attentional feature fusion. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. 2021, p. 3560–3569.

[25] Rouvreau, V. Alpha complex. In: GUDHI User and Reference Manual; 3.9.0 ed. GUDHI Editorial Board; 2023,URL: `https://gudhi.inria.fr/doc/3.9.0/group__alpha__complex.html`.

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

[27] Uy, MA, Pham, QH, Hua, BS, Nguyen, T, Yeung, SK. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019, p. 1588–1597.

[28] Zhang, Y, Rabbat, M. A graph-cnn for 3d point cloud classification. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE; 2018, p. 6279–6283.

[29] Xiang, T, Zhang, C, Song, Y, Yu, J, Cai, W. Walk in the cloud: Learning curves for point clouds shape analysis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021, p. 915–924.

[30] Fischler, MA, Bolles, RC. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 1981;24(6):381–395.

[31] Belkin, M, Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural computation 2003;15(6):1373–1396.