A persistent homology clustering approach for evaluating pollution severity worldwide

Camilla Brigandì, Francesco Lazzari, Paolo Meli, Matteo Pazzini, Riccardo Violano

Introduction

Our Idea was to find a new method to work with clustering problem, and while we were searching in the literature we discovered a new article of April 2024 that was speaking about the idea of using the Persistent Homology to create Cluster, and this inspired us and create curiosity on the topic.

Dataset

The dataset publicly available on Kaggle contained around 23.000 observation of pollution data from more than 150 countries world wide, thus allowing us to have a good baseline for both clustering algorithms and data visualization.
After minor preprocessing (mostly NA's removal), we added via a geolocalization API the coordinates of each (City, Country) as new columns, in order to then visually inspect the quality of each cluster by linking the city to the result of each clustering technique. We then mapped the categorical to numerical values for easier understanding of the different levels of pollution severity, according to official sources.

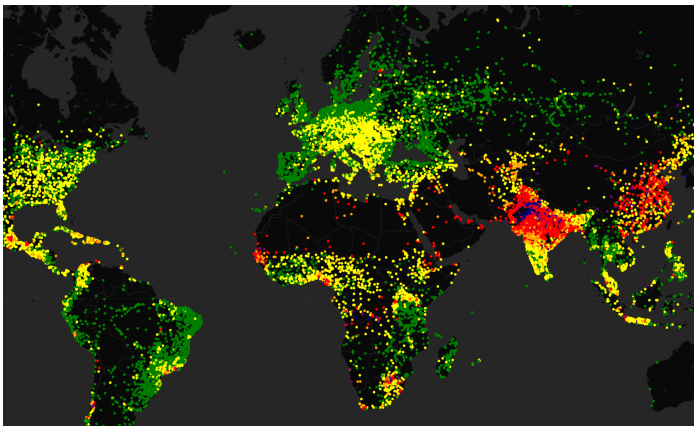

Figure 1: *AQI Categories of pollution data*

Persistent Homology

Persistent homology is an algebraic technique used to retrieve the topological features of shapes and functions. Features of small "persistence" are often regarded as noise, while the ones of big persistence are the "true" ones. A significant part of working with scientific datasets involves reducing noise or smoothing images and other observation records. However, what constitutes noise can be subjective, and even if a clear distinction is established, denoising is challenging due to dependencies that can cause unintended side effects.

The outcome of persistent homology is a collection of persistent pairs $(\sigma_{l(j)}, \sigma_j)$, where for each j, $\sigma_{l(j)}$ generates a component, and σ_j eliminates it. The persistence, or lifetime, of a pair is defined as $j - l(j)$. The k-dimensional persistence diagram consists of points with coordinates $(\alpha_{l(j)}, \alpha_j)$, where $\sigma_{l(j)}$ is a k-simplex, counted with multiplicity. Additionally, points along the diagonal $y = x$ are included with infinite multiplicity, which is useful for defining distances.

In this project, we're basically using only 0-persistent homology, with which we track the evolution of connected components that arise while exploring a filtered space. In particoular, the filtered space of interest for us is the **Nearest Neighbor Vietoris-Rips filtration** of a point cloud in $\mathbb{R}^d$ equipped with the Euclidean distance.

Given a metric space (X, d), its Nearest Neighbor Vietoris-Rips filtration $\mathrm{NNVR}(X, \alpha)$ is defined according to the following:

$$\forall x \in X, x \in \mathrm{NNVR}(X, \alpha) \iff \alpha \geq \inf_{y \in X, y \neq x} d(x, y).$$

The hardest challenge of this type of approach is choosing the right threshold to divide correctly the connect components, because if we choose a too high threshold everything will converge in an unique connected component, while if it's too low we're basically detecting noise.

Our Work

Our work consisted in implementing the clustering algorithm described in , testing it on some toy datasets (the same that have been used in the article, in order to be able to double-check the results we obtained), and then using it on the Air Pollution dataset described in the previous section.

