sensors

MDPI

Article

Discrete Geodesic Distribution-Based Graph Kernel for 3D Point Clouds

Mehmet Ali Balcı [1], Ömer Akgüller [1,*], Larissa M. Batrancea [2,*] and Lucian Gaban [3]

[1] Department of Mathematics, Faculty of Science, Muğla Sıtkı Koçman University, 48000 Muğla, Turkey
[2] Department of Business, Babeş-Bolyai University, 7 Horea Street, 400174 Cluj-Napoca, Romania
[3] Faculty of Economics, "1 Decembrie 1918" University of Alba Iulia, 510009 Alba Iulia, Romania
* Correspondence: oakguller@mu.edu.tr (Ö.A.); larissa.batrancea@ubbcluj.ro (L.M.B.)

Abstract: In the structural analysis of discrete geometric data, graph kernels have a great track record of performance. Using graph kernel functions provides two significant advantages. First, a graph kernel is capable of preserving the graph's topological structures by describing graph properties in a high-dimensional space. Second, graph kernels allow the application of machine learning methods to vector data that are rapidly evolving into graphs. In this paper, the unique kernel function for similarity determination procedures of point cloud data structures, which are crucial for several applications, is formulated. This function is determined by the proximity of the geodesic route distributions in graphs reflecting the discrete geometry underlying the point cloud. This research demonstrates the efficiency of this unique kernel for similarity measures and the categorization of point clouds.

Keywords: simplicial complex; Wasserstein distance; Kullback–Leibler information; point cloud processing

check for updates

Citation: Balcı, M.A.; Akgüller, Ö.; Batrancea, L.M.; Gaban, L. Discrete Geodesic Distribution-Based Graph Kernel for 3D Point Clouds. *Sensors* **2023**, *23*, 2398. https://doi.org/10.3390/s23052398

Academic Editors: Miaohui Wang, Guanghui Yue, Jian Xiong and Sukun Tian

Received: 12 January 2023
Revised: 16 February 2023
Accepted: 17 February 2023
Published: 21 February 2023

1. Introduction

Point clouds are one of the most direct representations of geometric datasets. One of the sources for obtaining point clouds is 3D shape acquisition devices such as laser range scanners, which also have applications in many disciplines. These scanners provide generally noisy raw data in the form of disorganized point clouds representing surface samples. Given the growing popularity and very wide applications of this data source, it is essential to work directly with this representation without having to go through an intermediate step that can add computational complexity and fitting errors. Another important area where point clouds are frequently used is the representation of high-dimensional manifolds. Such high-dimensional and general isodimensional data are found in nearly all disciplines, from computational biology to image analysis and financial data [1–5]. In this case, due to the high dimensionality, manifold reconstruction is challenging from a technological standpoint, and the relevant calculations have to be performed directly on the raw data, i.e., the point cloud.

Graph structures are important tools for representing discrete geometric data and the discrete manifold underlying the data, as they can reflect the structural and relational arrangements of objects [6–9]. In the classification of discrete graph-based geometric structures, the problem of accurately and effectively calculating the similarity of these data sets arises. Graph kernel functions are widely used to solve this type of problem [10–14]. Graph kernels have proven to be powerful tools for the structural analysis of discrete geometric data. There are two main advantages to using graph kernel functions. First, graph kernels can characterize graph properties in a high-dimensional space and therefore have the capacity to preserve the topological structures of the graph. Second, graph kernels make rapidly evolving machine-learning methods for vector data applicable to graphs.

The goal of this study was to construct a kernel function that generalizes across the topology and geometry of a 3D point cloud. A geodesic is a curve in differential geometry that depicts the shortest route between two points on a surface, or more broadly, the shortest path on a Riemannian manifold. In addition, it is intended to expand the idea of a ”straight line” on any differentiable manifold coupled to a more generic medium. Geodesics on piecewise linear manifolds, the foundation of discrete differential geometry, were first defined by [15] and then extended to polyhedral surfaces by [16]. In discrete differential geometric techniques, the underlying geometry of the point cloud is assumed to be known, and triangulation is used to include the locations between points into the geometry. Thus, noise reduction and subdivision techniques may also be implemented while using a raw 3D point collection. In the graph structure to be constructed from the 3D point cloud, the idea of a geodesic becomes the problem of the shortest path [17,18].

In general, point clouds should intensively sample the border of a smooth surface rebuilt using moving least squares [19–21], implicit [22], or Voronoi/Delaunay [23,24] methods. Since point clouds may describe 3D forms using graphs without the requirement for the explicit storing of the manifold connection, they have become a popular alternative surface representation to polygonal meshes [25–28]. Despite the fact that numerous particular graph types, such as the *k*-nearest neighbor graph [29], the Reeb graph [30], and the Gabriel graph [31], give methods to the geometry and topology of point clouds, higher-order topologies and submanifold topologies disregard topological characteristics. Graph representations of simplicial complex skeletons that preserve the submanifold topologies of point clouds and may potentially integrate higher-order topological information are used in this study. In general, skeleton-based representations provide a compact and expressive form abstraction that aims to imitate human intuition. Using concise, informative, and easily computable skeletal representations as opposed to full models may facilitate the comparison process. In practice, it may be hard to locate a query-like item in a database by comparing point clouds or stacks of hundreds of triangles.

This work presents an approach to the geodesic curves of the manifold by calculating the shortest paths on the graph structures defined by the simplicial complex skeleton of the submanifold from which the point cloud is taken. If the graph structures created by the simplicial complex skeleton are altered by noise, the approaches to the geodesic distributions of the submanifold are not significantly impacted. Consequently, a kernel function to be defined by the Wasserstein similarity of the distributions of discrete geodesics in the skeletons of 3D point clouds has shown to be an excellent assessment tool for point cloud similarity. In a variety of 3D applications, such as 3D object retrieval and inverse procedural modeling, measuring the similarity between 3D geometric objects is crucial. This study aimed to find and demonstrate the effectiveness of a kernel function that calculates the similarity of 3D point clouds while taking into consideration the discrete geometry and topology of the point cloud. The effective kernel function is obtained using the Wasserstein-1 distance by comparing the geodesic distributions in the graphs to those that are isomorphic to the skeleton of the simplicial complexes on the point cloud.

Direct comparison and classification are utilized to compare this newly constructed kernel function to graph models. Using the values of the kernel function, the intra-class and inter-class similarity matrices of the point clouds were constructed during the direct comparison procedure. Considering the geometric and topological aspects of the studied models, it has been noted that the Alpha complex skeletons are the networks on which the kernel function operates most effectively. Using the kernel function, the similarity of a point cloud to itself was stored in the matrices used in the classification procedures. At this, the graph communities acquired for each point are used. Using a convolutional neural network, supervised classification is performed according to the airplane, car, person, plant, and vase classes.

Related Works

As a collection of large points in three dimensions, point clouds may stand in for an object’s spatial distribution and surface properties. Model reconstruction [32,33], terrain monitoring [34,35], and resource monitoring and exploitation [36,37] are just a few examples of the various research and application sectors that rely on 3D point clouds obtained by stationary laser scanning. Accurate and efficient registration is crucial for obtaining a full scene or object from a collection of point clouds obtained via stationary laser scanning [34,38], and this has a knock-on effect on subsequent processing and applications such as segmenting and classifying the cloud as well as detecting and tracking objects within it. Moreover, semantic segmentation using point cloud data has made significant strides in recent years [39–43], thanks in large part to innovations such as point-cloud compression techniques [44,45]. When separating moving objects from LiDAR point clouds, semantic segmentation is essential. Existing semantic segmentation convolutional neural networks are good at predicting the semantic labels of point clouds such as automobiles, buildings, and people. Hence, new computational techniques on point clouds are needed because of their widespread use in many disciplines.

The kernel function of a pair of graphs may be described by decomposing the graphs and comparing the specified pairings of isomorphic substructures. Commonly used substructures are walks, paths, and spanning trees [46–48]. In [49], the authors evaluated the complex motions as decomposed spatio-temporal parts for each video using kernel functions defined with subtrees and created the corresponding binary trees. The resulting kernel function is defined by calculating the number of isomorphic subtree patterns. A kernel family to compare point clouds is proposed by [50]. These kernels are based on a newly developed local tree-walk kernel between subtrees, which is defined by factoring in the properly defined graph models of subtrees. The authors in [51] defined a graph kernel for motion recognition in videos. First, they describe actions in videos using directed acyclic graphs (DAGs). The resulting kernel is defined as an expanding random walking kernel by counting the number of isomorphic walks of the DAGs. [52] proposed a segmentation graph kernel for image classification. In this method, each image is represented by a segmentation graph; each vertex corresponds to a segmented region, and each edge connects a pair of neighboring regions. The resulting kernel function is calculated by counting the imprecise isomorphic subtree patterns between the segmentation graphs. Furthermore, some kernel functions are also effectively used for computer vision applications, such as the shortest path graph kernel [53], non-backward walking kernel [54], Lovas kernel [55], Weisfeiler–Lehman subtree kernel [56].

Although each of the kernel functions mentioned above gives effective results for different problems, they are very sensitive to noise and outliers in empirically obtained 3D point clouds. Furthermore, they do not generate reliable comparison information between isomorphic substructures. In other words, for graphs abstracted from 3D shapes, most of the available kernels cannot determine whether isomorphic substructures are located in the same regions based on the visual background. To overcome these shortcomings, [57] proposed an aligned subtree kernel. The proposed kernel function is calculated by counting the number of isomorphic subtrees rooted in aligned vertices, thus overcoming the shortcoming of neglecting positional or structural correspondences between isomorphic substructures that arise in most graph kernels. Although an aligned subtree kernel is effective on 3D shape classification problems, it cannot guarantee transitivity between aligned vertices. More specifically, given the vertices u, v, and w, if v and u and u and w align, the kernel function cannot guarantee that v and w are also aligned. On the other hand, [58] shows that the cascading alignment step is necessary to guarantee the positive precision of the vertex alignment kernel. Therefore, the aligned subtree kernel cannot be guaranteed as a positive-definite kernel. Furthermore, all the specified kernels reflect only graph properties for each graph pair under comparison and therefore ignore information from other graphs. These disadvantages limit the precision of kernel-based similarity measures.

This paper is organized as follows: In Section 2, we first present the basics of simplicial complexes and graph data obtained from their 1-skeleton. The method of obtaining the graph defined in this way is very important for the kernel function presented, since it will most effectively use the topological and geometric properties of point clouds. Then, the distributions of discrete geodesic curves on these graph structures are determined using Kullback–Leibler information and then a kernel function using the Wassertein-1 distance is introduced. In Section 3, we give basic computational results for this novel kernel function on the Princeton ModelNet-40 benchmark. The computational results measure the similarity of point clouds with each other and show the classification of point clouds. Finally, in Section 4, we present detailed conclusions.

2. Methodology

2.1. Simplicial Complexes

The convex body of the points with $v_0, v_1, \dots, v_d \in \mathbb{R}^n$ being $d+1$-affine independent points is called a d-symplex. The details about simplicial topology can be found in [59]. In this study, a d-simplex is denoted by $\sigma = conv\{v_0, \dots, v_n\} = [v_0, \dots, v_n]$. The convex body is simply a polyhedron with $d+1$-affine independent points as vertices. The face of a σ is defined by $conv\{S\}$ as $S \subset [v_0, \dots, v_n]$. The finite family of simplexes that provide the following properties is called a K complex:

- For $\sigma \in K$ and τ is face of σ, $\tau \in K$;
- When $\sigma, \sigma' \in K$, $\sigma \cap \sigma'$ is empty or is simultaneously a face of σ and σ'.

