ATOL: Measure Vectorization for Automatic Topologically-Oriented Learning

Martin Royer, Frédéric Chazal
Datashape, Inria Saclay, Palaiseau, France.

Clément Levrard
LPSM, Univ. Paris Diderot, Paris, France.

Umeda Yuhei, Ike Yuichi
Fujitsu Laboratories, AI Lab, Tokyo, Japan.

Abstract

Robust topological information commonly comes in the form of a set of persistence diagrams, finite measures that are in nature uneasy to affix to generic machine learning frameworks. We introduce a fast, learnt, unsupervised vectorization method for measures in Euclidean spaces and use it for reflecting underlying changes in topological behaviour in machine learning contexts. The algorithm is simple and efficiently discriminates important space regions where meaningful differences to the mean measure arise. It is proven to be able to separate clusters of persistence diagrams. We showcase the strength and robustness of our approach on a number of applications, from emulous and modern graph collections where the method reaches state-of-the-art performance to a geometric synthetic dynamical orbits problem. The proposed methodology comes with a single high level tuning parameter: the total measure encoding budget. We provide a completely open access software.

1 Introduction

Topological Data Analysis (TDA) is a field dedicated to the capture and description of relevant geometric or topological information from data. The use of TDA with standard machine learning tools has proved particularly advantageous in dealing with all sorts of complex data, meaning objects that are not or only partly Euclidean, for instance graphs, time series, etc. The applications are abundant, from social network analysis, bio and chemoinformatics, to physics, imaging and computer vision, to name a few. Recent examples include [Dindin et al., 2020], [Pike et al., 2019], [Duponchel, 2018], [Cole and Shiu, 2019], [Khasawneh et al., 2018]. Through Persistent Homology, a multi-scale analysis of the topological properties of the data, robust information is extracted. But the resulting features are commonly generated in the form of a persistence diagram whose structure does not easily fit the general machine learning input format. So TDA captures relevant information in a form that is challenging to handle – therefore it is generally combined to machine learning by way of an embedding method for persistence diagrams. This work is set in that trend.

Contributions. First we introduce a learnt, unsupervised vectorization method for measures in Euclidean spaces of any dimension (Section 2.1). Then we show how this method can be used for Topologically-Oriented Learning (Section 2.2), allowing for easy integration of topological features such as persistence diagrams into challenging machine learning problems. We illustrate our approach with sets of experiments that lead to state-of-the-art results on challenging problems (Section 3). We provide an open source implementation.

Our algorithm is simple and easy to use. It relies on a quantization of the space of diagrams that is statistically optimal. It is fast and practical for large scale and high dimensional problems. It is competitive and sometimes largely surpasses more sophisticated methods involving kernels, deep learning, or computations of Wasserstein distance. To the best of our knowledge, we introduce the first vectorization method for persis-

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

tence diagrams that is proven to be able to separate clusters of persistence diagrams. There is little to no tuning to this method, and no knowledge of TDA is required for using it.

Related work. Finding representations of persistence diagrams that are well-suited to be combined with standard machine learning pipeline is a problem that has attracted a lot of interest these last years. A first family of approaches consists in finding convenient vector representations of persistence diagrams. For instance it involves interpreting diagrams as images in [Adams et al., 2017], extracting topological signatures with respect to fixed points whose optimal position are supervisedly learnt in [Hofer et al., 2017], a square-root transform of their approximated pdf in [Anirudh et al., 2016]. Recently [Perea et al., 2019] introduced template functions, a mathematical framework to understand featurisation functions that integrates against the measure of a persistence diagram; our method is interpretable in this framework. A second family of approaches consists in designing specific kernel on the space of persistence diagrams, such as the multi-scale kernel of [Reininghaus et al., 2015], the weighted Gaussian kernel of [Kusano et al., 2016] or the sliced Wasserstein kernel of [Carrière et al., 2017]. Those techniques have state-of-the-art behaviour on problems, but for drawback they require another step for an explicit representation, and are known to scale poorly. A recent other line of work has managed to directly combine the uneasy structure of persistence diagrams to neural networks architectures [Zaheer et al., 2017], [Carrière et al., 2019]. Despite their successful performances, these neural networks are heavy to deploy and hard to understand. They are sometimes paired with a representation method as in [Hofer et al., 2017], [Hofer et al., 2019].

Persistent homology in TDA. Persistent homology provides a rigorous mathematical framework and efficient algorithms to encode relevant multi-scale topological features of complex data such as point clouds, time-series, 3D images... More precisely, persistent homology encodes the evolution of the topology of families of nested topological spaces $(F_\alpha)_{\alpha \in A}$, called filtrations, built on top of the data and indexed by a set of real numbers A that can be seen as scale parameters. For example, for a point cloud in a Euclidean space, F_α can be the union of the balls of radius α centered on the data points - see Figure 1. Given a filtration $(F_\alpha)_{\alpha \in A}$, its topology (homology) changes as α increases: new connected components can appear, existing connected components can merge, loops and cavities can appear or be filled, etc. Persistent homology tracks these changes, identifies features and associates, to each of them, an interval or lifetime from α_{birth} to α_{death}. For instance, a connected component is a feature that is born at the smallest α such that the component is present in F_α, and dies when it merges with an older connected component. As an interval can also be represented as a point in the plane with coordinates $(\alpha_{birth}, \alpha_{death})$, the set of intervals can be represented as an union of points called the persistence diagram - see [Edelsbrunner and Harer, 2010, Boissonnat et al., 2018] for a more detailed introduction.

