Cluster-based Data Reduction for Persistent Homology

Anindya Moitra, Nicholas O. Malott and Philip A. Wilsey
Dept. of EECS, University of Cincinnati, Cincinnati, OH 45221, USA
Email: moitraaa@mail.uc.edu, malottno@mail.uc.edu, philip.wilsey@uc.edu

***Abstract*—Persistent homology is used for computing topological features of a space at different spatial resolutions. It is one of the main tools from computational topology that is applied to the problems of data analysis. Despite several attempts to reduce its complexity, persistent homology remains expensive in both time and space. These limits are such that the largest data sets to which the method can be applied have the number of points of the order of thousands in $\mathbb{R}^3$. This paper explores a technique intended to reduce the number of data points while preserving the salient topological features of the data. The proposed technique enables the computation of persistent homology on a reduced version of the original input data without affecting significant components of the output. Since the run time of persistent homology is exponential in the number of data points, the proposed data reduction method facilitates the computation in a fraction of the time required for the original data. Moreover, the data reduction method can be combined with any existing technique that simplifies the computation of persistent homology. The data reduction is performed by creating small groups of *similar* data points, called nano-clusters, and then replacing the points within each nano-cluster with its cluster center. The persistence homology of the reduced data differs from that of the original data by an amount bounded by the radius of the nano-clusters. The theoretical analysis is backed by experimental results showing that persistent homology is preserved by the proposed data reduction technique.**

***Keywords*-topological data analysis; persistent homology; data reduction; k-means++; data mining**

I. Introduction

Topological Data Analysis (TDA) is an approach that focuses on studying the *'shape'* or topological structures of data in order to extract meaningful information. The ability of TDA to identify shapes despite certain deformations in the space renders it immune to noise and leads to discovering properties of data that are not discernible by conventional methods of data analysis [1], [2].

One of the principal methods for performing Topological Data Analysis is called persistent homology. Persistent homology can be informally defined as a process for computing topological features of data with increasing spatial resolutions. The input data for the computation of persistent homology is represented as a *point cloud*[1]. The output is a set of real number pairs, where each pair represents birth and death times of a topological feature. The pairs are usually plotted either as a set of lines, called *barcodes*, or as a set of points in the plane, called a *persistence diagram*.

The computation of persistent homology begins by associating a certain type of *complex* (simplicial, cubical, and CW complexes are some of the commonly used ones) to the point cloud. Unfortunately, even for data sets of moderate size, the size[2] of the complex becomes prohibitively large. For example, 3000 points sampled from the three-dimensional Stanford Dragon model [3] yields a filtered Vietoris–Rips complex (defined in Section II-C) of size 1.3×10^9 at a scale equal to the maximum distance between any two points in the data set [4]. This is because in the worst case, the Vietoris–Rips complex includes $N!/p!(N-p)!$ simplices of dimension p, where N is the number of points in the point cloud. This means that the number of simplices that constitute the complex can grow exponentially with the number of input data points.

The worst case time and space complexities of the computation of persistent homology are $O(N_K^3)$ and $O(N_K^2)$ respectively, where N_K is the total number of simplices in the filtration. In terms of the number of points N in the point cloud, the time complexity is $O(\exp(N))$. Although many optimizations of the algorithm have been proposed, the fastest one of them still has a run time of $O(N_K^{2.376})$. There is no known implementation to further improve the worst case time complexity [5]. For large data sets, this creates a major limitation in the computation of persistent homology.

A. Contribution

In this paper, we employ a data reduction technique that preserves the salient topological features of the data. The proposed technique enables the computation of persistent homology on a reduced version of the original input data without affecting significant components of the output. Since the run time of persistent homology is exponential in the number of data points, the proposed data reduction method facilitates the computation in a fraction of the time required for the original data. In other words, the method enables the computation of persistent homology on dramatically larger data sets than otherwise possible.

The data reduction is performed in two steps:

1) Create small groups of spatially *similar* data points, called *nano-clusters*. The similarity of the data points

[1]A point cloud $(P, dist)$ is a finite set P of N points, equipped with a distance function $dist$. P is assumed to be sampled from an underlying space $\mathbb{S}$.

[2]The size of a complex is represented in terms of its number of simplices.

is measured in terms of some, often Euclidean, distance function. The name "nano-clusters" is used due to their small sizes and large numbers (in most scenarios). The name also reflects the fact that nano-clusters are not intended to discover partitions in the data, which is the task in conventional data clustering problems.

2) Replace the points in a nano-cluster by their corresponding cluster center or centroid which is the arithmetic mean of the coordinates of the points in the nano-cluster.

