Persistent homology for pedestrian walk classification

Clotilde Pety, Yague Thiam, Mélisande Teng

March 8, 2018

1 Introduction, general setting

The aim of the project is to classify pedestrian walk, using time series representing the acceleration (3 coordinates) of 3 pedestrians A, B and C as data. Because the data was recorded using the accelerometer sensor of a smartphone carried in the pocket, it cannot be ensured that data can be matched to a single spatial system of axis: the pedestrians may well have put their phones in various positions in their pocket, making it irrelevant to compare the time series coordinate by coordinate. Persistence homology offers a way to classify pedestrian walk because it has coordinate invariance property.

Quick overview of the project We explore several methods using persistence homology to classify pedestrians.

- We start by computing persistence diagrams of the Rips filtrations. We hope to be able to discriminate the pedestrians by computing the matrices of pairwise bottleneck distance between the diagrams. This method would prove sufficient to classify the data if when representing the matrices, we obtained 3 distinct clusters of points, each cluster corresponding to a pedestrian.
- We also try the preceding method using alpha complexes to build the persistence diagrams.
- We compute persistence landscapes, which convert persistence diagrams into a well-behaved real-valued functions, and expect the pedestrians to be better classified than when using solely persistence diagrams. We compare the performances the 0-dimensional or 1-dimensional landscapes using a random forest algorithm.
- In order to check that persistent homology was indeed a useful tool for this classification task, we test our classifier with raw data as input.

The dataset The dataset is a list that contains 4 elements: the first three elements of the list encapsulate data about the walk of 3 pedestrians A, B and C that has been recorded using the accelerometer sensor of a smartphone carried in their pocket. This measure of acceleration for the 3 pedestrians generated 3 multivariate time series with 3D data, which correspond to the 3 coordinates of acceleration. Each of the 3 series are split in 100 time series of 200 consecutive points (the first time series for pedestrian A is plot in Fig. 1). The fourth element of the list specifies to which pedestrian the time series are associated with.

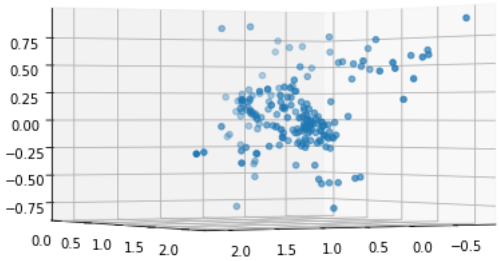


Figure 1: Acceleration of pedestrian A, first time series

Requirements The classifier was developed in Python 3.6, using library Gudhi.

2 Persistence diagrams

We build 0 and 1-persistence diagrams using Rips filtrations to get the most important topological features of the data corresponding to each pedestrian.

2.1 Rips filtrations and persistence diagrams

Let (P, D) be a metric space where P is a point set. Given $r > 0$, the Rips complex is the simplicial complex $R(P)$ constituted by the simplexes such that $d(p, q) \leq r$ for every pair of vertices in the simplex. Constructing the Rips complex helps capture the topology of the data set.
Choosing r can be difficult task because if r is too small, the complex is a discrete set, and if r is too large, the complex becomes a single high-dimensional complex. Given a filtration $(K_0, K_1, ..., K_n)$, the p-dimensional persistence diagram is the set of points (i,j) such that the number of p-dimensional homology classes born at K_i that die entering K_j is one.
Here the filtration will actually be a sequence of Rips complexes associated to the 3D point cloud for an increasing sequence of parameter values.

On alpha complexes The α-complex of the given set of points is the simplicial complex formed by the set of edges and triangles whose radii are at most $1/\alpha$.