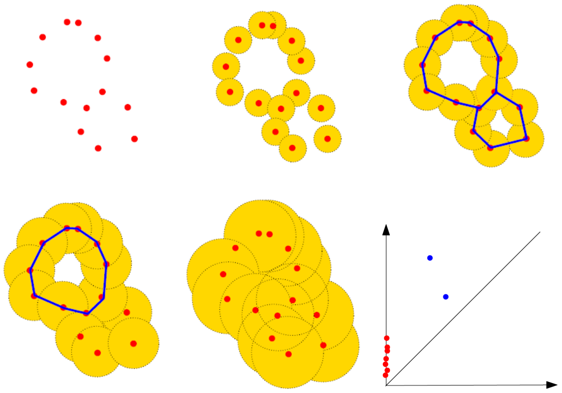

Figure 1: filtration by union of balls built on top of a 2-dimensional data set (red points) and its corresponding persistence diagram. As the balls radii increase (from left to right and top to bottom), the connected components (red points) are merged; two-cycles (blue points) appear and disappear along the filtration.

The classical main advantage of persistence diagrams is that: *(i)* they are proven to provide robust qualitative and quantitative topological information about the data [Chazal et al., 2016]; *(ii)* since each point of the diagram represents a specific topological feature with its lifespan, they are easily interpretable as features; *(iii)* from a practical perspective, persistence diagrams can be efficiently computed from a wide family of filtrations [The GUDHI Project, 2020]. However, as persistence diagrams come as unordered set of points with non constant cardinality, they cannot be immediately processed as standard vector features in machine learning algorithms. Considering diagrams as measures has proven beneficial in the literature before (see for instance [Chazal et al., 2016], [Chazal and Divol, 2018]) and allows to naturally encode the points multiplicity problems in the form of weighted measures.

Notations. Consider $\mathcal{M}_d$ the set of finite measures on the d-dimensional ball $\mathcal{B}_d(0, R)$ of the Euclidean space $\mathbb{R}^d$ with total mass smaller than M, for some given $M, R \in \mathbb{R}_+^2$. For $m \in \mathcal{M}_d$ and $\chi : \mathbb{R}^d \to \mathbb{R}$ borelian function, let $\chi \cdot m := \int_{x \in \mathbb{R}^d} \chi(x) m(dx)$ whenever $|\chi| \cdot m$

is finite.

Next, for $b \in \mathbb{N}^*$ we call a codebook $\mathbf{c} = (c_1, \dots, c_b) \in \mathcal{B}_d(0, R)^b$ the support of a distribution supported on b points and its associated Voronoï cells: $\mathrm{Vor}_j(\mathbf{c}) = \{x \in \mathbb{R}^d \mid \forall i < j, \|x - c_j\| < \|x - c_i\| \text{ and } \forall i > j, \|x - c_j\| \leqslant \|x - c_i\|\}$.

Finally, we assume that the set of input persistence diagrams comes as an i.i.d. sample from a distribution of uniformly bounded diagrams, that is given $M, R \in \mathbb{R}_+^2$, let $\mathcal{D}$ be the space of persistence diagrams with at most M points contained in the Euclidean disc $\mathcal{B}_2(0, R)$. The space $\mathcal{D}$ is considered as a subspace of the set $\mathcal{M}_2$ of finite measures on $\mathcal{B}_2(0, R)$ with total mass smaller than M: for any $D \in \mathcal{D}$, $D := \sum_{p \in D} \delta_p$ where δ_p is the Dirac measure centered at point p.

2 Methodology

In this section we introduce Atol, a simple unsupervised data-driven method for measure vectorization. Atol allows to automatically convert a distribution of persistence diagrams into a distribution of feature vectors that are well-suited for use as topological features in standard machine learning pipelines.

As an overview, given a positive integer b, Atol proceeds in two steps: it computes a discrete measure in $\mathbb{R}^d$ supported on b points that approximates the average measure of the distribution from which the input observations have been sampled. Then, it computes a set of well-chosen contrast functions centered on each point of the support of this measure, that are used to convert each observed measure into a vector of size b. This resulting vectorization can then be used in standard machine-learning problems such as clustering, classification, etc.

2.1 Measure vectorization through quantization

We now introduce Algorithm 1 Atol-featurisation: a featurisation method for elements of $\mathcal{M}_d$. The first step in our procedure is to use quantization in space $\mathcal{M}_d$. Starting from an i.i.d. sample of measures $X_1, \dots, X_n$ drawn from probability distribution $\mathcal{L}_X$ on $\mathcal{M}_d$ and given an integer budget $b \in \mathbb{N}^*$, we produce a compact representation for the mean measure $\mathbb{E}(X)$. That is, we produce a distribution $P_{\hat{\mathbf{c}}_n}$ supported on a fixed-length codebook $\hat{\mathbf{c}}_n = (c_1, \dots, c_b) \in \mathcal{B}_d(0, R)^b$ that aims to minimize over such distributions P based on b points the distorsion $R(P) := W_2^2(P, \mathbb{E}(X))$: the squared 2-Wasserstein distance to the mean measure. In practice, one considers the empirical mean measure $\bar{X}_n$ and the k-means problem for this $\bar{X}_n$ measure. Then the adaptation of Lloyd's [Lloyd, 1982] algorithm to measures can be used.

