Persistence Bag-of-Words for Topological Data Analysis

Bartosz Zieliński[1] , **Michał Lipiński**[1] , **Mateusz Juda**[1] ,
Matthias Zeppelzauer[2] and **Paweł Dłotko**[3]
[1]The Institute of Computer Science and Computer Mathematics,
Faculty of Mathematics and Computer Science, Jagiellonian University
[2]Media Computing Group, Institute of Creative Media Technologies,
St. Pölten University of Applied Sciences,
[3]Department of Mathematics and Swansea Academy of Advanced Computing,
Swansea University
{bartosz.zielinski, michal.lipinski, mateusz.juda}@uj.edu.pl, m.zeppelzauer@fhstp.ac.at,
p.t.dlotko@swansea.ac.uk

Abstract

Persistent homology (PH) is a rigorous mathematical theory that provides a robust descriptor of data in the form of persistence diagrams (PDs). PDs exhibit, however, complex structure and are difficult to integrate in today's machine learning workflows. This paper introduces persistence bag-of-words: a novel and stable vectorized representation of PDs that enables the seamless integration with machine learning. Comprehensive experiments show that the new representation achieves state-of-the-art performance and beyond in much less time than alternative approaches.

1 Introduction

Topological data analysis (TDA) provides a powerful framework for the structural analysis of high-dimensional data. A main tool of TDA is Persistent Homology (PH) [Edelsbrunner and Harer, 2010], which currently gains increasing importance in data science [Ferri, 2017]. It has been applied to a number of disciplines including, biology [Gameiro *et al.*, 2014], material science [Lee *et al.*, 2017], analysis of financial markets [Gidea and Katz, 2018]. Persistence homology is also used as a novel measure of GANs (Generative Adversarial Networks) performance [Khrulkov and Oseledets, 2018], and as a complexity measure for neural network architectures [Rieck *et al.*, 2018]. PH can be efficiently computed using various currently available tools [Bauer *et al.*, 2017; Dey *et al.*, 2019; Maria *et al.*, 2014]. A basic introduction to PH is given in the supplementary material (SM in the following)[1].

The common output representation of PH are persistence diagrams (PDs) which are multisets of points in $\mathbb{R}^2$. Due to their variable size, PDs are not easy to integrate within common data analysis, statistics and machine learning workflows. To alleviate this problem, a number of kernel functions and vectorization methods for PDs have been introduced. Kernel-based approaches have a strong theoretical background but in practice they often become inefficient when the number of training samples is large. As the entire kernel matrix must usually be computed explicitly (like in case of SVMs), this leads to roughly quadratic complexity in computation time and memory with respect to the size of the training set. Furthermore, such approaches are limited to kernelized methods, such as SVM and kernel PCA. Vectorized representations in contrast are compatible with a much wider range of methods and do not suffer from complexity constraints of kernels. Since they require a spatial quantization of the PD they might suffer from a loss in precision compared to kernels, especially since PDs are sparsely and unevenly populated structures.

In this work, we present a novel spatially adaptive and thus more accurate representation of PDs, which aims at combining the large representational power of kernel-based approaches with the general applicability of vectorized representations. To this end, we extend the popular bag-of-words (BoW) encoding (originating from text and image retrieval) to TDA to cope with the inherent sparsity of PDs [McCallum and Nigam, 1998; Sivic and Zisserman, 2003]. The proposed adaptation of BoW gives a universally applicable fixed-sized feature vector of low-dimension. It is, under mild conditions, stable with respect to a standard metric in PDs. Experiments demonstrate that our new representation achieves state-of-the-art performance and even outperforms numerous competitive methods while requiring orders of magnitude less time and being more compact. Due to the run-time efficiency of our approach it may in future enable the application of TDA for larger-scale data than possible today.

The paper is structured as follows. Section 2 reviews related approaches. In Section 3 we introduce persistence bag-of-words and prove its stability. Sections 5 and 6 present experimental setup, results and discussion. Please consider the SM[a] for additional information.

2 Background and Related Work

Different kernel based and vectorized representations have been introduced to make PDs compatible with statistical anal-

[1]Supplementary material: http://www.ii.uj.edu.pl/~zielinsb/papers/2019_ijcai_supplement.pdf

ysis and machine learning methods. The goal of kernel-based approaches on PDs is to define dissimilarity measures (also known as kernel functions) used to compare PDs and thereby make them compatible with kernel-based machine learning methods like Support Vector Machines (SVMs) and kernel Principal Component Analysis (kPCA). Li et al. [Li *et al.*, 2014] use the traditional bag-of-features (BoF) approach combining various distances between 0-dimensional PDs to generate kernels. On a number of datasets (SHREC 2010, TOSCA, hand gestures, Outex) they show that topological information is complementary to the information of traditional BoF. Reininghaus et al. [Reininghaus *et al.*, 2015] turns PDs into a continuous distribution by appropriate placing of Gaussian distributions and use the scalar product of the distributions to define a kernel function. Carrière et al. [Carrière *et al.*, 2017] propose a kernel based on sliced Wasserstein approximation of the Wasserstein distance. Le and Yamada [Le and Yamada, 2018] proposed a Persistence Fisher (PF) kernel for PDs. It has a number of desirable theoretical properties such as stability, infinite divisibility, and linear time complexity in the number of points in the PDs. Lacombe et al. [Lacombe *et al.*, 2018] reformulated the computation of diagram metrics as an optimal transport problem. This approach allows for efficient parallelization and scalable computations of a PD's barycenters. Another representation of PDs are Persistence Landscapes, PL [Bubenik, 2015]. PL is a transformation of PD into a sequence of piece-wise linear functions. L^p distance between those functions or their scalar products can be used to define kernels.