Since the points in a nano-cluster are spatially local, replacing them by their centroid has little effect on the 'shape' of the data. While this reduction removes points and consequently some of the simplices, the topological features lost from these removals are primarily those with short lifespans that are mostly insignificant to the broader goal of identifying significant features in the point cloud. Compared to the cost of computing persistent homology, the creation of nano-clusters and their replacement by respective cluster centers is computationally very simple, and can be done in a single scan over the data. For the experiments described in this paper, the nano-clusters are created using the k-means++ clustering algorithm [6].

The data reduction technique using nano-clusters can be considered a preprocessing step where the data is reduced before feeding it to the persistent homology algorithm. Thus the technique can be combined with other methods that attempts to simplify the computation of persistent homology. The data reduction method can also be viewed as a sampling technique where the set of centriods represents the data points sampled from the original point cloud.

In this paper, an upper bound on the error introduced by the data reduction method is established. The analysis is backed by experimental results on real-world data sets.

B. Related Work

Our approach is closely related to the subsampling method developed by Chazal *et al* [7] that proposes to take n independent samples of size k from the initial point cloud. Either the average persistence landscape [8] from the n samples or the landscape of the closest sample to the original point cloud (according to Hausdorff distance) is computed. The method requires one to either compute persistent homology n times or select the closest sample on which persistent homology is to be computed. Accordingly, computing the average landscape is $O(n\ \exp(k))$ and persistent homology of the closest sample is $O(nkN + \exp(k))$.

In contrast, our approach consists in taking only one sample from the original data, thereby saving $(n-1)$ computations of persistent homology and the generation of multiple samples. The data reduction by k-means++ and the subsequent computation of persistent homology is $O(kN + \exp(k))$. In spite of being computationally inexpensive, our method produces similar output to that of [7] (as shown in Section IV) and preserves all significant topological features. Indeed, the effectiveness of k-means as a data reduction method has been emphasized in [9] in the context of k-Nearest Neighbor classification. An additional advantage of the k-means++ reduction is that it permits us to partition the data and restore interesting subregions of the original point cloud for recomputing persistent homology only on the subregions that define topologically significant features. This subregion restoration is being called *upscaling*. While not explored fully in this paper, the upscaling method will ultimately permit one to iteratively refine the barcode birth/death times when the data sizes prevent computing persistent homology on the entire point cloud.

The remainder of this paper is organized as follows. Section II presents some of the background on persistent homology. Section III explores the data reduction technique using nano-clusters and establishes an upper bound on the error introduced by its use. Section IV presents the experimental results on several different data sets. Section V discusses about the method of upscaling. Finally, we conclude the paper with some remarks in Section VI.

II. Background

The purpose of this section is to briefly introduce the basic ideas of persistent homology. For a visual representation, the reader may watch an introductory video [10] by Matthew Wright. More technical details are available in [11], [12], [13], [14].

Homology can be thought of as a way of counting properties such as the number of connected components [13], holes or loops, and voids of a topological space. Persistent homology is a process of computing the lifespans those topological properties through increasing spatial resolutions. Computing the homology of arbitrary topological spaces can be very difficult. The solution is to approximate the spaces by combinatorial structures called *complexes* for which homology can be computed algorithmically [4]. Simplicial, cubical, and CW complexes [15], [4] are examples of the commonly used complexes. Since the simplicial complex is probably the most widely used with a richer theoretical foundation than others [4], we examine simplicial complexes and their use in persistent homology in the following subsections.

A. Simplicial Complex

A *simplex* is a generalization of the geometric objects such as a triangle or a tetrahedron. For example, a point is a 0-dimensional simplex, or simply, a 0-simplex; an edge between two points is a 1-simplex; a triangular face is a 2-simplex; a tetrahedron is a 3-simplex, and so on. A simplicial complex is obtained by combining more than one simplices together in such a way that their intersection is also a simplex. Formally, a *simplicial complex* is a set K of

finite sets such that if $\sigma \in K$ and $\tau \subseteq \sigma$, then $\tau \in K$ [16]. If $\sigma \in K$ is constructed of $p+1$ points, *i.e.,* $|\sigma| = p+1$, then σ is called a p-simplex of dimension p. A p-simplex becomes a vertex, an edge, a triangle, or a tetrahedron for $p = 0, 1, 2,$ and 3 respectively.

B. Filtration

A subset $L \subseteq K$ is called a *subcomplex* if L itself is a simplicial complex. A *filtration* of a complex K is a sequence of nested subcomplexes $\emptyset = K_0 \subseteq K_1 \subseteq K_2 \subseteq ... \subseteq K_m = K$. A complex with a filtration is called a *filtered complex* $K_{\mathcal{F}}$.

