Stochastic Multiresolution Persistent Homology Kernel

Xiaojin Zhu, Ara Vartanian, Manish Bansal, Duy Nguyen, Luke Brandl
Department of Computer Sciences, University of Wisconsin–Madison
1210 W. Dayton Street, Madison, Wisconsin, USA 53706

Abstract

We introduce a new topological feature representation for point cloud objects. Specifically, we construct a Stochastic Multiresolution Persistent Homology (SMURPH) kernel which represents an object's persistent homology at different resolutions. Under the SMURPH kernel two objects are similar if they have similar number and sizes of "holes" at these resolutions. Our multiresolution kernel can capture both global topology and fine-grained topological texture in the data. Importantly, on large point clouds the SMURPH kernel is more computationally tractable compared to existing topological data analysis methods. We demonstrate SMURPH's potential for clustering and classification on several applications, including eye disease classification and human activity recognition.

1 Introduction and Background

There has been growing interest in bringing topological data analysis (TDA) into machine learning [Bubenik, 2015; Chazal *et al.*, 2015b; Reininghaus *et al.*, 2015; Ahmed *et al.*, 2014; Li *et al.*, 2014; Chazal *et al.*, 2015a; Pachauri *et al.*, 2011]. However, no method exists that is simultaneously rich in topological information, efficient in computation, and easy to plug into a machine learning system. We take a step in this direction by proposing a Stochastic MUltiResolution Persistent Homology (SMURPH) kernel. Our kernel can be viewed as a topological feature extractor for machine learning that compares the number and sizes of "holes" in two point clouds. Unlike prior TDA methods, SMURPH captures both the global topology and fine-grained "topological texture" in point cloud objects. Using SMURPH for machine learning requires little background in topology, as it just produces a kernel matrix.

Persistent homology is a TDA method for studying homology classes, or "topological holes." A thorough tutorial is out of scope of this paper and can be found in e.g. [Edelsbrunner and Harer, 2010; Nanda and Sazdanović, 2014; Carlsson, 2009; Zhu, 2013]. We will focus on first-order homology over Z_2 (binary) coefficients. For this purpose the following background knowledge suffices. On a point cloud X in $\mathbb{R}^d$ we construct an ϵ-hypergraph by creating edges (known as 1-simplices) between all pair of points $x_i, x_j \in X$ within distance ϵ. We also create triangle faces (2-simplices) among all x_i, x_j, x_k whose pairwise distances are within ϵ. This hypergraph is known as a simplicial complex. Following the terminology of [Edelsbrunner and Harer, 2010] we define a first-order homology group whose linearly independent generators represents holes (cycles that are not filled in by triangles) in the graph.

Now imagine we repeat this hypergraph construction separately for all thresholds $\epsilon > 0$. This series of hypergraphs is called a *Vietoris-Rips* filtration. As ϵ increases more edges and triangles will be created. The homology group changes by gaining or losing generators as holes are born (when a cycle is established) at some ϵ_1 and die (the cycle is filled) at some $\epsilon_2 \geq \epsilon_1$. Persistent homology tracks such birth and death events at critical ϵ values. This information is traditionally stored as a persistence diagram (PD) in the TDA literature. Intuitively, a PD is a multiset of N points (b_i, d_i) in 2D for the birth and death time (i.e. ϵ threshold) of the N holes encountered during Vietoris-Rips filtration.

The space of PDs is a metric space under the Wasserstein distance but not a vector space (hence not a Hilbert space). We instead build our kernels on the recently-proposed persistence landscape (PL) representation, which is a benign function space [Bubenik, 2015]. To obtain the PL from PD, one first rotates the PD so that the diagonal becomes the x-axis, and scale it by $1/\sqrt{2}$. In the new coordinate system, a (birth, death) pair (b_i, d_i) takes the coordinate (s_i, t_i) where $s_i = (b_i + d_i)/2, \quad t_i = (d_i - b_i)/2$ for $i = 1 \ldots N$. One defines a "tent function" Λ_i at each (s_i, t_i): $\Lambda_i(s) = \max(t_i - |s - s_i|, 0), \quad s \in \mathbb{R}$. The PL is a collection of piecewise linear functions $\mathbb{N} \times \mathbb{R} \mapsto \mathbb{R}$ indexed by level l: $\lambda(l, s) = \mathrm{lmax}_{i=1}^{N} \Lambda_i(s)$ where lmax returns the lth largest value in the set, or 0 if the set has less than l items.

2 The SMURPH Kernel

We will define three kernels on n point cloud objects $X_1 \ldots X_n$. The first kernel $K^\flat$ captures global persistent homology but does not scale to large point clouds. The second kernel $K^\sharp$ introduces multi-resolution but also does not scale. The SMURPH kernel K solves the scalability issue of $K^\sharp$ with Monte Carlo sampling.