Vectorized representations of PDs can be used directly as an input to most machine learning methods. Adams et al. [Adams *et al.*, 2017], propose the persistence image (PI) building upon earlier work [Donatini *et al.*, 1998; Ferri *et al.*, 1998]. PI first fits a distribution to the PD and then samples it at regular intervals to obtain a vectorized representation. Anirudh et al. [Anirudh *et al.*, 2016] propose an approach based on Riemannian manifold (RM). It transforms PD into a Gaussian kernel being a Riemannian manifold with Fisher-Rao metric and subsequently into a vector space which is reduced by PCA to a fixed-size representation.

Recently, a third type of approach has been introduced. It aims at learning areas/points in the PD which are of particular importance for a given task in a supervised manner [Hofer *et al.*, 2017]. Despite promising properties this approach can be used only in cases where supervisory information (labels) are available, requires a large training set and long training time. Additionally, it requires specifically adapted architectures for different data sets and data types [Hofer *et al.*, 2017].

3 Persistence Bag of Words

In this section, we adopt the Bag-of-Words (BoW) model [McCallum and Nigam, 1998; Sivic and Zisserman, 2003], introduced in text and image retrieval, for the quantization of PDs. The idea behind BoW is to quantize variable length input data into a fixed-size representation by a so called *codebook*. The codebook is generated from the input data in an unsupervised manner by clustering. The basic assumption behind BoW is that the clusters (i.e. codewords) capture the intrinsic structure of the data and thereby represent a suitable vocabulary for the quantization of the data. Given a codebook C, every input point P (in a potentially high-dimensional space) is encoded by assigning points from P to the nearest codeword from C. In traditional BoW this encoding leads to a codeword histogram where each codeword from C is a bin and counts how many points from P are closest to it.

For BoW approaches, three important hyperparameters need to be set: (1) the clustering algorithm used to generate the codebook, (2) the size of the codebook, i.e., the number of clusters, and (3) the type of proximity encoding which is used to obtain the final descriptors, i.e. hard or soft assignment. We employ k-means and Gaussian Mixture Models (GMM) for clustering. The size of each codebook is optimized to maximize performance. In the following sections, we will describe two ways of generating codeword histograms based on the proximity of points from P to codewords in C.

The overall approach is visualized in Fig. 1. The input consists of a set of PDs extracted from all instances of a given dataset and transformed into *birth-persistence* coordinates using $(b, d) \rightarrow (b, d-b)$ transformation for each point in the PD. Next, all PDs are consolidated into one diagram which may then be sub-sampled to reduce the influence of noise. Based on the consolidated diagram, the codebook C is generated by clustering. For a given codebook we generate codeword histograms by different quantization strategies.

3.1 Persistence Bag of Words

In order to directly adapt BoW [Baeza-Yates and Ribeiro-Neto, 1999; Sivic and Zisserman, 2003] to PDs, let us consolidate a collection of training diagrams $D_1, D_2, \ldots, D_n$ into $D = D_1 \cup D_2 \cup \ldots \cup D_n$ in order to obtain a codebook by using k-means clustering on D. Let $\{\mu_i \in \mathbb{R}^2, i = 1, \ldots, N\}$ denote the centers of obtained clusters. Given a new PD $B = \{x_t \in \mathbb{R}^2\}_{t=1}^{T}$ let $NN(x_t)$ be equal to i if $d(x_t, \mu_i) \leq d(x_t, \mu_j)$ for all $j \in \{1, \ldots, N\}$. We define a *persistence bag of words* (PBoW) as a vector:

$$\mathbf{v}^{\mathbf{PBoW}}(B) = \left(v_i^{PBoW}(B)\right)_{i=1,\ldots,N}, \tag{1}$$

where $v_i^{PBoW}(B) = card\{x_t \in B \mid NN(x_t) = i\}$. In other words, v_i^{PBoW} captures the number of points from B, which are closer to μ_i than to any other μ_j. Next, $\mathbf{v}^{\mathbf{PBoW}}(B)$ is normalized by taking the square root of each component (preserving the initial sign) and dividing it by the norm of the vector. This is a standard normalization for BoW [Perronnin *et al.*, 2010] used to reduce the influence of outliers.

This direct adaptation of BoW to PDs is, however, not 1-Wasserstein stable. To show this, let us assume that we have two clusters with centers $\mu_1 = (0, 0), \mu_2 = (1, 0) \in \mathbb{R}^2$, and PD B containing only one point $x_1 = (\frac{1}{2} + \epsilon, 0)$, for some small $\epsilon > 0$. Then, $\mathbf{v}^{\mathbf{PBoW}}(B) = [0, 1]$, because x_1 is closer to μ_2 than μ_1. However, a small perturbation in B, e.g. by -2ϵ, changes the assignment of x_1 from μ_2 to μ_1. In this case $B' = \{(\frac{1}{2} - \epsilon, 0)\}$ and $\mathbf{v}^{\mathbf{PBoW}}(B') = [1, 0]$. In order to be stable in 1-Wasserstein sense, PBoW should fulfill the following condition:

$$2 = |\mathbf{v}^{\mathbf{PBoW}}(B) - \mathbf{v}^{\mathbf{PBoW}}(B'))| < C|x_1 - y_1| < 2C\epsilon,$$