C. Computation of Persistent Homology

The computation of persistent homology begins with associating a complex to the point cloud. There are several ways to associate a simplicial complex to a point cloud $(P, dist)$. The construction of a Vietoris–Rips complex is one of the most common ways of achieving this. For a real number $\epsilon \geq 0$, the Vietoris–Rips complex $V(P, \epsilon)$ at scale ϵ is defined as:

$$V(P, \epsilon) = \{\, \sigma \subset P \mid dist\,(x, y) \leq \epsilon \text{ for all } x, y \in \sigma \,\}$$

Thus, in a Vietoris–Rips complex any two points within a distance ϵ are connected. For $\epsilon_1 \leq \epsilon_2$, we have $V(P, \epsilon_1) \subseteq V(P, \epsilon_2)$. Therefore, if we consider increasing (or decreasing) values of the scale parameter ϵ, we obtain a filtration $K_{\mathcal{F}}$.

The topological structure of the filtration $K_{\mathcal{F}}$ changes with the scale parameter ϵ. At $\epsilon = 0$, all data points are disconnected. As ϵ increases, the points become connected to one another by edges. As more and more points are connected, existing connected components are merged into one another, holes or voids appear (or, are born) and eventually get filled (or, die). Persistent homology is a mechanism to track these changes as ϵ grows from 0 to a user-specified threshold. More specifically, persistent homology records the birth and death times of the connected components and holes as they appear and disappear with increasing ϵ. The difference between the death and birth times is the lifespan of a topological feature. The basic intuition of persistence is that the significant topological features have much longer lifespans than those of noise [17], [18].

The output of persistent homology is a set of pairs of real numbers $(\epsilon_{birth}, \epsilon_{death})$. A topological feature is born at $\epsilon = \epsilon_{birth}$, and dies at $\epsilon = \epsilon_{death}$. The pairs $(\epsilon_{birth}, \epsilon_{death})$ are shown as a set of lines or bars, called *barcodes*. As salient features persist longer than noise, long bars represent important features and short bars represent noise [13].

The set of pairs $(\epsilon_{birth}, \epsilon_{death})$ can also be plotted on a plane, with the horizontal and vertical axes representing the birth and death times of topological features respectively. This plot is called a *persistence diagram*. As significant features have longer lifespans, their death times are much greater than birth times. Therefore, the points that represent significant features lie far away from the 45° line that passes through the origin of the persistence diagram. Noise points, on the other hand, reside close to or on the 45° line.

III. Data Reduction Using Nano-Clusters

This section provides more details on the proposed data reduction method and the error introduced by it.

A. Creation of Nano-clusters

Nano-clusters are created using a standard k-means++ clustering algorithm [6]. The input to k-means++ algorithm is the original data set having N points. The number of points in the reduced data set is specified by the parameter k. The output of k-means++ is a set of k cluster centers that represent the centroids of the nano-clusters. If one selects any $k < N$, one obtains a reduced data set of k points from the original data set of N points.

Creation of nano-clusters and their substitution by respective centroids, essentially, constitute a 'knowledgeable' sampling mechanism. We call it 'knowledgeable' because the sampled centroids follow the 'shape' of the original data and preserve the topological notions of "closeness" of things in the data set. k-means++ seeks to minimize the maximum distance to any point from the centroid of a cluster. Moreover, the algorithm employs an intelligent mechanism to select initial cluster centers. k-means++ exhibits an improved accuracy in including the closest set of points in each nano-cluster, generating centroids along the shape of the data.

B. Error Introduced by Data Reduction

The error introduced in data reduction manifests itself as changes in the position of points on the persistence diagram (or equivalently, in the lifespan of bars on the barcodes). This subsection establishes a bound on the error that is represented as the bottleneck distance W_∞ [13] between the persistence diagrams of the original data and the reduced data.

The radius r of a nano-cluster is defined as the maximum distance to a point in the cluster from its centroid. Let r_{max} denote the maximum radius among all nano-clusters. For all practical purposes, both the original data $\mathbf{D}$ and the reduced data $\mathbf{C}$ can be considered as compact subsets of the same metric space $\mathbb{S}$ equipped with a distance function $dist$. If $\mathrm{Dg}_{\mathbf{D}}$ and $\mathrm{Dg}_{\mathbf{C}}$ are persistence diagrams resulting from $\mathbf{D}$ and $\mathbf{C}$ respectively, then, according to the stability theorem of persistence diagrams [19], [7]:

$$W_\infty(\mathrm{Dg}_{\mathbf{D}}, \mathrm{Dg}_{\mathbf{C}}) \leq 2H(\mathbf{D}, \mathbf{C})$$

where

$$H(\mathbf{D}, \mathbf{C}) = \max\left\{ \max_{x \in \mathbf{D}} \min_{c \in \mathbf{C}} dist(x, c), \max_{c \in \mathbf{C}} \min_{x \in \mathbf{D}} dist(x, c) \right\}$$