From this quantization our aim is to derive spatial information on measures in order to discriminate them. Much like one would compactly describe a point cloud with respect to its barycenter in a PCA procedure, we describe measures based on a number of reduced difference to our mean measure approximate. To this end, our second step is to tailor b individual contrast functions each based on the estimated codebook that individually describe the space with respect to a certain viewpoint. In other words we set to find regions of the space where measures seem to aggregate on average, and build a dedicated descriptor for those regions. We define and use the following contrast family $\mathbb{R}^d \to \mathbb{R}_+$, for $i \in [b]$:

$$\Psi_i(\cdot, \hat{\mathbf{c}}_n) : x \mapsto \exp\Big[-\frac{\|x - c_i\|_2}{\sigma_i(\hat{\mathbf{c}}_n)}\Big], \tag{1}$$

$$\text{with} \qquad \sigma_i(\hat{\mathbf{c}}_n) := \min_{j \in [b], j \neq i} \|c_i - c_j\|_2 / 2. \tag{2}$$

These specific contrast functions are chosen to decrease away from the approximate mean centroid in a Laplacian fashion and we choose the scale to correspond to the minimum distance to the closest Voronoi cell in the corresponding codebook $\hat{\mathbf{c}}_n$. To our knowledge there is nothing that prevents other, well designed contrast families to be substituted in their place, and to that regard some intuition is provided in the ablation study of Section 3.2.

Given a mean measure codebook approximate $\hat{\mathbf{c}}_n$, element $X \in \mathcal{M}_d$ can now be compactly described through the integrated contribution to each contrast functions: $\Psi_i(\cdot, \hat{c}_{n,b}) \cdot X$. Our algorithm concatenates into a vector each of those contributions.

It is worth mentioning that the ATOL-vectorization scheme can be straightforwardly applied to weighted persistence diagrams – weigthed sums of Dirac masses. Such a case is encoutered whenever some diagram points are interpreted as topological noise: for instance points that are close to the diagonal would be assigned smaller weights than others.

This algorithm is practical for large scale and high dimensional problems: it has a running time in $O(n \times M \times b \times d)$, therefore it is able to handle difficult problems as long as corresponding measures are found. If necessary, a single-pass quantization step can be derived as a minibatch adaptation of the [MacQueen, 1967] MacQueen algorithm (we refer to [Chazal et al., 2020]), and then combined with the contrast functions vectorization. Therefore Algorithm 1 is simple, fast and automatic once the desired length for vectorization has been chosen. Now let us introduce how it appears in machine-learning contexts.

Algorithm 1: Atol-vectorization

```
Data: Collection of measures
      $X_1, \dots, X_n \in (\mathcal{M}_d)^n$.
Parameters: budget $b \in \mathbb{N}^*$.
Quantization algorithm of the mean measure with
  fixed-length support (Lloyd's adaptation to
  measures): sample $\mathbf{c} = (c_1, \dots, c_b)$ from $\bar{X}_n$. ;
while $\mathbf{c}^{new} = (c_1^{new}, \dots, c_b^{new}) \neq \mathbf{c}$ do
    $\mathbf{c} = \mathbf{c}^{new}$ ;
    $\forall i \in [b], \quad c_i^{new} :=$
      $\left[u \mapsto u 1_{\mathrm{Vor}_i(\mathbf{c})}(u)\right] \cdot \frac{1}{\bar{X}_n(\mathrm{Vor}_i(\mathbf{c}))} \bar{X}_n$ ;
let $\hat{\mathbf{c}}_n$ be the resulting codebook, define the $b$
  measurable contrast functions
  $(\Psi_1(\cdot, \hat{\mathbf{c}}_n), \dots, \Psi_b(\cdot, \hat{\mathbf{c}}_n))$ to compute
  featurisation map:
  $v_{\text{Atol}} : X \mapsto \Big[\Psi_i(\cdot, \hat{\mathbf{c}}_n) \cdot X\Big]_{i \in [b]}$. ;
Result: vectorization map $v_{\text{Atol}} : \mathcal{M}_d \to \mathbb{R}^b$.
```

2.2 Topological learning with Atol

Set in the context of a standard learning problem, we introduce Algorithm 2 Atol: Automatic Topologically-Oriented Learning. Let $\Omega := (X, y)$ with given observations X in some space $\mathcal{X}$ corresponding to a known, partially available or hidden label $y \in \mathcal{Y}$. Assume that one has a way to extract toplogical features from $\mathcal{X}$ (for example a collection of diagrams associated to those elements), and let $\kappa : \mathcal{X} \to \mathcal{M}_d$ be the corresponding map. Then applying Algorithm 1 to the resulting collection of descriptors provides some simplified topological understanding on elements X of this problem.

Algorithm 2: Atol: Automatic Topologically-Oriented Learning

```
Data: Learning problem $\Omega := (X, y)$ with $X \in \mathcal{X}$
      collections and $y \in \mathcal{Y}$ labels.
Parameters: $\kappa : \mathcal{X} \to \mathcal{M}_d$ yielding topological
            descriptors, and budget $b \in \mathbb{N}^*$.
Compute intermediate learning problem
  $\Omega_{\text{Topo}} := ((X, \kappa(X)), y) \in (\mathcal{X} \times \mathcal{M}_d) \times \mathcal{Y}$ with
  topological features, potentially unfit for general
  machine learning routines;
Use Algorithm 1 to derive Euclidean
  representations of those features, i.e. transform
  $\Omega_{\text{Topo}}$ into a generic machine learning problem
  $\widetilde{\Omega} := ((X, v_{\text{Atol}} \circ \kappa(X)), y) \in (\mathcal{X} \times \mathbb{R}^b) \times \mathcal{Y}$. ;
Result: Enhanced problem
        $\widetilde{\Omega} := ((X, v_{\text{Atol}} \circ \kappa(X)), y)$ where
        $v_{\text{Atol}} \circ \kappa(X) \in \mathbb{R}^b$.
```

This algorithm is integrated in the open source topological library GUDHI [The GUDHI Project, 2020] accessible at `https://gudhi.inria.fr/python/latest/representations.html`. We point out that as embedding map v_{Atol} is automatically computed without knowledge of a learning task, its derivation is fully unsupervised. The representation is learned since it is data-dependent, but it is also agnostic to the task and eventually only depends on getting a glimpse at an average persistence diagram.