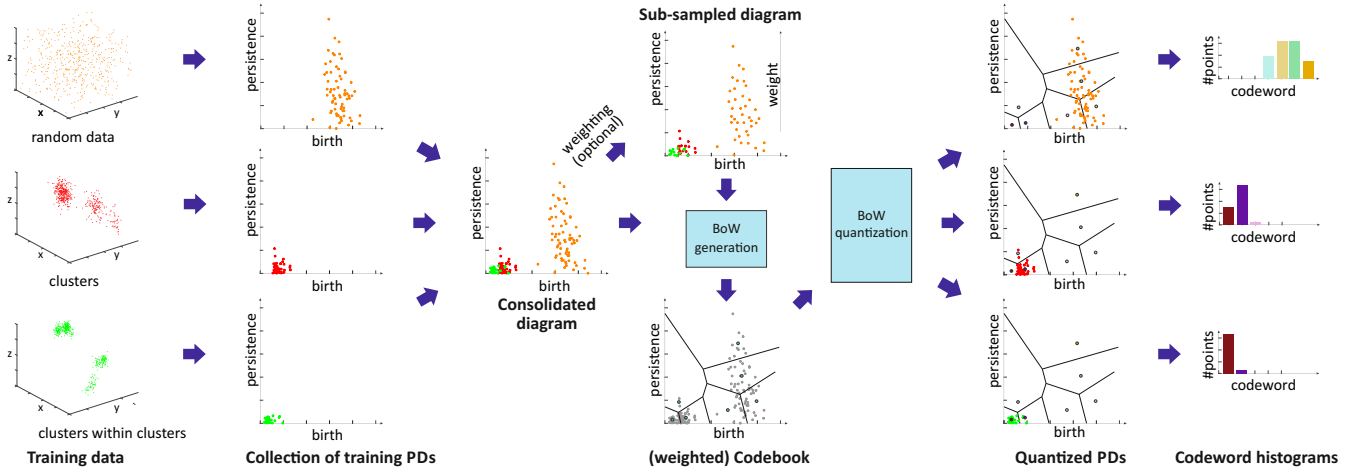

Figure 1: Persistence bag-of-words: An illustration of the entire pipeline for codebook generation and the extraction of codeword histograms. From the input data we compute PDs in birth-persistence coordinates and combine them into one consolidated diagram. Next, a subset of points is obtained from this diagram by either weighted or unweighted sub-sampling. Subsequently, we cluster the sub-sampled consolidated diagram to derive a codebook. Finally, the individual points for each input PD are encoded by the codewords (BoW quantization). This illustration shows a hard assignment of points to codewords (for better illustration). In practice, a soft assignment strategy is recommended for stability reasons. The result is a codeword histogram for each input PD that represents how many points fall into which cluster of the codebook. These histograms represent a compact and fixed-size vectorial representation.

therefore $C > 1/\epsilon$. As $\epsilon > 0$ can be arbitrarily small, there does not exist a constant C that meets this condition. Therefore PBoW is not stable. In the next section, we adopt BoW to better fit the structure of PDs and to deal with the instability.

3.2 Stable Persistence Bag of Words

In this section we present two important adaptations of BoW for PDs. Firstly, we enforce the codeword selection to be preferential to higher persistence points. Secondly, we adopt soft assignment of points to the clusters and prove that such an approach guarantees stability of the representation.

A consequence of the stability theorem for PDs is that points with higher persistence are typically considered more important than points with lower persistence. Therefore, when selecting the cluster centers μ_i in BoW, preference should be given to higher persistence points. To integrate this into codebook generation we perform the clustering on a subset of points obtained by a weighted sampling of D. For the sub-sampling we use a piece-wise linear weighting function $w_{a,b} : \mathbb{R} \to \mathbb{R}$. Given $a < b$:

$$w_{a,b}(t) = \begin{cases} 0 & \text{if } t < a \\ (t-a)/(b-a) & \text{if } a \leq t < b \\ 1 & \text{if } b \leq t \end{cases} \qquad (2)$$

and use it to weight the second coordinate (persistence) of the points in the PD. In our experiments we set a and b to the persistence values corresponding to 0.05 and 0.95 quantiles of the persistence coordinate of the points in D. Consequently, persistence points having higher values for function w are more likely to be sampled. Note that the sub-sampling does not guarantee that the points of highest persistence will be selected as centers of clusters, but it makes the probability of such an event considerably larger.

To account for the instability of PBoW (see Section 3.1), we propose *Stable Persistence Bag of Words* (sPBoW). Similarly to PBoW, we first consolidate PDs in the initial step of construction. Next, we generate a GMM based on the sub-sampled points (by expectation maximization [Nasrabadi, 2007]). This approach was originally introduced in [Van Gemert *et al.*, 2008]. Let the parameters of the fitted GMM be $\lambda = \{w_i, \mu_i, \Sigma_i, i = 1, \ldots, N\}$, where w_i, μ_i and Σ_i denote the weight, mean vector and covariance matrix of Gaussian i and N denotes the number of Gaussians. Given a PD B its stable PBoW is defined as:

$$\mathbf{v}^{\mathbf{sPBoW}}(B) = \left(v_i^{sPBoW} = w_i \sum_{x_t \in B} p_i(x_t|\lambda) \right)_{i=1,\ldots,N}, \qquad (3)$$

where $w_i > 0$, $\sum_{i=1}^{N} w_i = 1$, and $p_i(x_t|\lambda)$ is the likelihood that observation x_t was generated by Gaussian i:

$$p_i(x_t|\lambda) = \frac{exp\{-\frac{1}{2}(x_t - \mu_i)'\Sigma_i^{-1}(x_t - \mu_i)\}}{2\pi|\Sigma_i|^{\frac{1}{2}}}.$$

See SM[a] for a complexity analysis of PBoW and sPBoW.