2.2 Feature importance

As r increases, the topology of the filtration of the metric space changes. New connected components may appear while existing ones may merge. Persistent diagrams are a convenient tool that keep track of these changes in the topology of the filtration of the metric space, detect features and allocate a lifetime to these features. For example, a feature can be a connected component born for a given r that later died by merging with another connected component as r increases. For 0-persistence, the features are connected components, while for 1-persistence, we also look for cycles. The notion of lifetime associated to a feature is key since the longer the lifetime, the more important the feature. The lifetime associated to a single feature can be represented as a segment whose extremities are given by the value of r when the feature was born and by the value of r when the feature died. As a result, each feature contained in the metric space is associated with a bar and a metric space can be represented by a barcode that could be obtained under varying Rips filtration. And the longer the bar, the longer the lifetime and, as a consequence, the more important the feature associated to the bar.
A more convenient way to represent the birth and the death of a feature under Rips filtrations is to represent the bar as a point in a x-y plane where the abscissae of the point would be given by the value of r at birth while the ordinate of the point would take the value of r at death. Since death occurs after birth, all the points represented using the above-mentioned process are above the line y = x. Using the same underlying logic for the barcode and the associated lifetime of a feature, the furthest a point in a persistent diagram is from the diagonal y = x, the longer the lifetime of a feature associated to this point is, which means the more persistent a feature is and the more important it is.

2.3 Implementation

In order to draw the 0 and 1-dimensional persistence diagrams of the Rips filtrations, we use `gudhi.RipsComplex`on the 3D point clouds.

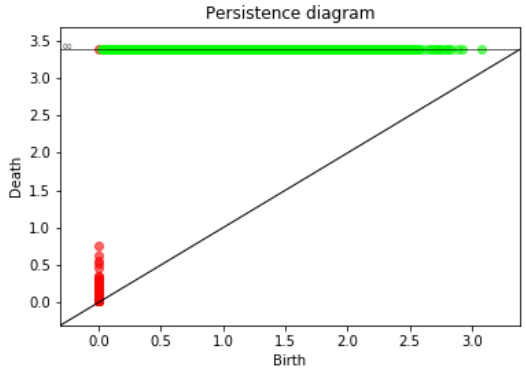
Parameter tuning In order to choose a relevant r (maximum edge length in a simplex) parameter, we compute the maximum and minimum distances between data points in each time series in each data set A, B and C, which will bound the range in which we can best capture the topology of the dataset. These are reported in the table below.

	Data A	Data B	Data C
minimum distance	$3.14\ 10^{-3}$	$9.62\ 10^{-4}$	$4.08\ 10^{-3}$
maximum distance	3.12	4.11	4.58

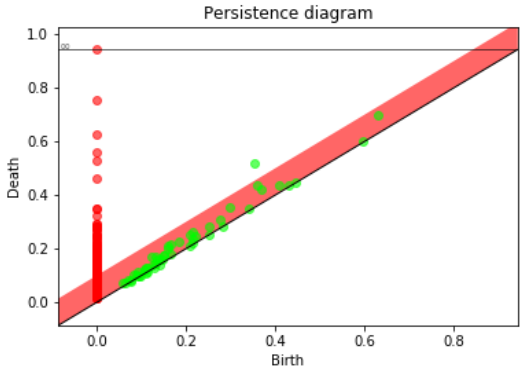
Thus, we want to choose a value of r in [$9.62\ .10^{-4}$, 4.58].
Unless specified otherwise, the parameter chosen for all our computations was 3.95. This happens to be roughly the maximum over sets A, B, C of the maximum pairwise distances minus the standard deviation for each set. There is no theoretical justification for that choice, but we wanted to aim for r closer to the upper bound than the lower bound.

We also tried very small values (0.2) and very large values (5) to check that we indeed obtained worse results for the classification task.

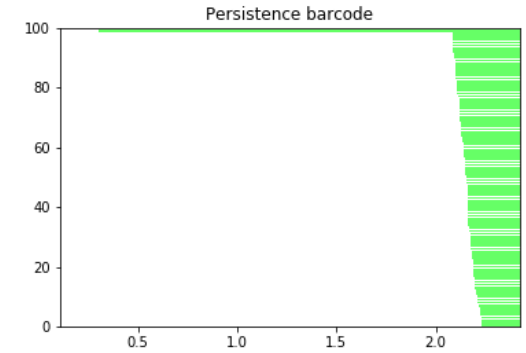
Results A sample of the persistence diagrams obtained is represented in Fig. 2, for the first time series of pedestrian A data. For pedestrian A, for 0-persistence (2a), the most important features are of dimension 1, whereas the most significant features for 1-persistence (2b) are of dimension 0. As for pedestrians C and B (Fig. 3), we can notice that 1-dimension features are more are significant than for pedestrian A in the 1-persistence diagrams. However, it is quite a hard task to compare the persistence diagrams in this form, therefore we will compare the diagrams using the bottleneck distance in the next section.