Atol in dimension 2 for persistence diagrams. We now specialise this algorithm to the context of persistent homology that is usually set in dimension $d = 2$. Applying Algorithm 2 to a collection from $\mathcal{M}_2$ such as persistence diagrams, as $\mathcal{D} \subset \mathcal{M}_2$, is straightforward and allows to embed the complex, unstructured space $\mathcal{M}_2$ in Euclidean terms.

Now let us assume that the measures in $\mathcal{M}_2$ come from distinct sources: that observed measures $D_1, \dots, D_n$ are sampled with noise from a mixture model $D = \sum_{l=1}^{L} \pi_l D^{(l)}$ of distinct measures $D^{(1)}, \dots, D^{(L)}$ (by that we mean that any two measures in this set differ in support by at least one point). Let Z the latent variable of the mixture so that $D | Z = l \sim D^{(l)}$. The following results ensures that v_{Atol} has separative power, i.e. that the vectorization clearly separates the different sources:

Theorem 1 (Separation with Atol)**.**
For a given noise level assuming $\mathbb{E}(D)$ satisfies some (explicit) margin condition and for n and b large enough there exists a non-empty segment for $\sigma_1, \dots, \sigma_b$ in Equation (1) *such that for all $i, j \in [n]^2$, with high probability:*

$$Z_i = Z_j \implies \|v_{\text{Atol}}(D_i) - v_{\text{Atol}}(D_j)\|_\infty \leqslant 1/4, \quad (3)$$

$$Z_i \neq Z_j \implies \|v_{\text{Atol}}(D_i) - v_{\text{Atol}}(D_j)\|_\infty \geqslant 1/2. \quad (4)$$

Since the contrast functions Ψ_i's are smooth, our vectorization scheme is inherently stable with respect to Wasserstein W_1 perturbations. The first point of Theorem 1 builds upon this remark, for a suitable choice of bandwiths.

To our knowledge it is the first time that a measure vectorization method (or a persistence diagram vectorization method) has been proven to separate clusters. This result follows from Corollary 19 in [Chazal et al., 2020] that studies theoretical properties of Atol-like procedures. The explicit statement of the assumptions and margin conditions are standard and rather technical.

But the theory behind it uses an idealistic framework (including the so-called margin condition) under which such procedures will succeed in separating different sources. Based on this framework, the requirements of Theorem 1 cannot be checked in practice: apart from the technical margin condition, the prescribed bounds on budget b are unknown and they theoretically grow quite large with the number of underlying centers + covering number, and the bounds for bandwidths $\sigma_1, \ldots, \sigma_b$ are heavily dependent on the structure of source model D.

In practice we prove it needs not be so difficult: for intuition we refer to the ablation study on the influence of b exposed in Table 3, that shows it is easy to choose a low budget for efficient results – so we leave it as the only parameter of the algorithm. For full automaticity, a simple adaptive strategy would be to try a range of budgets during the training task, since the combination of Algorithm 1 and a standard learning algorithm such as random forests runs very fast. Furthermore, the adaptive strategy of Equation (2) for bandwidths $\sigma_1, \ldots, \sigma_b$ proves efficient, see the bandwiths variation study of Section 3.2 Figure 3.

In dimension 2 this vectorization is conceptually close to two other recent works. [Hofer et al., 2017] computes a persistence diagram vectorization through a deep learning layer that adjusts Gaussian contrast functions used to produce topological signatures. So in essence our approach substitutes quantization to deep learning, with no need of supervision and allowing to provide mathematical guarantees. Next, the bag of word method of [Zieliński et al., 2019] uses an ad-hoc form of quantization for the space of diagrams, then count functions as contrast functions to produce histograms as topological signatures. Those are in fact sensible differences, that will ultimately translate in terms of effectiveness: Section 3.1 shows the Atol-featurisation to produce state-of-the-art mean accuracy on two difficult multi-class classification problems (67.1 % on `REDDIT5K` and 51.4 % on `REDDIT12K`) that are also analysed by those papers: [Hofer et al., 2017] report a mean accuracy of respectively 54.5% and 44.5%, and [Zieliński et al., 2019] report an accuracy of respectively 49.9% and 38.6%.

3 Competitive TDA-Learning

In this section we show the Atol framework to be competitive, sometimes greatly improving the state-of-the-art, but also versatile and easy to use. This section presents experiments on two sorts of classification problems (graphs and point clouds), and another applied experiment on time series is provided in the Supplementary Materials.

Algorithm 2 transforms the initial problems into a typically standard machine-learning problem, so the problem although transformed remains to be solved. In the instances below we use the `scikit-learn` [Pedregosa et al., 2011] random-forest classification tool with 100 trees and all other parameters set as default. We use random forests as a ready-to-use tool, but comparable performances can be obtained from using a linear SVM classifier or a neural network classifier, depending on the problem. It is a light choice that requires no particular infrastructure or tuning efforts that would produce overly design-dependent results – as our ambition is to show an ability to perform well overall.

3.1 Graph Classification

As learning problems involving graph data are receiving a strong interest at the moment, consider a standard graph classification framework: $\Omega := (G, y) \in \mathcal{G} \times \mathcal{Y}$ is a finite family of graphs and available labels and one learns to map $\mathcal{G} \to \mathcal{Y}$.