The algorithm skeleton is the following:

- **Input**: a finite metric space (X, d);
- Compute the filtration $\mathrm{NNVR}(X)$ and its 0-dimensional persistence diagram;
- Choose a birth threshold and a persistence threshold;
- Mark all points whose birth dates are above the birth threshold as outliers. Let Y be the remaining points;
- Compute 0-dimensional persistent homology on $\mathrm{NNVR}(Y)$, but do not add edges that would merge components whose persistence is above the persistence threshold;
- **Output**: a simplicial complex, where each connected component is a cluster.

Following this algorithm, we implemented some functions needed to construct the NNVR filtration starting from both a point cloud and a matrix of distances, the maxjump algorithm, a function that detects the outliers and removes it, and a function that computes the clusters on the dataset after the outlier removal step.

Once our code was written, we tested it on the same artificial dataset contained in the article.
On the first toy example (a very small graph dataset containing 9 nodes) we obtained the exact same results.

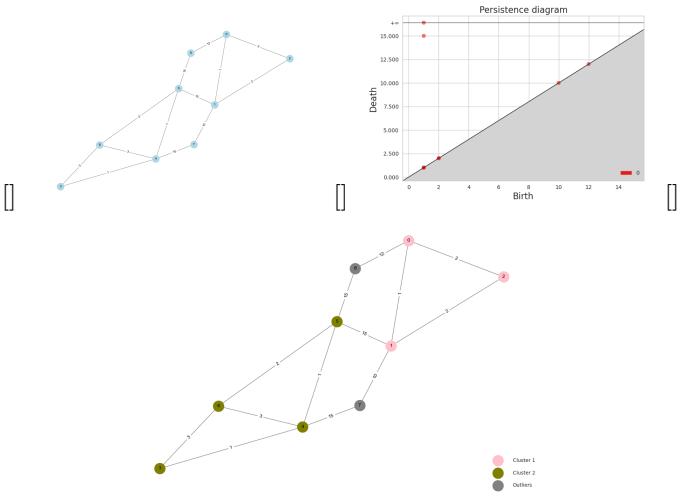

Figure 2: *Original toy dataset; Persistence diagram; clusters on toy dataset*

The second toy example was the karate club graph but with the weights on the edges set to the inverse of the original ones and using as a distance the shortest path distance between the nodes. Unlike the authors of the article, we didn't add any outliers to the original graph. In this case, the results we obtained are slightly different from the original ones, but we know the reason behind it: to compute 0-persistence homology, a minimum spanning tree is computed, so the order with which the algorithm explores the graph edges changes the resulting MST (since it's not unique). The problem basically lies in the fact that in the MST computed by our algorithm, the edges that link the nodes that are in the "wrong" cluster with the "right" one are not present at all since these nodes are equally distant from the two clusters.

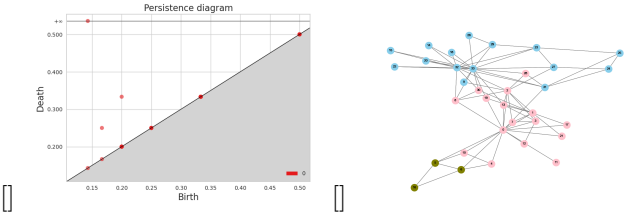

Figure 3: *Persistence diagram on karate club graph; clusters on karate club graph*

The third toy example is the one on the point cloud dataset formed by two noisy circles with 4 outliers manually added. In this case, both the clusters and the outliers are detected perfectly with , but we saw that increasing the noise parameter in the dataset generation or putting the outliers nearer to the actual dataset makes the algorithm with this thresholds fail.

At this point we were sure that our algorithm worked correctly, so we computed the clusters on the air pollution dataset described in the first section.

We computed the clusters on the dataset after applying PCA. The PCA application was done in order to be able to visualize the results also on a scatterplot to have a better understanding of how the algorithm worked, since it sees the points as actual points in $\mathbb{R}^d$ and not points on a map.

The results we obtained with this algorithm were compared to the ones obtained with some other algorithms. In particoular, we used:

- Hierarchical Clustering - Ward distance
- K-means
- Optics
- H-DBSCAN
- Mean-Shift
- Spectral Clustering

The results obtained were visualized both on an interactive map and on a scatterplot for the PCA results.