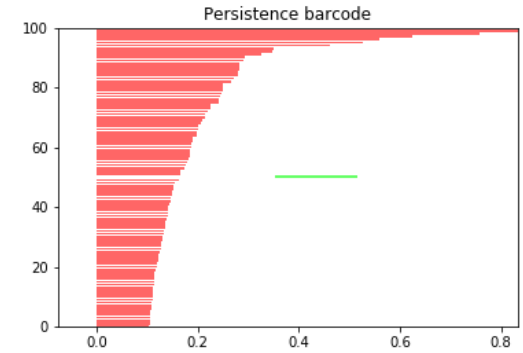
(a) 0-persistence diagram



(b) 1-persistence diagram

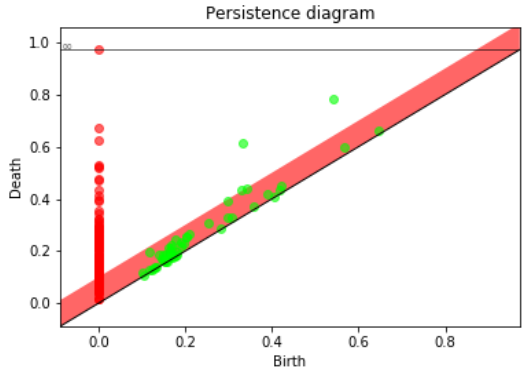


(c) 0-persistence barcodes



(d) 1-persistence barcodes

Figure 2: Persistence diagrams of the Rips filtrations ($r = 3.95$) and persistence barcodes for the first time series of `data A`



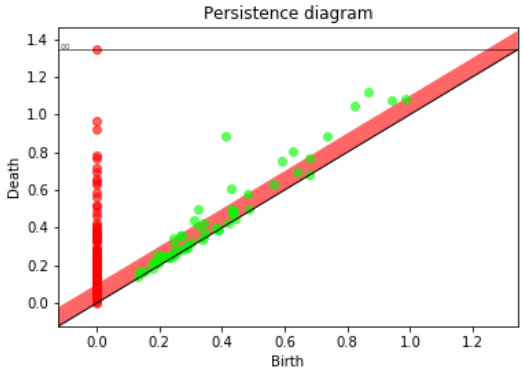


Figure 3: 1-persistence diagram sample for pedestrian B (left) and pedestrian C (right)

When using alpha complex with parameter alpha = 2.7 (4) to build 1-persistence diagrams, we observe that neither dimension 1 nor dimension 2 features are important (they are very close to the diagonal),

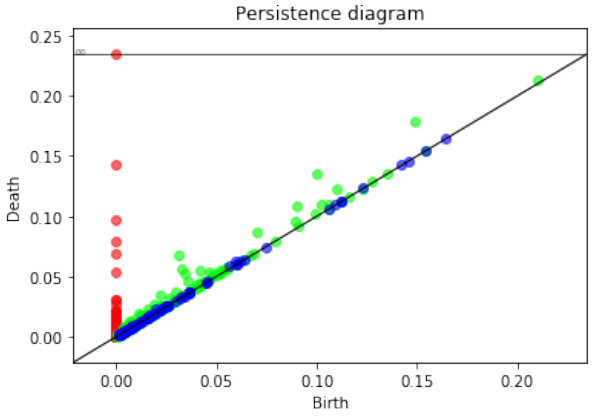


Figure 4: 1-Persistence diagram for pedestrian A with an alpha complex (alpha = 2.7)

Note on outliers We also tried removing the outliers as persistence diagrams of Rips filtrations are very sensitive to noise and outliers, but the results for classification with random forests obtained were not nearly as good as those obtained when keeping the full dataset, so we did not further investigate the question of outliers.