A collection $\{\sigma \in K \mid dim(\sigma) \leq j\}$ of a simplicial complex K is called the j-skeleton of K and is denoted by $K^{(j)}$. In this definition, $dim(\sigma)$ denotes the dimension of σ. The 1-skeleton of any K complex is $K^{(1)} = \{\sigma \in K \mid dim(\sigma) \leq 1\}$, that is, the set of vertices and edges of the simplex that form the complex. Hence, for $V = \{v_0, \dots, v_n\}$ and $E = \{[v_i, v_k] \mid [v_i, v_k] \in K\} \subseteq V \times V$, there exists a simple graph $G = (V, E)$ with $K^{(1)} \equiv G$. An example of a 2-dimensional K complex and the corresponding $K^{(1)}$ skeleton is given in Figure 1. It is straightforward to see that $K^{(1)} \equiv G = (V, E)$ with $V = \{v_1, \dots, v_{10}\}$.

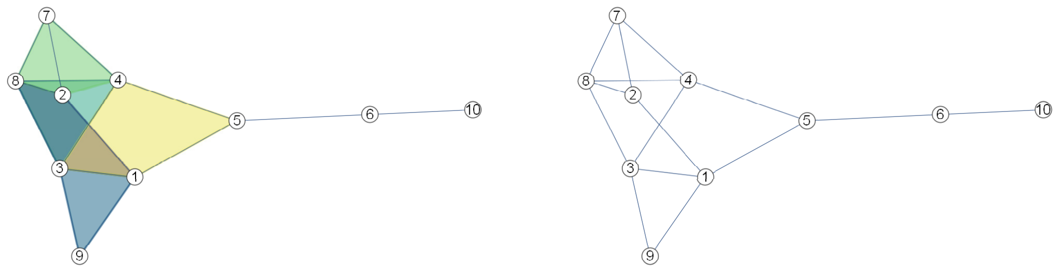

Figure 1. A 2-dimensional complex (**left**) and its 1-skeleton (**right**), where the numbers denote the vertex index.

With a K complex to be formed on a 3D point cloud, it is possible to capture the topological features of the underlying manifold of the point cloud, such as connectivity and holes. Moreover, it is possible to apply various graph algorithms to the simple graph which the skeleton $K^{(1)}$ is isomorphic. Let us consider the shortest graph paths $\ell_1 = v_7, v_8, v_3, v_9$ and $\ell_2 = v_7, v_2, v_1, v_9$ between the vertices v_7 and v_9 on the $K^{(1)}$ skeleton in Figure 1. It is obvious that the ℓ_1 and ℓ_2 are homotopic on $K^{(1)} \equiv G$. Thus, when comparing two skeletons $K_1^{(1)} \equiv G_1$ and $K_2^{(1)} \equiv G_2$ in terms of topological similarity, examining the approximation distributions of discrete geodesics corresponding to the shortest paths yields effective results.

Let us now consider the methods of obtaining various geometric or abstract complexes, which are widely used in practice. In particular, the input is often a series of points from which some hidden field is sampled or approximated. This set of points is called a point cloud. Point clouds have no topology other than a discrete topology, and some connections and some topologies are applied to them. Let a point cloud be $P = \{p_1, \dots, p_n\}$ and the Euclidean ball be $B(p, r)$ with $p_i \in P$ at the center and r is the radius. We introduce the simplicial complex forming algorithms that we discussed in this study as follows:

Definition 1. *For a point cloud $P \subset \mathbb{R}^3$, a simplex $\sigma = [p_{i_0}, \dots, p_{i_d}]$ is in the Delaunay complex $Del(P)$ if and only if there is a ball B whose boundary contains the vertices of σ and does not include the other points of the point cloud.*

As an indicative for the simplicial complexes employed in the research, Figure 2 provides a point cloud and an example of the Delaunay complex derived from it.

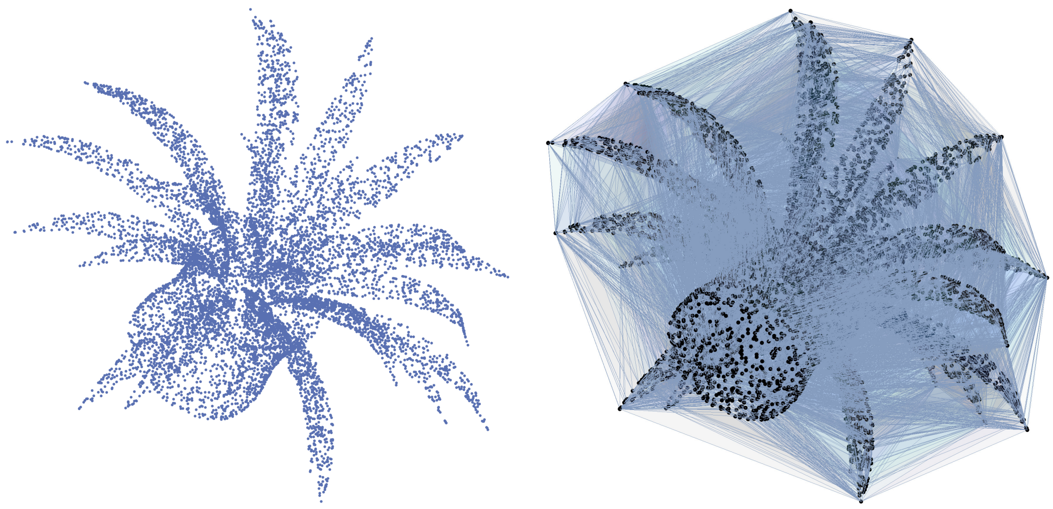

Figure 2. A point cloud example (**left**) and its Delaunay complex (**right**).

Delaunay complexes have very rich geometric features in 2D and 3D [60,61]. Only computing first-dimensional Delaunay complexes does not seem to be asymptotically faster than the computation of the full Delaunay complex. This makes the complex less attractive for high-dimensional data analysis. In this situation, Delaunay complexes' subcomplex Alpha complexes become a key tool for topological data processing. The Delaunay triangulation of a point cloud includes a subset of the faces, which together make up the Alpha complex $A_r(P)$, a d-dimensional simplicial complex [62].

Definition 2. *For a point cloud $P \subset \mathbb{R}^3$ and the given real number $r > 0$, a simplex $\sigma = [p_{i_0}, \ldots, p_{i_d}]$ is in the Alpha complex $A_r(P)$ if and only if $\bigcap_{p \in Q \subset P} Vor_r(p, P) \neq \emptyset$, where $Vor_r(p, P)$ is the Voronoi ball of p defined by the intersection of the open ball centered at p with radius r and the Voronoi cell of $p \in P$.*

Definition 3. *For a point cloud $P \subset \mathbb{R}^3$ and the given real number $r > 0$, a simplex $\sigma = [p_{i_0}, \ldots, p_{i_d}]$ is in the Čech complex $C_r(P)$ if and only if $\bigcap_{0 \leq j \leq d} B(p_{i_j}, r) \neq \emptyset$.*

It should be noted that the definition of the Čech complex includes a parameter r, which may be useful in practice. Specifically, we can think of creating a sphere at each p_i in the point cloud and looking at the convergence of the spheres on the r scale.

Definition 4. *For a point cloud $P \subset \mathbb{R}^3$ and the given real number $r > 0$, a simplex $\sigma = [p_{i_0}, \ldots, p_{i_d}]$ is in the Vietoris Rips complex $VR_r(P)$ if and only if $\forall j, j' \in [0, d]$, $B(p_{i_j}, r) \cap B(p_{i'_j}, r) \neq \emptyset$.*

In other words, the points $p_{i_0}, \ldots, p_{i_k}$ span a d-simplex if and only if the Euclidean balls with radius r centered at these points have a pairwise intersection.

Definition 5. *Let $\forall i \in [0, d]$ and $q \in Q = P \setminus \{q_0, \ldots, q_d\}$. If $d(q_i, x) \leq d(q, x)$, then it is said that the simplex $\sigma = [q_0, \ldots, q_d]$ is weakly witnessed by point x and if $d(q_i, x) = d(q, x)$, then it is said that the simplex $\sigma = [q_0, \ldots, q_d]$ is strongly witnessed by the point x. For a point cloud $P \subset \mathbb{R}^3$ and $Q \subset P$, the witness complex $W(Q, P)$ is the simplicial complex whose vertices are from the set Q and all faces are weakly witnessed by a point in P.*

Detailed information for these complexes, which are frequently used in the literature, can be found in [63–65]. Moreover, the following features can be given from the same studies:

$$W(Q,P) \subseteq Del(P) \tag{1}$$

and

$$C_r(P) \subseteq VR_r(P) \subseteq C_{2c}(P). \tag{2}$$

A comparison for the complexes used in this study is given in Table 1.

Table 1. Comparison for the complex-forming methods.

K	Dimension	Time Complexity	Guarantee
$Del(P)$	$2^{\mathcal{O}(\lvert P\rvert)}$	$\mathcal{O}(\lvert P\rvert + \lvert P\rvert \log \lvert P\rvert)$	Approx. Geometry
$C_r(P)$	$2^{\mathcal{O}(\lvert P\rvert)}$	$\mathcal{O}(\lvert P\rvert^{d+1})$	Nerve Theorem
$VR_r(P)$	$2^{\mathcal{O}(\lvert P\rvert)}$	$\mathcal{O}(M(\lvert P\rvert))$	Approx. $C_r(P)$
$W(Q,P)$	$2^{\mathcal{O}(\lvert Q\rvert)}$	$\mathcal{O}\left(\frac{\lvert P\rvert}{\mu^{d^2}}\right)$	Approx. Geometry

In this study, temporal modified methods introduced by [66,67] are used for Vietoris Rips and Witness complexes, respectively. The time complexities of these methods are given in Table 1. $M(|P|)$ represents the complexity of the product of a matrix of type $|P| \times |P|$ and μ represents the time complexity of the sparsity function of the marker of the subset Q.

2.2. Kernel Function

Modeling and computing object similarity is one of the most difficult tasks in machine learning. When it comes to graphs, graph kernels have gotten a lot of press in recent years and have emerged as the most popular method for learning from graph-structured data. A graph kernel is a symmetric, positive semi-definite function defined on the space of graphs. This function can be expressed as an inner product in some Hilbert spaces. In particular, given a kernel κ, there exists a transformation $\varphi : \mathcal{G} \to \mathcal{H}$ mapping a graph space $\mathcal{G}$ to a Hilbert space $\mathcal{H}$ such that for each $G_1, G_2 \in \mathcal{G}, \kappa(G_1, G_2) =< \varphi(G_1), \varphi(G_2) >$.

Graph kernels handle the challenge of graph comparison by attempting to efficiently capture as much of the graph's topology as possible. One of the primary reasons for the widespread adoption of graph kernel methods is that they enable a vast array of techniques to operate directly on graphs. Thus, graph kernels enable the application of machine learning methods to real-world situations using graph-structured data.

First, information on the distributions will be presented before the definition of a kernel function κ defined by the distribution of the geodesics of the graphs $K_1^{(1)} \equiv G_1$ and $K_2^{(1)} \equiv G_2$.

Kullback–Leibler information is a measure of how far apart two pieces of information are in terms of probability [68]. Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability measures with densities of p and q, respectively. Then, the Kullback–Leibler information is defined by

$$\mathcal{L}_X(\mathbb{P},\mathbb{Q}) = \int \log\left(\frac{p(x)}{q(x)}\right) p(x) d\mu(x) = \mathbb{E}_p\left[\log\left(\frac{p(x)}{q(x)}\right)\right]. \tag{3}$$

Since Kullback–Leibler information is not symmetric, it is not a metric. However $\mathcal{L}_X(\mathbb{P},\mathbb{Q}) + \mathcal{L}_X(\mathbb{Q},\mathbb{P})$ is symmetric and is called the Kullback–Leibler divergence [69].