is the Hausdorff distance between $\mathbf{D}$ and $\mathbf{C}$. Here, $\max_{x \in \mathbf{D}} \min_{c \in \mathbf{C}} dist(x, c)$ is the maximum of the distances from data points to their nearest centroids. This distance is the maximum radius r_{max} among all the nano-clusters. On the other hand, $\max_{c \in \mathbf{C}} \min_{x \in \mathbf{D}} dist(x, c)$ is the maximum of the distances from the centroids to their closest data points. Let us denote this distance by l_{max}. Clearly, $r_{max} > l_{max}$. It follows that

$$H(\mathbf{D}, \mathbf{C}) = \max \left\{ r_{max},\ \ l_{max} \right\} = r_{max}$$

Hence,

$$W_\infty(\mathrm{Dg}_{\mathbf{D}}, \mathrm{Dg}_{\mathbf{C}}) \leq 2\, r_{max}. \tag{1}$$

Thus up to certain limits of data reduction, r_{max} will be much smaller than the lifespan of any significant feature of the data and the impact on the birth/death times of the barcodes of these significant features is bounded by r_{max}. It is apparent then that the data reduction has little to no impact on the goal of finding significant topological features of data using persistent homology. Of course, minor features of size less than r_{max} may be lost. That said, in general, the average cluster radius (r_{avg}) will be the bound for topological feature preservation. Thus, most features having lifespans of r_{avg} or more will be preserved (especially if r_{max} is an outlier size among the nano-clusters).

IV. Experimental Results

In this section, we describe and evaluate the data reduction technique through two sets of experiments. First, we reduce the instances in the point cloud with k-means++ to present run time and memory performance improvements for computing persistent homology. We examine the error in the reduced point cloud and persistence intervals, complementing the statement that this approach preserves the salient topological structures within the point cloud without significant impact on the barcodes of filtered Rips complexes. Second, we show comparisons of multiple triangulated mesh objects with reduced points through k-means++ to analyze the dissimilarity of the shapes with respect to persistent homology and the output landscapes. Analysis of the dissimilarity matrix provides experimental results detailing the topological differences between objects after significant reduction of the point cloud with k-means++. The results can provide a basis for more efficient computation of persistent homology for object recognition and comparison.

The first set of experiments were carried out on two selected data sets and the second set of experiments were evaluated on four triangulated mesh data sets. Programs were executed on an Intel(R) Xeon(TM) E5-2670 CPU @ 2.60GHz with 64GB of RAM. For each data set, we reduced the original data with k-means++. GUDHI library [20] was used to compute persistent homology of the resultant data sets. However, any other persistent homology library could be used. To reduce computation time, persistence barcodes were computed for topological features of dimensions 0, 1, and 2 only. See Section III for explanation of the generalization of shifts in persistence intervals in dimension n. The output lifespans were plotted as barcodes and persistence diagrams. Several levels of data reductions were compared for each data set.

A. Performance and Error Characterization

For the first set of experiments, run time data were collected for k-means++ and the GUDHI persistent homology algorithm. In the case of no data reduction, only the run time for persistent homology was recorded.

Characterization data were collected from the nano-clusters to relate the error introduced by the data reduction to the maximum and average radius of nano-clusters, denoted as r_{max} and r_{avg} respectively. These calculations give us the worst-case and average-case estimates for the error introduced by data reduction.

Two error index measures, the *Wasserstein* and *Bottleneck* distances, were evaluated for each output. The Wasserstein distance represents the total difference in paired mappings between two sets of intervals for each set of dimensional intervals. The Bottleneck distance is a variation of the Wasserstein distance that measures the length of the maximum difference in mappings between two intervals for each dimensional set. Both measures give a quantitative degree of change between the original and reduced barcodes.

The total run time on a data set is the sum of run times for k-means++ and persistent homology. The accuracy trade off of k-means++ as a pre-processing technique benefits the total run time as the reduction percentage is increased. Significant run time improvements can be attained from the pre-processing method even for small reduction percentages. The worst case time and space complexities of the computation are $O(n_K^3)$ and $O(n_K^2)$, respectively, showing the benefit of the data reduction method for large data sets.

Real-world data were used to demonstrate the technique on topological features where persistent homology is computationally complex. The first example is from the Gesture Phase Segmentation data available in the UCI machine learning repository. The data set describes a post-processed gesture research video recorded on a Microsoft Kinect sensor consisting of 1,743 instances with 32 attributes. The processed attributes measure vectorial and scalar velocities of the hands and wrists of the user. This data source has been previously used in SimBa [21] for topological data analysis along with several other machine learning and segmentation experiments. Run time performance and characterization data can be found in Table I. Results show a slight increase in the Wasserstein and Bottleneck distances with increased reduction amounts, however this value is larger relative to the scale of the original data set. No major collapsing of the point cloud occurs from the reductions performed.