3 Pairwise bottleneck distances and visualization

In order to compare the persistence diagrams obtained, we compute the matrices of bottleneck distances between the 0 and 1-persistence diagrams.

On the bottleneck distance The bottleneck distance measures the similarity between two persistence diagrams. It is the shortest distance d for which there exists a perfect matching between the points of the two diagrams such that any couple of matched points are at distance at most d. The cost of matching, i.e. taking a point p of the first diagram to a point p' of the second diagram corresponds the minimum between moving p to p' and moving both points on the diagonal.

3.1 Implementation

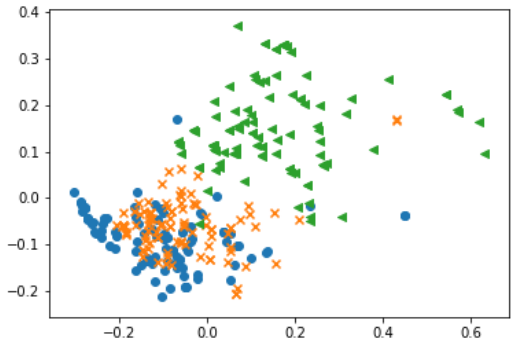
We compute the matrices of bottleneck distances between the 0 and 1-persistence diagrams, using `gudhi.bottleneck_distance`, and represent them in 2D and 3D using `sklearn.manifold.MDS` .

Representation in 2D and 3D While representing directly datasets A, B, and C does not lead to 3 distinct clusters (because the phones were carried in various positions), persistent homology enable us to visualize the three datasets clearly (Fig. 5). This is encouraging with regards to the next tasks, and leads us to believe that persistent homology i indeed relevant for classification in this context.
The graphs obtained lead us to think that using 1-persistence will lead to better classification than 0-persistence, as the clusters seem to be more clearly separated one from the other for 1-persistence.

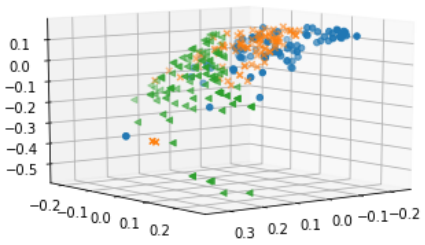
4 Persistence landscapes

So far, we have described the topology of our dataset using persistence diagrams of Rips (0 and 1 persistence diagrams) and Alpha Shape filtrations. And by computing the pairwise bottleneck distance of all the diagrams (3 pedestrians), we managed to separate 3 distinct classes. This guarantee the possibility to build a classifier when using the right data representation. To build the classifier, we will use another descriptor of the topology of the dataset called persistence landscapes. The latter has a real advantage in our problem.

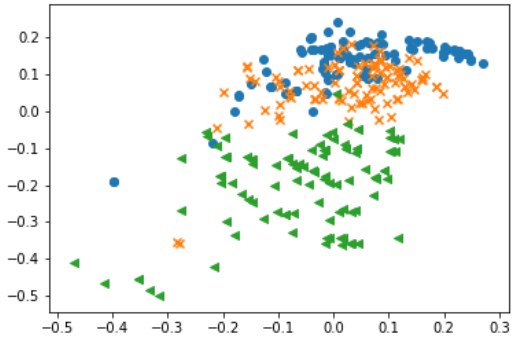
On Persistence landscapes Persistence landscape can be defined as a function from NxR to R. It converts a chosen dimension of a persistence diagrams into sequences of piecewise linear functions and hence allows fast computation when exploiting the data. This is very useful to our classification problem. It also have good stability properties and provides a lower bound for the bottleneck distance.