Let us consider a random variable X with a mean μ_X and a metric function defined by $d(x;\mu_X) = f(x - \mu_X)$. For $\mu_X, \mu'_X \in \mathbb{R}$, a function defined by $g(x;\mu_X) = d(x;\mu'_X) - d(x;\mu_X)$ is Jensen equal; that is, $\mathbb{E}[g(x)] = g(\mathbb{E}[x])$. Let $c \in \mathbb{R}$ and $(\nu,\omega) \in \mathbb{R} \times \mathbb{R}^-$. For a function $f(x;\mu_X)$ with $\mathbb{E}[X] = c$, with the density function

$$f(x;\mu_X) = \nu \exp(\omega d(x;\mu_X)) = g(d(x;\mu_X)) \tag{4}$$

$\mathcal{L}_X((\mathbb{P},\mathbb{Q})$ becomes a metric [70].

Let us look at how to use Kullback–Leibler information to explain geodesic distributions on graphs. The geodesic distance between two vertices u and v in a graph G is defined as the number of edges of the shortest path linking u and v and denoted as $d_G(u,v)$. For example, in Figure 1, $d_{K^{(1)}}(2,5) = 2$. The greatest geodesic distance between u and any other vertex is called the eccentricity of u and denoted by ε. It can be thought of as a measure of how far a vertex is from the furthest vertex in the graph. The eccentricity property allows the radius and diameter of a graph to be defined. The radius of a graph is the minimum eccentricity between the vertices of the graph. The diameter of a graph is the maximum eccentricity of any vertex of the graph. Hence, the diameter is the greatest distance between any pair of vertices. In order to find the diameter of a graph, we first find the shortest path between each pair of vertices using the landmark-based method presented in [71]. The greatest distance of any path is the diameter of the graph. Using the eccentricity of a graph, it is possible to define its two subgraphs. A central subgraph of $K^{(1)}$ is the graph with n vertices of degree α and the smallest eccentricity, and is denoted by $K^{(1),C}_{n,\alpha}$. An orbital subgraph of $K^{(1)}$ is the graph with n vertices of degree β and with n vertices and is denoted by $K^{(1),O}_{n,\beta}$.

Considering the Kullback–Leibler divergence, we can define two types of geodetic distributions for Jensen equal functions:

Definition 6. *Central geodesic density function of $K^{(1)}$ is*

$$f_C(u,\alpha,\nu) = \nu \exp\left(-d_G\left(u, K^{(1),C}_{n,\alpha}\right)\right), \tag{5}$$

and the orbital geodesic density function of $K^{(1)}$ is

$$f_O(u,\beta,\nu) = \nu \exp\left(-d_G\left(u, K^{(1),O}_{n,\beta}\right)\right), \tag{6}$$

where $\nu \geq 1$ is a normalization factor.

On a space of probability measures, the Wasserstein distances give a natural metric. They intuitively assess the least amount of effort necessary to change one distribution into another. The Wasserstein distances, in general, do not permit closed-form formulations; however, for $\mathbb{R}$, we have the explicit form as

$$W_1(\rho_1,\rho_2) = \int_{\mathbb{R}} |F_{\rho_1}(t) - F_{\rho_2}(t)|\, dt, \tag{7}$$

where $F_{\rho_1}(t)$ and $F_{\rho_2}(t)$ are the cumulative distribution functions of ρ_1 and ρ_2 [72]. It is feasible to compare the geodesic distributions with the kernel function below using the Wasserstein-1 distance function:

Definition 7. *Let $K^{(1)}_1 = (V_1,E_1)$ and $K^{(1)}_2 = (V_2,E_2)$. Then, we have a kernel function*

$$\kappa_J\left(K^{(1)}_1, K^{(1)}_2\right) = \frac{W_1\left(\cup_{i=1}^{|V_1|}, f_C(u_i,\alpha_1,\nu_1), \cup_{j=1}^{|V_2|}, f_C(u_j,\alpha_2,\nu_2)\right)}{W_1\left(\cup_{i=1}^{|V_1|}, f_O(u_i,\beta_1,\nu_1), \cup_{j=1}^{|V_2|}, f_O(u_j,\beta_2,\nu_2)\right)}, \tag{8}$$

where W_1 is Wassterstein-1 distance.

We shall note that the kernel function defined in Equation (8) is symmetric and positive definite.

3. Results

3.1. Data Set

Princeton ModelNet (Internet Access: https://modelnet.cs.princeton.edu/, accessed on 10 January 2023) seeks to give researchers in computer vision, computer graphics, robotics, and cognitive science a thorough and organized library of 3D CAD models. The researchers used information from the SUN database to create a list of the most widespread object types on the globe, which served as the foundation of the dataset. After developing an object vocabulary, the 3D CAD models of each item category were gathered by searching web search engines for each category phrase. Then, using custom-made tools with quality control, human employees were hired at Amazon Mechanical Turk to manually determine whether each CAD model fits into the designated categories [73].

Two methods are used to assess the effectiveness of the graph kernel function given in this paper. The first strategy is to assess the extent to which the topologies of the underlying point cloud geometries have an impact on the kernel function. In this study, the kernel function is used to determine the extent to which point clouds are similar or dissimilar. Point cloud samples containing 10,000 points are taken from the ModelNet-40 dataset for the first method. These point cloud examples belong to the airplane, car, person, plant, and vase classes as they have different inter- and intra-class topologies. Ten sample point clouds are taken from each class, and the kernel function presented in this study is run between these samples.

The second strategy involves using the provided kernel function to create a classification that is based on machine learning. On the ModelNet-40 dataset, this classification methodology is used, and it is compared to other approaches.

3.2. Point Cloud Comparison

To produce graphs from point clouds, each of the simplicial complex derivation techniques described in Section 2.1 are applied to the sample set, and the 1-skeletons of the complexes were selected as the representation graph. Equation (8) was used in this part to generate the graphs for each data group as well as the kernel matrices of the data subgroups. The point clouds used in this approach are presented in Figures A1–A5.

The relevant parameters are selected in the formation of simplicial complexes as the minimum values that connect the 1-skeletons derived from the point cloud. Using a step size of $h = 0.001$ and beginning at 0, the least parameter is found. It is anticipated that different parameters will be found for each sample, both inter- and intra-classes. In addition to this, no discernible variation in the computational time complexity of the simplicial complexes' generation procedures was found. Moreover, better computational times could result from differentiating the approach taken when determining the lowest parameter.

Figures 3–7 contain similarity matrices obtained by using the kernel function to compare point clouds with each other. The samples from each class are complexed using the previously discussed simplex techniques, and the kernel function is applied to the graph structures that were built using the 1-skeletons of the samples.

Similarity matrices generated using the kernel function encode the measurements of the similarity between the point clouds. In the context of the distribution of discrete geodesics, the near-zero values of the kernel function provided by Equation (8), whose inputs are two graphs produced from point clouds, indicate that the two graphs are highly similar. Likewise, the graphs diverge in terms of similarity within the same context for large numbers. This study used four of the most fundamental simplicial complex approaches to determine the similarity metrics within each class. The analysis of the similarity matrices reveals that the common points of the point clouds in the discrete geometry environment within each class are most evident in the graphs produced with Alpha complexes. The geometries underlying the point clouds of the Airplane models are most comparable in the case of the Airplane 5 model. The matrix given in Figure 3 clearly demonstrates that the similarity value derived from the graph of Alpha complexes for this model is close to zero. When a comparable course is followed, the Airplane 4 and 8 models entirely diverge in terms of similarity, as shown by the matrix's high values. In the similarity matrices provided in Figures 4–7, the models most geometrically comparable to other models for the circumstance produced with the Alpha complex have very low values.

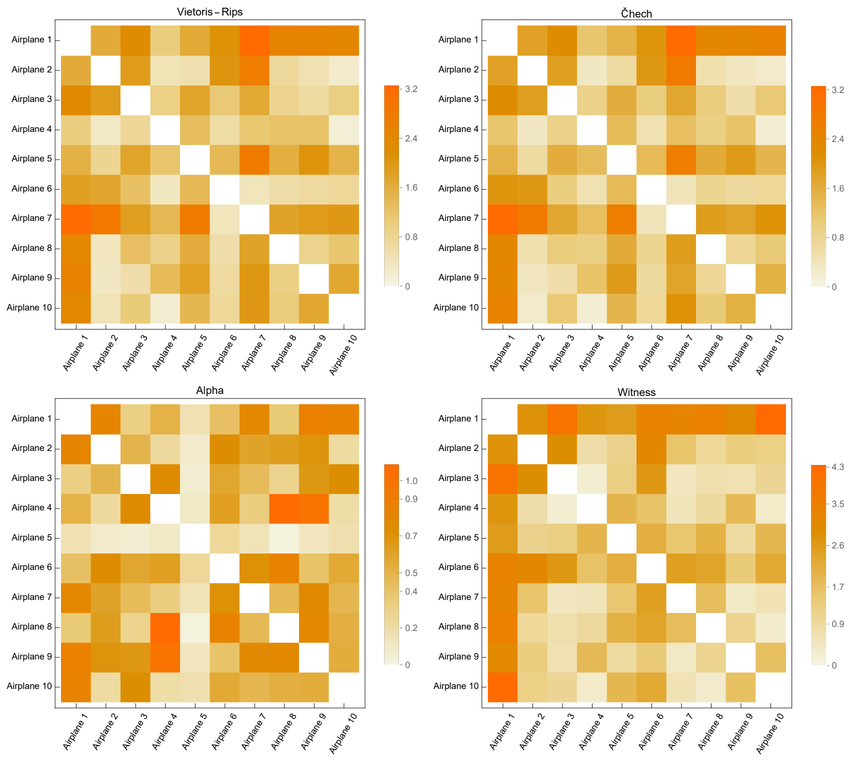

Figure 3. Similarity Matrices for Airplane Models.

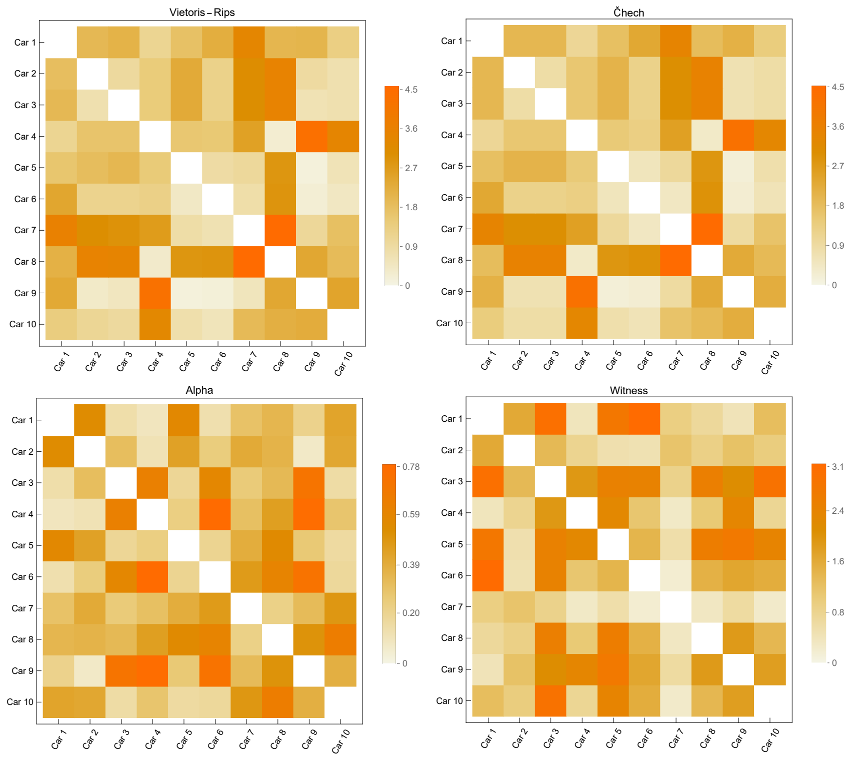

Figure 4. Similarity Matrices for Car Models.