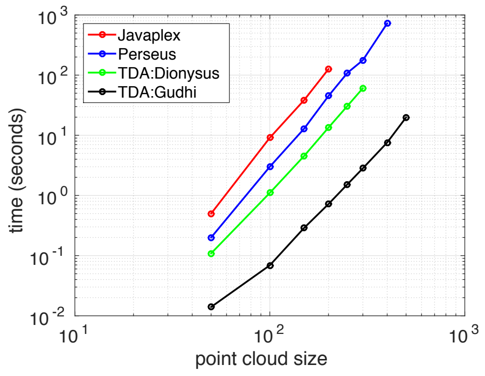

Figure 1: Time to compute a PD vs. point cloud size, log-log scale

2.1 A Global Persistent Homology Kernel $K^\flat$

For any fixed level l, the lth landscape function $\lambda(l,s) \in L^2$ and is 1-Lipschitz. We define the inner product between two PL functions λ, λ' by

$$\langle \lambda, \lambda' \rangle = \sum_l \int_{-\infty}^{\infty} \lambda(l,s)\lambda'(l,s)ds. \tag{1}$$

We will use $|X|$ to denote the number of points in point cloud X. The summation over level l is finite, since the maximum nonzero level in a PL is upper bounded by $\max(|X|,|X'|)$. It has been noted that this space of PL functions is a Hilbert space [Bubenik, 2015]. Therefore, we can define a positive definite topological kernel $K^\flat$ between two point clouds X, X' via the inner product of their PL functions $\lambda_X, \lambda_{X'}$:

$$K^\flat(X,X') = \langle \lambda_X, \lambda_{X'} \rangle. \tag{2}$$

While $K^\flat$ captures global topological information, there is a major roadblock in using it for data analysis. Figure 1 shows the time to compute PD, the building block of PL and $K^\flat$, versus the point cloud size $|X|$. We compared several popular TDA software: JavaPlex [Tausz *et al.*, 2011], Perseus [Nanda, 2013], the R TDA package [Fasy *et al.*, 2014] with Dionysus [Morozov, 2007] and GUDHI [Maria, 2014] solvers, respectively. The reported time is in seconds on a typical desktop computer for the software to compute 1^{st} persistence diagram. The slopes of these log-log time curves suggest that computation time scales as $|X|^3$ to $|X|^4$ for 1^{st} persistent homology. Objects with more than a few hundred points are computationally prohibitive.

A naive solution to the scalability issue is to sample b points from X to form a sparser point cloud $b(X)$, as suggested in [Chazal *et al.*, 2015b; Chazal *et al.*, 2015a]. Repeating the process s times, we obtain point clouds $b_1(X) \dots b_s(X)$. A bootstrapping PL function can then be computed as $\bar{\lambda}^{sb} = \frac{1}{s}\sum_j \lambda_{b_j(X)}$, where the average applies to each PL level separately.

Unfortunately, this bootstrap procedure introduces a loss of resolution. As an example, consider the two point clouds in Figure 2. At the global level (a) they look the same, but zooming in (b) we notice that the top one is made of numerous small rings while the bottom one is simply an annulus.

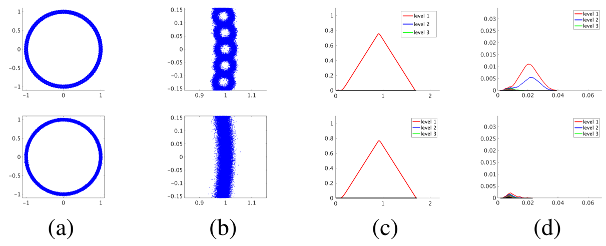

Figure 2: (a) top: the ring-of-rings dataset, bottom: the annulus dataset. (b) zoom. The two datasets look identical in their global PL $\bar{\lambda}_r^{msb}$ with $r = \mathrm{diam}(X) = 2$ in (c), but can be distinguished at a finer resolution $r = 0.06$ in (d).

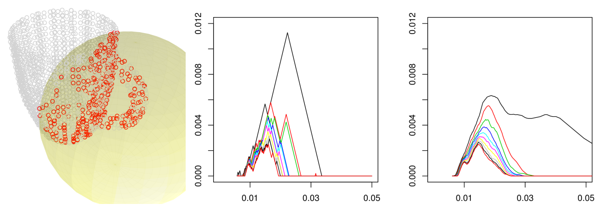

Figure 3: A mug point cloud inside a ball, its persistence landscape $\lambda_{\mathbb{B}(c,r)}$, and its average persistence landscape $\bar{\lambda}_r^{msb}$ over $m = 20, s = 1$ samples.

Subsampling globally (c) ignores this difference – justifiably so if one is only concerned with the overall shape of λ_X, since the finer resolution only contributes minimally to the global PL. Nonetheless, this difference in "topological texture" may be important for classifying or clustering. For better topological data analysis, we need the ability to identify differences in localized topology.