Gesture					
Reduction %	r_{max}	r_{avg}	Wasserstein	Bottleneck	Runtime (s)
0	0.0	0.0	0.0	0.0	1483.43
10	0.0017	0.0002	0.0035	0.0024	1399.64
25	0.0045	0.0005	0.0034	0.0030	462.60
50	0.0106	0.0106	0.0017	0.0033	149.19
75	0.0193	0.0193	0.0051	0.0038	13.67
90	0.0461	0.0461	0.0145	0.0056	1.73
Camel					
Reduction %	r_{max}	r_{avg}	Wasserstein	Bottleneck	Runtime (s)
0	0.0	0.0	0.0	0.0	213.14
10	0.0137	0.0013	0.0014	0.0040	157.11
25	0.0193	0.0040	0.0060	0.0087	85.86
50	0.0270	0.0100	0.0186	0.0092	20.61
75	0.0440	0.0187	0.3820	0.3477	14.02
90	0.0866	0.0310	0.4180	0.3616	8.65

Table I
DIFFERENCES IN THE PERSISTENT DIAGRAMS AT PERCENTAGES OF NANO-CLUSTER REDUCTION.

The second set of results in Table I are from Camel data, a subsampling from the publicly available triangulated shapes database [22], consisting of 21,877 instances with 3 attributes describing a geometric camel shape in $\mathbb{R}^3$. Data sets from the triangulated shapes database have been previously used in [7] for subsampling methods to reduce data instances.

The original Camel point cloud was extracted from the camel-reference object file, saving only the vertices for the triangulated mesh. Subsampling of the original camel point cloud was performed with k-means++ to preserve the model's geometry. In the same method as the nano-cluster reduction technique described above, the data was clustered into 1,000 centroids prior to experimentation. Again, as shown by the small Wasserstein and Bottleneck distances, the results have nominal impact on the barcode computation.

The data reduction technique provides significant performance benefits while preserving the relevant topological features of the point cloud. This fact is evident from the run times shown in Table I. A speed up of 157 times is achieved with the Gesture data set at 90% reduction without introducing significant error as demonstrated by the Wasserstein and Bottleneck distances.

As noted in Section III-B, r_{max} can play a role in determining the error induced into the barcodes and persistence diagram from the data reduction. For noisy data sets, r_{max} can be skewed when outliers are assigned to clusters. The value contains information on the worst-case error that can result from the data reduction with k-means++ as a preprocessing technique, but does not quantify the additional error in the Wasserstein and Bottleneck distances attributed to the reduction of points, loss of connected components, and removal of short persistence intervals.

Reduction of the number of instances in the point cloud will remove short persistence intervals, as described in Section III. Nano-clustering will also alter the distance between points in the point cloud by no more than $2\,r_{max}$. Using r_{avg} to estimate the overall change in shape of the cluster provides a closer approximation of quantitative error introduced through k-means++ nano-clustering.

B. Dissimilarity Analysis of Persistence Landscapes

The data reduction technique provides performance benefits while preserving the significant topological features of the point cloud. This technique can be useful for faster classification of multidimensional objects and comparing the topological features between different point clouds. As the percentage of reduction continues to increase, an eventual loss of distinguishing features will occur. These features may be significant in classifying the differences between barcodes of different multidimensional objects. The increased performance for point cloud reduction provides considerable reason to explore at what levels of reduction objects become indistinguishable.

To analyze the effect of the reduction on classification of multidimensional objects, experiments were performed and analyzed through comparison of persistence landscapes. Landscapes are defined to characterize the relative locations and lengths of different barcodes discovered through persistent homology. Landscapes correspond directly to the output of persistent homology and give an alternative graphical representation of the topological features present in the point cloud.

Four data sets were chosen to demonstrate the reduction and dissimilarity of topological features: Camel, Flamingo, Lion, and Elephant. Each model represents a triangulated point cloud representation of an animal, from the publicly available triangulated shapes database [22]. Data sets from the triangulated shapes database have been previously used in [7] for validation of the subsampling method to simplify the computation of persistent homology. For each data set, k-means++ was used to reduce the total number of data points

between 500 points to 100 points. GUDHI library was used to compute the persistent homology at each reduction level.

The original Camel model consists of 21,877 points; Flamingo consists of 26,906 points, Lion consists of 4,999 points, and Elephant consists of 42,320. All of these triangulated mesh models are represented in $\mathbb{R}^3$. Due to the sizes of the larger point clouds (Flamingo and Elephant), the associated simplicial complexes cannot fit in the system memory. This constraint prohibits the computation of persistent homology on the original point clouds and requires an approximate or reduced analysis to be performed.