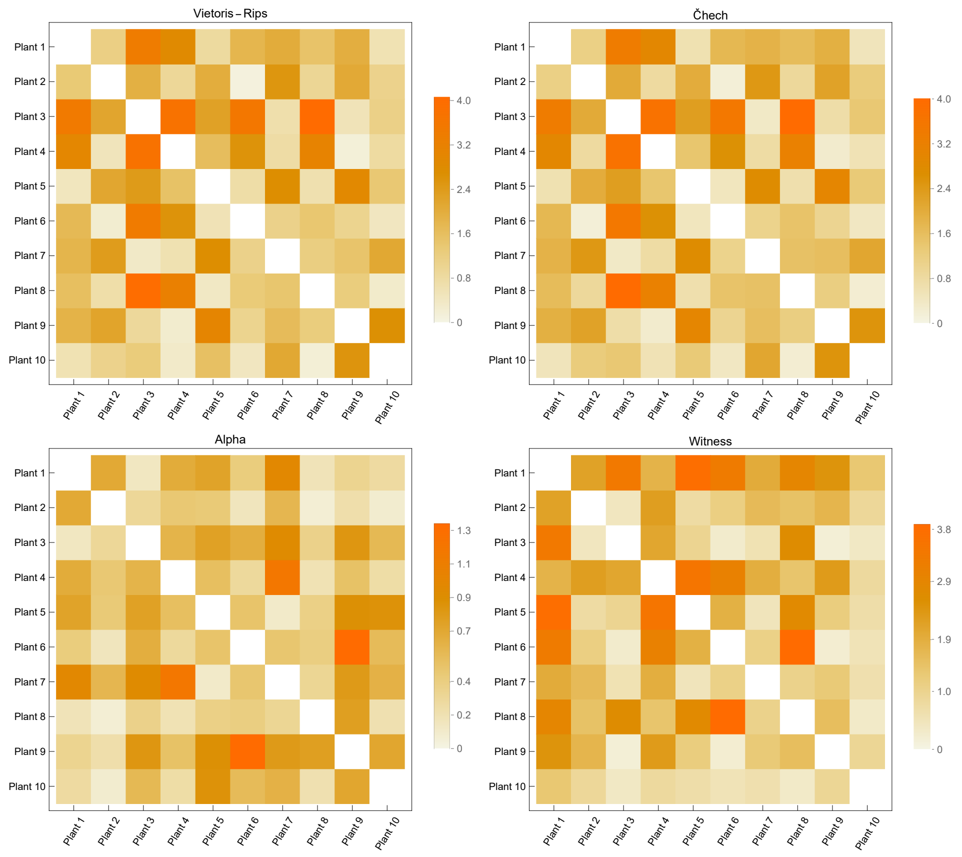

Figure 5. Similarity Matrices for Plant Models.

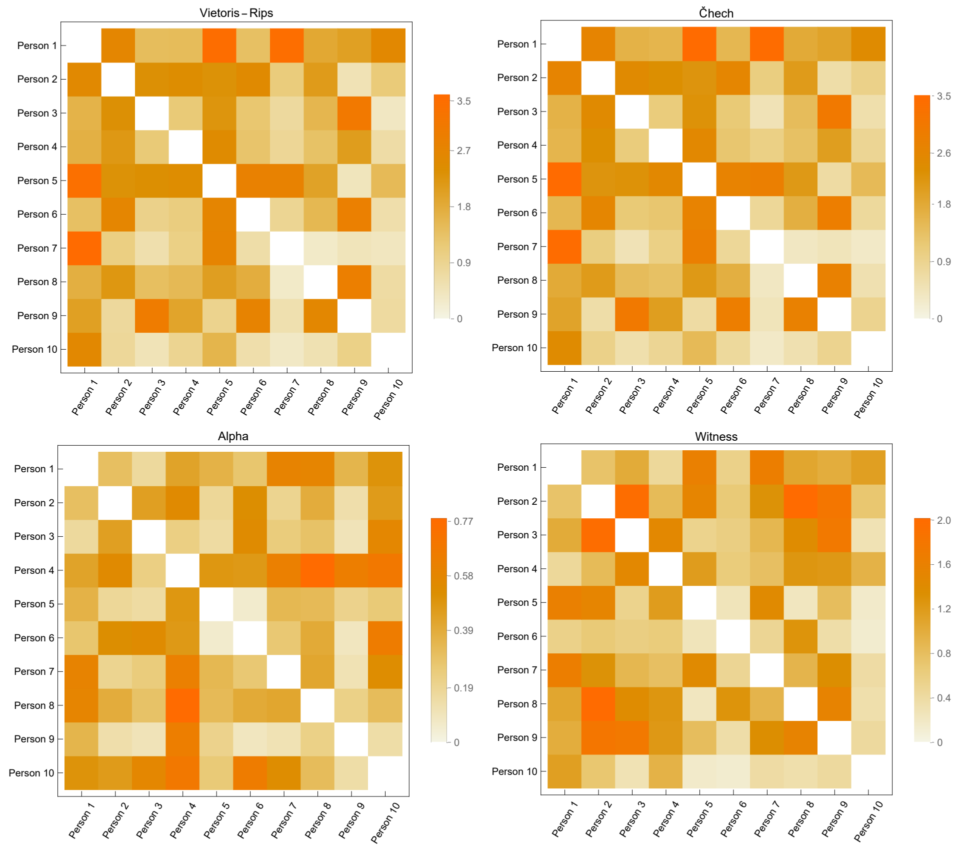

Figure 6. Similarity Matrices for Person Models.

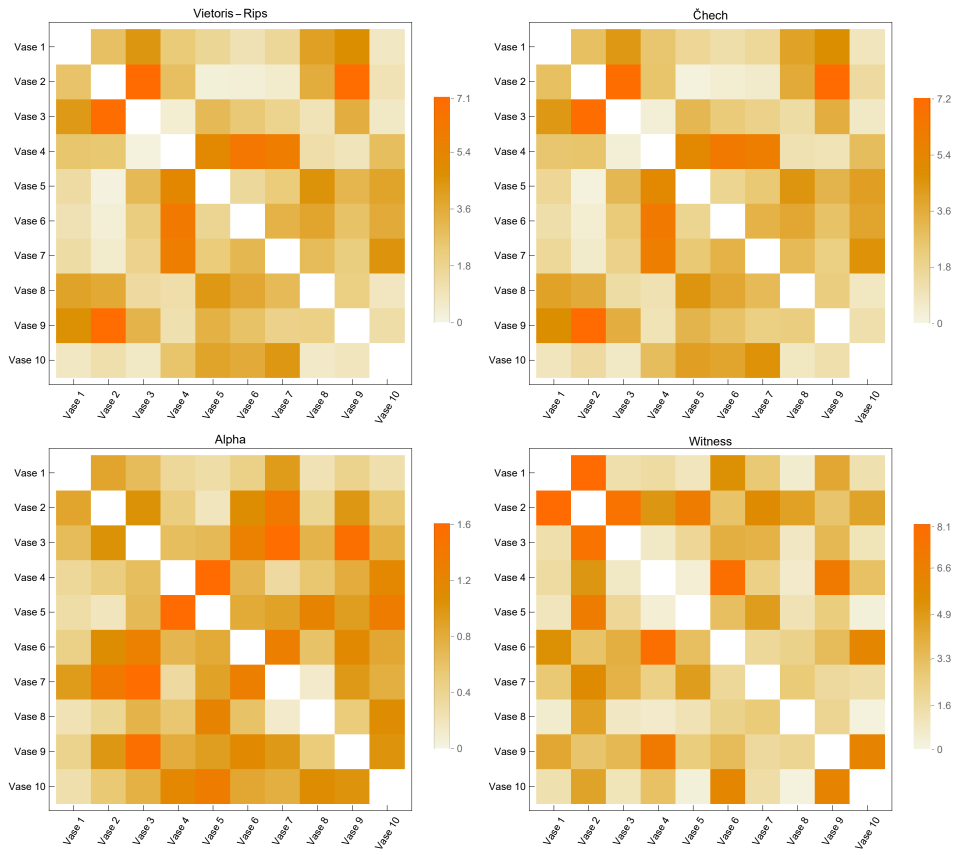

Figure 7. Similarity Matrices for Vase Models.

The graphs generated using the Vietoris Rips and Čech complexes reveal that the kernel function values are quite similar. The graphs formed by the Witness complex have the most inconsistent similarity values among these four techniques. When examining the underlying geometries of each point cloud, the kernel function assigns large values to very comparable geometries. Even if the Vase 2 model is physically comparable to models 1 and 3, the inputs of the Witness complex's similarity matrix are relatively high. The observation that the kernel function computed on the graphs produced with the Vietoris Rips and Čech complexes has a large value when it is low is another result of the calculations performed on the graphs acquired with the Alpha complexes.

Considering all of these circumstances, it is feasible to conclude that the 1-skeleton of Alpha complexes are the graphs in which the kernel function developed in this research performs the best. Additionally, it indicates that these values vary when the underlying geometries have sharp edges. Since geodesics become geometrically singular at sharp endpoints, it is said that the presented kernel function provides better geometrically correct results when the Alpha complex is utilized to generate graphs.

In order to apply this kernel function to assess the similarities between distinct model classes, the similarities between the classes are analyzed in our research using the graphs derived from the Alpha complexes. Within the scope of this precision, Figure 8 depicts the inter-class similarity matrix constructed by applying the kernel function to the graphs generated using the Alpha complexes of five distinct models. This matrix also displays graphs with comparable discrete geodesy distributions whose values are close to zero. There are two remarkable commonalities between the two courses. The Car and Vase models, as well as the Airplane and Person models, are among the most different models. In other words, the kernel function in these models has high values.

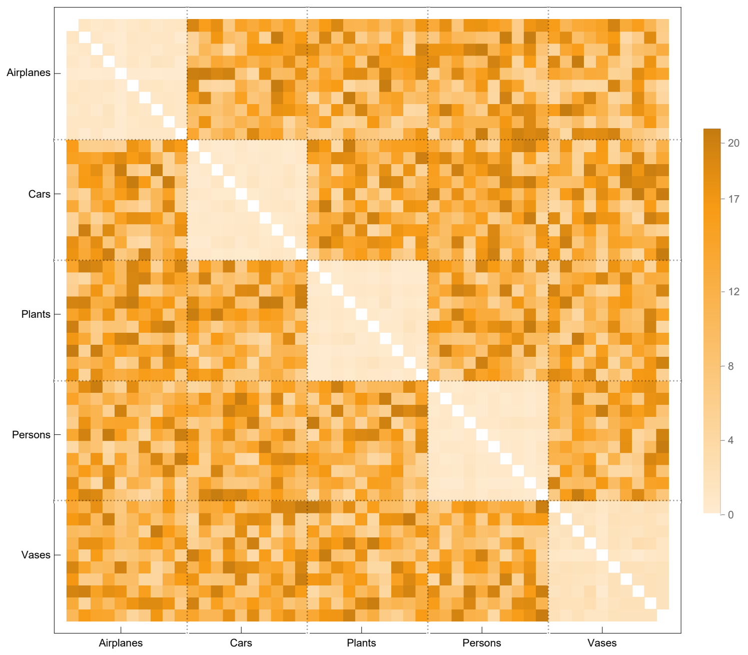

Figure 8. Interclass similarity matrix for graphs obtained by the Alpha complex.