2.2 A Multiresolution Kernel $K^\sharp$

Inspired by multiresolution data analysis, especially the work on local relative homology in [Ahmed *et al.*, 2014], we propose a multiresolution persistent homology description of X to compute localized topology like those in Figure 2(d). To this end consider balls of a particular radius r, which we call the resolution. Let $\mathbb{B}(c,r) = \{x \in X : \|x - c\| \le r\}$ be the set of points in the ball centered at c with radius r. Let $\lambda_{\mathbb{B}(c,r)}$ be the corresponding PL function computed by performing a filtration on $\mathbb{B}(c,r)$. As an example, Figure 3(left) shows a 3D point cloud of a mug. The ball is shown in yellow, $\mathbb{B}(c,r)$ shown in red[1], and $\lambda_{\mathbb{B}(c,r)}$ in Figure 3(mid).

We draw centers c from a probability distribution $\mathcal{P}_C$. In this paper, we assume $\mathcal{P}_C$ is supported on X; that is, each ball must center on one of the data points in X. A simple choice of $\mathcal{P}_C$ is the uniform distribution over points in X, though a domain expert can design other $\mathcal{P}_C$ to emphasize certain interesting regions. Our quantity of interest is the expected PL function at resolution r:

$$\pi_r := \mathbb{E}_{c\sim\mathcal{P}_C}[\lambda_{\mathbb{B}(c,r)}]. \tag{3}$$

[1]Actually a subset of $\mathbb{B}(c,r)$, see section 2.3

When $\mathcal{P}_C$ is the uniform distribution over X, this quantity is simply $\pi_r = \frac{1}{|X|}\sum_{c\in X}\lambda_{\mathbb{B}(c,r)}$. Let the diameter of X be $\mathrm{diam}(X) = \max_{i,j\in X}\|i-j\|$. Noting that if $r \geq \mathrm{diam}(X)$, $\mathbb{B}(c,r) = X$ for all $c \in X$, we can establish that $\forall r \geq \mathrm{diam}(X)$, $\pi_r = \lambda_X$. Thus $r \geq \mathrm{diam}(X)$ recovers the global PL function. To obtain topological information of X at finer resolutions, we consider L decreasing radii $r_1 > \ldots > r_L$ and compute the corresponding expected PL functions $\pi_{r_1}, \ldots, \pi_{r_L}$.

Definition 1 *The multiresolution persistent homology representation of X at radii $r_1 \ldots r_L$ is $(\pi_{r_1}, \ldots, \pi_{r_L})$.*

We now have all the ingredients to define a multiresolution persistent homology kernel $K^\sharp$. Specifically, we define homology kernel $K^\sharp$ between two point cloud objects X and X' as

$$K^\sharp(X, X') = \sum_{i=1}^{L} w_i\langle\pi_{r_i}, \pi'_{r_i}\rangle \tag{4}$$

where $w_1, \ldots, w_L$ are nonnegative weights to combine different resolutions. A particularly useful weighting scheme is to set

$$w_i = (r_1/r_i)^3,\ i = 1\ldots L. \tag{5}$$

To understand this scheme, imagine two holes both born at $b = 0$ and die at $d = r_1$. The inner product (1) between their PLs is $r_1^3/12$. Now imagine two other holes both with $b = 0, d = r_i$. Their inner product is $r_i^3/12$. The weighting scheme thus scales up the finer resolution so that all resolutions contribute equally to the kernel.

However, the kernel $K^\sharp$ (4) is defined without regard to computation. In fact $K^\sharp$ is more costly to compute than $K^\flat$ due to two issues: 1. The expectation inside π requires enumerating all possible centers $c \in X$; 2. $\mathbb{B}(c,r)$ may still contain many points, making persistent homology software slow (recall Figure 1). We address both issues with sampling next.

2.3 The SMURPH Kernel K

The procedure below is carried out independently at each resolution r. We sample m centers $c_1 \ldots c_m \sim \mathcal{P}_C$. For each center c, we generate $j = 1\ldots s$ bootstrap samples within the ball $\mathbb{B}(c,r)$. Each bootstrap sample $b_j(c,r)$ consists of b points sampled with replacement from $\mathbb{B}(c,r)$. Instead of computing $\lambda_{\mathbb{B}(c,r)}$, we compute $\lambda_{b_j(c,r)}$. The value of b is chosen with computation speed in mind, so that state-of-the-art persistent homology software can finish in a reasonable amount of time. We define *average persistence landscape* (APL), an estimator of π_r, as follows:

$$\bar{\lambda}_r^{msb} = \frac{1}{ms}\sum_{i=1}^{m}\sum_{j=1}^{s}\lambda_{b_j}(c_i, r). \tag{6}$$

The superscripts msb remind the reader that $\bar{\lambda}_r^{msb}$ is subject to three kinds of randomness: m balls, s bootstraps per ball, b points per bootstrap.

To illustrate, we go back to Figure 2. Both the ring-of-rings point cloud and the annulus point cloud in (a) contain one million points. This poses no computational difficulty for $\bar{\lambda}_r^{msb}$ in (c) and (d) if we choose e.g. $b = 300$, which is easily handled by state of the art TDA software, and $m = 20, s = 1$. As another example, Figure 3(right) shows the average persistence landscape on the mug. The most significant persistent homology class (due to the handle) survives the averaging, even though we do not expect every random ball to contain the handle.