Recently [Carrière et al., 2019] have introduced a powerful way of extracting topological information from graph structures. They make use of heat kernel signatures (HKS) for graphs [Hu et al., 2014], a spectral family of signatures (with diffusion parameter $t > 0$) whose topological structure can be encoded in the extended persistence framework, yielding four types of topological features with exclusively finite persistence. We replicate their methodology, and on both HKS and extended persistence we refer to Sections 4.2 and 2 from [Carrière et al., 2019]. Schematically for diffusion time $t > 0$ and graph $G(V, E)$ (with V, E the sets of vertices and edges), the topological descriptors are computed as:

$$\kappa_t := g \circ h, \tag{5}$$

where

$$g : G(V, E) \in \mathcal{G} \xrightarrow[\text{signatures}]{\text{heat kernel}} \mathrm{HKS}_t(G) \in \mathbb{R}^{|V|}, \tag{6}$$

$$h : \mathrm{HKS}_t(G) \xrightarrow[\text{persistence}]{\text{extended}} \mathrm{PD}(\mathrm{HKS}_t(G)) \in \mathcal{D}^4. \tag{7}$$

For the entire set of problems to come we choose to use the same two HKS diffusion times to be $t_1 = .1$ and $t_2 = 10$, so that Algorithm 2 is used with topological map $\kappa := \kappa_{t_1} + \kappa_{t_2} : \mathcal{G} \to \mathcal{D}^8$, such that all in all 8 persistence diagrams are computed and considered per graph. For budget in Algorithm 2 we choose $b = 80$ for all experiments, which means Algorithm 1 will rely on approximating the mean measure on ten points per diagram type and HKS time filtration. We make no use of graph attributes on edges or vertices that some datasets do possess, and no other sort of features are collected, so that our results are solely based on the toplogical graph structure of the problems. To sum-up, Algorithm 2 here simply consists in reducing

method	RetGK	FGSD	WKPI	GNTK	PersLay	Atol
problem						
REDDIT (5K, 5 classes)	56.1±.5	47.8	59.5±.6	—	55.6±.3	**67.1±.3**
REDDIT (12K, 11 classes)	48.7±.2	—	48.5±.5	—	47.7±.2	**51.4±.2**
COLLAB (5K, 3 classes)	81.0±.3	80.0	—	83.6±.1	76.4±.4	**88.3±.2**
IMDB-B (1K, 2 classes)	71.9±1.	73.6	75.1±1.1	**76.9±3.6**	71.2±.7	74.8±.3
IMDB-M (1.5K, 3 classes)	47.7±.3	**52.4**	48.4±.5	**52.8±4.6**	48.8±.6	47.8±.7

Table 1: Mean accuracy and standard deviations for Large Social Network datasets.

the original problem from Ω to $\widetilde{\Omega} := (v_{\text{Atol}} \circ \kappa(G), y)$ with $v_{\text{Atol}} \circ \kappa(G) \in \mathbb{R}^{80}$. The embedding map v_{Atol} from Algorithm 1 is computed using only 10% of all diagrams from the training set, without supervision.

On each problem we perform a 10-fold cross-validation procedure and average the resulting accuracies; we report accuracies and standard deviations over ten such experiments. We use two sets of graph classification problems for benchmarking, one of Social Network origin and one of Chemoinformatics and Bioinformatics origin. They include small and large sets of graphs (MUTAG has 188 graphs, REDDIT12K has 12000), small and large graphs (IMDB-M has 13 nodes on average, REDDIT5K has more than 500), dense and sparse graphs (FRANKENSTEIN has around 12 edges per nodes, COLLAB has more than 2000), binary and multi-class problems (REDDIT12K has 11 classes), all available in the public repository [Kersting et al., 2016]. Computations are run on a single laptop (i5-7440HQ 2.80 GHz CPU), in batch version for datasets smaller than a thousand observations and mini-batch version otherwise. Average computing time of Algorithm 1 (the average time to calibrate the vectorization map on the training set then compute the vectorization on the entire dataset), are: less than 1 second for datasets with less than a thousand observations, less than 5 seconds for datasets that have less than 5 thousand observations, 7.5 seconds for REDDIT-5K, and less than 16 seconds for the largest REDDIT-12K and the densest problem COLLAB.

We compare performances to the top scoring methods for these problems, to the best of our knowledge. Those methods are mostly graph kernel methods tailored to graph problems: two graph kernel methods based on random walks (RetGK1, RetGK11 from [Zhang et al., 2018]), one graph embedding method based on spectral distances (FGSD from [Verma and Zhang, 2017]), two topological graph kernel method (WKPI-kM and WKPI-kC from [Zhao and Wang, 2019]), one graph kernel combined with a graph neural network (GNTK from [Du et al., 2019]). Finally PersLay from [Carrière et al., 2019] is a topological vectorization method learnt by a neural network that encodes most topological frameworks from the literature – landscapes, silhouettes, persistence images, etc. In particular [Carrière et al., 2019] demonstrated that the aforementioned method of [Hofer et al., 2017] can be replicated by and compared with the PersLay framework and results.

As for Atol the comparisons to PersLay were computed with the exact same persistence diagrams input – except for a few cases where the authors used those same two HKS diffusion times then discarded one with no loss of performances. The total budget for Atol ($b = 80$) is a magnitude below that required to build the PersLay architecture (several hundreds nodes before the optimisation phase). Competitor accuracy are quoted from their respective publication and should be interpreted as follows: for RetGK and WKPI and PersLay the evaluating procedure is done over ten 10-fold, just as ours is so the results directly compare; for FGSD the average accuracy over a single 10-fold is reported, and for GNTK the average accuracy and deviations is reported over a single 10-fold as well. When there are two or more methods under one label (e.g. RetGK1 and RetGK11), we always favorably reported the best outcome.

Our results Table 1 are state-of-the-art or substantially improving the state-of-the-art on the Large Social Network datasets that are difficult multi-class problems. The REDDITs and COLLAB datasets all see major improvements in the mean accuracies, and those three datasets can readily be considered the most difficult problems (by size, graph density and number of classes) in the entire series.