4 Stability Proof

Theorem. Let B and B' be persistence diagrams with a finite number of non-diagonal points. Stable persistence bag of words, sPBoW with N words is stable with respect to 1-Wasserstein distance between the diagrams, that is

$$\left\| \mathbf{v}^{\mathbf{sPBoW}}(B) - \mathbf{v}^{\mathbf{sPBoW}}(B') \right\|_\infty \leq C \cdot W_1(B, B'),$$

where C is a constant.

Proof. Let $\eta : B \to B'$ be an optimal matching in the definition of 1-Wasserstein distance. For a fixed $i \in \{1, \ldots, N\}$ we have:

$$\left\|v_i^{sPBoW}(B) - v_i^{sPBoW}(B')\right\|_\infty = \left\|w_i \sum_{x\in B}(p_i(x|\lambda) - p_i(\eta(x)|\lambda)\right\|_\infty \leq |w_i| \sum_{x\in B} \|(p_i(x|\lambda) - p_i(\eta(x)|\lambda))\|_\infty$$

Note that $p_i : \mathbb{R}^2 \to \mathbb{R}$ are Lipschitz continuous (as they are 2-dimensional Gaussian distributions). Let L_i be their Lipschitz constant. We get

$$|w_i| \sum_{x\in B} \|(p_i(x|\lambda) - p_i(\eta(x)|\lambda))\|_\infty \leq |w_i| \sum_{x\in B} \|(L_i(x - \eta(x)))\|_\infty = |w_i\ L_i| \sum_{x\in B} \|(x - \eta(x))\|_\infty = |w_i\ L_i|\ W_1(B, B')$$

Consequently for $C = max_i|w_i\ L_i|$ we have

$$\left\|\mathbf{v^{sPBoW}}(B) - \mathbf{v^{sPBoW}}(B')\right\|_\infty \leq C\ W_1(B, B').$$

□

5 Experimental Setup

5.1 Datasets

We incorporate datasets which cover a wide range of different retrieval problems. Firstly, to provide a proof-of-concept, we evaluate all approaches on a set of synthetically generated shape classes from [Adams *et al.*, 2017]. It consists of six shape classes represented by point clouds of the geometrical objects. The task is to differentiate them by the derived representations. Additionally, we evaluate the approaches on real-world datasets for geometry-informed material recognition (GeoMat) [DeGol *et al.*, 2016], classification of social network graphs (reddit-5k, reddit-12k) [Hofer *et al.*, 2017], analysis of 3D surface texture (PetroSurf3D) [Zeppelzauer *et al.*, 2017], and 3D shape segmentation [Carrière *et al.*, 2017]. Where available, we have used pre-computed PDs available with datasets to foster reproducibility and comparability. More detail on the datasets is provided in the SM^a.

5.2 Compared Approaches

We compare our bag-of-word approaches with both kernel-based techniques and vectorized representations. Kernel-based approaches include: 2-Wasserstein distance[2] (2Wd) [Kerber *et al.*, 2017], the multi-scale kernel[3] (MK) of [Reininghaus *et al.*, 2015], and sliced Wasserstein kernel[4] (SWK) [Carrière *et al.*, 2017]. Furthermore, we employ the persistence landscape[5,6] (PL) representation and generate a kernel matrix by the distance metric defined in [Bubenik, 2015]. Vectorized PD representations include: persistence image[7] (PI) [Adams *et al.*, 2017] and the Riemannian manifold approach[8] (RM) [Anirudh *et al.*, 2016].

5.3 Setup

For all datasets, we aim at solving a supervised classification task. The classification pipeline is as follows: for the kernel-based approaches we take the PDs as input and compute the kernel matrices for the training and test samples. Next we train an SVM from the explicit kernel matrices and evaluate it on the test data. For the vectorized representations we compute the respective feature vectors from the PDs and feed them into a linear SVM for training. This procedure allows for directly comparing kernel-based approaches and vectorized representations. To enhance the comparability we employ (if available) the original train/test division of the datasets. To find optimal parameters for each evaluated approach, we run a grid search including cross-validation over the hyperparameters of all approaches (see Table 1 in SM^a).

As the computation times for some of the considered methods, especially for kernel-based approaches, do not scale well with the sizes of datasets (computation time and space required for explicit kernel matrices grows exponentially), we have decided to split the evaluation into two parts: EXP-A uses all related approaches on smaller randomly subsampled versions of the datasets (see SM^a for details on subsampling), while EXP-B operates only on the vectorized representations and uses larger datasets.

The mean accuracy is obtained as an average over 5 runs with the same train/test divisions used by the compared methods. A Wilcoxon signed-rank tests is used to show the statistical significance of differences between the scores of the best and the remaining methods. As the number of repetitions is small, the p-value is set to 0.1 and no multiple testing correction is applied.

