Persistence kernels for classification A comparative study

Cinzia Bandiziol^{*}, Stefano De Marchi[†]

Dipartimento di Matematica "Tullio Levi-Civita" University of Padova

Keywords: TDA, persistent homology, support vector machine, classification, kernel

Abstract: The aim of the present work is a comparative study of different persistence kernels applied to various classification problems. After some necessary preliminaries on homology and persistence diagrams, we introduce five different kernels that are then used to compare their performances of classification on various datasets. We also provide the Python codes for the reproducibility of results.

1 Introduction

In the last two decades, with the increasing need to analyze big amounts of data, which are usually complex and of high dimension, it was revealed meaningful and helpful to discover further methodologies to provide new information from data. This has brought to the birth of Topological Data Analysis (TDA), whose aim is to extract intrinsic, topological features, related to the so-called "shape of data". Thanks to its main tool, Persistent Homology (PH), it can provide new qualitative information that it would be impossible to extract in any other way. These kinds of features that can be collected in the so-called Persistence Diagram (PD), have been winning in many different applications, mainly related to applied science, improving the performances of models or classifiers, as in our context. Thanks to the strong basis of algebraic topology behind it, the TDA is very versatile and can be applied to data with a priori any kind of structure, as we will explain in the following. This is the reason why there is a wide range of fields of applications like chemistry [39], medicine [4], neuroscience [26], [16], finance [25] and computer graphics [6] only to name a few.

An interesting and relevant property of this tool is its stability to noise [9], which is a meaningful aspect for applications to real-world data. On the other hand, since the space of PDs is only metric one, to use methods that require data to live in a Hilbert space, such as SVM and PCA, it is necessary to introduce the notion of kernel or better Persistence Kernel (PK) that maps PD to space with more structure where it is possible to apply techniques that need a proper definition of inner product.

The goals of the present paper are: first we investigate how to choose values for parameters related to different kernels, then we collect tools for computing PD starting from different kinds of data and finally we compare performances of the main kernels in the classification context. As far as we know, the content of this study is not already present in literature.

^{*} cinzia. bandiziol@phd.unipd.it

[†]stefano.demarchi@unipd.it

The paper is organized as follows: in Section 2 we recall the basic notion related to persistent homology, in Section 3 we describe the problem of classification and solve it using Support Vector Machine (SVM), Section 4 lists the main PK available in literature, Section 5 collects all numerical tests that we have run and in Section 6 we outline conclusions.

2 Persistent Homology

This brief introduction does not claim to be exhaustive therefore we invite interested readers to refer, for instance, to the works [17], [32], [13], [14] and [7]. The first ingredient needed is the concept of *filtration*. The most common choice in applications is to consider a function $f: X \to \mathbb{R}$, where X is a topological space that varies based on different contexts, and then take into account the filtration based on the sub-level set given by $f^{-1}(-\infty, a), a \in \mathbb{R}$. For example, such an f can be chosen as the distance function in the case of point cloud data, the gray-scale values at each pixel for images, the heat kernel signature for datasets as SHREC14 [29], the weight function of edges for graphs, and so on. We now recall the main theoretical results related to point cloud data but all of them can be easily apply to other contexts.

We assume to have a set of points $\mathcal{X} = {\mathbf{x}_k}_{k=1,\dots,m}$ that, we suppose to live in an open set of a manifold \mathcal{M} . The aim is to be able to capture relevant intrinsic properties of the manifold itself and this is achieved through Persistent Homology (PH) applied to such discrete information. To understand how PH has been introduced, first, we have to mention the simplicial homology, which represents the extension of homology theory to structures called simplicial complexes.



Figure 1: An example of a valid simplicial complex (left) and an invalid one (right)

Definition 1. A simplicial complex K consists of a set of simplices of different dimensions and has to meet the following conditions:

- Every face of a simplex σ in K must belong to K
- The non-empty intersection of any two simplices $\sigma_1, \sigma_2 \in K$ is a face of both σ_1 and σ_2

The dimension of K is the maximum dimension of simplices that belong to K.

In application, data analysts usually compute the Vietoris-Rips complex.

Definition 2. Let (\mathcal{X}, d) denote a metric space from which the samples are taken. The **Vietoris-Rips complex** for \mathcal{X} , associated to the parameter ϵ , denoted by $VR(\mathcal{X}, \epsilon)$, is the simplicial complex whose vertex set is \mathcal{X} and $\{\mathbf{x}_0, \ldots, \mathbf{x}_k\}$ spans a k-simplex if and only if $d(\mathbf{x}_i, \mathbf{x}_j) \leq 2\epsilon$ for all $0 \leq i, j \leq k$.

If $K := VR(\mathcal{X}, \bar{\epsilon})$, we can divide all simplices of this set K, into groups based on their dimension k and we can enumerate them using Δ_i^k . If $G = (\mathbb{Z}, +)$ is the well known Abelian group, we may build linear combinations of simplices with coefficients in G, and so we introduce the following

Definition 3. An object of the form $c = \sum_{i} a_i \Delta_i^k$ with $a_i \in \mathbb{Z}$ is a **integer valued** k-dimensional chain.

Linearity allows to extend the previous definition to any subsets of simplices of K with dimension k,

Definition 4. The group $C_k^{\overline{\epsilon}}(\mathcal{X})$ is called the **group of k-dimensional simplicial** integer-valued chains of the simplicial complex K.

It is then possible to associate to each simplicial complex, the corresponding set of Abelian groups $C_0^{\bar{\epsilon}}(\mathcal{X}), \ldots, C_n^{\bar{\epsilon}}(\mathcal{X})$.

Definition 5. The boundary $\partial \Delta^k$ of an oriented simplex Δ^k , is the sum of all its (k-1)-dimensional faces taken with a chosen orientation. More precisely

$$\partial \Delta^k = \sum_{i=0}^k (-1)^k \Delta_i^{k-1}.$$

In the general setting, we can extend the boundary operator by linearity to a general element of $C_{\bar{k}}^{\bar{\epsilon}}(\mathcal{X})$, obtaining a map $\partial_k : C_k^{\bar{\epsilon}}(\mathcal{X}) \to C_{k-1}^{\bar{\epsilon}}(\mathcal{X})$.