The results on the Chemoinformatics and Bioinformatics datasets Table 2 are on par with or sometimes sub state-of-the-art, with a significant achievement on DHFR. Further comparison with [Zhao and Wang, 2019] is instructive: using discrete Ricci curvatures (or the Jaccard function) instead of HKS functions with Atol improves performances but never more than by 1 or 2 percentage point, although [Zhao and Wang, 2019] report results that are 5 to 10 points higher. Therefore Atol is not limited by the type of input measures but by its analysis of contrasting to the mean-measure.

method problem (size)	RetGK	FGSD	WKPI	GNTK	PersLay	Atol
MUTAG (188)	90.3±1.1	**92.1**	88.3±2.6	90.0±8.5	89.8±.9	88.3±.8
COX2 (467)	**81.4±.6**	—	—	—	80.9±1.	79.4±.7
DHFR (756)	81.5±.9	—	—	—	80.3±.8	**82.7±.7**
PROTEINS (1113)	**78.0±.3**	73.4	**78.5±.4**	75.6±4.2	74.8±.3	71.4±.6
NCI1 (4110)	84.5±.2	79.8	**87.5±.5**	84.2±1.5	73.5±.3	78.5±.3
NCI109 (4127)	—	78.8	**87.4±.3**	—	69.5±.3	77.0±.3
FRNKNSTN (4337)	**76.4±.3**	—	—	—	70.7±.4	72.9±.3

Table 2: Mean accuracy and standard deviations for Chemoinformatics and Bioinformatics datasets, all binary classification problems.

It is not surprising that Atol is not always on par with the state-of-the-art, especially on the smaller binary classification datasets where considering the mean measure can potentially be too simple a model, easily refined upon – recall that contrary to competitors, Atol does not build a kernel or a neural network. Quantisation without supervision makes the learning process stricter, heavily dependent on the measure input: Atol is only capable of interpreting behavior with respect to the mean measure, therefore if some discriminant feature in a problem is found at a border of the measure space, as could be happening on the PROTEINS and NCIs datasets, it shall not be captured and there is no learning-room to change that. This is a liability, as well as a virtue of the method.

We surmise that the Atol performances can be interpreted as a general optimal score for discriminative capacity with respect to the mean, in a problem. So for instance on the IMDB datasets, potentially whatever is gained on top of this baseline is obtained through supervision, at the cost of general discriminative power. The performance differences between Tables 1 and 2 can also suggest that the bio-chemical problems are less stochastic in essence than the social graphs network ones.

The simplicity and absence of tuning indicate robustness and generalisation power. Overall these results are especially positive seeing how Algorithm 1 has been employed with a universal configuration.

3.2 A measured look at discrete dynamical systems

We now show the modularity capacity of the Atol framework, as well as its efficiency in compactly encoding information. [Adams et al., 2017] use a synthetic, discrete dynamical system (used to model flows in DNA microarrays) with the following property: the resulting chaotic trajectories exhibit distinct topological characteristics depending on a parameter $r > 0$.

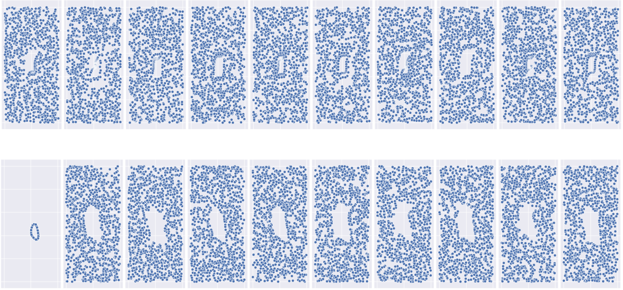

Figure 2: Synthetised orbits x, y coordinates in $[0,1]^2$ for parameter 4.0 (top) and 4.1 (bottom).

The dynamical system is: $x_{n+1} := x_n + r y_n(1 - y_n) \mod 1$, and $y_{n+1} := y_n + r x_{n+1}(1 - x_{n+1}) \mod 1$. With random initialisation and five different parameters $r \in \{2.5, 3.5, 4, 4.1, 4.3\}$, a thousand iterations per trajectory and a thousand orbits per parameter, a datasets of five thousand orbits is built. Figure 2 shows a few orbits generated with parameters $r \in \{4.0, 4.1\}$. For orbits generated with parameter $r = 4.1$, it happens that the initialisation spawns close to an attractor point that gives it the special shape as in the leftmost orbit. The problem of classifying this datasets according to their underlying parameter is rather uneasy and challenging. This dataset is commonly used for evaluating topological methods under the following experimental setup: a learning phase with a 70/30 split, and accuracy with standard deviation computed over a hundred such experiments. The state-of-the-art accuracy of 87.7±1.0 with persistence diagrams is reported in [Carrière et al., 2019].

Since those discrete orbits can be seen as measures in $[0,1]^2$, we instead decide to directly apply Algorithm 2 on the observed point cloud, using the modularity of our framework – so in this instance κ is the identity map. Therefore Atol is used here as a purely spatial approach and in this context, it is alike an image classification algorithm where instead of a fixed grid we have learnt center points to perform measurements. We present results in the form of a short ablation study Table 3 designed to illustrate influence of the small number of parameters from Algorithm 1.

In this study we consider varying parameter $b \in \mathbb{N}^*$ for describing the measure space; replacing contrast functions Ψ_i in Equation (1) with $\Phi_i(\cdot, \hat{\mathbf{c}}_n) : x \mapsto \exp\left[-\|x - c_i\|_2^2/\sigma_i^2(\hat{\mathbf{c}}_n)\right]$ for vectorization of the quantised space; and lastly changing the proportion of training observations used for deriving the quantization, with 10% indicating that a random selection of a tenth of the measures from the training set were used to calibrate Algorithm 1. We measure accuracies over 10 70/30 splits and for comparison purpose we also compute results for a 2D-grid quantization scheme labeled `grid`, that uses the same contrast family and a regular grid of size $\lfloor\sqrt{b}\rfloor \times \lfloor\sqrt{b}\rfloor$.