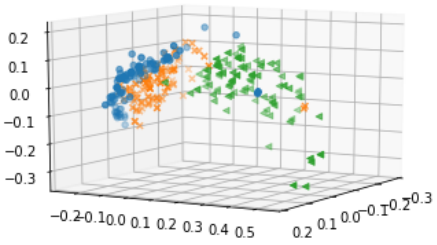
(a) 0-persistence bottleneck distance matrix 2D representation



(b) 0-persistence bottleneck distance matrix 3D representation



(c) 1-persistence bottleneck distance matrix 2D representation



(d) 1-persistence bottleneck distance matrix 3D representation

Figure 5: Bottleneck distance matrices representation for 0 and 1 persistence diagrams. For both 0 and 1 persistence, pedestrian C data (green) seems to be easier to classify (though less "compact") than pedestrian A (blue) and pedestrian B (orange) data which seem to form closer clusters

4.1 Results

Function definition We create a function that transform a diagram (in the Gudhi format) into a grid of size nbld x nbnodes where nbld represent the number of landscapes retained and nbnodes refers to the discretization degree of the axis where lie points of the diagram. It also take as argument an integer k for the dimension of topological features under consideration.

Example The 5 first landscapes on 400 nodes of the 1-persistence diagram corresponding to the first multivariate time series of pedestrian A is computed below (k is fixed to 0).

```
computeLandscapes(diagA_1[0],0,15,400,5,0)
```

```
array([[ 0.    ,  0.0375,  0.075 , ..., 14.8875, 14.925 , 14.9625],
       [ 0.    ,  0.0375,  0.075 , ...,  0.    ,  0.    ,  0.    ],
       [ 0.    ,  0.0375,  0.075 , ...,  0.    ,  0.    ,  0.    ],
       [ 0.    ,  0.0375,  0.075 , ...,  0.    ,  0.    ,  0.    ],
       [ 0.    ,  0.0375,  0.075 , ...,  0.    ,  0.    ,  0.    ]])
```

5 Classification with random forest

In this section, we will discuss the result of a random forest classifier in predicting the pedestrian task. We will use the persistence landscapes build using the 0-persistent diagram, the 1-persistent diagram and the Alpha shape diagram. For each of these 3 diagrams, we build as many persistence landscapes as possible(k <= dimension of the diagram) and for each we run the classifier and record the accuracy. But

before we build the persistence landscapes, we need to choose a relevant range [Xmin,Xmax] that is used to compute the landscapes.

5.1 Selection of the relevant interval

As stated in section 4, the persistence landscapes of a diagram is a matrix of dimension nlbd x nbnodes. In order to capture the topological information of the diagram, we would like to make the matrix less parse possible i.e. with a few null values. To do so, we need to localize the range where the couples (birth,death) of the persistence diagrams lie in and select the interval accordingly. Below we compute some statistics for the 0-persistence and 1-persistence diagrams that will help us pick a relevant interval.

0-persistence diagrams	min birth	max death
Pedestrian A	0.0	1.209
Pedestrian B	0.0	1.241
Pedestrian C	0.0	1.469
1-persistence diagrams		
Pedestrian A	0.0	1.209
Pedestrian B	0.0	1.241
Pedestrian C	0.0	1.608

Xmin and Xmax We compute Xmin as the minimum of the column "min birth" for both minus a small constant and Xmax as the maximum of the column "max death" to which we add the same small constant.

5.2 Implementation and Results

Before we start training the classifiers, we transform each grid (representing the features of an instance) into a vector of dimension nlbd x nbnodes by flattening it. This trick ease the usage of the python library we use to run a random forest.

5.2.1 Implementation through cross validation

For each dataset we concatenate the persistence landscapes of pedestrians and split it randomly between 80% of training set and 20% of testing set. In a cross validation style, we repeat this step 20 times and return the average accuracy.