No matter what the value of k is, it is a linear map. Therefore we can take into account its kernel, for instance, the group of k-cycles, $Z_k^{\bar{\epsilon}}(\mathcal{X}) := \ker(\partial_k)$ and the image, the group of k-boundaries, $B_{k+1}^{\bar{\epsilon}}(\mathcal{X}) := \operatorname{im}(\partial_k)$. Then $H_k^{\bar{\epsilon}}(\mathcal{X}) = Z_k^{\bar{\epsilon}}(\mathcal{X})/B_{k+1}^{\bar{\epsilon}}(\mathcal{X})$ is the k-homology group and represents the k-dimensional holes that can be recovered from the simplicial structure. We briefly recall here that, for instance, 0-dimensional holes correspond to connected components, 1-dimensional holes are cycles, and 2-dimensional holes are cavities/voids. Since they are algebraic invariants, they collect qualitative information regarding the topology of the data. The most crucial aspect is highlighting the best value for ϵ to obtain a simplicial complex K that faithfully reproduces the original manifold's topological structure. The answer is not straightforward and the process reveals unstable, therefore the PH analyzes not only one simplicial complex but a nested sequence of them, and, following the evolution of such structure, it notes down features that gradually emerge. From a theoretical point of view, letting $0 < \epsilon_1 < \cdots < \epsilon_l$ be an increasing sequence of real numbers, we obtain the filtration

$$\emptyset \subset K_1 \subset K_2 \subset \cdots \subset K_l$$

with $K_i = VR(\mathcal{X}, \epsilon_i)$ and then

Definition 6. The **p-persistent homology group** of K_i is the group defined as

$$H_k^{i,p} = Z_k^i / (B_k^{i+p} \cap Z_k^i)$$

This group contains all stable homology classes in the interval i to i + p: they are born before the time/index i and are still alive after p steps. The persistent homology classes alive for large values of p are stable topological features of S (see [3]). Along the filtration, the topological information appears and disappears, thus it means that they may be represented with a couple of indexes. If p is such a feature, it must be born in some K_i and die in K_j so it can be described as (i, j), i < j. We underline here that j can be equal to $+\infty$, since some features can be alive up to the end of the filtration. Hence, all such topological invariants live in the extended positive plane, that here is denoted by $\mathbb{R}^2_+ = \mathbb{R}_{\geq 0} \times \{\mathbb{R}_{\geq 0} \cup \{+\infty\}\}$. Another interesting aspect to highlight is that some features can appear more than once and accordingly such collection of points are called multisets. All of these observations are grouped into the following

Definition 7. A **Persistence Diagram (PD)** $D_r(\mathcal{X}, \boldsymbol{\varepsilon})$ related to the filtration $\emptyset \subset K_1 \subset K_2 \subset \cdots \subset K_l$ with $\boldsymbol{\varepsilon} := (\epsilon_1, \ldots, \epsilon_l)$ is a multiset of points defined as

$$D_r(\mathcal{X}, \boldsymbol{\varepsilon}) := \{(b, d) | (b, d) \in P_r(\mathcal{X}, \boldsymbol{\varepsilon})\} \cup \Delta$$

where $P_r(\mathcal{X}, \boldsymbol{\varepsilon})$ denotes the set of *r*-dimensional birth-death couples that came out along the filtration, each (b, d) is considered with its multiplicity, while points of $\Delta = \{(x, x) | x \geq 0\}$ with infinite multiplicity. One may consider all $P_r(\mathcal{X}, \boldsymbol{\varepsilon})$ for every *r* together, obtaining the total PD denoted here by $D(\mathcal{X}, \boldsymbol{\varepsilon})$, that we will usually consider in the following sections.

Each point $(b, d) \in D_r(\mathcal{X}, \varepsilon)$ is called **generator** of the persistent homology, and represents a topological property which appears at K_b and disappears at K_d . The difference d - b is called **persistence** of the generator, represents its lifespan and shows the robustness of the topological property.



Figure 2: An example of Persistence Diagram with features of dimensions 0,1 and 2 (image taken from a financial application as discussed in [44])

Figure 2 is an example of total PD collecting features of dimension 0 (in blue), of dimension 1 (in orange), and of dimension 2 (in green). Points close to the diagonal represent features with a short lifetime, and so usually they are concerned with noise, instead features far away are indeed relevant and meaningful and, based on applications, one can decide to consider both or only the most interesting ones. At the top of the Figure, there is a dashed line that indicates the infinity and allows to plot also couples as $(i, +\infty)$.

In the previous definition, the set Δ is added to finding out proper bijections between sets, that without Δ could not have the same number of points. It makes it possible to compute the proper distance between PDs.

2.1 Stability

A key property of PDs is stability under perturbation of the data. First, we recall two famous distances for sets,

Definition 8. For two nonempty sets $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^d$ with the same cardinality, the **Hauss-dorff distance** is

$$d_H(\mathcal{X}, \mathcal{Y}) := \max\{\sup_{x \in \mathcal{X}} \inf_{y \in \mathcal{Y}} \|x - y\|_{\infty}, \sup_{y \in \mathcal{Y}} \inf_{x \in \mathcal{X}} \|y - x\|_{\infty}\}$$

and **bottleneck distance** is defined as

$$d_B(\mathcal{X}, \mathcal{Y}) := \inf_{\gamma} \sup_{x \in \mathcal{X}} \|x - \gamma(x)\|_{\infty}$$
(1)

where we consider all possible bijection of multisets $\gamma : \mathcal{X} \to \mathcal{Y}$. Here, we use

$$||v - w||_{\infty} = \max\{|v_1 - w_1|, |v_2 - w_2|\}, \text{ for } v = (v_1, v_2), w = (w_1, w_2) \in \mathbb{R}^2$$



Figure 3: Example of bottleneck distance between two PDs in red and blue (image from [42])

We try to explain better how to compute the bottleneck distance. We have to take all possible ways to move points from \mathcal{X} to \mathcal{Y} in a bijective manner and then one can compute properly the distance. Figure 3 shows two different PDs overlapped, that consist of Δ joined with 2 points in red and 11 points in blu respectively. First, in order to apply the definition (1), we need two sets with same cardinality. For this aim, it is necessary to add points of Δ , more precisely the orthogonal projection onto the diagonal of the 9 blue points closer to it, to reach 11. Lines between points and Δ represent the bijection that realizes the best matching between points in definition (1).

Proposizione 2.1. Let \mathcal{X} and \mathcal{Y} be finite subset in a metric space (M, d_M) . Then the Persistence Diagrams $D(\mathcal{X}, \boldsymbol{\varepsilon})$, $D(\mathcal{Y}, \boldsymbol{\varepsilon})$ satisfy

$$d_B(D(\mathcal{X}, \boldsymbol{\varepsilon}), D(\mathcal{Y}, \boldsymbol{\varepsilon})) \leqslant d_H(\mathcal{X}, \mathcal{Y}).$$

For any further details see for example [32].

3 SVM

3.1 Classification problem

Let $\Omega \subset \mathbb{R}^d$ and $\{\mathbf{x}_1, ..., \mathbf{x}_m\} \subset \mathcal{X} \subset \Omega$ be the set of input data with $d, m \in \mathbb{N}$. We have a training set, composed by the couples (\mathbf{x}_i, y_i) with i = 1, ..., m and $y_i \in \mathcal{Y} = \{-1, 1\}$. The **binary supervised learning task** consists in finding a function $f : \Omega \longrightarrow \mathcal{Y}$, the model, such that it can predict, in a satisfactory way, the label of an unseen $\tilde{\mathbf{x}} \in \Omega \setminus \mathcal{X}$.

The aim is to find the hyperplane that can separate, in the best possible way, points that belong to different classes and from here the name separating hyperplane. The best possible way means that it separates the two classes with the higher margin, that is the distance between the hyperplane and the points of both classes.