3.3. Classification

Figure 8 illustrates how useful similarity matrices between classes are when they are produced using the graph kernel function discussed in this study. This part describes how to use this kernel function to infer the class to which a given point cloud belongs. The goal is to create a model that learns from the input graphs $K_1^{(1)}, K_2^{(1)}, \ldots, K_N^{(1)}$ and predicts the label of new, unobserved graphs using this dataset of input graphs.

With the usage of graph kernel functions in convolution processes, particularly with the advancements in GCN networks, several deep learning architectures have been built in recent years [74–78]. However, these systems often utilize vectors between graph vertices or edges that correlate with different physical qualities. Additionally, kernel-based classification techniques use kernel functions created across different subgraphs to quantify the similarities between two vertices. It is important to build a matrix between the vertex clusters, namely graph communities, and not between vertex pairs, in the classification process since the kernel function taken into account in this study takes into account the discrete geometries underlying the 3D point clouds.

It is known that the matrices of kernel functions created between two graph classes include the structures utilized to embed graphs in Hilbert space. Therefore, it is inappropriate to apply the in-class kernel function directly to solve the graph classification issue. In this study, communities of graphs that are specific to each point cloud were identified, and matrices with the kernel function were produced among the subgraphs generated by the communities. The Fluid Communities [79] approach was used in this study to find the communities of each graph. The propagation mechanism on which Fluid Communities is based is state of the art in terms of computing cost and scalability. Despite being very effective, Fluid Communities can locate communities in artificial graphs with a degree of accuracy that is competitive with the best options. The first propagation-based method that can recognize a changeable number of communities in the network is Fluid Communities. Figure 9 depicts the 10 graph communities found in the 1-skeleton of the Alpha complex using the Fluid Communities approach.

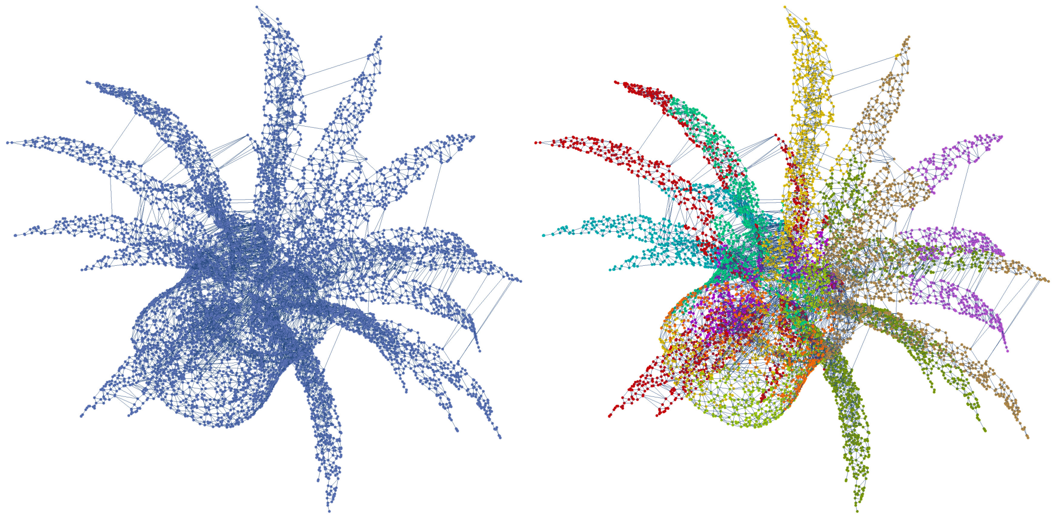

Figure 9. The 1-skeleton of an Alpha complex emerging from plant point cloud and 10 communities obtained using the Fluid Communities approach.

Each initial point cloud sample contains the communities of graphs derived from Alpha complexes for the classification process. The number of communities is determined to be 10, as the community discovery technique utilized in this study needs that number to be predetermined. The kernel function created in the research among the subgraphs discovered by the communities is then used to realize the commonalities between these communities. In order to represent the inherent geometry of each point cloud, a 10×10 matrix is produced. Then, a CNN network with three convolution layers is used. After the first convolution layer, Max Pooling was performed, followed by Average Pooling for the second and third layers. While ReLU activation functions were used for the first two Fully Connected layers, softmax was utilized for classification after the second layer. The ModelNet-40 dataset's whole sample pool was used in the classification procedure. The ModelNet-40 dataset includes 2468 test models and 9843 training models divided into 40 classes. We uniformly choose 1024 points from each model and normalize them to a unit sphere in order to compare our results with those of other approaches in the literature. We give accurate data from the classification procedure and comparisons to alternative approaches in Table 2.

Table 2. Classification results on ModelNet-40 benchmark.

Method	Input	No. Points	Accuracy
[80]	xyz	1024	% 86.1
[81]	xyz	1024	% 87.1
[82]	xyz	1024	% 87.4
[83]	xyz	1024	% 89.2
[84]	xyz	1024	% 90.0
[85]	xyz	1024	% 90.2
[86]	xyz	1024	% 90.6
[87]	xyz	1024	% 90.7
[88]	xyz	1024	% 91.0
[89]	xyz	1024	% 92.2
[90]	xyz	1024	% 92.3
[91]	xyz	1024	% 93.6
Ours	xyz	1024	% 93.7

4. Discussion and Conclusions

Graphs are maybe one of the most comprehensive data structures used in machine learning. Complex objects may be represented using graphs as a collection of items and their connections, each of which can be annotated with information such as category or vectorial vertex and edge properties. As graph states, both unstructured vector data and structured data types such as time series, images, volumetric data, point clouds, and asset bags may be understood. Most importantly, the extra flexibility that graph-based representations provide is helpful for a variety of applications.

Kernel approaches are being increasingly used as a method for determining how comparably organized items are. In general, kernel approaches compare two objects using a kernel function that corresponds to an inner product in a kernel Hilbert space. Finding an appropriate kernel function that can be traced computationally while capturing the structure's semantics is challenging for kernel approaches. In this paper, a kernel function based on the distribution of discrete geodesics on graphs is used to solve the point cloud comparison problem, one of the most important topics in computer vision. Creating this kernel function requires the description of a technique sensitive to isometries and topological transformations.

The dataset was chosen based on the Princeton ModelNet-40 benchmark. Similarity matrices were generated by applying the kernel function to 10 point clouds across five distinct categories. When the similarity matrices are assessed, it is discovered that the graphs of the 1-skeletons of the Alpha complexes provide the highest internal similarity of the categories. Among the point clouds within each category, the similarity value (lower kernel value) between models that are more topologically connected, i.e., structural holes and bottleneck structures, is rather high. The kernel value distributions of the Airplane, Car, Person, Plant, and Vase models attain lower values (more similarity), notably in the Car and Vase models. When the point clouds of both models are analyzed, it becomes evident that these two groups have several topological similarities. Therefore, we may conclude that the kernel function discussed in this research performs well when applied to topological similarities, particularly when the Alpha complex is considered.

The classification of graphs is an issue that regularly arises in machine learning research. Kernel functions that are defined by structural measurements between the related subgraphs of a graph are commonly used for graph classification problems. In this study, the kernel function suggested for graph classification is studied. Within the graphs of each point cloud, the input matrices for a simple convolution neural network model are generated. Utilizing the aggregates of graphs with vertex clusters, the category to which a graph in vector form belongs was determined. For each network model, the Fluid Community approach was utilized to integrate a fixed cluster size while identifying the communities. The kernel function was used to generate inherent similarity matrices between the subgraphs generated by graph communities. This method yielded classification results with an accuracy rating of 91.1%. Even though this percentage is not the greatest reported in the literature, the results were quite effective in comparison to other approaches.

This kernel function, which was designed to compare point clouds with their geometries and topologies, is independent of the dimension of the point cloud, and therefore may be utilized in future research with 2D or higher dimensional point clouds. In addition, it is anticipated that the deep learning architecture used in the classification method provided in this paper will be refined, resulting in improved classification outcomes.

Author Contributions: Conceptualization, Ö.A. and M.A.B.; Formal analysis, Ö.A., M.A.B. and L.M.B.; Funding acquisition, L.G.; Investigation, Ö.A. and M.A.B.; Methodology, Ö.A., M.A.B. and L.M.B.; Project administration, Ö.A.; Resources, Ö.A., M.A.B., L.M.B. and L.G.; Supervision, Ö.A. and M.A.B.; Writing—original draft, Ö.A., M.A.B., L.M.B. and L.G.; Writing—review and editing, Ö.A., M.A.B., L.M.B. and L.G. All authors have read and agreed to the published version of the manuscript.

Funding: This study was conducted with financial support from the scientific research funds of "1 Decembrie 1918" University of Alba Iulia, Romania. Moreover, analyses and results were funded by TUBITAK grant number 121E031.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A

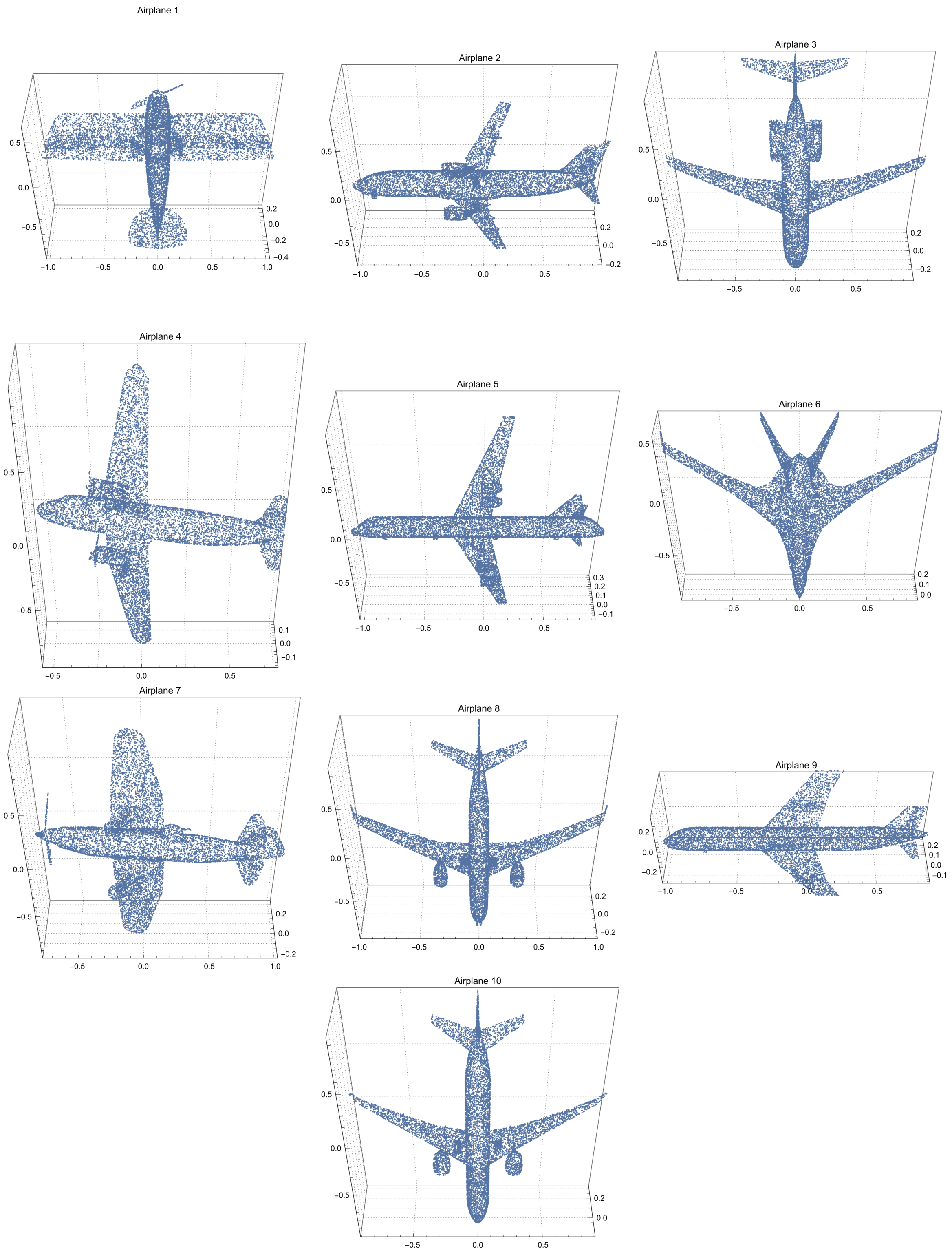

Figure A1. Airplane Models.