Results

In this section we're going to analyze the results obtained with some of the previously mentioned clustering algorithms, besides the ones obtained with the new method.

NNVR Filtration based Clustering - As said in the previous section, the algorithm was runned on the dataset after applying principal component analysis as dimensionality reduction. The persistence diagram obtained from the NNVR filtration constructed on this dataset is in 4.

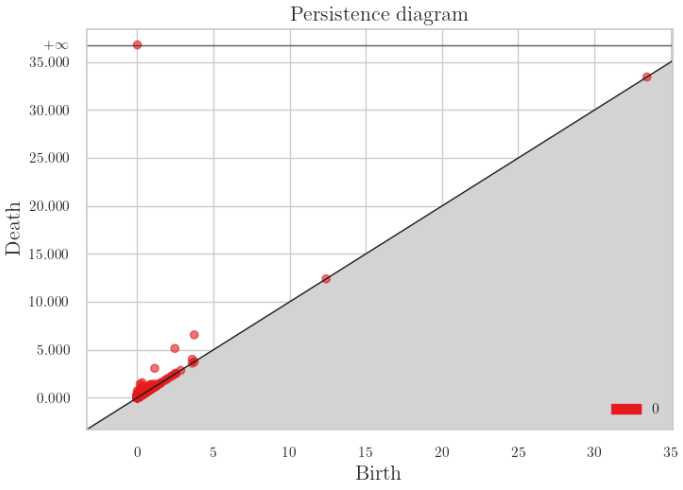

Figure 4: *Persistence diagram before outliers removal*

At first, we used *maxjump* algorithm to compute both the birth and the persistence threshold, then we removed the outliers and computed the clusters. With these thresholds we obtained only two points as outliers and 29 clusters. Since the number of obtained clusters was too high (we expected between 3 and 6 clusters, since there are 6 Air Quality Index categories), we then decided to inspect the persistence diagram obtained from the NNVR filtration of point cloud without the outliers (5) to set a persistence threshold that would give us back a number of clusters in line with our expectations. The thresholds used are in table 1 .

Persistence threshold (maxjump):	0.38
Persistence threshold (6 clusters):	1
Birth threshold (maxjump):	10.53

Table 1: *Persistence and birth thresholds*

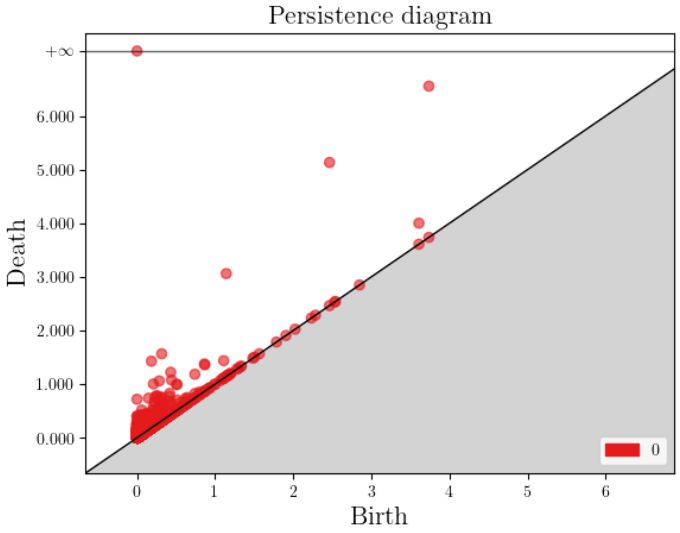

Figure 5: *Persistence diagram after outliers removal*

With the second threshold we obtained exactly six clusters, that we visualized on the map using colours that shouldn't recall the levels of AQI. The results can be seen in 6.