More formally, if we assume to be in a space \mathscr{F} with dot product, for instance \mathscr{F} can be a subset of \mathbb{R}^d with $\langle \cdot, \cdot \rangle$, since a generic hyperplane can be defined as

$$\{\mathbf{x}\in\mathscr{F}|\langle\mathbf{w},\mathbf{x}\rangle+b=0\}\ w\in\mathscr{F},b\in\mathbb{R}$$

one can introduce

Definition 9. We call

$$\rho_{w,b}(\mathbf{x}, y) := \frac{y(\langle w, \mathbf{x} \rangle + b)}{\|w\|}$$

the **geometrical margin** of the point $(\mathbf{x}, y) \in \mathscr{F} \times \{-1, 1\}$. Instead, the minimum value,

$$\rho_{w,b} := \min_{i=1,\dots,m} \rho_{w,b}(\mathbf{x}_i, y_i)$$

shall be call the **geometrical margin** of $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$.

From a geometrical point of view, this margin measures effectively the distance between samples and the hyperplane itself. Then SVM is looking for a suitable hyperplane that, intuitively realizes the maximum of such margin. For any further details, see for example [34]. The precise formalization brings to an optimization problem that, thanks to the Lagrange multipliers and Karush-Kuhn-Tucker conditions, it turns out to have the following formulation, as **SVM optimization problem**,

$$\max_{\alpha \in \mathbb{R}^m} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

s. to
$$\sum_{i=1}^m \alpha_i y_i = 0$$
$$0 \le \alpha_i \le C \ \forall i = 1, \dots, m$$

where $[0, C]^m$ is the bounding box $C \in [0, +\infty)$ and $\alpha_i > 0$ are called **Support Vectors**. From here and what follows the name Support Vector Machine is shortened in SVM and $\langle \cdot, \cdot \rangle$ denotes the inner product in \mathbb{R}^d . This formulation can face satisfactorily the classification task if data are linearly separable. In applications, it doesn't happen frequently and so it is needed to introduce some nonlinearity and move in higher dimensional space where, hopefully, that can happen. This can be achieved with the use of kernels. Starting from the original dataset \mathcal{X} , the theory tells to introduce a feature map $\Phi : \mathcal{X} \to \mathcal{H}$ that moves data from \mathcal{X} to a Hilbert space of function \mathcal{H} , the so-called **feature space**. The kernel is then defined as $\kappa(\mathbf{x}, \bar{\mathbf{x}}) := \langle \Phi(\mathbf{x}), \Phi(\bar{\mathbf{x}}) \rangle_{\mathcal{H}}$ (kernel trick). Thus the optimization problem becomes

$$\max_{\alpha \in \mathbb{R}^m} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

s. to
$$\sum_{i=1}^m \alpha_i y_i = 0$$
$$0 \le \alpha_i \le C \ \forall i = 1, \dots, m$$

where kernel represents a generalization of the inner product in \mathbb{R}^d . We are interested in classifying PDs and obviously, we need suitable definitions for kernels for PDs, the so-called **Persistence Kernels (PK)**.

4 Persistence Kernels

In what follows we denote with \mathfrak{D} the set of the total PDs.

4.0.1 Persistence Scale-Space Kernel (PSSK)

The first kernel was described in [31]. The main idea is to compute the feature map as the solution of the Heat equation. We consider $\Omega_{ad} = \{\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2 : x_2 \ge x_1\}$ and we denote with $\delta_{\mathbf{x}}$ the Dirac delta centered at \mathbf{x} . For a given $D \in \mathfrak{D}$, we consider the solution $u : \Omega_{ad} \times \mathbb{R}_{\ge 0} \to \mathbb{R}$, $(\mathbf{x}, t) \mapsto u(\mathbf{x}, t)$ of the following PDE:

$$\Delta_{\mathbf{x}} u = \partial_t u \quad \text{in } \Omega_{ad} \times \mathbb{R}_{\geq 0}$$
$$u = 0 \quad \text{on } \partial\Omega_{ad} \times \mathbb{R}_{\geq 0}$$
$$u = \sum_{\mathbf{y} \in D} \delta_{\mathbf{y}} \quad \text{on } \Omega_{ad} \times 0$$

The feature map $\Phi_{\sigma} : \mathfrak{D} \to L^2(\Omega_{ad})$ at scale $\sigma > 0$ at D is defined as $\Phi_{\sigma}(D) = u|_{t=\sigma}$. This map yields the **Persistence Scale Space Kernel (PSSK)** K_{σ} on \mathfrak{D} as:

$$K_{\sigma}(D, E) = \langle \Phi_{\sigma}(D), \Phi_{\sigma}(E) \rangle_{L^{2}(\Omega_{ad})}.$$

But since it is known as an explicit formula for the solution u, the kernel takes the form

$$K_{\sigma}(D, E) = \frac{1}{8\pi\sigma} \sum_{\mathbf{x}\in D, \mathbf{y}\in E} \exp(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{8\sigma}) - \exp(-\frac{\|\mathbf{x} - \bar{\mathbf{y}}\|^2}{8\sigma})$$

where $\mathbf{y} = (a, b), \, \bar{\mathbf{y}} = (b, a), \, \text{for any } D, E \in \mathfrak{D}.$

4.0.2 Persistence Weighted Gaussian Kernel (PWGK)

In [21], the authors introduce a new kernel whose idea is to replace each PD with a discrete measure. Starting with a strictly positive definite kernel, as for example the gaussian one $\kappa_G(\mathbf{x}, \mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}}, \sigma > 0$ we denote the corresponding Reproducing Kernel Hilbert Space \mathcal{H}_{κ_G} .

If $\Omega \subset \mathbb{R}^d$, we denote with $M_b(\Omega)$ the space of finite signed Radon measures and

$$E_{\kappa_G}: M_b(\Omega) \to \mathcal{H}_{\kappa_G}, \mu \mapsto \int_{\Omega} \kappa_G(\cdot, \mathbf{x}) d\mu(\mathbf{x})$$

For any $D \in \mathfrak{D}$, if $\mu_D^w = \sum_{\mathbf{x} \in D} w(\mathbf{x}) \delta_{\mathbf{x}}$, where the weight function satisfies w(x) > 0 for all $\mathbf{x} \in D$ then

$$E_{\kappa_G}(\mu_D^w) = \sum_{\mathbf{x}\in D} w(\mathbf{x})\kappa_G(\cdot, \mathbf{x})$$

where

$$w(\mathbf{x}) = \arctan(C_w pers(\mathbf{x})^p)$$

and $pers(\mathbf{x}) = x_2 - x_1$.

The Persistence Weight Gaussian Kernel (PWGK) is defined as

$$K_{G}^{w}(D, E) = \exp\left(-\frac{1}{2\tau^{2}} \|E_{\kappa_{G}}(\mu_{D}^{w}) - E_{\kappa_{G}}(\mu_{E}^{w})\|_{\mathcal{H}_{\kappa_{G}}}^{2}\right), \tau > 0$$

for any $D, E \in \mathfrak{D}$.

4.0.3 Sliced Wasserstein Kernel (SWK)

Another possible choice for κ has been introduced in [8].