Visual inspection of the result of data reduction gives an immediate perspective of the method's benefit in preserving topological features. Figure 2 shows the original Flamingo point cloud with 26,906 points in $\mathbb{R}^3$. Reduction of the point cloud to 300 points is shown in Figure 3, showing the preservation of the general shape of the Flamingo triangulated mesh. Persistent homology on the reduced point clouds is feasible to generate barcodes with minimal error introduced into the output.

After generation of barcodes for each reduced point cloud, landscapes were calculated to compare and measure the dissimilarity of the objects with respect to the reduction levels and with respect to other models at the same reduction percentage. The dissimilarity is computed as L_∞ distance between pairs of persistence landscapes. A dissimilarity matrix was plotted to provide a visual representation of each comparison and analyzed for patterns within the reduction levels and the different objects being evaluated.

The Camel data set provides significant insight into the effects of data reduction. The Camel dissimilarity matrix is shown in Figure 1. The dissimilarity matrix shows a slight reduction in similarity as the number of points is reduced. For the Camel model, the value gradually increases as the number of points decreases, due to reduction through k-means++. This increase in the dissimilarity value signifies a larger change in the resultant barcodes at the 100 point reduction due to loss of significant features from reduction.

Comparing all of the models at different reduction levels shows how the objects can be distinguished from one another through persistent homology. In Figure 4, the objects show significant dissimilarity, enough to match and distinguish the different models used. The reduction method introduces some bounded error into the models as reductions increase while preserving the discrimination of features at greater reduction levels.

Previous studies on the subsampling method for persistent homology from [7] have examined the camel, elephant, flamingo, and lion model landscapes to generate a dissimilarity matrix at a single reduction level through subsampling. The method presented in this paper is similar to the subsampling technique described and the dissimilarity matrix over all models is qualitatively similar. However, since the dissimilarity matrix presented in [7] does not include quantitative values for each comparison, the method presented cannot be directly compared. Figure 4 can be utilized to qualitatively evaluate the previous subsampling method for dissimilarity of the triangulated mesh datasets.

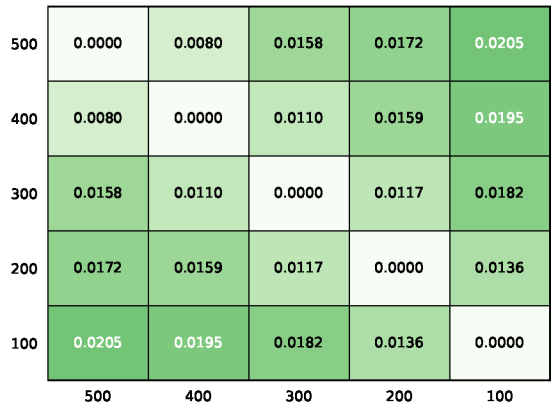

Figure 1. Dissimilarity matrix of the camel model at different reduction levels.

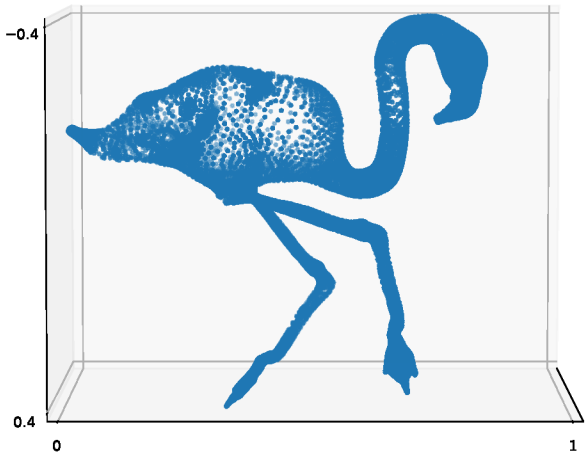

Figure 2. Plot of original Flamingo point cloud of 26,906 points.

Further analysis of error induced through the reduction technique with k-means++ nano-clusters on pattern matching and classification of data sets will provide a basis for the degree to which data can be reduced to preserve the relevant topological features within complex, high-dimensional point clouds. Recognition at increased performances is an obvious benefit to computation and feasibility of topological data analysis as a means to classify and distinguish high-dimensional patterns not found through traditional data analysis methods.