It is expected that a higher budget for vectorising the measure space will yield a better description of said space, and this intuition is confirmed by Table 1. Although the differences are small, there is a slight advantage for operating ATOL than a fixed `grid` at lower budgets, which is coherent with the intuition that the mean measure performs better than other procedures as a first approximation. Next, there does not seem to be significant differences from using Gaussian over Laplacian contrast functions on this experiment, although it can be the case on other problems. Understanding the ability of such contrast functions to describe some particular observation space is challenging and left for future work. The percentage of observations used in the calibration part of the algorithm does not have a strong influence on the final result either (it does have a significant influence when the budget is lower then 80). This tells us that the calibration step is rather stable for a given level of information in a problem, and that our procedure is well-designed for dealing with problems online. Finally we report that when using budgets greater than 250 (i.e. finer than 12 × 12 for the regular-grid), both methods reach comparable mean accuracies that are over 95%. This indicates that this problem can be precisely described by a purely spatial approach, without topological descriptors.

Lastly, using the default parameters in Table 3 we compare the adaptive strategy of Equation (2) for bandwiths $\sigma_1, \dots, \sigma_b$ to using identical constant values for those bandwiths. For investigating constant values we use the array $\mu \times 10^{[-2,-1.5,-1,-.5,-.2,-.1,0,.1,.2,.5,1,1.5,2]}$ where μ is the average distance between codebook points of $\hat{\mathbf{c}}_n$. The results are shown Figure 3. This experiment shows that if a constant value for bandwiths can be found for optimal results, the adaptive strategy introduced in this paper already produces competitive results effortlessly.

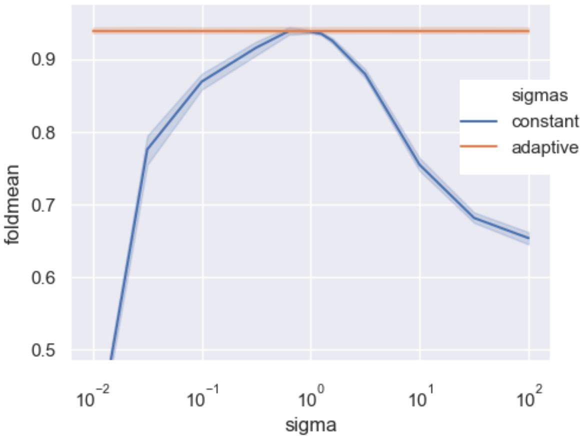

Figure 3: Classification accuracy and deviations for `Orbit5K` as $\sigma_1, \dots, \sigma_b = \sigma$ is varied (in blue), compared to the adaptive strategy of Equation (2) (orange).

4 Conclusion

This paper introduces an unsupervised vectorization method for measures in Euclidean spaces based on optimal quantization procedures, then shows how this method can be employed in machine learning contexts to exploit topological information. ATOL is fast with a simple design, multifaceted and ties theoretical guarantees to practical efficiency. In particular it does not a specific design of noise: it can be externally introduced through the use of weights, or by adapting robust quantization algorithms such as [Garcia-Escudero and Gordaliza, 1999] to the collection of measures case – this falls beyond the scope of the paper.

From a practical viewpoint, we can guess that our method is less prone to bias and overfitting for two reasons: the centers are designed unsupervisedly (thus no possible overfitting) and the dimension of our vectorization is credibly low. Furthermore, effective insight can be gained as the method is interpretable: once the mean measure is computed one can observe its location (e.g. for diagrams, are centers close to the diagonal or not) and further derive information from it (e.g. in a classification task, center importance). For instance on a using a particular dataset of diagrams, one can learn if low persistence points are meaningful signal or not, with no preconceived hypothesis. Lastly ATOL only depends on a simple parameter: the size b of the codebook.

	Budget				Contrast functions		Calibration	
	$b = 4$	$b = 16$	$b = 36$	$b = 100$	Φ-Gaussian	Ψ-Laplacian	10%	100%
Atol	56.3±1.6 2.4 s	83.1±2.2 3.1 s	89.6±1.3 5.5 s	93.8±.8 12.7 s	93.8±.5 12.7 s	93.8±.8 12.7 s	93.8±.8 12.7 s	93.6±.4 50.2 s
grid	55.8±1.1	82.7±.8	88.9±1.0	93.8±.7	94.2±.5	93.8±.7	93.8±.7	93.8±.7

Table 3: Mean accuracy, deviation and vectorization time (including calibration step) over 10 experiments for ORBIT5K. Blue indicate parameters by default; only one parameter is varied at a time.

Acknowledgment

One of the author was partly funded by the French ANR Chair in AI TopAI.

References

[Adams et al., 2017] Adams, H., Emerson, T., Kirby, M., Neville, R., Peterson, C., Shipman, P., Chepushtanova, S., Hanson, E., Motta, F., and Ziegelmeier, L. (2017). Persistence images: a stable vector representation of persistent homology. *Journal of Machine Learning Research*, 18(8).

[Anirudh et al., 2016] Anirudh, R., Venkataraman, V., Ramamurthy, K. N., and Turaga, P. (2016). A riemannian framework for statistical analysis of topological persistence diagrams. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1023–1031.

[Boissonnat et al., 2018] Boissonnat, J.-D., Chazal, F., and Yvinec, M. (2018). *Geometric and Topological Inference*, volume 57. Cambridge University Press.

[Carrière et al., 2019] Carrière, M., Chazal, F., Ike, Y., Lacombe, T., Royer, M., and Umeda, Y. (2019). PersLay: A Simple and Versatile Neural Network Layer for Persistence Diagrams. *AISTATS 2020*, page arXiv:1904.09378.