We consider μ and ν two nonnegative measures on \mathbb{R} such that $\mu(\mathbb{R}) = r = |\mu|$ and $\nu(\mathbb{R}) = r = |\nu|$, we recall that the 1-Wasserstein distance for nonnegative measures is defined as

$$\mathcal{W}(\mu,\nu) = \inf_{P \in \Pi(\mu,\nu)} \int \int_{\mathbb{R} \times \mathbb{R}} |x-y| dP(x,y)$$

where $\Pi(\mu, \nu)$ is the set of measures on \mathbb{R}^2 with marginals μ and ν .

Definition 10. Given $\theta \in \mathbb{R}^2$ with $\|\theta\|_2 = 1$, let $L(\theta)$ denote the line $\{\lambda\theta|\lambda \in \mathbb{R}\}$ and let $\pi_{\theta} : \mathbb{R}^2 \to L(\theta)$ be the orthogonal projection onto $L(\theta)$. Let $D, E \in \mathfrak{D}$ and let $\mu_D^{\theta} := \sum_{\mathbf{x} \in D} \delta_{\pi_{\theta}(\mathbf{x})}$ and $\mu_{D\Delta}^{\theta} := \sum_{\mathbf{x} \in D} \delta_{\pi_{\theta} \circ \pi_{\Delta}(\mathbf{x})}$ and similarly for μ_E^{θ} and $\mu_{E\Delta}^{\theta}$ where π_{Δ} is the orthogonal projection onto the diagonal. Then, the **Sliced Wasserstein distance** is

$$SW(D,E) = \frac{1}{2\pi} \int_{\mathbb{S}_1} \mathcal{W}(\mu_D^{\theta} + \mu_{E\Delta}^{\theta}, \mu_E^{\theta} + \mu_{D\Delta}^{\theta}) d\theta$$

Thus, the Sliced Wasserstein Kernel (SWK) is defined as

$$K_{SW}(D,E) := \exp\left(-\frac{SW(D,E)}{2\eta^2}\right), \, \eta > 0$$

for any $D, E \in \mathfrak{D}$.

4.0.4 Persistence Fisher Kernel (PFK)

In [22], the authors describe a kernel based on Fisher Information geometry.

A persistence diagram $D \in \mathfrak{D}$ can be considered as a discrete measure $\mu_D = \sum_{u \in D} \delta_u$, where δ_u is the Dirac's delta centered in u. Given a bandwidth $\sigma > 0$, and a set Θ , one can smooth and normalize μ_D as follows

$$\rho_D := \frac{1}{Z} \sum_{u \in D} N(\mathbf{x}; u, \sigma I)$$

where N is a Gaussian function, $Z = \int_{\theta} \sum_{u \in D} N(x; u, \sigma I) dx$ and I is the identity matrix. Thus, using this measure, any PD can be regarded as a point in $\mathbb{P} = \{\rho | \int \rho(\mathbf{x}) d\mathbf{x} = 1, \rho(\mathbf{x}) \ge 0\}.$

Given two element in $\rho_i, \rho_j \in \mathbb{P}$, the Fisher Information Metric is

$$d_{\mathbb{P}}(\rho_i, \rho_j) = \arccos\left(\int \sqrt{\rho_i(\mathbf{x})\rho_j(\mathbf{x})} d\mathbf{x}\right)$$

Inspiring by the Sliced Wasserstein Kernel construction, we have the following

Definition 11. Let D, E be two finite and bounded persistence diagrams. The Fisher information metric between D and E, is defined as

$$d_{FIM}(D, E) := d_{\mathbb{P}}(\rho_{D \cup E_{\Delta}}, \rho_{E \cup D_{\Delta}})$$

where $D_{\Delta} := \{\Pi_{\Delta}(u) | u \in D\}, E_{\Delta} := \{\Pi_{\Delta}(u) | u \in E\}$ and Π_{Δ} is the orthogonal projection on the diagonal $\Delta = \{(a, a) | a \ge 0\}.$

The Persistence Fisher Kernel (PFK) is then defined as

$$K(D, E) := \exp(-td_{FIM}(D, E)), t > 0$$
, for any $D, E \in \mathfrak{D}$.

4.0.5 Persistence Images (PI)

The main reference is [1]. If $D \in \mathfrak{D}$ we introduce a change of coordinates, $T : \mathbb{R}^2 \to \mathbb{R}^2$ given by T(x, y) = (x, y - x) and let T(D) be the transformed multiset in firstpersistence coordinates. Let $\phi_u : \mathbb{R}^2 \to \mathbb{R}$ be a differentiable probability distribution with mean $u = (u_x, u_y) \in \mathbb{R}^2$, usually $\phi_u = g_u$, where g_u is the 2-dimensional Gaussian with mean u and variance σ^2 , defined as

$$g_u(x,y) = \frac{1}{2\pi\sigma^2} e^{-[(x-u_x)^2 + (y-u_y)^2]/2\sigma^2}.$$

Fix a weight function $f : \mathbb{R}^2 \to \mathbb{R}$, that is $f \ge 0$, it is equal zero on the horizontal axis, continuous and piecewise differentiable. A possible choice is a function that depends only to the persistence coordinate y, a function $f(x, y) = w_b(y)$ where

$$w_b(t) = \begin{cases} 0 & \text{if } t \le 0, \\ \frac{t}{b} & \text{if } 0 < t < b, \\ 1 & \text{if } t \ge b \end{cases}$$

Definition 12. Given $D \in \mathfrak{D}$, the corresponding **persistence surface** $\rho_D : \mathbb{R}^2 \to \mathbb{R}$ is the function

$$\rho_D(x,y) = \sum_{u \in T(D)} f(u)\phi_u(x,y).$$

If we divide the plane in a grid with n^2 pixels $(P_{i,j})_{i,j=1,\dots,n}$, we have the following

Definition 13. Given $D \in \mathfrak{D}$, its **persistence image** is the collection of pixels

$$PI(\rho_D)_{i,j} = \int \int_{P_{i,j}} \rho_D(x,y) dx dy$$

Thus, through persistence image, each persistence diagram is turned into a vector $PIV \in \mathbb{R}^{n^2}$ that is $PIV(D)_{i+n(j-1)} = PI(D)_{i,j}$, then it is possible to introduce the following kernel

$$K_{PI}(D, E) = \langle PIV(D), PIV(E) \rangle_{\mathbb{R}^{n^2}}$$

5 Shape paramenters analysis

Each aforementioned kernel has some parameters, that have been chosen through the cross-validation phase. As in the context of RBF as explained in [15], also in Machine Learning framework, it is better to tune the parameters accordingly to the so-called trade-off principle, for instance, such that the condition number of Gram matrix is not so high and on the other hand the accuracy is satisfactorily high. The aim here is to run such analysis to kernels presented in this paper.

The **PSSK** has only one parameter to tune σ . Typically the users consider $\sigma \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$. We run the CV phase for different shuffles of a dataset and plot the results in terms of the condition number of the Gram matrix related to the training samples and the accuracy. For our analysis, we consider $\sigma \in \{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 500, 800, 1000\}$ and run tests on some datasets cited in the following. The results are similar in each case so we decided to report ones about SHREC14 dataset.