Definition 2 *The stochastic multiresolution persistent homology representation of X at radii $r_1 \ldots r_L$ is $(\bar{\lambda}_{r_1}^{msb}, \ldots, \bar{\lambda}_{r_L}^{msb})$.*

Finally, we define the Stochastic Multi-Resolution Persistent Homology (SMURPH) kernel as:

$$K(X, X') = \sum_{i=1}^{L} w_i\langle\bar{\lambda}_{r_i}^{msb}(x), \bar{\lambda}_{r_i}^{\prime msb}\rangle. \tag{7}$$

Note that K is stochastic due to sampling but given the samples it is a positive semi-definite kernel matrix. During test time when one needs to compute the kernel $K(X_i, X^*)$ between a training object X_i and a new point cloud object X^*, it is important that one uses the same representation $\left(\bar{\lambda}_{r_1}^{msb}, \ldots, \bar{\lambda}_{r_L}^{msb}\right)$ for X_i during training to ensure that the kernel matrix is well-defined.

Algorithm 1 specifies the computation of SMURPH kernel K. If we take the state-of-the-art persistent homology time complexity to be $O(b^3)$ as indicated by Figure 1, the complexity of Algorithm 1 is $O(nLmsb^3 + n^2)$. We also note that the inner product (1) can be efficiently computed in closed-form since PL is piecewise linear as a consequence of lmax over tent functions. In practice, computing each object's stochastic multiresolution persistent homology representation $(\bar{\lambda}_{r_1}^{msb}, \ldots, \bar{\lambda}_{r_L}^{msb})$ takes a few seconds for modest b in the hundreds.

Algorithm 1 SMURPH Kernel

```
Input: $n$ point cloud objects $X_1, \ldots X_n$
Parameters: radius scheme $(r_1, \ldots, r_L)$, kernel weight scheme $(w_1, \ldots, w_L)$, center distribution $\mathcal{P}_C$, number of centers $m$, bootstrap sample size $b$, number of bootstraps $s$, filtration $F$.
for object $X \in \{X_1 \ldots X_n\}$ do
  for resolution $r \in \{r_1 \ldots r_L\}$ do
    for center $i = 1 \ldots m$ do
      Sample $c_i \sim \mathcal{P}_C$
      for bootstrap $j = 1 \ldots s$ do
        Sample $b$ points within $\mathbb{B}(c_i, r)$ to form $b_j(c_i, r)$
        Apply filtration $F$ on $b_j(c_i, r)$ to compute PL $\lambda_{b_j}(c_i, r)$
      end for
    end for
    $\bar{\lambda}_r^{msb} = \frac{1}{ms}\sum_{i=1}^{m}\sum_{j=1}^{s}\lambda_{b_j}(c_i, r)$
  end for
  Represent object $X$ by $\left(\bar{\lambda}_{r_1}^{msb}, \ldots, \bar{\lambda}_{r_L}^{msb}\right)$
end for
Define the $n \times n$ resolution-$r$ kernel matrix as $K_r(X_i, X_j) = \langle\bar{\lambda}_r^{msb}(X_i), \bar{\lambda}_r^{msb}(X_j)\rangle$
Output: SMURPH kernel matrix $K = \sum_{i=1}^{L} w_i K_{r_i}$
```

For theoretical consideration, define the $n \times n$ SMURPH kernel matrix $K = [K_{ij}]$ where $K_{ij} = K(X_i, X_j)$ for

$i, j = 1, \ldots, n$. Similarly, we define $K^\sharp = \left[K^\sharp_{ij}\right]$. Following the technique in [Chazal *et al.*, 2015a; 2013], we can show that K approximates $K^\sharp$: under mild conditions $\mathbb{E}||\pi_r - \bar{\lambda}_r^{msb}||_\infty \leq \mathcal{O}\left[\left(\frac{\log b}{b}\right)^{1/\beta}\right]$ for some constant β. This means that $\bar{\lambda}_r^{msb}$ is an asymptotically unbiased estimator of π_r. Furthermore, we can bound $\mathbb{E}||K^\sharp - K||_{\max}$ where $||.||_{\max}$ is the max matrix norm. If s, m are of the order $\left(\frac{b}{\log b}\right)^{2/\beta}$, then $\mathbb{E}||K^\sharp - K||_{\max} \leq \mathcal{O}\left[\left(\frac{\log b}{b}\right)^{1/\beta}\right]$.

3 Experiments

We present three applications in clustering and classification to demonstrate the potential of the SMURPH kernel.

3.1 Pots and Pans

We demonstrate SMURPH kernel's ability to embed point cloud objects in a meaningful way based on persistent homology features. The dataset consists of 41 point cloud kitchen utensils (e.g. pans, cups, bottles, knives) [Neumann *et al.*, 2013]. We compute SMURPH kernel matrix using a radius of $r = 0.1$, $m = 20$ centers per point cloud, $s = 1$ samples per center, and a budget of $b = 350$ points per sample.