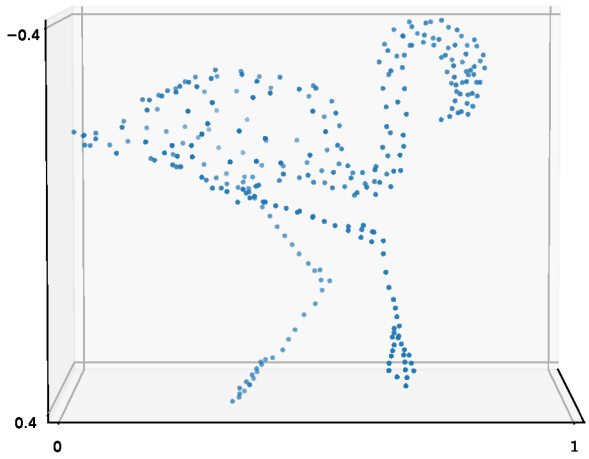

Figure 3. Plot of Flamingo point cloud reduced to 300 points.

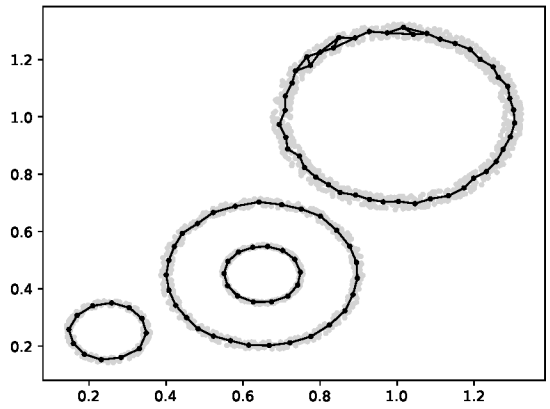

Figure 5. Boundaries identified at 90% reduction with ϵ=0.07

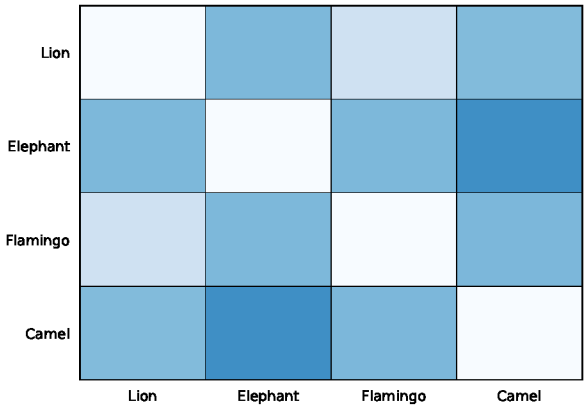

Figure 4. Dissimilarity of all models at 300 point reduction.

V. Upscaling and Iterative Refinement

Data reduction by nano-clusters is a highly effective mechanism to speedup the persistent homology computation and enable the use of persistent homology for the analysis of much larger data sets than previously possible. However, this speedup comes at the cost of small, but not fully predictable changes in the birth/death times of the barcodes of significant features. In some cases this may not be an acceptable trade off, however in other cases, it may be necessary to produce more accurate barcode birth/death times. Fortunately one could upscale the data in the region about the feature for which the "approximate" barcode is generated and then re-compute the persistent homology for that subregion of the original point cloud. That is, we can retrieve the nano-clusters on the boundary of the feature and reproduce all the points from the nano-clusters (upscale) on that boundary and recompute from that set of points. Should that upscaling produce too many points, an iterative refinement that reduces the points in that subregion using nano-clusters followed by upscaling and so on until the problem converges to increasingly more focused subregions of the point cloud that can be processed for persistent homology computations. Each barcode feature can be independently (and concurrently) upscaled and iterated over to quickly gain accurate barcode computations.

To gather the boundaries for upscaling, an initial identification needs to be carried out on the reduced data set. Data is replaced by an assigned centroid, reducing data by a significant percentage. Persistent homology can be evaluated relatively fast on the reduced data set. The boundary matrix is transformed into reduced row echelon form to determine constituent centroids of the boundary.

Figure 5 shows an example of the boundary resolution. The light gray points show the original point cloud of 2000 points. The data was reduced through k-means++ by 90%, leaving 200 data points shown as black dots. Edges are shown from simplex construction at $\epsilon = 0.07$. Persistent homology still distinguishes 4 holes in the reduced data set.

The boundary centroids are examined at a higher resolution to calculate a more accurate barcode. Each boundary centroid is upscaled using the k-means++ labels to expand to a larger point cloud. This can be accomplished in parallel, focusing on each feature realized by the reduced persistent homology as an independent point cloud. Ideally the persistent homology for each feature can be computed at full scale. For higher dimensional features the size of the entire data set is no longer a constraint. The data is partitioned into different features and upscaled boundaries that can be evaluated in parallel to accelerate the process of determining exact barcodes.