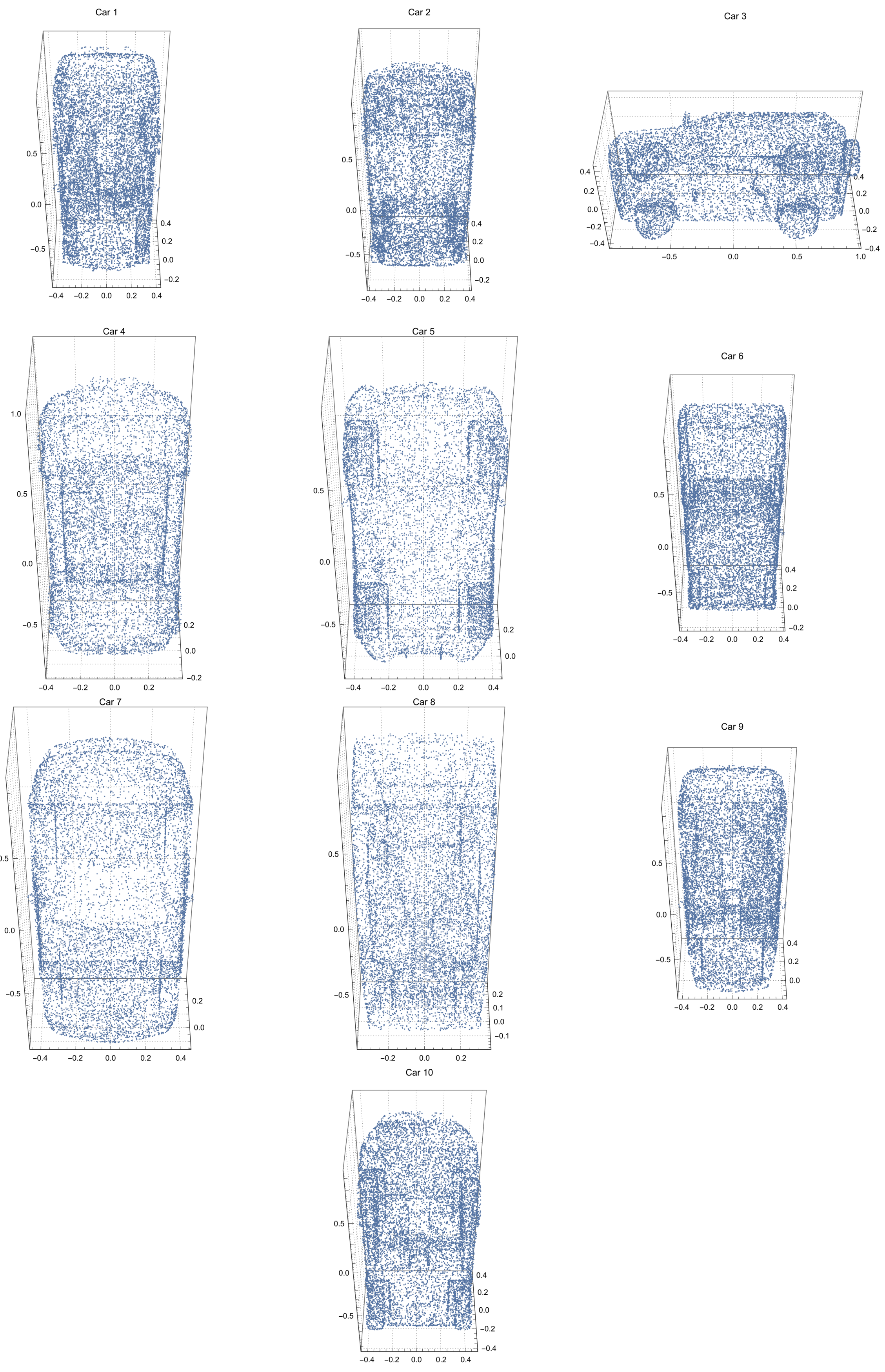

Figure A2. Car Models.

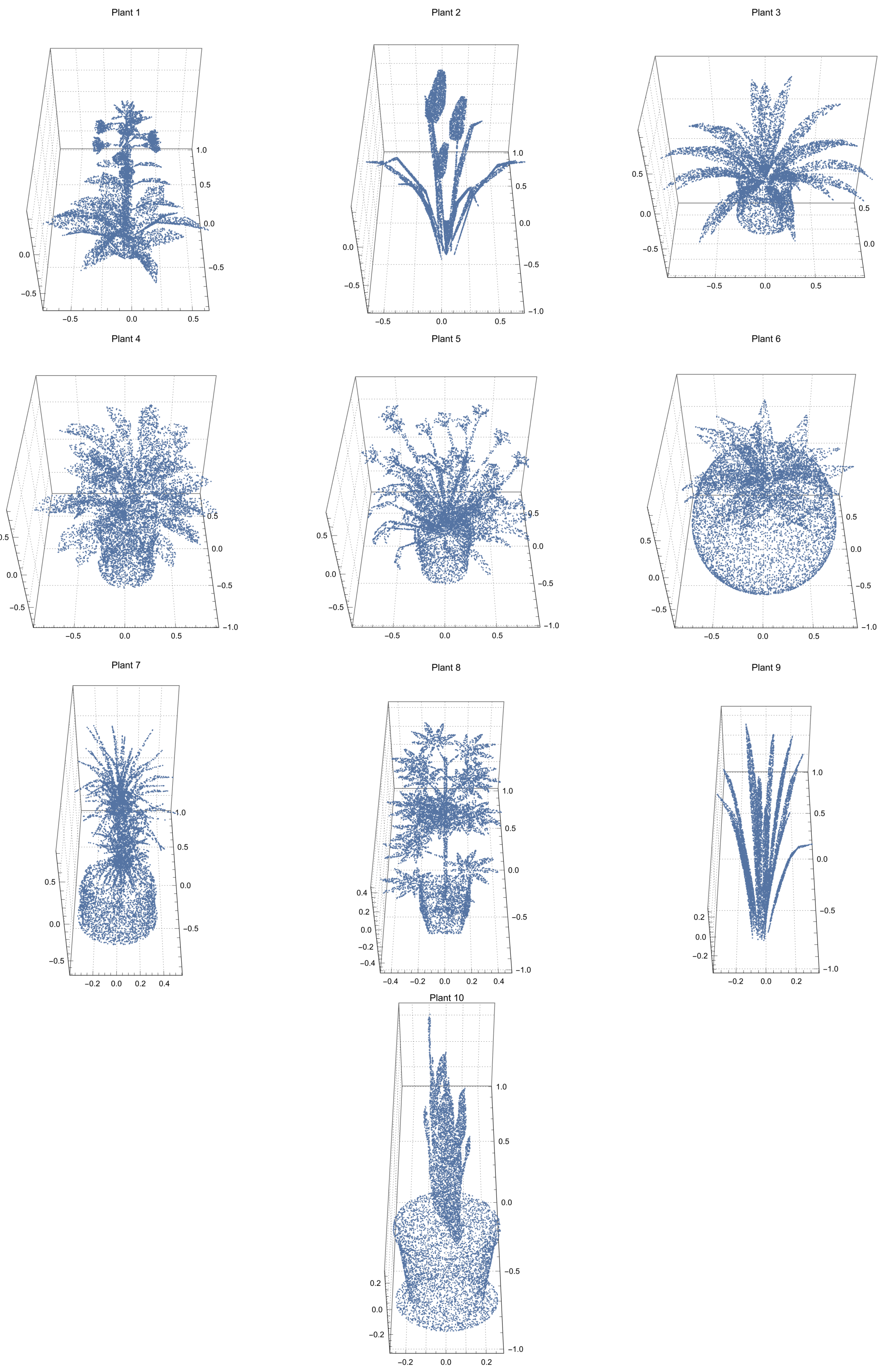

Figure A3. Plant Models.

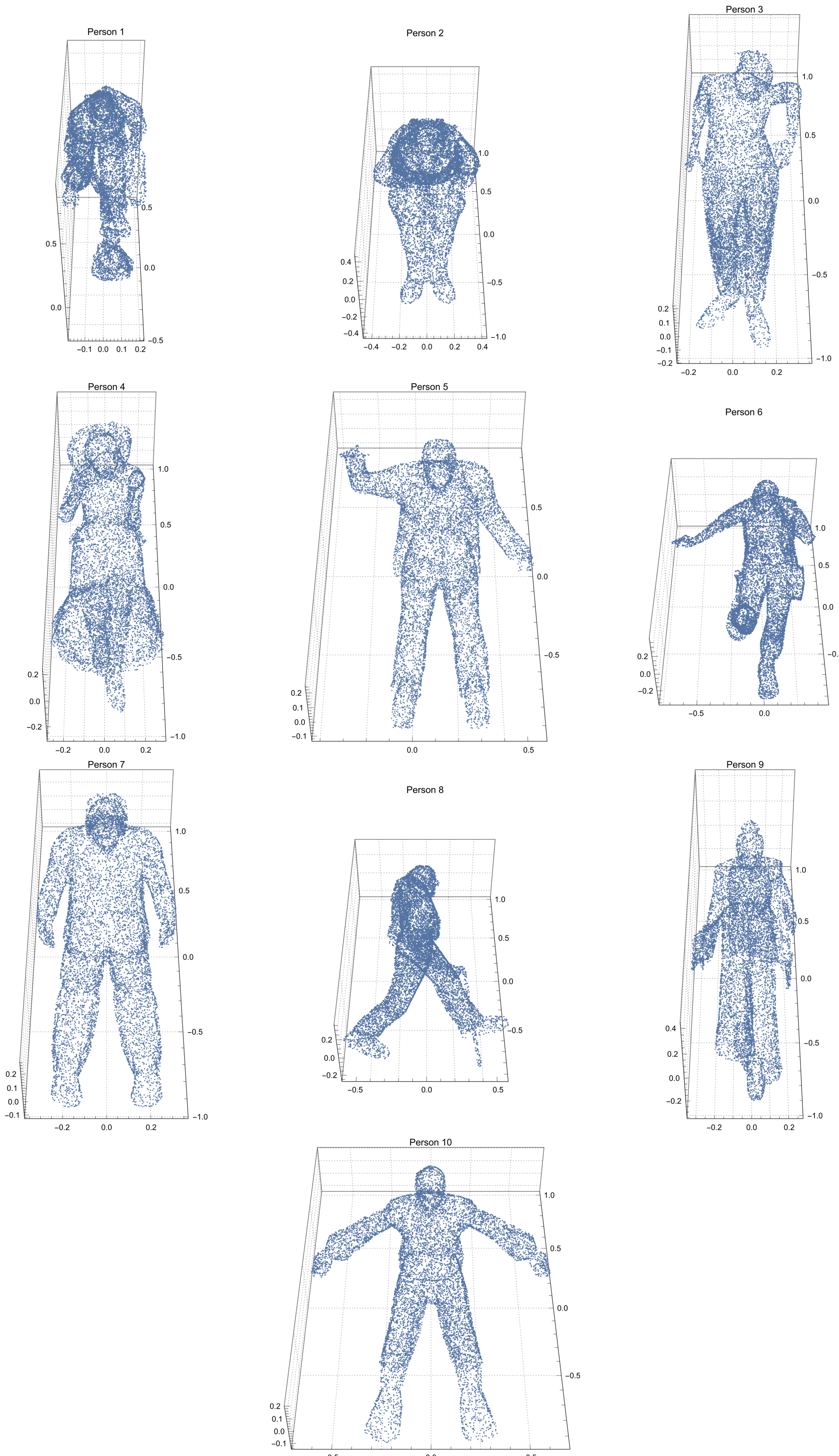

Figure A4. Person Models.