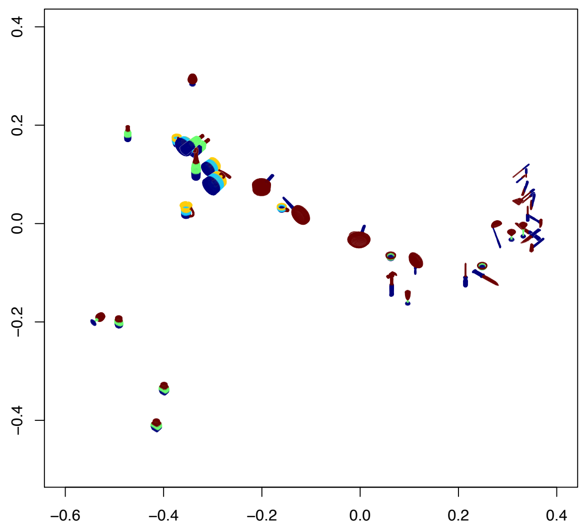

Figure 4: Kernel PCA embedding with SMURPH kernel matrix on the pots and pans dataset

We embed the 41 objects in 2D using kernel PCA on the SMURPH kernel matrix. For better visualization we draw each object centered on its embedding coordinates, see Figure 4. All objects are drawn at the same scale. We identify three salient groups with similar persistent homology in this embedding: (1) mugs with a handle, pots with handles, and long bottles near $(-0.4, 0.2)$; (2) ladle, knifes, screw drivers near $(0.3, 0)$; (3) small cans near $(-0.4, -0.4)$. We now explain why the grouping is topologically meaningful.

Most objects in the group (1) have handles. We have shown one such mug in Figure 3. Another mug is shown in Figure 5(left). Colors are solely for better visualization. Handles produce large holes in Vietoris-Rips filtration, whose effect is preserved in the average persistence landscape as a tall and long hump. Recall the SMURPH kernel is computed as inner products between these APL functions. The grouping in kPCA embedding space reflects the overall similarity between APL functions. Therefore, objects with handles are grouped close.

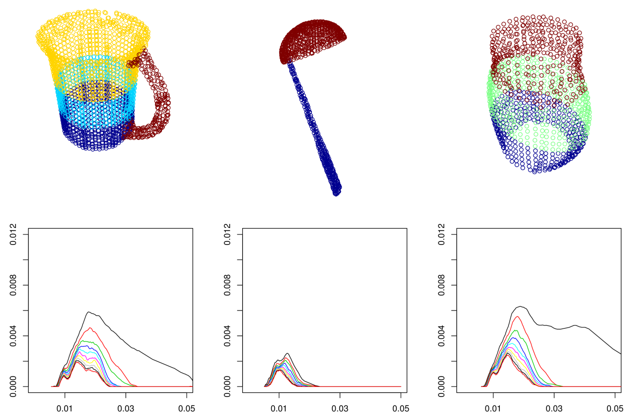

Figure 5: Three representative point clouds from the Pots and Pans dataset and their average persistence landscape $\bar{\lambda}_r^{msb}$.

In contrast, objects in group (2), such as the ladle shown in Figure 5(middle), have no intrinsic holes. Their average persistence landscape is almost flat and distinct from group (1). Note their APL is not exactly the zero line – this is an artifact of sampling. To see why, imagine a regular 2D grid of points in a ball $\mathbb{B}(c, r)$. SMURPH samples b points from the ball. When b is small relative to the number of points in the ball, spurious holes will be created during Vietoris-Rips filtration. In practice, however, this artifact has limited effect and does not prevent kPCA from producing meaningful embedding.

Finally, group (3) reveals an interesting property of SMURPH. First, the three cans are defective and missing the bottom in their point cloud (Figure 5 right), making them similar to the wine glass at $(-0.6, -0.2)$. Second, when SMURPH samples from a ball centered near the bottom, the ball would contain the whole can except the cap. The points in the ball then form a cylinder, which has a large hole. This homology feature *will* enter APL. In other words, SMURPH reflects both the overall topological structure of the object and that of its parts. We call the latter topological texture.[2] The topological texture of the cans and the wine glass has its own distinct APL signature, and is responsible for group (3).

[2]Such topological texture is also responsible for the long bottles, which has no intrinsic 1st order holes, to enter group (1).

3.2 Drusen Detection in Fundus Photography

Drusen are small yellowish hyaline deposits that develop beneath the retinal pigment epithelium. While drusen are natural phenomena that occur with aging, The presence of numerous, larger drusen is predictive of macular degeneration. Some drusen sites form small holes in skeletonized images (Figure 6 right). This suggests SMURPH kernel can help in classifying drusen vs. non-drusen images.

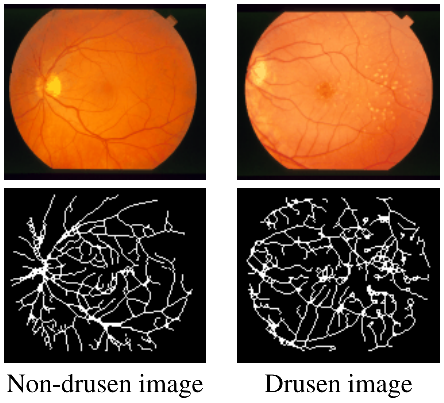

Figure 6: Fundus photographs and the point clouds after skeletonizing.