From the map, it's easy to see that the even if the Silhouette score obtained is very high (approximately 0.79) algorithm doesn't perform well on this data. Indeed, we have a very big cluster containing almost all the points and the other five containing the remaining ones. One possible explanation for this behaviour of the algorithm is the fact that the data after the PCA form a sort of cone in $\mathbb{R}^3$ whose top part contains most of the data points, that are really close one to the other, making it impossible for the algorithm to separate them. Besides, if we look at the persistence diagram we can see that some components we are considering as clusters have a very high birth time, indicating that they consist of points really far one from another and whose nearest neighbor is also really far, indicating the fact that we could be distinguishing some clusters made only of outliers that weren't detected with the previous birth threshold.

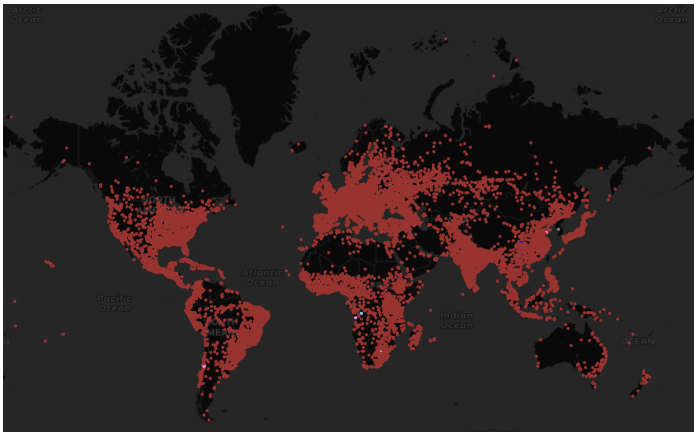

Figure 6: *NNVR Filtration based clustering, 6 clusters*

At this point we decided to visualize the results obtained with both thresholds obtained using *maxjump* algorithm but using a scatterplot of the dataset.

From the scatterplot we can see that also in this case there's a big cluster composed of the points near the top of the cone. The other 28 clusters seem to detect noise and/or to pair "outliers".

After seeing these results, we also tried to run the

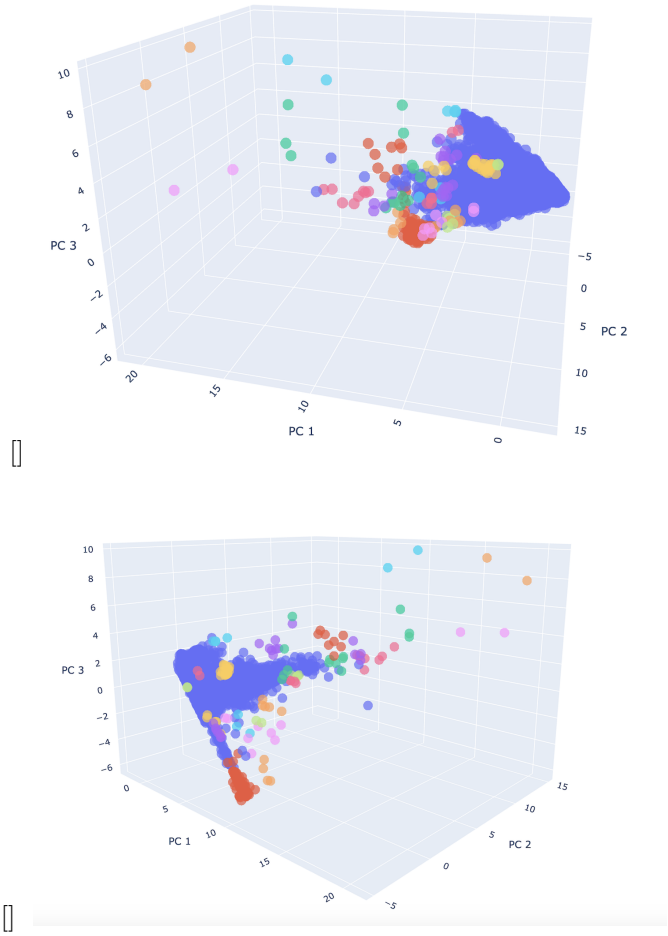

Figure 7: *Scatterplot of data pints colored by clusters; clusters obtained using NNVR filtration algorithm with persistence threshold = 0.38*

algorithm on the dataset without running PCA on it, but only removing the PM2.5 Value variable, which is highly correlated to the AQI Value one, but the results we obtained were even worse: only one point was detected as outlier and using the persistence threshold obtained with*maxjump* we obtained 377 clusters which lead to a negative silhouette score: the threshold was so low that the algorithm was basically dividing noisy components. When we forced the threshold in order to have an acceptable number of clusters, the Silhouette score was higher, but we incurred in the same problem as the one of the PCA data: the algorithm gave back one cluster containing most of the points. We decided not to run the comparison algorithm on the unmodified data.