The code of our experiments is implemented in Matlab[9]. For external approaches we use the publicly available implementations of the original authors. For bag-of-words we employ the VLFeat library [Vedaldi and Fulkerson, 2008].

6 Results and Discussion

Table 1 summarizes the results obtained in our experiments for EXP-A and EXP-B. For each combination of dataset and approach we provide the obtained classification accuracy (including the standard deviation) and the processing time needed to construct the representations (excluding the time for classification). Note that for the synthetic dataset and the 3D shape segmentation dataset, results of EXP-A and EXP-B are equal, as no sub-sampling was needed to perform EXP-A.

Overall, for all experiments in EXP-A and EXP-B, PBoW or sPBoW achieve state-of-the-art performance or even above. From EXP-A we further observe that vectorized representations (including the proposed ones) in all cases outperform kernel-based approaches. Among the compared vector-

[2]https://bitbucket.org/grey_narn/hera

[3]https://github.com/rkwitt/persistence-learning

[4]code obtained from Mathieu Carrière

[5]https://www.math.upenn.edu/~dlotko

[6]https://github.com/queenBNE/Persistent-Landscape-Wrapper

[7]https://github.com/CSU-TDA/PersistenceImages

[8]https://github.com/rushilanirudh/pdsphere

[9]https://github.com/bziiuj/pcodebooks

EXP-A

Descr.	Synthetic		GeoMat		Reddit-5k		Reddit-12k		PetroSurf3D		3D Shape Segm.	
	Score	Time (sec.)	Score	Time (sec.)	Score	Time (sec.)	Score	Time (sec.)	Score	Time (sec.)	Score	Time (sec.)
2Wd	**98.0 ± 1.3**	133.4	**24.7 ± 3.1**	14740.6	29.6 ± 5.6	6198.8	23.3 ± 2.4	4979.8	**79.9 ± 6.6**	63822.5	71.0 ± 0.6	6300.8
MK	92.3 ± 2.8	44.0	9.1 ± 3.1	10883.0	32.7 ± 3.1	7251.7	26.7 ± 4.6	4222.3	**80.2 ± 3.6**	53831.0	88.5 ± 0.5	860.3
SWK	**97.4 ± 1.9**	33.3	22.0 ± 2.7	1069.6	39.2 ± 6.1	373.8	30.9 ± 5.0	329.7	**78.0 ± 6.3**	3913.0	94.2 ± 0.5	1995.4
PL	95.1 ± 2.2	81.0	17.9 ± 3.8	1563.7	30.4 ± 6.2	791.3	24.7 ± 4.2	719.0	**78.2 ± 6.7**	7991.7	92.6 ± 0.8	2668.3
PI	**98.3 ± 1.6**	10.1	11.1 ± 2.4	342.3	**48.4 ± 5.7**	1250.9	**35.6 ± 4.6**	111.8	**80.0 ± 3.7**	1877.0	**94.9 ± 0.3**	291.2
RM	92.3 ± 2.6	**0.9**	19.1 ± 2.7	10.3	39.2 ± 9.2	7.8	28.7 ± 2.0	17.9	**77.3 ± 6.3**	128.8	72.3 ± 0.4	6.6
PBoW	**98.0 ± 2.2**	**0.8**	**26.7 ± 2.5**	**0.3**	**46.8 ± 4.8**	**0.5**	**32.7 ± 2.2**	**0.3**	**79.9 ± 4.6**	**1.8**	90.8 ± 0.7	**5.4**
sPBoW	**96.9 ± 1.6**	4.0	22.2 ± 2.6	2.2	**45.6 ± 5.4**	0.7	31.6 ± 2.8	**1.3**	**78.8 ± 3.7**	31.5	**94.4 ± 0.6**	14.7

EXP-B

PI	same as EXP-A		22.45 ± 0.0	7243.1	**49.3 ± 2.8**	4758.7	**38.4 ± 0.9**	11329.0	**80.4 ± 4.9**	12137.0	same as EXP-A	
RM			9.4 ± 0.0	222.6	46.3 ± 3.1	108.1	32.6 ± 1.1	215.5	**79.9 ± 5.0**	1451.0		
PBoW			**29.0 ± 0.4**	**5.2**	**49.9 ± 3.3**	**1.5**	**38.6 ± 0.9**	**4.8**	**80.3 ± 5.3**	**28.5**		
sPBoW			27.8 ± 0.7	29.7	46.6 ± 3.3	5.2	35.3 ± 1.4	29.3	**79.9 ± 5.2**	161.8		
SoA	97.3 [Adams *et al.*, 2017]		22.3 ± 0.8 [DeGol *et al.*, 2016]		49.1 [Hofer *et al.*, 2017]		38.5 [Hofer *et al.*, 2017]		n/a		n/a	

Table 1: Results of EXP-A and EXP-B averaged over 5 runs. We provide average classification accuracy, standard deviation and computation time. Results with statistically significant improved performance are highlighted bold. The first four approaches are kernels (2Wd: 2-Wasserstein Distance, MK: Multiscale Kernel, SWK: Sliced Wasserstein Kernel, PL: Persistence Landscape). The remaining approaches are vectorized representations evaluated in EXP-A and EXP-B (PI: Persistence Image, RM - Riemmanian Manifold, (s)PBoW: (stable) Persistence Bag-of-Words). Column "Time" shows the computation times for kernel matrices or vectorized representations using the optimal parameters obtained in grid search (excluding cross-validation and classification). Times highly depend on those parameters (e.g. the resolution of PI), which is further investigated in Section 6.1 and in the SM^a. Times (esp. in EXP-A) further vary significantly with the dataset that influences the number of points in the PDs. Row "SoA" shows comparable state-of-the-art results from the literature where available.

ized representations, PI in most cases outperforms RM and will thus serve as the primary approach for further comparison. When comparing the stable vs. unstable variants of PBoW, we observe that PBoW in most cases outperforms its stable equivalent. This is further studied in Section 6.1.