5.2.2 Results

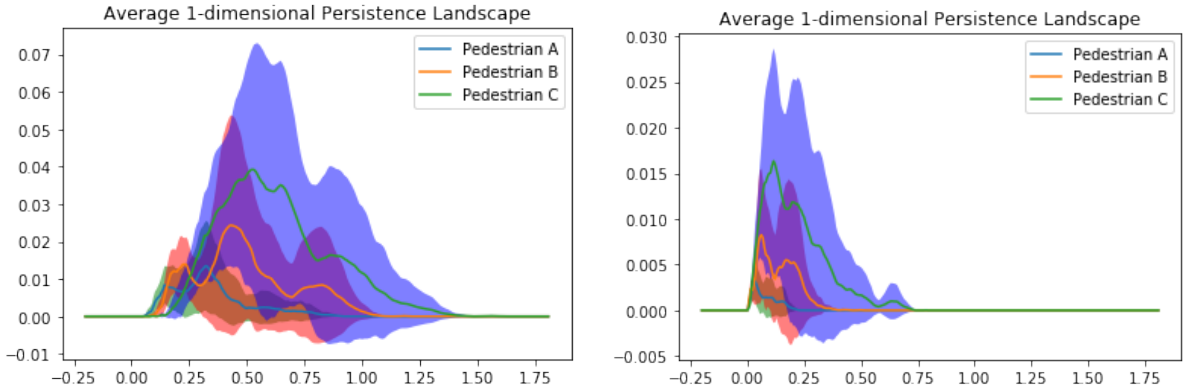
The accuracies of the same classifier trained and tested using different persistence landscapes as input are listed in the table below.

Dataset	Prediction accuracy
0-dimensional persistence landscapes with the 0-dimensional-persistence diagram	84.6%
1-dimensional persistence landscapes with the 0-dimensional-persistence diagram	91.1%
0-dimensional persistence landscapes with the 1-dimensional-persistence diagram	83.1%
1-dimensional persistence landscapes with the 1-dimensional-persistence diagram	98.5%
2-dimensional persistence landscapes with the 1-dimensional-persistence diagram	30.1%
0-dimensional persistence landscapes with the Alpha shape diagram	86.9%
1-dimensional persistence landscapes with the Alpha shape diagram	97.9%
2-dimensional persistence landscapes with the Alpha shape diagram	82.4%

Comments From the table above, we see that the persistence landscapes build with the 1-dimensional-persistence diagram and the Alpha shape diagram give us the highest accuracy. This results confirm what we observe in section 3. We can visualize the average persistence landscapes for these two datasets and for the 3 pedestrians (Fig. 6). Now that we have identified the best datasets, we can try to enhance the results by optimizing the hyper-parameters of the random forest classifier. We do it by using again the grid search of scikit-learn. The results slightly improve.

Dataset	Accuracy with Grid Search
1-dimensional persistence landscapes with the 1-dimensional-persistence diagram	99.6%
1-dimensional persistence landscapes with the Alpha shape diagram	99.6%

We also to concatenate the features of the best persistence landscapes for the 0 & 1-dimensional persistent diagrams. We obtain an accuracy of 96.4%.



(a) 0- dimensional persistence landscapes build with 1-dimensional persistence diagram

(b) 1-dimensional persistence landscapes build with the alpha shape diagram

Figure 6: Average persistence landscapes for the 3 pedestrians.

6 Classification using raw data

In this section we repeat the task of classifying pedestrian using the same random forest algorithm. But instead of using topological descriptors as dataset, we use the raw data directly. We obtain an accuracy of 89%. This value must be taken as the benchmark of the classification problem. We can now state the importance of extracting the topology structure of the dataset when training the algorithm. We improve the performance of the classifier by almost 10%. One should pick the topological descriptor with care as some may give poor performance.

7 Conclusion

In this project, we showed the benefit of coordinate invariance offered by persistence homology, making it possible to successfully complete the task of classification of 3D pedestrian walk acceleration data. The best results were obtained with 1-dimensional persistence landscapes with 1-dimensional persistence diagrams (using either Rips filtrations of alpha shape complexes to build the diagrams), reaching an accuracy of 99.6 %.

References

[1] *Convergence Rates for Persistence Diagram Estimation in Topological Data Analysis*, Frederic Chazal, Catherine Labruere and Bertrand Michel, in Journal of Machine Learning Research 16 (2015) 3603-3635.

[2] *Statistical Topological Data Analysis using Persistence Landscapes*, Peter Bubenik, in Journal of Machine Learning Research 16 (2015) 77-102.