Hierarchical clustering, ward distance - This method performs well on our dataset. We obtained seven clusters of the following cardinalities:

Cluster Label	Number of points
Noise	0
0	1
1	1613
2	78
3	15702
4	727
5	1347
6	2799

Table 2: *Hierarchical clustering, clusters cardinality*

By looking at the centroids of the clusters, we were able to gain some insight on the six detected clusters: Cluster 0 features a high Air Quality Index (AQI) with slightly above-average ozone levels, while Cluster 1 is notable for its extremely high carbon monoxide (CO) values, along with elevated levels of AQI and nitrogen dioxide (NO2). Cluster 2 has pollutant levels below average, representing the city with the cleanest air. In contrast, Cluster 3 is defined by high ozone levels. Cluster 4 shows a high presence of NO2 and above-average CO, and finally, Cluster 5 is marked by pollutant values that are very close to the average.

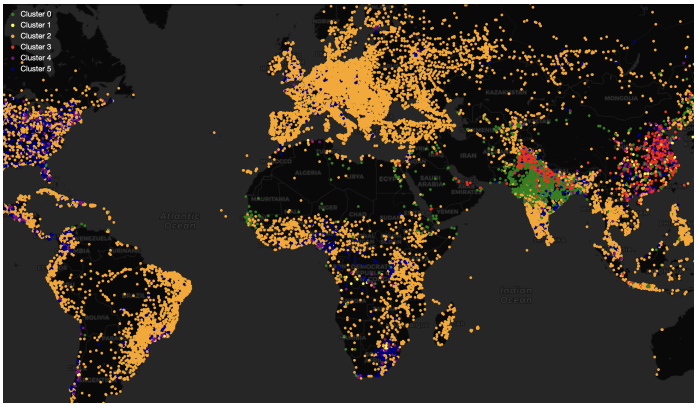

Figure 8: *Hierarchical clustering with Ward distance results*

Besides, if we look at the map in 8, we can see that the algorithm highlights the heavily polluted areas (look at India, for example) while flattening a bit the less polluted ones.

Kmeans clustering - We decided to include also the results from the Kmeans algorithm since is the one that seems to perform better on this dataset.

In this case, we found six clusters with the following carinalities:

Cluster Label	Number of points
0	7290
1	961
2	264
3	2306
4	10129
5	1316

Table 3: *K-means clustering, clusters cardinality*

The clusters are distrubuted as in the map in 9

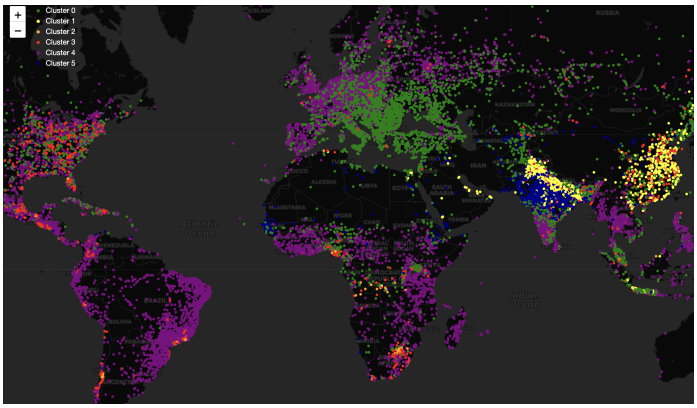

Figure 9: *K-means clustering results*

To better understand the content of the clusters, we can say that Cluster 0 has all values relatively close to the average, while Cluster 1 is characterized by high ozone levels and a relatively high AQI value. Cluster 2 stands out with extremely high levels of CO, NO2, and AQI value. Cluster 3 is also close to the average, almost mirroring Cluster 1, suggesting two types of situations near the average. Cluster 4 shows low values for all variables, representing the cluster with the cleanest air compared to the others. Lastly, Cluster 5 has a high AQI value but remains relatively close to the average for the other variables.