Data. Our sample consists of 67 retinal images, collected as part of the STARE [Hoover and Goldbaum, 2013] project. Of these, 35 have been labeled by experts as images without drusen and 32 as images with drusen. We pre-process the images by first binarizing (setting all non-zero pixel values to 1) and then skeletonizing with MATLAB's `bwmorph`. Our point cloud is then the coordinates of the nonzero pixels of this skeletonized image. To compute SMURPH kernel, we sample patch centers with a uniform distribution over the point cloud. We choose a resolution level of $r = 8$ and $m = 50, s = 1, b = 300$.

Method. We compare SMURPH to two baselines: linear kernel and radial basis function (RBF) kernel. While more sophisticated computer vision processing can undoubtedly enhance the baselines, our goal is to demonstrate the potential of SMURPH without heavy feature engineering. We tune all parameters (regularization C for SVM and bandwidth σ for RBF kernel) using an inner cross validation (CV) inside the outer CV training portion. The tuning grid is $C \in \{2^{-7}, 2^{-6}, \dots, 2^7\}$ and $\sigma \in \{10^0, 10^1, \dots, 10^3\}$. We feed the kernels to an SVM (kernlab [Karatzoglou *et al.*, 2004]) for classification and report 5-fold CV accuracy.

Results. Table 1 summarizes our results. SMURPH kernel outperforms the linear and RBF kernel baselines, demonstrating that SMURPH captures topological features that help to detect drusen. To assess the significance of our result, we perform a t-test over the accuracy of our classifiers across the CV folds, testing the accuracy of the SMURPH kernel against the RBF kernel and the linear kernel. As reported in Table 1, for either baseline we can reject the null hypothesis that accuracy does not differ at the 0.05 level.

Kernel	Parameters	Accuracy	p-value
SMURPH	$C = 32$	70.2%	-
RBF	$C = 2, \sigma = 10$	52.2%	0.040
Linear	$C = 2$	43.3%	0.007

Table 1: 5-fold cross validation accuracy of SVM classifiers on Fundus data set

3.3 Human Activity Recognition

We now demonstrate SMURPH kernel's ability to handle point clouds from a state space. In an unsupervised learning setting we show how kernel PCA embeds the n objects; in a supervised learning setting we will use activity as the class label and perform classification.

Data. The "daily and sports activities" data set [Altun *et al.*, 2010] contains sensor data of several everyday activities, among which we choose five representative ones: sitting (A1), walking (A9), running (A12), jumping (A18), and playing basketball (A19). Each activity is performed by 8 people in their own style for five minutes. Each person's data is a time series measured at 25 Hz, for a total of $5*60*25 = 7500$ measurements. Each measurement is 45-dimensional: 5 sensor units placed on torso, right arm, left arm, right leg, and left leg; each sensor unit contains x,y,z accelerometers, x,y,z gyroscopes, and x,y,z magnetometers.

We treat each activity-person combination as an object for a total of $n = 5 * 8 = 40$ objects. Each object contains 7500 points. Each point has a time stamp and a 45-dimensional measurement. No processing is performed on the measurements.

Filtration with Timeline. Since each object here is a time series, we present a special filtration design which is of independent interest. A similar filtration has been used to model sequences of natural language paragraphs [Zhu, 2013]. Let the point cloud object be $X = \{x = (t, \mathbf{z})\}$ where t is the time stamp and $\mathbf{z}$ is the measurement vector at time t. Our filtration has two unique steps:

(1) The ball is defined by t, not by $\mathbf{z}$. That is, p is the uniform distribution over time stamps in X. Once we sample a center in time: $c \sim P_t$, we define the ball $\mathbb{B}(c, r) = \{x = (t, \mathbf{z}) \in X : \|t - c\| \leq r\}$, which is in fact a time interval.

(2) We add a "timeline" (a set of edges) before Vietoris-Rips filtration begins. Within $\mathbb{B}(c, r)$, we randomly sample b points $x_1 \dots x_b$. Sort these points by time so that $t_1 < \dots < t_b$. We create the timeline by connecting points (x_i, x_{i+1}) adjacent in time for $i = 1 \dots b - 1$. The Vietoris-Rips filtration then proceeds as usual but uses the distance metric on $\mathbf{z}$. The final complex is the union of the timeline and the filtration. All in all, the timeline acts as a pre-existing temporal skeleton to help enhance the filtration. We compute a $b \times b$ distance matrix D where $D_{i,j} = \|z_i - z_j\|$ and set $D_{i,i+1} = 0$ for $i = 1 \dots b - 1$. Even though D no longer satisfies the triangle inequality it can still serve as a filtration function.