Figure 4: Comparison results about PSSK for SHREC14 in terms of condition number (left) and accuracy (right) with different σ

From the plot, it is evident how large values of σ bring to unstable matrix and less accuracy. Then in what follows, we will take into account only $\sigma \in \{0.00001, 0.0001, 0.001, 0.001, 0.01, 0.01, 0.1, 1, 10\}$.

PWGK is the kernel with a higher number of parameters to tune, then it is not so evident what are the best-set values to take into account. We choose reasonable starting sets as: $\tau \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}, \rho \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}, p \in \{1, 5, 10, 50, 100\}, C_w \in \{0.001, 0.01, 0.1, 1\}$. Due to a large number of parameters, we first ran some experiments varying (ρ, τ) with fixed (p, C_w) , and then we reversed the roles.

We report here in Figure 5 only a plot for fixed C_w and p because it highlights how high values of τ (for example $\tau = 1000$) are to be excluded. We find this behavior for different values of C_w, p and various datasets, here the case $C_w = 1, p = 10$ and MUTAG dataset. Therefore we decide to vary the parameters as follows: $\tau \in$ $\{0.001, 0.01, 0.1, 1, 10, 100\}, \rho \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}, p \in \{1, 5, 10, 50, 100\},$ $C_w \in \{0.001, 0.01, 0.1, 1\}$. Unluckily there is no other evidence that can guide the choices, except for τ , where values $\tau = 1000$ always have bad accuracy, as one can see below in the case of MUTAG with shortest path distance.

In the case of **SWK**, there is only one parameter η . In [8], the authors propose to consider values starting from the first and last decile and the median value of the gram matrix of the training samples flatten in order to obtain a vector, then they multiply these three values for 0.01, 0.1, 1, 10, 100. For our analysis, we have decided to study the



Figure 5: Comparison results about PWGK for MUTAG in terms of accuracy with different τ and ρ

behavior of such kernel considering the same set of values, independently from the specific dataset. We consider $\eta \in \{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 500, 800, 1000\}$.



Figure 6: Comparison results about SWK for DHFR in terms of condition number (left) and accuracy (right) with different η

We run tests on some datasets and the plot, related to the DHFR dataset, reveals evidently that large values for η are to be excluded. So, we will decide to take η only in $\{0.00001, 0.001, 0.001, 0.01, 0.1, 1, 10\}$.

PFK has two parameters: the variance σ and t. In [22], the authors exhibit the procedure to follow in order to obtain the corresponding set of values. It shows that the choice of t depends on σ . Instead, our aim in this paper is to carry out an analysis that is dataset-independent and that turns out to be strictly connected only to the definition of kernel itself. First, we take different values for (σ, t) and we plot the corresponding accuracies, here in the case of MUTAG with shortest path distance, but the same behavior holds true also for other datasets.

The condition numbers are indeed high for every choice of parameters and therefore we avoid reporting here because it would be meaningless. From the plot, it is evident that it is convenient to set σ lower or equal to 10 instead t should be set bigger or



Figure 7: Comparison results about PFK for MUTAG in terms of accuracy with different t and σ

equal to 0.1. Thus in what follows, we take into account $\sigma \in \{0.001, 0.01, 0.1, 1, 10\}$ and $t \in \{0.1, 1, 10, 100, 1000\}$.

In the case of **PI**, we considered a reasonable set of values for the parameter $\sigma \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$. The results are related to BZR with shortest path distance.



Figure 8: Comparison results about PI for BZR in terms of condition number (left) and accuracy (right) with different σ

As in the previous kernels, it seems that the accuracy is better for small values of σ . For this reason, we set $\sigma \in \{0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$.

6 Numerical Tests

For what concerns the computation of simplicial complexes and persistence diagrams, we use some Python libraries available online as gudhi [42], ripser [38], giotto-tda [43] and persim [33]. To all datasets, we have performed a random splitting (70%/30%) for training and test and applied a 10-fold Cross Validation on the training set for the

hyperparameters tuning. Then we averaged the results over 10 runs. For balanced datasets, we have measured the performances of classifier through the accuracy for binary and multiclass problems,

 $accuracy = \frac{number of test samples correctly classify}{all test samples}$

Instead, in the case of imbalanced datasets, we have adopted the balanced accuracy as explained in [19], if for every class i we define the related recall as

$$recall_i = \frac{\text{test samples of class i correctly classify}}{\text{all test samples of class i}}$$

then the balanced accuracy, in case of n different classes is,

balanced_accuracy =
$$\frac{\sum_{i=1}^{n} recall_i}{n}$$

This definition is able to effectively quantify how accurate is the classifier even in the case of the smallest classes. In tests, we use the implementation of SVM provided by the Scikit [27] library of Python. For PFK, we precomputed the Gram matrices using a Matlab (Matlab R2023b) routine because it is faster than the Python one. The values for C belong to $\{0.001, 0.01, 0.1, 1, 10, 100\}$. For each kernel, we have considered the following values for the parameters:

- **PSSK**: $\sigma \in \{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$
- **PWGK**: $\tau \in \{0.001, 0.01, 0.1, 1, 10, 100\}, \rho \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}, p \in \{1, 5, 10, 50, 100\}, C_w \in \{0.001, 0.01, 0.1, 1\}$ and for kernel we chose the Gaussian one.
- **SWK**: $\eta \in \{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$
- **PFK**: $\sigma \in \{0.001, 0.01, 0.1, 1, 10\}$ and $t \in \{0.1, 1, 10, 100, 1000\}$
- **PI**: $\sigma \in \{0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$ and number of pixel 0.1.

All codes have been run using Python 3.11 on a 2.5 GHz Dual-Core Intel Core i5, 32 Giga RAM. They are available in the GitHub page

https://github.com/cinziabandiziol/persistence_kernels

6.1 Point cloud data and shapes

6.1.1 Protein

This is the Protein Classification Benchmark dataset PCB00019 [36]. It sums up information for 1357 proteins corresponding to 55 classification problems. The data are highly imbalanced and therefore we apply the classifier to one of them, where the imbalance is slightly less evident. Persistence diagrams were computed for each protein by considering the 3-D structure or better the (x, y, z) position of any atoms in each of the 1357 molecules, as a point cloud in \mathbb{R}^3 . Finally using **ripser** we compute the persistence diagrams only of dimension 1.



6.1.2 SHREC14 - Synthetic data

The dataset is related to the problem of non-rigid 3D shape retrieval. It collects exclusively human models in different body shapes and 20 poses. It consists of 15 different human models, about man, woman, and child, each with its own body shape. Each of these models exists in 20 different poses making the dataset composed of 300 models. For each shape, the meshes are given with about 60000 vertices and, using the Heat Kernel Signature (HKS) introduced in [37], over different values of t_i as [31], we have computed the persistence diagrams of the induced filtrations in dimensions 1.

6.1.3 Orbit recognition

We consider the dataset proposed in [1]. We take into account the linked twisted map, which models fluid flows. The orbits can then be computed through the following discrete dynamical system

$$\begin{cases} x_{n+1} &= x_n + ry_n(1 - y_n) \mod 1\\ y_{n+1} &= y_n + rx_{n+1}(1 - x_{n+1}) \mod 1 \end{cases}$$

where the starting point $(x_0, y_0) \in [0, 1] \times [0, 1]$ and r > 0 is a real parameter that influences the behavior of the orbits, as appears in images.