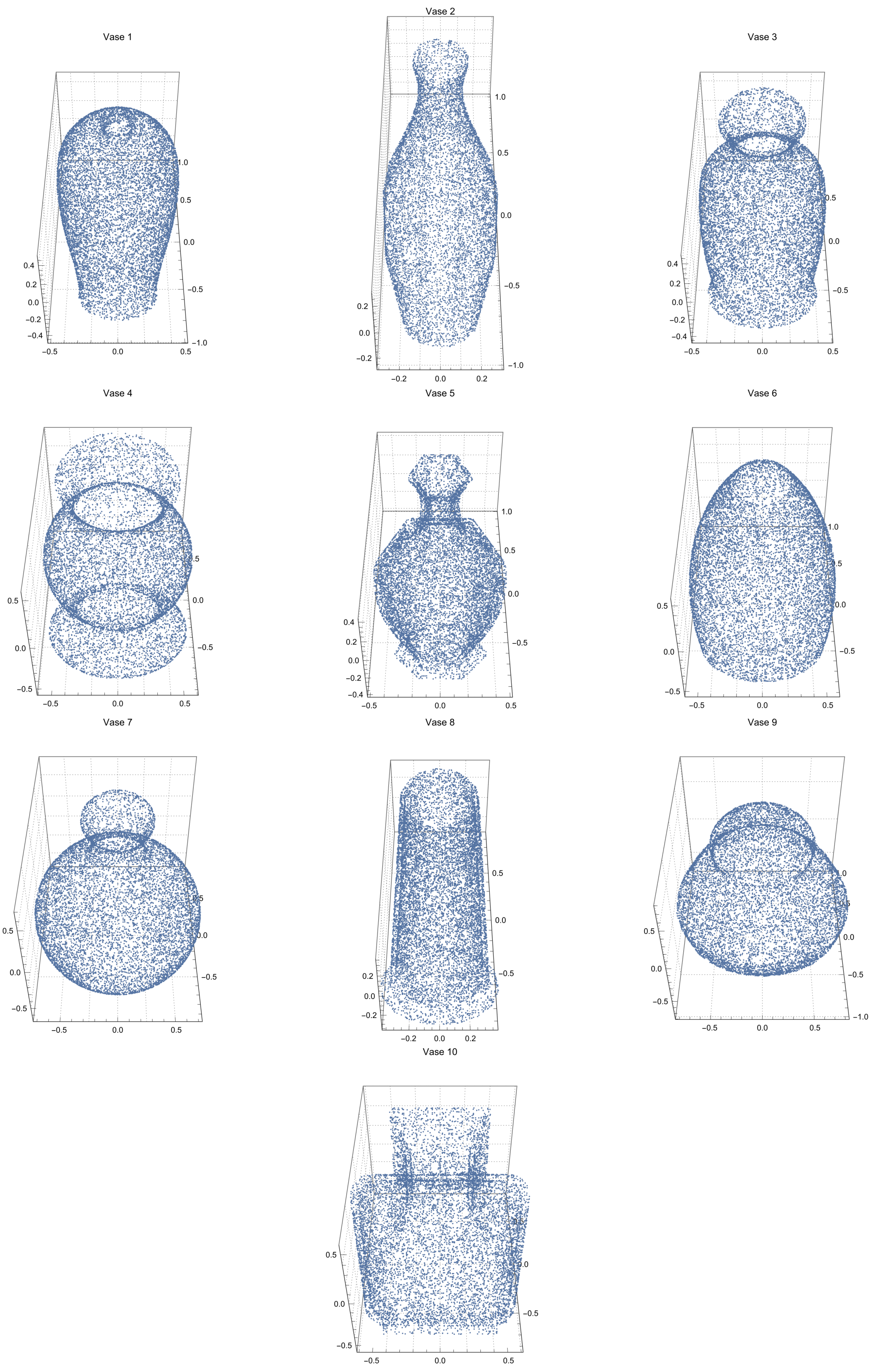

Figure A5. Vase Models.

References

1. Chougule, V.; Mulay, A.; Ahuja, B. Three dimensional point cloud generations from CT scan images for bio-cad modeling. In Proceedings of the International Conference on Additive Manufacturing Technologies—AM2013, Bengaluru, India, 7–8 October 2013 Volume 7, p. 8.
2. Fu, Y.; Lei, Y.; Wang, T.; Patel, P.; Jani, A.B.; Mao, H.; Curran, W.J.; Liu, T.; Yang, X. Biomechanically constrained non-rigid MR-TRUS prostate registration using deep learning based 3D point cloud matching. *Med Image Anal.* **2021**, *67*, 101845. [CrossRef]
3. Gidea, M.; Katz, Y. Topological data analysis of financial time series: Landscapes of crashes. *Phys. A Stat. Mech. Its Appl.* **2018**, *491*, 820–834. [CrossRef]
4. Ismail, M.S.; Hussain, S.I.; Noorani, M.S.M. Detecting early warning signals of major financial crashes in bitcoin using persistent homology. *IEEE Access* **2020**, *8*, 202042–202057. [CrossRef]
5. Liu, J.; Guo, P.; Sun, X. An Automatic 3D Point Cloud Registration Method Based on Biological Vision. *Appl. Sci.* **2021**, *11*, 4538. [CrossRef]
6. Barra, V.; Biasotti, S. 3D shape retrieval and classification using multiple kernel learning on extended Reeb graphs. *Vis. Comput.* **2014**, *30*, 1247–1259. [CrossRef]
7. Chen, L.M. *Digital and Discrete Geometry: Theory and Algorithms*; Springer: Berlin/Heidelberg, Germany, 2014.
8. Song, M. A personalized active method for 3D shape classification. *Vis. Comput.* **2021**, *37*, 497–514. [CrossRef]
9. Wang, F.; Lv, J.; Ying, G.; Chen, S.; Zhang, C. Facial expression recognition from image based on hybrid features understanding. *J. Vis. Commun. Image Represent.* **2019**, *59*, 84–88. [CrossRef]
10. Boulch, A.; Puy, G.; Marlet, R. FKAConv: Feature-kernel alignment for point cloud convolution. In Proceedings of the Asian Conference on Computer Vision, Kyoto, Japan, 30 November–4 December 2020.
11. Gou, J.; Yi, Z.; Zhang, D.; Zhan, Y.; Shen, X.; Du, L. Sparsity and geometry preserving graph embedding for dimensionality reduction. *IEEE Access* **2018**, *6*, 75748–75766. [CrossRef]
12. Meltzer, P.; Mallea, M.D.G.; Bentley, P.J. PiNet: Attention Pooling for Graph Classification. *arXiv* **2020**, arXiv:2008.04575.
13. Wang, T.; Liu, H.; Li, Y.; Jin, Y.; Hou, X.; Ling, H. Learning combinatorial solver for graph matching. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 7568–7577.
14. Wu, J.; Pan, S.; Zhu, X.; Zhang, C.; Philip, S.Y. Multiple structure-view learning for graph classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *29*, 3236–3251. [CrossRef]
15. Mitchell, J.S.; Mount, D.M.; Papadimitriou, C.H. The discrete geodesic problem. *SIAM J. Comput.* **1987**, *16*, 647–668. [CrossRef]
16. Polthier, K.; Schmies, M. Straightest geodesics on polyhedral surfaces. In *ACM SIGGRAPH 2006 Courses*; Association for Computing Machinery: New York, NY, USA, 2006; pp. 30–38.
17. Atici, M.; Vince, A. Geodesics in graphs, an extremal set problem, and perfect hash families. *Graphs Comb.* **2002**, *18*, 403–413. [CrossRef]
18. Bernstein, M.; De Silva, V.; Langford, J.C.; Tenenbaum, J.B. *Graph Approximations to Geodesics on Embedded Manifolds*; Technical Report; Citeseer: University Park, PA, USA, 2000.
19. Alexa, M.; Behr, J.; Cohen-Or, D.; Fleishman, S.; Levin, D.; Silva, C.T. Point set surfaces. In Proceedings of the Visualization, 2001—VIS'01, San Diego, CA, USA, 21–26 October 2001; pp. 21–29.
20. Amenta, N.; Kil, Y.J. Defining point-set surfaces. *ACM Trans. Graph. (TOG)* **2004**, *23*, 264–270. [CrossRef]
21. Levin, D. Mesh-independent surface interpolation. In *Geometric Modeling for Scientific Visualization*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 37–49.
22. Adamson, A.; Alexa, M. Approximating and intersecting surfaces from points. In Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, Aachen, Germany, 23–25 June 2003; pp. 230–239.
23. Amenta, N.; Choi, S.; Kolluri, R.K. The power crust, unions of balls, and the medial axis transform. *Comput. Geom.* **2001**, *19*, 127–153. [CrossRef]
24. Dey, T.K.; Goswami, S. Provable surface reconstruction from noisy samples. *Comput. Geom.* **2006**, *35*, 124–141. [CrossRef]
25. Attene, M.; Patanè, G. Hierarchical structure recovery of point-sampled surfaces. *Comput. Graph. Forum* **2010**, *29*, 1905–1920. [CrossRef]
26. Chen, X. Hierarchical rigid registration of femur surface model based on anatomical features. *Mol. Cell. Biomech.* **2020**, *17*, 139. [CrossRef]
27. Nakarmi, U.; Wang, Y.; Lyu, J.; Liang, D.; Ying, L. A kernel-based low-rank (KLR) model for low-dimensional manifold recovery in highly accelerated dynamic MRI. *IEEE Trans. Med. Imaging* **2017**, *36*, 2297–2307. [CrossRef]
28. Zhang, S.; Cui, S.; Ding, Z. Hypergraph spectral analysis and processing in 3D point cloud. *IEEE Trans. Image Process.* **2020**, *30*, 1193–1206. [CrossRef]
29. Connor, M.; Kumar, P. Fast construction of k-nearest neighbor graphs for point clouds. *IEEE Trans. Vis. Comput. Graph.* **2010**, *16*, 599–608. [CrossRef] [PubMed]
30. Natali, M.; Biasotti, S.; Patanè, G.; Falcidieno, B. Graph-based representations of point clouds. *Graph. Model.* **2011**, *73*, 151–164. [CrossRef]
31. Klein, J.; Zachmann, G. Point cloud surfaces using geometric proximity graphs. *Comput. Graph.* **2004**, *28*, 839–850. [CrossRef]
32. Fan, H.; Wang, Y.; Gong, J. Layout graph model for semantic façade reconstruction using laser point clouds. *Geo-Spat. Inf. Sci.* **2021**, *24*, 403–421. [CrossRef]