Unsupervised kernel PCA Results. We use two resolution levels: $r_1 = 125$ and $r_2 = 25$ which correspond to 10-second and 2-second intervals. We set the parameters $m = 10, s = 1, b = 100$. We compute the 40×40

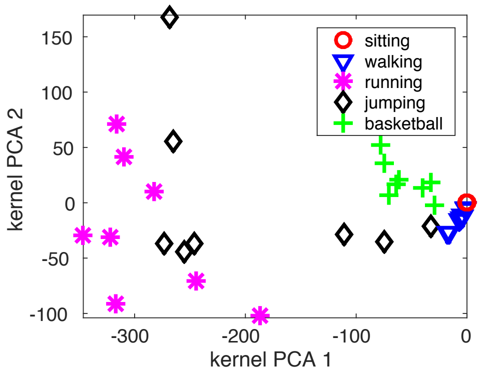

Figure 7: Kernel PCA with the SMURPH kernel on eight people $\times$ five activities

SMURPH kernel matrix with weights $w_1 = w_2 = 1$ in (7). We then perform kernel PCA on this kernel matrix to embed the 40 objects in 2D. Figure 7 shows the embedding with annotated activities. Overall the activities form clear clusters. All eight participants' sitting activities are overlapping at the origin. This is expected because each sitting PL is nearly empty. The participants' walking activities are similar and form a tight cluster next to sitting. The cluster of playing basketball is more spread out than walking but tighter than running. We speculate that although basketball is more physical, it also lacks clear periodicity, hence the in-between embedding. Running as a cluster occupies an extreme region in kernel PCA due to both strong activities and clear periodicity. Finally, jumping is the only activity that spread out in the embedding across the eight people. The original data collection in [Altun *et al.*, 2010] instructed people to perform each activity in free style. We suspect that the cluster spread may be attributed to a large variation in how consistently people jumped for five minutes, given that it is a physically demanding activity.

Supervised activity classification. We perform a classification task to predict the five activities. The input is the 40×40 SMURPH kernel matrix. We measure 8-fold CV error, where in each fold we use 7 people (total of 35 activities) as the training data, and leave all 5 activities from one person out as test data. We use SVM^{light} [Joachims, 2008] with our SMURPH kernel. This SVM problem has a single regularization parameter C. We tune C by an inner CV within the first training fold of 7 people. On a parameter grid $C \in \{10^{-5}, 10^{-4}, \ldots, 10^{5}\}$, this inner CV selects $C = 100$. We fix this C for all other outer cross validation folds. The SVM CV accuracy with the SMURPH kernel is 95.0%. In contrast, with a linear kernel the SVM CV accuracy is 62.5%.

4 Related Work

This paper is related to but differs from several recent work in topological data analysis:

	multi-resolution	single-resolution
kernel	ours	KHNLB15, RHBK15, PHCJS11
non-kernel	AFW14	Bubenik15, LOC14

The closest work AFW14 is the local persistent homology proposed by Ahmed, Fasy, and Wenk [2014]. They computed local relative homology based on PD at various resolutions to compare street maps. SMURPH differs in that we build on PL; we address scalability issue with Monte Carlo; and we construct kernels to enable a much broader range of machine learning applications.

Also closely related are KHNLB15 [Kwitt *et al.*, 2015] and RHBK15 [Reininghaus *et al.*, 2015]'s multi-scale kernel. An important difference is that the "scale" in their multi-scale kernel means different amount of heat diffusion. Therefore, their kernels always describe global topology but with varying amount of smoothing. In contrast SMURPH kernels are defined over varying spatial scales, which allows us to see both global topology and fine-grained "topological texture." In addition our PL-based kernel avoids the need to perform heat diffusion on PDs as they do, and enjoys both conceptual simplicity and computational savings. Both kernels exhibits stability [Bubenik, 2015; Reininghaus *et al.*, 2015] and can be used for machine learning.

Bubenik [2015] pointed out that PL is a Hilbert space. LOC14 [Li *et al.*, 2014] used both PL and PD to measure the global topological distance between point clouds. Their methods did not aim to define kernels, nor did they study multi-resolution. PHCJS11 [Pachauri *et al.*, 2011] proposed a topological kernel for studying Alzheimer's disease. That kernel is based on kernel density estimation within a PD. However, this is a heuristic because a PD is not a 2D Euclidean space. Our paper inherits Chazal *et al.*'s bootstrap approach and approximates the persistent homology on a full point cloud with that on a random sample [Chazal *et al.*, 2015b; Chazal *et al.*, 2015a]. Their bootstrap approach loses resolution with sampling; our multi-resolution approach ameliorates this issue.

5 Conclusion

We have presented a novel kernel that compares the topology of point cloud objects. Our SMURPH kernel is multiresolutional and can handle large point clouds. It serves as a new bridge between topology and machine learning to encourage further research that can benefit both communities.

There are many ways one can extend SMURPH. The experiments in this paper are only proof-of-concept. One important task is to identify applications where SMURPH, in conjunction with standard kernels, can significantly improve the performance of machine learning applications. Another important task is to go beyond kernel methods and bring the same topological information into probabilistic models.