Large differences exist in processing times of the different approaches. The slowest vectorized approach is PI. The runtimes, however, vary strongly, depending on the resolution used for PI, which depends on the optimally estimated resolution during grid search. The RM representation is one to two magnitudes faster than PI[10]. All kernel methods are about one magnitude or more slower then PI. (s)PBoW outperform almost all state of the art approaches in runtime for all datasets (in EXP-A and EXP-B). The gain in runtime efficiency ranges from one to up to four orders of magnitude. For the largest dataset (PetrSurf3D), for example, the PBoW and sPBoW require 29 and 162 seconds while RM requires 1.451 seconds and PI 12.136 seconds. In the following, we analyze selected aspects of the proposed representations in greater detail.

6.1 Accuracy vs. Codebook Size

The most important parameter for BoW representations is the codebook size N, i.e. the number of clusters. There is no commonly agreed analytic method to estimate the optimal codebook size, thus the estimation is usually performed empirically. To investigate the sensitivity of (s)PBoW and their performance on the codebook size, we evaluate both approaches for a range of values of N for each dataset. Results are shown in Fig. 2, both without (solid lines) and with weighted sub-sampling (dashed lines) of the consolidated PD.

On the synthetic dataset we observe that without weighting PBoW outperforms sPBoW. However, when codebook weighting is introduced, the stable variant sPBoW starts to work equally well. The trend shows that larger codebook sizes are better than small ones but also that with already 20 words a high performance can already be achieved. Using weighting and the stable formulation of PBoW clearly improves the performance on this dataset.

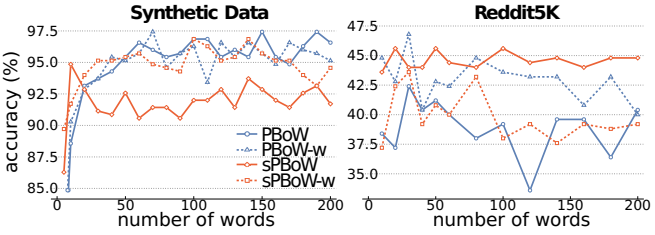

Figure 2: Accuracy vs. codebook size for datasets from EXP-A without (solid lines) and with codebook weighting (dashed lines). See plots for the remaining datasets in Fig. 3 of the SM^a.

Unlike to the synthetic data, in the case of social network graphs (reddit-5k) there is no recognizable increase in performance for larger codebook sizes. We assume that this is caused by the larger variation between training and test data in real data sets. More precisely, larger codebooks result in codewords which tend to overfit on the training PDs. Such codewords do not generalize well to test data, which can result in the observed behavior.

Over all datasets we observe that the benefit of weighting is dataset dependent and that no general recommendation can be derived (see results for the remaining datasets in SM^a). The stable formulation of PBoW, however, performs equally well or even better than the unstable one in 4/6 datasets.

6.2 Qualitative Analysis

In this section we investigate the proposed representations with a special focus on their discriminative abilities. We employ the synthetic dataset as proof-of-concept and GeoMat as example of a complex real-world dataset. We compute PBoW with $N = 20$ clusters for the synthetic dataset and visually analyze the codeword histograms obtained by (hard)

[10]For both representations the original implementations are used

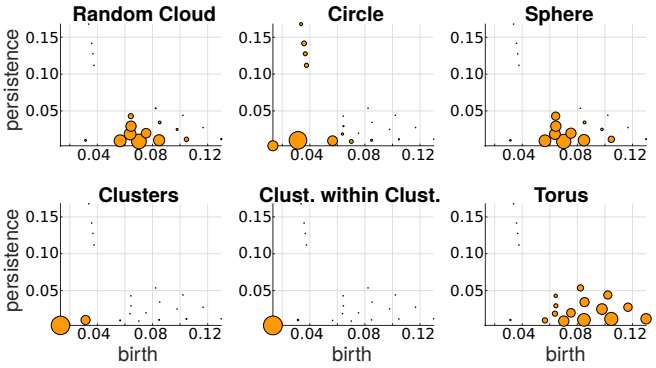

Figure 3: Average codebook histograms computed for each of the six shape classes of the synthetic dataset. The cluster center of each codeword is presented as a circle in the birth-persistence domain. The area of the circles reflects the histogram values of the specific class. For all classes the same codebook (same clustering) is employed, thus, dot locations are the same on all plots. Class differences are thus reflected by different sizes of the circles.