Figure 9: Orbits composed by the first 1000 iterations of the twisted map with r = 3.5, 4.1, 4.3 from left to right

As in [1], $r \in 2.5, 3.5, 4, 4.1, 4.3$ and it is strictly connected to the label of the corresponding orbit. For each of them, we compute the first 1000 points of 50 orbits, with starting points chosen randomly. The final dataset is composed of 250 elements. We compute PD considering only the 1-dimensional features. Since each PD has a huge number of topological features, we decide to consider only the first 10 most persistent, as done in [12].

Kernel	PROTEIN	SHREC14	DYN SYS
PSSK	0.561	0.933	0.829
PWGK	0.538	0.923	0.819
SWK	0.531	0.935	0.841
PFK	0.556	0.935	0.784
PI	0.560	0.934	0.777

Table 1: Accuracy related to point cloud and shape datasets

First, the high difference in performances through different datasets is probably due to the high imbalance of the PROTEIN one with respect to the perfect balance of the other ones. It is well known that, if the classifier has not enough samples for each class, as in the case of the imbalanced dataset, it has to face high issues in classifying correctly elements of the minor classes. Except for PROTEIN where the PSSK shows slightly better performances, for SHREC14 and DYN SYS the best accuracy has been achieved by SWK.

6.2 Images

All the definitions introduced in Section 2 can be extended to another kind of simplicial complex, the cubical complex. It is useful when one deals with images or objects based on meshes, for example. More precise from [24]

Definition 14. An elementary cube $Q \subset \mathbb{R}^d$ is defined as a product $Q = I_1 \times \cdots \times I_d$ where each I_j is either a singleton set $\{m\}$ or a unit-length interval [m; m+1] for some integers $m \in \mathbb{Z}$. The number k of the unit-length intervals in the product of Q is called the dimension of cube Q and we call Q a k-cube. If Q and \overline{Q} are two cubes and $Q \subset \overline{Q}$, then Q is said to be a face of \overline{Q} . A cubical complex X in \mathbb{R}^d is a collection of k-cubes $(0 \leq k \leq d)$ such that:

- every face of a cube in X is also in X;
- the intersection of any two cubes of X is either empty or a face of each of them.



Figure 10: Cubical simplices

6.2.1 MNIST and FMNIST

MNIST [23] is very common in the classification framework. It consists of 70000 handwritten digits, in grayscale, which one could try to classify into 10 different classes. Each image can be viewed as a set of pixels with a value between 0 and 256 (black and white) as in the figure.



Figure 11: Example of an element in MNIST dataset

Starting from this kind of dataset, we have to compute the corresponding persistent features. According to the approach proposed in [18] coming from [5], we first binarize each image, for instance, we replace each grayscale image with a white/black one, then we use as filtration function the so-called **Height filtration** $\mathcal{H}(p)$ in [18]. For cubical complex, for a chosen vector $v \in \mathbb{R}^d$ of unit norm, it is defined as

$$\mathcal{H}(p) = \begin{cases} \langle p, v \rangle & \text{if p is black,} \\ H_{\infty} & \text{otherwise} \end{cases}$$

where H_{∞} is a big default value chosen by the user. As in [5] we have chosen 4 different vectors for p: (1,0), (-1,0), (0,1), (0,-1) and we have computed 0 and 1-dimensional persistent features both using tda-giotto and gudhi libraries. Finally, we concatenate them. For the current experiment, we decided to focus the test on a subset of the original MNIST, composed of only 10000 samples. This is a balanced dataset. Due to some memory issues, we have to consider for this dataset a pixel size of 0.5 and for PWGK only $\tau \in \{0.001, 0.01, 0.1, 1, 10, 100\}, \rho \in \{0.001, 0.1, 10, 1000\}, p = 10, C_w \in \{0.001, 0.01, 0.1, 1\}.$

Another example of a grayscale image dataset is the FMNIST [40], which contains 28 x 28 grayscale images related to the fashion world.

To deal with it, we follow another approach proposed in [3], where the authors apply padding, median filter, shallow thresholding, and canny edges and then compute the usual filtration to the image obtained. Due to some memory issues, we have to consider for this dataset a pixel size of 1 and for PWGK only $\tau \in \{0.001, 0.01, 0.1, 1, 10, 100\}, \rho \in \{0.001, 0.1, 10, 1000\}, p = 10, C_w \in \{0.001, 0.01, 0.1, 1\}.$

Kernel	MNIST	FMNIST
PSSK	0.729	0.664
SWK	0.802	0.709
PWGK	0.754	0.684
PFK	0.734	0.671
PI	0.760	0.651

Table 2: Accuracy related to MNIST and FMNIST



Figure 12: Example of an element in FMNIST dataset

Both datasets are balanced and probably the results are better in the case of MNIST due the fact that it is easier to classify handwritten digits instead of images of cloths. SWK shows slightly best performances.

6.3 Graphs

In many different contexts from medicine to chemistry, data can have the structure of graphs. Graphs are couples of set (V, E) where V is the set of vertices and E is the set of edges. The graph classification is the task of attaching a label/class to each whole graph. In order to compute the persistent features, we need to build a filtration. In the context of graphs, as in other cases, there are different definitions, see for example [2].

We consider the Vietoris Rips filtration, where starting from the set of vertices, at each step we add the corresponding edge whose weights are less or equal to a current value ϵ . This turns out to be the most common choice and the software available online allows us to build it after providing the corresponding adjacency matrix. In our experiments, we consider only undirected graphs but, as in [2], building a filtration is possible also for directed graphs. Once defining the kind of filtration to use, one needs again to choose the corresponding weights. We decide to take into account: first, the shortest path distance and then the Jaccard index as for example in [41].

Given two vertices $u, v \in V$ the **shortest path distance** is defined as the minimum number of different edges that one has to meet going from u to v, or vice versa since the graphs here are considered as undirected. In graphs theory, it is a widely use metric.

Instead, the Jaccard index is a good measure of edge similarity. Given an edge $e = (u, v) \in E$ then the corresponding **Jaccard index** is computed as

$$\rho(u,v) = \left| \frac{NB(u) \cap NB(v)}{NB(u) \cup NB(v)} \right|$$

where NB(u) is the set of neighbours of u in the graph. This metric recover local information of nodes in the sense that two nodes are considered similar if their neighbor sets are similar.

In both cases, we consider the sub-level set filtration and we collect 0 and 1 dimensional persistent features both.

We take 6 of such sets among the graph benchmark datasets, all undirected. They are

- **MUTAG**: it is a collection of nitroaromatic compounds and the goal is to predict their mutagenicity on Salmonella typhimurium
- **PTC**: is a collection of chemical compounds represented as graphs that report the carcinogenicity of rats
- **BZR**: it is a collection of chemical compounds and one has to classify them as active or inactive
- **ENZYMES**: it is a dataset of protein tertiary structures obtained from the BRENDA enzyme database and the aim is to classify each graph into 6 enzymes.
- **DHFR**: it is a collection of chemical compounds and one has to classify them as active or inactive
- **PROTEINS**: in each graph nodes represent the secondary structure elements and the task is to predict whether a protein is an enzyme or not.