Acknowledgments The authors thank Frédéric Chazal and Brittany Fasy for helpful discussions. Support for this research was provided in part by NSF grant IIS 0953219.

References

[Ahmed *et al.*, 2014] Mahmuda Ahmed, Brittany Terese Fasy, and Carola Wenk. Local persistent homology based distance between maps. In *SIGSPATIAL*. ACM, Nov. 2014.

[Altun *et al.*, 2010] Kerem Altun, Billur Barshan, and Orkun Tuncel. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recognition*, 43(10):3605 – 3620, 2010.

[Bubenik, 2015] Peter Bubenik. Statistical topological data analysis using persistence landscapes. *Journal of Machine Learning Research*, 16:77–102, 2015.

[Carlsson, 2009] Gunnar Carlsson. Topology and data. *Bulletin (New Series) of the American Mathematical Society*, 46(2):255–308, 2009.

[Chazal *et al.*, 2013] F. Chazal, B. T. Fasy, F. Lecci, A. Rinaldo, A. Singh, and L. Wasserman. On the Bootstrap for Persistence Diagrams and Landscapes. *arXiv preprint arXiv:1311.0376*, November 2013.

[Chazal *et al.*, 2015a] F. Chazal, B. T. Fasy, F. Lecci, B. Michel, A. Rinaldo, and L. Wasserman. Subsampling Methods for Persistent Homology. In *ICML15, 32nd International Conference on Machine Learning*, Lille, France, 2015.

[Chazal *et al.*, 2015b] Frédéric Chazal, Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, and Larry A. Wasserman. Stochastic convergence of persistence landscapes and silhouettes. *Journal of Computational Geometry*, 2015. To appear.

[Edelsbrunner and Harer, 2010] H. Edelsbrunner and J. Harer. *Computational Topology: An Introduction*. Applied mathematics. Amer Mathematical Society, 2010.

[Fasy *et al.*, 2014] Brittany T. Fasy, Jisu Kim, Fabrizio Lecci, and Clement Maria. TDA: Statistical tools for topological data analysis. http://cran.r-project.org/web/packages/TDA, 2014.

[Hoover and Goldbaum, 2013] Adam Hoover and Michael Goldbaum. Stare public onine database, 2013. http://www.ces.clemson.edu/~ahoover/stare/.

[Joachims, 2008] Thorsten Joachims. SVM^{light} support vector machine, 2008. http://svmlight.joachims.org/.

[Karatzoglou *et al.*, 2004] Alexandros Karatzoglou, Alex Smola, Kurt Hornik, and Achim Zeileis. kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20, 2004.

[Kwitt *et al.*, 2015] Roland Kwitt, Stefan Huber, Marc Niethammer, Weili Lin, and Ulrich Bauer. Statistical topological data analysis - a kernel perspective. In C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3052–3060. Curran Associates, Inc., 2015.

[Li *et al.*, 2014] Chunyuan Li, Maks Ovsjanikov, and Frederic Chazal. Persistence-based structural recognition. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[Maria, 2014] C. Maria. Gudhi, simplicial complexes and persistent homology packages. https://project.inria.fr/gudhi/software/, 2014.

[Morozov, 2007] Dmitriy Morozov. Dionysus, a c++ library for computing persistent homology. http://www.mrzv.org/software/dionysus/, 2007.

[Nanda and Sazdanović, 2014] Vidit Nanda and Radmila Sazdanović. Simplicial models and topological inference in biological systems. In Natasa Jonoska and Masahico Saito, editors, *Discrete and Topological Models in Molecular Biology*. Springer, 2014.

[Nanda, 2013] Vidit Nanda. Perseus, the persistent homology software. http://www.sas.upenn.edu/~vnanda/perseus, 2013. Accessed Aug. 6, 2014.

[Neumann *et al.*, 2013] M. Neumann, P. Moreno, L. Antanas, R. Garnett, and K. Kersting. Graph Kernels for Object Category Prediction in Task-Dependent Robot Grasping. In *Proceedings of the Eleventh Workshop on Mining and Learning with Graphs (MLG–2013)*, Chicago, US, 2013.

[Pachauri *et al.*, 2011] Deepti Pachauri, Chris Hinrichs, Moo K. Chung, Sterling C. Johnson, and Vikas Singh. Topology-based kernels with application to inference problems in alzheimer's disease. *IEEE Trans. Med. Imaging*, 30(10):1760–1770, 2011.

[Reininghaus *et al.*, 2015] Jan Reininghaus, Stefan Huber, Ulrich Bauer, and Roland Kwitt. A stable multi-scale kernel for topological machine learning. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[Tausz *et al.*, 2011] Andrew Tausz, Mikael Vejdemo-Johansson, and Henry Adams. Javaplex: A research software package for persistent (co)homology. Software available at http://appliedtopology.github.io/javaplex/, 2011.

[Zhu, 2013] Xiaojin Zhu. Persistent homology: An introduction and a new text representation for natural language processing. In *The 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.