[Carrière et al., 2017] Carrière, M., Cuturi, M., and Oudot, S. (2017). Sliced Wasserstein kernel for persistence diagrams. In *International Conference on Machine Learning*, volume 70, pages 664–673.

[Chazal et al., 2016] Chazal, F., de Silva, V., Glisse, M., and Oudot, S. (2016). *The structure and stability of persistence modules.* SpringerBriefs in Mathematics. Springer.

[Chazal and Divol, 2018] Chazal, F. and Divol, V. (2018). The density of expected persistence diagrams and its kernel based estimation. In *SoCG 2018 - Symposium of Computational Geometry*, Budapest, Hungary. Extended version of the SoCG proceedings, submitted to a journal.

[Chazal et al., 2020] Chazal, F., Levrard, C., and Royer, M. (2020). Optimal quantization of the mean measure and application to clustering of measures. *arXiv (preprint)*, page arXiv:2002.01216.

[Cole and Shiu, 2019] Cole, A. and Shiu, G. (2019). Topological data analysis for the string landscape. *Journal of High Energy Physics*, 2019.

[Dindin et al., 2020] Dindin, M., Umeda, Y., and Chazal, F. (2020). Topological data analysis for arrhythmia detection through modular neural networks. In Goutte, C. and Zhu, X., editors, *Advances in Artificial Intelligence*, pages 177–188, Cham. Springer International Publishing.

[Du et al., 2019] Du, S. S., Hou, K., Salakhutdinov, R. R., Poczos, B., Wang, R., and Xu, K. (2019). Graph neural tangent kernel: Fusing graph neural networks with graph kernels. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 5724–5734. Curran Associates, Inc.

[Duponchel, 2018] Duponchel, L. (2018). Exploring hyperspectral imaging data sets with topological data analysis. *Analytica Chimica Acta*, 1000:123 – 131.

[Edelsbrunner and Harer, 2010] Edelsbrunner, H. and Harer, J. (2010). *Computational Topology: An Introduction.* AMS.

[Garcia-Escudero and Gordaliza, 1999] Garcia-Escudero, L. A. and Gordaliza, A. (1999). Robustness properties of k means and trimmed k means. *Journal of the American Statistical Association*, 94(447):956–969.

[Hofer et al., 2019] Hofer, C., Kwitt, R., and Niethammer, M. (2019). Graph filtration learning.

[Hofer et al., 2017] Hofer, C., Kwitt, R., Niethammer, M., and Uhl, A. (2017). Deep learning with topological signatures. In *Advances in Neural Information Processing Systems*, pages 1634–1644.

[Hu et al., 2014] Hu, N., Rustamov, R., and Guibas, L. (2014). Stable and informative spectral signatures for graph matching. In *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, pages 2305–2312.

[Kersting et al., 2016] Kersting, K., Kriege, N. M., Morris, C., Mutzel, P., and Neumann, M. (2016). Benchmark data sets for graph kernels. `http://graphkernels.cs.tu-dortmund.de`.

[Khasawneh et al., 2018] Khasawneh, F. A., Munch, E., and Perea, J. A. (2018). Chatter classification in turning using machine learning and topological data analysis. *IFAC-PapersOnLine*, 51(14):195 – 200. 14th IFAC Workshop on Time Delay Systems TDS 2018.

[Kusano et al., 2016] Kusano, G., Hiraoka, Y., and Fukumizu, K. (2016). Persistence weighted Gaussian kernel for topological data analysis. In *International Conference on Machine Learning*, volume 48, pages 2004–2013.

[Lloyd, 1982] Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Trans. Inform. Theory*, 28(2):129–137.

[MacQueen, 1967] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Sympos. Math. Statist. and Probability (Berkeley, Calif., 1965/66)*, pages Vol. I: Statistics, pp. 281–297. Univ. California Press, Berkeley, Calif.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Perea et al., 2019] Perea, J., Munch, E., and Khasawneh, F. A. (2019). Approximating continuous functions on persistence diagrams using template functions. *ArXiv*, abs/1902.07190.

[Pike et al., 2019] Pike, J. A., Khan, A. O., Pallini, C., Thomas, S. G., Mund, M., Ries, J., Poulter, N. S., and Styles, I. B. (2019). Topological data analysis quantifies biological nano-structure from single molecule localization microscopy. *Bioinformatics*, 36(5):1614–1621.

[Reininghaus et al., 2015] Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R. (2015). A stable multiscale kernel for topological machine learning. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[The GUDHI Project, 2020] The GUDHI Project (2020). *GUDHI User and Reference Manual*. GUDHI Editorial Board, 3.3.0 edition.

[Verma and Zhang, 2017] Verma, S. and Zhang, Z.-L. (2017). Hunt for the unique, stable, sparse and fast feature learning on graphs. In *Advances in Neural Information Processing Systems*, pages 88–98.

[Zaheer et al., 2017] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R., and Smola, A. (2017). Deep sets. In *Advances in Neural Information Processing Systems*, pages 3391–3401.

[Zhang et al., 2018] Zhang, Z., Wang, M., Xiang, Y., Huang, Y., and Nehorai, A. (2018). RetGK: Graph Kernels based on Return Probabilities of Random Walks. In *Advances in Neural Information Processing Systems*, pages 3968–3978.

[Zhao and Wang, 2019] Zhao, Q. and Wang, Y. (2019). Learning metrics for persistence-based summaries and applications for graph classification. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 9855–9866. Curran Associates, Inc.

[Zieliński et al., 2019] Zieliński, B., Lipiński, M., Juda, M., Zeppelzauer, M., and Dłotko, P. (2019). Persistence bag-of-words for topological data analysis. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4489–4495. International Joint Conferences on Artificial Intelligence Organization.