Their properties are summarized in table 3, where the IR index is the so-called **Imbalanced Ratio** (IR), that denotes the imbalance of the dataset, and it is defined as, a sample size of the major class over sample size of the minor class.

Dataset	N° Graphs	N° classes	IR
MUTAG	188	2	125:63
PTC	344	2	192:152
BZR	405	2	319:86
ENZYMES	600	6	100:100
DHFR	756	2	461:295
PROTEINS	1113	2	663:450

Table 3: Graph datasets

Computations of adjacency matrix and PDs are made using functions implemented in tda-giotto.

The performances achieved with two edge weights are reported in tables,

Kernel	MUTAG	BZR	DHFR	PROTEINS	ENZYMES	PTC
PSSK	0.868	0.606	0.557	0.668	0.281	0.545
PWGK	0.858	0.644	0.655	0.694	0.329	0.510
SWK	0.872	0.712	0.656	0.686	0.370	0.511
PFK	0.842	0.682	0.656	0.694	0.341	0.534
PI	0.863	0.585	0.519	0.691	0.285	0.542

Table 4: Balanced Accuracy related to graph datasets using shortest path distance (Accuracy only for ENZYMES dataset)

Thanks to these results two conclusions can be taken. The first one is that, as expected, the performance of the classifier depends on the particular filtration used for the computation of persistent features. The second one is related to the fact that SWK and PFK seem to work slightly better than the other kernels: in the case of

Kernel	MUTAG	BZR	DHFR	PROTEINS	ENZYMES	PTC
PSSK	0.865	0.704	0.717	0.675	0.298	0.490
PWGK	0.859	0.720	0.727	0.699	0.355	0.516
SWK	0.858	0.703	0.726	0.689	0.406	0.523
PFK	0.874	0.704	0.743	0.678	0.400	0.554
PI	0.846	0.670	0.712	0.690	0.280	0.478

Table 5: Balanced Accuracy related to graph datasets using Jaccard Index (Accuracy only for ENZYMES dataset)

shortest path distance, SWK is to be preferred while PFK seems work better in the case of Jaccard index. In the case of PROTEINS, in both cases PWGK provides best balanced accuracy.

6.4 1-Dimensional Time Series

In many different applications, one can deal with 1-dimensional time series. A 1dimensional time series is a set $\{x_t \in \mathbb{R} | t = 1, \ldots, T\}$. In [30] authors provide different approaches to build a filtration upon this kind of data. We decide to adopt the most common one. Thanks to the Taken's embedding, these data can be translated into point clouds. With suitable choices for two parameters: $\tau > 0$ the delay parameter and d > 0 the dimension, it is possible to compute a subset of points in \mathbb{R}^d composed by $v_i = \{x_i, x_{i+\tau}, \ldots, x_{i+(d-1)\tau}\}$ for $i = 1, \ldots, T - (d-1)\tau$. The theory mentioned above related to point clouds can now be applied to signals, as points in \mathbb{R}^d . For how to choose values for the parameters, see [30]. The dataset for tests is taken from the UCR Time Series Classification Archive (2018) [10], which consists of 128 datasets of time series from different worlds of application. In the archive there is the splitting into test and train sets but, for the aim of our analysis, we don't take care of it and we consider train and test data as a whole dataset and then codes provide properly the subdivision.

Dataset	N° time series	N° classes	IR
ECG200	200	2	133:67
SONY	621	2	349:272
DISTAL	876	2	539:337
STRAWBERRY	983	2	632:351
POWER	1096	2	549:547
MOTE	1272	2	685:587

Table 6: Time series datasets

Using giotto-tda we compute the persistent features of dimensions 0,1,2 and join them together. The final results of the datasets are reported here.

As in the previous examples, SWK is winning and provides slightly best performances in terms of accuracy.

7 Conclusions

In this paper, we have compared the performance of five Persistent Kernels applied to data of different nature. The results show how different PK are indeed comparable

Kernel	ECG200	SONY	DISTAL	STRAWBERRY	POWER	MOTE
PSSK	0.642	0.874	0.658	0.814	0.720	0.618
PWGK	0.726	0.888	0.696	0.840	0.769	0.633
SWK	0.731	0.892	0.723	0.898	0.784	0.671
PFK	0.707	0.895	0.676	0.892	0.750	0.652
PI	0.717	0.841	0.662	0.793	0.712	0.606

Table 7: Balanced Accuracy related to time series datasets

in terms of accuracy and there is not a PK that emerges clearly above the others. However, in many cases the SWK and PFK perform slightly better. In addition, from a purely computational point of view, SWK is to be preferred since, by construction, the preGram matrix is parameter-independent. Therefore, in practice, the user has to compute such a matrix on the whole dataset only once at the beginning and then choose a suitable subset of rows and columns to perform the training, cross-validation, and test phases. This aspect is relevant and reduces the computational costs and time compared with other kernels. Another aspect to be considered, as in the case of graphs, is how to choose the function f that provides the filtration. The choice of such a function is still an open problem and an interesting field of research. The right choice in fact would guarantee to be able to better extract the intrinsic information from data, improving, in this way, the classifier's performances. For the sake of completeness, we recall here that in literature there is also an interesting direction of reasearch whose aim is to build a new PK strarting from the main 5. From one of the PKs mentioned in previous sections, the authors in [12] studied how to modify them obtaining the so-called Variably Scale Persistent Kernels, which are Variably Scaled Kernels applied to the classification context. The results reported by the authors are indeed promising, thus it could be another interesting direction for further analysis.

Acknowledgments. This research has been achieved as part of RITA "Research ITalian network on Approximation" and as part of the UMI topic group "Teoria dell'Approssimazione e Applicazioni". The authors are members of the INdAM-GNCS Research group. The project has also been funded by the European Union-Next Generation EU under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 1.1 - Call PRIN 2022 No. 104 of February 2, 2022 of Italian Ministry of University and Research; Project 2022FHCNY3 (subject area: PE - Physical Sciences and Engineering) "Computational methods for Medical Imaging (CEMI)".