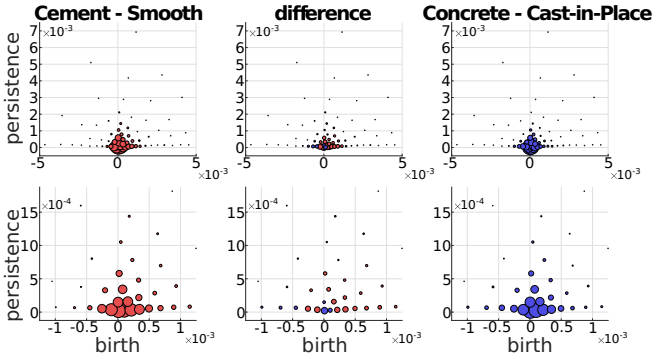

Figure 4: Comparison of averaged PBoW histograms for class "cement smooth" (left, red) and "concrete cast-in-place" (right, blue) from GeoMat (top row: total view, bottom row is zoomed in). The plot in the center shows the difference between the classes where red color means that the left class has stronger support for this cluster and blue means that the right class has stronger support.

assignment. For each of the six shape classes we compute the average codebook histogram (over all samples of each class) to obtain one representative PBoW vector per class. The averaged PBoW histograms for each classes are presented in Fig. 3. Instead of only providing the histograms themselves, we plot for each codeword of the histogram the corresponding cluster center as a circle in the original birth-persistence domain and encode the number of assigned codeworks (the actual values of the histograms) in the area of the circles, i.e. the larger the count for a cluster, the larger the circle. The advantage of this representation is that the spatial distribution of the codewords in the PD is preserved.

From Fig. 3 we can see that except for the classes "random cloud" and "sphere" (which are difficult to differentiate) all classes generate strongly different cluster distributions. Class "circle", for example, uniquely activates four clusters with strong persistence (top-left corner) and the "torus" class distributes its corresponding code words across a large number of clusters representing less persistent components.

Fig. 3 further illustrates an important property of persistence bag-of-words, namely its sparse nature. More specifically, areas with no points in the consolidated persistence diagram will contain no codewords (clusters). In Fig. 3, for example, no codeword is obtained in the upper-right quadrant of the diagram, since no components are located there for the underlying data. Thus these unimportant areas are neglected and not encoded into the final representation. This not only reduces the dimension of the final representation but further makes the representation adaptive to the underlying data and increases the information density of the representation.

We further investigate the performance on the GeoMat dataset. For GeoMat (s)PBoW significantly outperforms all other representations (see Table 1), first and foremost PI, which is the best related approach on this dataset in EXP-B. To this end, we generate confusion matrices for PI and PBoW that show that PBoW yields a better class separation (see Fig. 6 in SM^a). Averaged PBoW histograms for two example classes ("cement smooth" and "concrete cast-in-place") are shown in Fig. 4. For both classes the histograms are on the first sight similar (upper row in Fig. 4). However, by zooming-in towards the birth-persistence plane in Fig. 4 (bottom row), differences become visible. The plots in the center illustrate the difference between the class distributions (red color means left class is stronger, blue means right class is stronger for this cluster). The classes distinguish themselves by fine-grained spatial differences. The set of three blue points around birth time of 0 (which are characteristic for class "concrete cast-in-place") surrounded by red points (which are characteristic for class "cement smooth") illustrates this well (see lower central plot). For the discrimination of these two classes a particularly fine-grained codebook with many clusters is needed. The PI has problems with such fine-grained structures, because due to its limited resolution, all topological components in the most discriminative area would most likely fall into one pixel. Therefore, an extraordinary high resolution would be necessary to capture the discriminative patterns between those two classes. The bag-of-words model makes our approaches independent of the resolution and enables to efficiently capture such fine-grained differences. More examples from the GeoMat dataset can be found in SM^a in Section 5.4 together with additional evaluations of (s)PBoW on computation time, dataset size and accuracy, see Sections 5.2 and 5.3.

Acknowledgements

This work was supported by the National Science Centre, Poland under grants no. 2015/19/D/ST6/01215, 2015/19/B/ST6/01819 and 2017/25/B/ST6/01271, the Austrian Research Promotion Agency FFG under grant no. 856333 and 866855, the Lower Austrian Research and Education Company NFB under grant no. LSC14-005 and by EPSRC grant no. EP/R018472/1.

References

[Adams *et al.*, 2017] Henry Adams, Sofya Chepushtanova, Tegan Emerson, Eric Hanson, Michael Kirby, Francis Motta, Rachel Neville, Chris Peterson, Patrick Shipman, and Lori Ziegelmeier. Persistence images: a stable vector representation of persistent homology. *Journal of Machine Learning Research*, 18(8):1–35, 2017.

[Anirudh *et al.*, 2016] Rushil Anirudh, Vinay Venkataraman, Karthikeyan Natesan Ramamurthy, and Pavan Turaga. A riemannian framework for statistical analysis of topological persistence diagrams. In *Proc. of IEEE CVPR - Workshops*, pages 68–76, 2016.

[Baeza-Yates and Ribeiro-Neto, 1999] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[Bauer *et al.*, 2017] Ulrich Bauer, Michael Kerber, Jan Reininghaus, and Hubert Wagner. Phat–persistent homology algorithms toolbox. *Journal of Symbolic Computation*, 78:76–90, 2017.

[Bubenik, 2015] Peter Bubenik. Statistical topological data analysis using persistence landscapes. *JMLR*, 16(1):77–102, 2015.

[Carrière *et al.*, 2017] Mathieu Carrière, Marco Cuturi, and Steve Oudot. Sliced wasserstein kernel for persistence diagrams. In *ICML*, 2017.

[DeGol *et al.*, 2016] Joseph DeGol, Mani Golparvar-Fard, and Derek Hoiem. Geometry-informed material recognition. In *Proc. of CVPR*, pages 1554–1562, 2016.