H-DBSCAN and OPTICS clustering - After carrying out a grid search to find the best parameters giving a reasonable number of clusters (between 3 and 6), we were able to derive three clusters with the cardinality highlighted in table 4.

Cluster Label	Number of points
Noise	4766
0	444
1	16691
2	365

Table 4: *H-DBSCAN clustering, clusters cardinality*

Already from this information it is clear that the result is far from being optimal, however, if represented graphically a few things can be noted: the algorithm succeeds in identifying some of the most polluted areas (cluster 0 and cluster 1) quite well, but fails in correctly assigning all those points in between with cluster 1 (good air quality), which have been incorrectly classified as noise. This is especially evident from the inspection of the 3D scatterplot in 10.

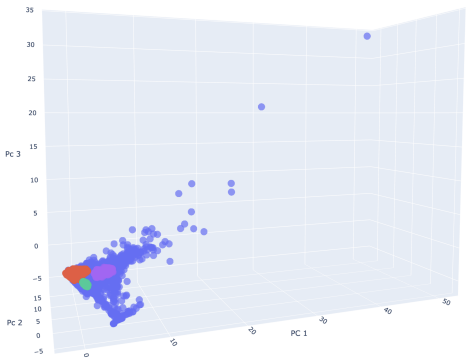

Figure 10: *H-DBASCAN clustering scatterplot. In blue are highlighted the noise point which are not correctly assigned to a cluster*

A similar result was also obtained with the OPTICS algorithm, proving the similarity between the two models used.

Mean-Shift and Spectral clustering - The latter algorithms fail in no way to identify a reasonable number of distinct clusters (by cardinality). On the contrary, they tend to assign almost all the points to the same cluster, effectively failing to differentiate the variability of the data. The motivation behind the poor performance of the models is attributable, again, to the very dense conical shape of the points, which we have already explained in the analysis of the NNVR Filtration based algorithm.

Conclusions

To summarize, we can say that this new clustering algorithm works very well in cases in which the clusters are well separated also with the "automatic" thresholds, while these thresholds start to give problems even if the clusters are well separated but containing noise, or the outliers are not distant enough from the "clean" dataset (see the comments about the two noisy circles dataset). Sometimes, these problems are solvable by inspecting the persistence diagram of the dataset and manually setting both the birth and the persistence threshold. In some cases leveraging some previous knowledge about the cluster numbers can be helpful in identifying the right ones. On the other hand, the situation is critical when this algorithm works on dataset in which the cluster are not well separated and equally distant from each other, like in our case. Besides, working on large datasets can be challenging since the persistence diagram in this cases could be a little messy, and leveraging previous knowledge about the clusters number can be tricky. Besides, to be sure to remove all the outliers, one can do the outlier remove step more than one time, but this could also lead in the removal of some "non-outlier" points. In conclusion, after this analysis it seems that the algorithm may not perform well on some real data, since the "optimal conditions" for its application (at least in its non-parametric version) are really hard to find. Besides, for this dataset the algorithm is not appropriate and the best performances on it are obtained using hierarchical clustering and K-means.

References

[1] Laurent, O., Alexandre, B., Brian, T. "Persistence-based clustering with outlier-removing filtration". *Frontiers in Applied Mathematics and Statistics*, (2024).

[2] Data scraped from `https://www.elichens.com/`. "Global Air pollution dataset". Available at: `https://www.kaggle.com/datasets/hasibalmuzdadid/global-air-pollution-dataset/data` (2022).

[3] Edelsbrunner, H., Harer, J. "Computational Topology: An Introduction". *American Mathematical Society*, XII, 241p. (2010).