References

- H. Adams, T. Emerson, M. Kirby, R. Neville, C. Peterson, P. Shipman, S. Chepushtanova, E. Hanson, F. Motta, L. Ziegelmeier, *Persistence images: A stable vector representation of persistent homology*, Journal of Machine Learning Research 18 (2017)
- [2] M. E. Aktas, E. Akbas, A. El Fatmaoui, Persistent Homology of Networks: Methods and Applications, Appl Netw Sci 4, 61 (2019). https://doi.org/10.1007/s41109-019-0179-3
- [3] D. Ali, A. Asaad, M. Jimenez, V. Nanda, E. Paluzo-Hidalgo, M. Soriano-Trigueros A Survey of Vectorization Methods in Topological Data Analysis, IEEE Transac-

tions on Pattern Analysis and Machine Intelligence, Dec. 2023, pp. 14069-14080, vol. 45

- [4] A. Asaad, D. Ali, T. Majeed, R. Rashid, Persistent Homology for Breast Tumor Classification Using Mammogram Scans Mathematics 2022, 10(21), 4039; https://doi.org/10.3390/math10214039
- [5] D. Barnes, L. Polanco, J. A. Peres, A Comparative Study of Machine Learning Methods for Persistence Diagrams, Frontiers in Artificial Intelligence, (2021) Volume 4, Article 681174
- [6] R. Brüel-Gabrielsson, V. Ganapathi-Subramanian, P. Skraba, L. J. Guibas, Topology-Aware Surface Reconstruction for Point Clouds, arXiv:1811.12543
- [7] G. Carlsson Topology and data, Bulletin of the American Mathematical Society vol.46 n° 2 pp. 255-308 (2009)
- [8] M. Carriere, M. Cuturi, S. Oudot, Sliced Wasserstein kernel for persistent diagrams, International Conference on Machine Learning, PMLR 2017, pp.664-673
- [9] D. Cohen-Steiner, H. Edelsbrunner, J. Harer, *Stability of persistence diagrams*, Discrete & computational geometry 37 (1) (2007) 103–120.
- [10] H. A. Dau, E. Keogh, K. Kamgar, C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, Yanping, B. Hu, N. Begum, A.y Bagnall, A. Mueen, G. Batista, and Hexagon-ML, October 2018. https://www.cs.ucr.edu/ eamonn/time series data 2018/
- [11] S. De Marchi, F. Lot, F. Marchetti, Kernel-based mathods for persistent homology and their applications to Alzheimer's Disease (2021)
- [12] S. De Marchi, F. Lot, F. Marchetti, D. Poggiali, Variably Scaled Persistence Kernels (VSPKs) for persistent homology applications, Journal of Computational Mathematics and Data Science, Volume 4, August 2022
- [13] H. Edelsbrunner, J. Harer, Persistent homology a survey, Contemporary Mathematics, 453:257-282, 2008
- [14] H. Edelsbrunner, J. Harer, Computational topology: An introduction, American Mathematical Society, 2010
- [15] G. E. Fasshauer, *Meshfree approximation with MATLAB*, World scientific, (2007)
- [16] M. Flammer, Persistent Homology-Based Classification of Chaotic Multi-variate Time Series: Application to Electroencephalograms, SN COMPUT. SCI. 5, 107 (2024). https://doi.org/10.1007/s42979-023-02396-7
- [17] A. T. Fomenko, Visual geometry and topology, Springer Science and Business Media (2012)
- [18] A. Garin, G. Tauzin, A Topological "Reading" Lesson: Classification of MNIST using TDA International Conference on 18 October 2019 Computer Science, Mathematics
- [19] M. Grandini, E. Bagli, G. Visani, *Metrics for multi-class classification: an overview*,

- [20] M. Guillemard, A. Iske, Interactions between kernels, frames and persistent omology, Recent Applications of Harmonic Analysis to Function Spaces, Differential Equations and- Data Science, Springer pp. 861-888 (2017)
- [21] G. Kusano, K. Fukumizu, Y. Hiraoka, Kernel method for persistence diagrams via kernel embedding and weight factor, The Journal of Machine Learning Research vol. 18, n° 1 (2017), pp. 6947-6987
- [22] T. Le, M. Yamada, Persistence fisher kernel: A riemannian manifold kernel for persistence diagrams, arXiv preprint arXiv:1802.03569 (2018)
- [23] Y. LeCun and C. Cortes, MNIST handwritten digit database, 2010
- [24] D. Lee, S.-H. Lee, J.-H. Jung, The effects of topological features on convolutional neural networks—an explanatory analysis via Grad-CAM, Mach. Learn.: Sci. Technol. 4 (2023) 035019
- [25] S. Majumdar, A. K. Laha, Clustering and classification of time series using topological data analysis with applications to finance, Expert Systems with Applications, Volume 162, 30 December 2020, 113868 https://doi.org/10.1016/j.eswa.2020.113868
- [26] D. Pachauri, C. Hinrichs, M.K. Chung, S.C. Johnson, V. Singh, Topology based Kernels with Application to Inference Problems in Alzheimer's disease, in IEEE Transactions on Medical Imaging vol. 30, no. 10 (2011), pp. 1760-1770
- [27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830
- [28] J. M. Phillips, S. Venkatasubramanian, A Gentle* Introduction to the Kernel Distance arXiv preprint arXiv:1103.1625v2 (2011)
- [29] D. Pickup et al. SHREC' 14 Track: Shape Retrieval of Non-Rigid 3D Human Models, Proceedings of the 7th Eurographics workshop on 3D Object Retrieval, EG 3DOR'14. Eurographics Association, 2014
- [30] N. Ravinshanker, R. Chen An introduction to persistent homology for time series WIREs Computational Statistics, 04 February 2021 https://doi.org/10.1002/wics.1548
- [31] J. Reininghaus, S. Huber, U. Bauer, R. Kwitt A Stable Multi-Scale Kernel for Topological Machine Learning, Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4741-4748 (2015)
- [32] J. J. Rotman, An introduction to Algebraic Topology, Springer (1988)
- [33] N. Saul, C. Tralie, Scikit-tda: Topological data analysis for python (2019). doi:10.5281/zenodo.2533369. URL https://doi.org/10.5281/zenodo.2533369
- [34] B. Scholkopf, A. J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond, The MIT Press (2002)
- [35] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press (2009)

- [36] P. Sonego, M. Pacurar, S. Dhir, A. Kertész-Farkas, A. Kocsor, Z. Gáspári, J.A.M. Leunissen, S. Pongor, A Protein Classification Benchmark collection for machine learning Nucleic Acids Research, 2006, Vol. 00, Database issue D1–D5, doi:10.1093/nar/gkl812
- [37] J. Sun, M. Ovsjanikov, L. Guibas, A Coincise and Provably Informative Multi-Scale Signature Based on Heat Diffusion, 28-Issue 5 SGP09: Eurographics Symposium on Geometry Processing
- [38] C. Tralie, N. Saul, R. Bar-On, Ripser.py: A lean persistent homology library for python, The Journal of Open Source Software 3 (29) (2018) 925. doi:10.21105/joss.00925. URL https://doi.org/10.21105/joss.00925
- [39] J. Townsend, C. P. Micucci, J. H. Hymel, V. Maroulas, K. D. Vogiatzis Representation of molecular structures with persistent homology for machine learning applications in chemistry, Nat Commun 11, 3230 (2020). https://doi.org/10.1038/s41467-020-17035-5
- [40] H. Xiao, K. Rasul, and R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, arXiv preprint arXiv:1708.07747, 2017.
- [41] Q. Zhao, Y. Wang, Learning metrics for persistence-based summaries and applications for graph classification
- [42] The GUDHI Project, GUDHI User and Reference Manual, 3.5.0 Edition, GUDHI Editorial Board, 2022. URL https://gudhi.inria.fr/doc/3.5.0/
- [43] Giotto-tda 0.5.1 documentation https://giotto-ai.github.io/gtdadocs/0.5.1/library.html
- [44] https://quantdare.com/understanding-the-shape-of-data-ii/