[Dey *et al.*, 2019] Tamal K. Dey, Dayu Shi, and Yusu Wang. Simba: An efficient tool for approximating rips-filtration persistence via simplicial batch collapse. *J. Exp. Algorithmics*, 24(1):1.5:1–1.5:16, January 2019.

[Donatini *et al.*, 1998] Pietro Donatini, Patrizio Frosini, and Alberto Lovato. Size functions for signature recognition. In *Proc. SPIE*, volume 3454, pages 178–183, 1998.

[Edelsbrunner and Harer, 2010] Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

[Ferri *et al.*, 1998] Massimo Ferri, Patrizio Frosini, Alberto Lovato, and Chiara Zambelli. Point selection: A new comparison scheme for size functions (with an application to monogram recognition). In *Computer Vision - ACCV'98. LNCS.*, volume 1351, pages 329–337. Springer, 1998.

[Ferri, 2017] Massimo Ferri. Persistent topology for natural data analysis — a survey. In *Towards Integrative Machine Learning and Knowledge Extraction*, pages 117–133, Cham, 2017. Springer International Publishing.

[Gameiro *et al.*, 2014] Marcio Gameiro, Yasuaki Hiraoka, Shunsuke Izumi, Miroslav Kramàr, Konstantin Mischaikow, and Vidit Nanda. A topological measurement of protein compressibility. *Japan J.of Industrl. and Appl. Mathem.*, 32(1):1–17, 2014.

[Gidea and Katz, 2018] Marian Gidea and Yuri Katz. Topological data analysis of financial time series: Landscapes of crashes. *Physica A: Statistical Mechanics and its Applications*, 491:820–834, 2018.

[Hofer *et al.*, 2017] Christoph Hofer, Roland Kwitt, Marc Niethammer, and Andreas Uhl. Deep learning with topological signatures. In *Advances in Neural Information Processing Systems*, pages 1633–1643, 2017.

[Kerber *et al.*, 2017] Michael Kerber, Dmitriy Morozov, and Arnur Nigmetov. Geometry helps to compare persistence diagrams. *Journal of Experimental Algorithmics (JEA)*, 22:1–4, 2017.

[Khrulkov and Oseledets, 2018] Valentin Khrulkov and Ivan Oseledets. Geometry score: A method for comparing generative adversarial networks. In *Proc. of the 35th ICML*, volume 80, pages 2621–2629. PMLR, 2018.

[Lacombe *et al.*, 2018] Théo Lacombe, Marco Cuturi, and Steve Oudot. Large scale computation of means and clusters for persistence diagrams using optimal transport. *arXiv:1805.08331*, 2018.

[Le and Yamada, 2018] Tam Le and Makoto Yamada. Persistence fisher kernel: A riemannian manifold kernel for persistence diagrams. In *Adv. in Neural Inf. Proc. Sys.*, pages 10028–10039, 2018.

[Lee *et al.*, 2017] Yongjin Lee, Senja D. Barthel, Paweł Dłotko, S. Mohamad Moosavi, Kathryn Hess, and Berend Smit. Quantifying similarity of pore-geometry in nanoporous materials. *Nature Communications*, 8(15396), 2017.

[Li *et al.*, 2014] Chunyuan Li, Maks Ovsjanikov, and Frédéric Chazal. Persistence-based structural recognition. In *IEEE CVPR*, pages 2003–2010. IEEE, 2014.

[Maria *et al.*, 2014] Clément Maria, Jean-Daniel Boissonnat, Marc Glisse, and Mariette Yvinec. The gudhi library: Simplicial complexes and persistent homology. In *International Congress on Mathematical Software*, pages 167–174. Springer, 2014.

[McCallum and Nigam, 1998] Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48, 1998.

[Nasrabadi, 2007] Nasser M. Nasrabadi. Pattern recognition and machine learning. *J. of electronic imaging*, 16(4):049901, 2007.

[Perronnin *et al.*, 2010] Florent Perronnin, Jorge Sénchez, and Yan Liu Xerox. Large-scale image categorization with explicit data embedding. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2297–2304. IEEE, 2010.

[Reininghaus *et al.*, 2015] Jan Reininghaus, Stefan Huber, Ulrich Bauer, and Roland Kwitt. A stable multi-scale kernel for topological machine learning. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4741–4748, June 2015.

[Rieck *et al.*, 2018] Bastian Rieck, Matteo Togninalli, Christian Bock, Michael Moor, Max Horn, Thomas Gumbsch, and Karsten Borgwardt. Neural persistence: A complexity measure for deep neural networks using algebraic topology. *arXiv:1812.09764*, 2018.

[Sivic and Zisserman, 2003] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *IEEE ICCV, 2003*, pages 1470–1477. IEEE, 2003.

[Van Gemert *et al.*, 2008] Jan C. Van Gemert, Jan-Mark Geusebroek, Cor J. Veenman, and Arnold W. M. Smeulders. Kernel codebooks for scene categorization. In *European conference on computer vision*, pages 696–709. Springer, 2008.

[Vedaldi and Fulkerson, 2008] Andrea Vedaldi and Brian Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

[Zeppelzauer *et al.*, 2017] Matthias Zeppelzauer, Bartosz Zieliński, Mateusz Juda, and Markus Seidl. A study on topological descriptors for the analysis of 3d surface texture. *Comp. Vision and Image Underst.*, 2017.