This method can be used in a single pass over the data but may be utilized to iteratively scale back to the largest point

cloud the persistent homology can be computed on with available system resources. A system with limited resources can continuously iterate through the data to extrapolate the actual barcodes without requiring large amounts of memory to store the entire simplicial complex. This method can lead to development of an algorithm that scales with resources of the computer and would be more useful for the computation of persistent homology on large data sets.

VI. Conclusions

The performance of the computation of persistent homology can be substantially improved by the use of nanoclusters to reduce the size of the input data set. This can be achieved with little to no impact on the significant topological features of a point cloud. Since the data reduction technique allows efficient and sufficiently accurate computation of persistent homology on much larger data sets than otherwise possible, the proposed method facilitates the application of persistent homology to solve the conventional problems of data mining or machine learning, such as classification and clustering. We have demonstrated the benefits of our approach in detecting dissimilarities of features in greatly reduced point clouds. Furthermore, the possibility of upscaling the data around discovered features should permit the recomputation of more accurate barcodes on subsets of the original data, should that be desirable.

Acknowledgment

Support for this work was provided in part by the National Science Foundation under grant ACI–1440420.

References

[1] P. Y. Lum, G. Singh, A. Lehman, T. Ishkanov, M. Vejdemo-Johansson, M. Alagappan, J. Carlsson, and G. Carlsson, "Extracting insights from the shape of complex data using topology," *Scientific Reports*, vol. 3, Feb. 2013.

[2] R. Ghrist, "Barcodes: The persistent topology of data," *Bulletin of the American Mathematical Society*, vol. 45, no. 1, pp. 61–75, 2008.

[3] S. U. C. G. Laboratory, "The stanford 3d scanning repository," 2014. [Online]. Available: https://graphics.stanford.edu/data/3Dscanrep

[4] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington, "A roadmap for the computation of persistent homology," *EPJ Data Science*, vol. 6, no. 1, Aug. 2017.

[5] K. Mischaikow and V. Nanda, "Morse theory for filtrations and efficient computation of persistent homology," *Discrete & Computational Geometry*, vol. 50, no. 2, pp. 330–353, 2013.

[6] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '07. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[7] F. Chazal, B. T. Fasy, F. Lecci, B. Michel, A. Rinaldo, and L. Wasserman, "Subsampling methods for persistent homology," in *International Conference on Machine Learning (ICML 2015)*, Lille, France, Jul. 2015. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01073073

[8] P. Bubenik, "Statistical topological data analysis using persistence landscapes," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 77–102, Jan. 2015.

[9] S. Ougiaroglou and G. Evangelidis, "A simple noise-tolerant abstraction algorithm for fast k-nn classification," in *Hybrid Artificial Intelligent Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 210–221.

[10] M. Wright, "Introduction to persistent homology," (last viewed Jan 2018). [Online]. Available: https://www.youtube.com/watch?v=2PSqWBIrn90

[11] F. Chazal and B. Michel, "An introduction to topological data analysis: fundamental and practical aspects for data scientists," *ArXiv e-prints*, Oct. 2017.

[12] H. Edelsbrunner and J. Harer, "Persistent homology – a survey," *Surveys on Discrete and Computational Geometry*, vol. 453, pp. 257–282, 2008.

[13] ——, *Computational Topology, An Introduction*. American Mathematical Society, 2010.

[14] R. Ghrist, *Elementary Applied Topology*. Createspace, 2014.

[15] R. Forman, "Morse theory for cell complexes," *Advances in Mathematics*, vol. 134, no. 1, pp. 90–145, 1998.

[16] A. Zomorodian, "Fast construction of the vietoris–rips complex," *Computer and Graphics*, pp. 263–271, 2010.

[17] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological persistence and simplification," in *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, ser. FOCS '00. Washington, DC, USA: IEEE Computer Society, 2000.

[18] A. Zomorodian and G. Carlsson, "Computing persistent homology," *Discrete & Computational Geometry*, vol. 33, no. 2, pp. 249–274, Feb. 2005.

[19] F. Chazal, V. de Silva, M. Glisse, and S. Oudot, "The structure and stability of persistence modules," *arXiv preprint arXiv:1207.3674*, 2012.

[20] C. Maria, J.-D. Boissonnat, M. Glisse, and M. Yvinec, "The gudhi library: Simplicial complexes and persistent homology," INRA, Tech. Rep. RR-8548, 2014. [Online]. Available: https://hal.inria.fr/hal-01005601v2

[21] T. K. Dey, D. Shi, and Y. Wang, "Simba: An efficient tool for approximating rips-filtration persistence via simplicial batch-collapse," *24th Annual European Symposium on Algorithms (ESA 2016)*, 2016.

[22] R. W. Sumner and J. Popovic, "Mesh data from deformation transfer for triangle meshes," 2004. [Online]. Available: https://people.csail.mit.edu/sumner/research/deftransfer/data.html