33. Wang, Y.; Ma, Y.; Zhu, A.x.; Zhao, H.; Liao, L. Accurate facade feature extraction method for buildings from three-dimensional point cloud data considering structural information. *ISPRS J. Photogramm. Remote Sens.* **2018**, *139*, 146–153. [CrossRef]
34. Wang, Y.; Li, H.; Liao, L. DEM Construction Method for Slopes Using Three-Dimensional Point Cloud Data Based on Moving Least Square Theory. *J. Surv. Eng.* **2020**, *146*, 04020013. [CrossRef]
35. Wu, B.; Yang, L.; Wu, Q.; Zhao, Y.; Pan, Z.; Xiao, T.; Zhang, J.; Wu, J.; Yu, B. A Stepwise Minimum Spanning Tree Matching Method for Registering Vehicle-Borne and Backpack LiDAR Point Clouds. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–13. [CrossRef]
36. Guo, Q.; Wang, Y.; Yang, S.; Xiang, Z. A method of blasted rock image segmentation based on improved watershed algorithm. *Sci. Rep.* **2022**, *12*, 7143. [CrossRef]
37. Wang, Y.; Tu, W.; Li, H. Fragmentation calculation method for blast muck piles in open-pit copper mines based on three-dimensional laser point cloud data. *Int. J. Appl. Earth Obs. Geoinf.* **2021**, *100*, 102338. [CrossRef]
38. Wang, Y.; Zhou, T.; Li, H.; Tu, W.; Xi, J.; Liao, L. Laser point cloud registration method based on iterative closest point improved by Gaussian mixture model considering corner features. *Int. J. Remote Sens.* **2022**, *43*, 932–960. [CrossRef]
39. Alaba, S.Y.; Ball, J.E. A survey on deep-learning-based lidar 3d object detection for autonomous driving. *Sensors* **2022**, *22*, 9577. [CrossRef] [PubMed]
40. Song, F.; Shao, Y.; Gao, W.; Wang, H.; Li, T. Layer-wise geometry aggregation framework for lossless lidar point cloud compression. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 4603–4616. [CrossRef]
41. Wang, S.; Jiao, J.; Cai, P.; Wang, L. R-pcc: A baseline for range image-based point cloud compression. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 10055–10061.
42. Xiong, J.; Gao, H.; Wang, M.; Li, H.; Ngan, K.N.; Lin, W. Efficient geometry surface coding in V-PCC. *IEEE Trans. Multimed.* 2022, *early access*. [CrossRef]
43. Zhao, L.; Ma, K.K.; Liu, Z.; Yin, Q.; Chen, J. Real-time scene-aware LiDAR point cloud compression using semantic prior representation. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 5623–5637. [CrossRef]
44. Sun, X.; Wang, S.; Wang, M.; Cheng, S.S.; Liu, M. An advanced LiDAR point cloud sequence coding scheme for autonomous driving. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA 12–16 October 2020; pp. 2793–2801.
45. Sun, X.; Wang, M.; Du, J.; Sun, Y.; Cheng, S.S.; Xie, W. A Task-Driven Scene-Aware LiDAR Point Cloud Coding Framework for Autonomous Vehicles. *IEEE Trans. Ind. Inform.* 2022, *early access*. [CrossRef]
46. Filippone, M.; Camastra, F.; Masulli, F.; Rovetta, S. A survey of kernel and spectral methods for clustering. *Pattern Recognit.* **2008**, *41*, 176–190. [CrossRef]
47. Kriege, N.M.; Johansson, F.D.; Morris, C. A survey on graph kernels. *Appl. Netw. Sci.* **2020**, *5*, 1–42. [CrossRef]
48. Nikolentzos, G.; Siglidis, G.; Vazirgiannis, M. Graph Kernels: A Survey. *J. Artif. Intell. Res.* **2021**, *72*, 943–1027. [CrossRef]
49. Gaidon, A.; Harchaoui, Z.; Schmid, C. A time series kernel for action recognition. In Proceedings of the BMVC 2011-British Machine Vision Conference, Dundee, UK, 29 August–2 September 2011; pp. 63.1–63.11.
50. Bach, F.R. Graph kernels between point clouds. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 25–32.
51. Wang, L.; Sahbi, H. Directed acyclic graph kernels for action recognition. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 3168–3175.
52. Harchaoui, Z.; Bach, F. Image classification with segmentation graph kernels. In Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 17–22 June 2007; pp. 1–8.
53. Borgwardt, K.M.; Kriegel, H.P. Shortest-path kernels on graphs. In Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05), Houston, TX, USA, 27–30 November 2005; p. 8.
54. Aziz, F.; Wilson, R.C.; Hancock, E.R. Backtrackless walks on a graph. *IEEE Trans. Neural Netw. Learn. Syst.* **2013**, *24*, 977–989. [CrossRef]
55. Johansson, F.; Jethava, V.; Dubhashi, D.; Bhattacharyya, C. Global graph kernels using geometric embeddings. In Proceedings of the International Conference on Machine Learning, Beijing, China, 22–24 June 2014; pp. 694–702.
56. Shervashidze, N.; Schweitzer, P.; Van Leeuwen, E.J.; Mehlhorn, K.; Borgwardt, K.M. Weisfeiler-lehman graph kernels. *J. Mach. Learn. Res.* **2011**, *12*, 2539–2561.
57. Bai, L.; Rossi, L.; Zhang, Z.; Hancock, E. An aligned subtree kernel for weighted graphs. In Proceedings of the International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 30–39.
58. Fröhlich, H.; Wegner, J.K.; Sieker, F.; Zell, A. Optimal assignment kernels for attributed molecular graphs. In Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 7–11 August 2005; pp. 225–232.
59. Hatcher, A. *Algebraic Topology*; Cambridge University Press: Cambridge, UK, 2002.
60. Dey, T.K. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*; Cambridge University Press: Cambridge, UK, 2006; Volume 23.
61. Shewchuk, J.R. Lecture Notes on Delaunay Mesh Generation. 1999. Available online: https://people.eecs.berkeley.edu/~jrs/meshpapers/delnotes.pdf (accessed on 10 January 2023).
62. Edelsbrunner, H.; Mücke, E.P. Three-dimensional alpha shapes. *ACM Trans. Graph. (TOG)* **1994**, *13*, 43–72. [CrossRef]
63. De Floriani, L.; Hui, A. Data Structures for Simplicial Complexes: An Analysis Furthermore, A Comparison. In Proceedings of the Symposium on Geometry Processing, Vienna, Austria, 4–6 July 2005; pp. 119–128.

64. Edelsbrunner, H.; Harer, J. Persistent homology-a survey. *AMS Contemp. Math.* **2008**, *453*, 257–282.
65. Kahle, M. Topology of random simplicial complexes: A survey. *AMS Contemprory Math.* **2014**, *620*, 201–222.
66. Zomorodian, A. Fast construction of the Vietoris-Rips complex. *Comput. Graph.* **2010**, *34*, 263–271. [CrossRef]
67. Boissonnat, J.D.; Guibas, L.J.; Oudot, S.Y. Manifold reconstruction in arbitrary dimensions using witness complexes. *Discret. Comput. Geom.* **2009**, *42*, 37–70. [CrossRef]
68. Burnham, K.P.; Anderson, D.R. Kullback–Leibler information as a basis for strong inference in ecological studies. *Wildl. Res.* **2001**, *28*, 111–119. [CrossRef]
69. Pérez-Cruz, F. Kullback–Leibler divergence estimation of continuous distributions. In Proceedings of the 2008 IEEE International Symposium on Information Theory, Toronto, ON, Canada, 6–11 July 2008; pp. 1666–1670.
70. Yu, S.; Mehta, P.G. The Kullback–Leibler rate pseudo-metric for comparing dynamical systems. *IEEE Trans. Autom. Control* **2010**, *55*, 1585–1598.
71. Potamias, M.; Bonchi, F.; Castillo, C.; Gionis, A. Fast shortest path distance estimation in large networks. In Proceedings of the 18th ACM Conference on Information and Knowledge Management, Hong Kong, China, 2–6 November 2009; pp. 867–876.
72. Panaretos, V.M.; Zemel, Y. Statistical Aspects of Wasserstein Distances. *Annu. Rev. Stat. Its Appl.* **2019**, *6*, 405–431. [CrossRef]
73. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
74. Kriege, N.M.; Giscard, P.L.; Wilson, R. On valid optimal assignment kernels and applications to graph classification. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 1623–1631.
75. Kudo, T.; Maeda, E.; Matsumoto, Y. An application of boosting to graph classification. *Adv. Neural Inf. Process. Syst.* **2004**, *17*, 1–8.
76. Ma, T.; Shao, W.; Hao, Y.; Cao, J. Graph classification based on graph set reconstruction and graph kernel feature reduction. *Neurocomputing* **2018**, *296*, 33–45. [CrossRef]
77. Wu, J.; He, J.; Xu, J. Net: Degree-specific graph neural networks for node and graph classification. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 406–415.
78. Yao, H.R.; Chang, D.C.; Frieder, O.; Huang, W.; Lee, T.S. Graph Kernel prediction of drug prescription. In Proceedings of the 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI), Chicago, IL, USA, 19–22 May 2019; pp. 1–4.
79. Parés, F.; Gasulla, D.G.; Vilalta, A.; Moreno, J.; Ayguadé, E.; Labarta, J.; Cortés, U.; Suzumura, T. Fluid communities: A competitive, scalable and diverse community detection algorithm. In Proceedings of the International conference on Complex Networks and Their Applications, Lyon, France, 29 November–1 December 2017; pp. 229–240.
80. Hua, B.S.; Tran, M.K.; Yeung, S.K. Pointwise convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 984–993.
81. Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R.R.; Smola, A.J. Deep sets. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 3391–3401.
82. 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, Honolulu, HI, USA, 21–26 July 2017; pp. 3693–3702.
83. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
84. Xie, S.; Liu, S.; Chen, Z.; Tu, Z. Attentional shapecontextnet for point cloud recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4606–4615.
85. Groh, F.; Wieschollek, P.; Lensch, H. Flex-convolution (million-scale point-cloud learning beyond grid-worlds). *arXiv* **2018**, arXiv:1803.07289.
86. Klokov, R.; Lempitsky, V. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 863–872.
87. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–14.
88. Shen, Y.; Feng, C.; Yang, Y.; Tian, D. Mining point cloud local structures by kernel correlation and graph pooling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4548–4557.
89. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic graph cnn for learning on point clouds. *Acm Trans. Graph. (TOG)* **2019**, *38*, 1–12. [CrossRef]
90. Atzmon, M.; Maron, H.; Lipman, Y. Point convolutional neural networks by extension operators. *arXiv* **2018**, arXiv:1803.10091.
91. Liu, Y.; Fan, B.; Xiang, S.; Pan, C. Relation-shape convolutional neural network for point cloud analysis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 8895–8904.