# Bridge Aware Clustering with Noise Detection

## Summary

**Candidate**:
Christian Paesante
**Supervisors**:
prof. Paolo Garza
prof. Luca Cagliero
ing. Luca Colomba

July 2021

Clustering is a set of techniques aiming to select and group homogeneous elements in a dataset. Clustering is an unsupervised technique that allows to discover patterns and distributions similarities between points in a dataset. Clustering algorithms are organized in few categories such as:

1. partition-based
2. hierarchical
3. density-based

Between its various applications it allows to group individuals in groups with similar behaviour or characteristics.

Density-based algorithms are algorithms that rely on detecting highly dense regions in order to aggregate points in those regions under the same cluster. They allow to identify clusters of any shape. Indeed partition-based clustering algorithms fail to clusterize non-globular shaped clusters. An example is the "Banana" cluster shown in Figure 1 and Figure 2.
On the other hand density-based clutering algorithms have some issues in dealing with multi-density datasets.

Bridge Aware Clustering is a density-based algorithm that relies on identifying bridge points inside a connectivity graph as points separating different clusters. These bridge points are identified as points with a high Local Outlier Factor, computed by means of the homonym Local Outlier Factor outlier detection, which is a density-based algorithm. The issues of Bridge Aware Clustering are that it doesn't scale very well on large scale datasets, it doesn't detect noise points and occasionally suffers of very light cluster fragmentation.

This work focuses on improving a density-based algorithm called Bridge Aware Clustering in terms of scalability by means of implementing a distributed version of it, improving its cluster fragmentation robustenss by exploring an existing cluster fusion technique and improving its noise robustness by means of implementing a noise detection strategy.

The scalability improvements were achieved by implementing a distributed version of Bridge Aware Clustering on Spark using Python. The algorithm relies on [1] for the bridge detection and using GraphFrame connected components for labeling clusters.
The cluster fragmentation robustness improvements were conducted by integrating a cluster fusion technique documented in the literature and testing it over different benchmark datasets.

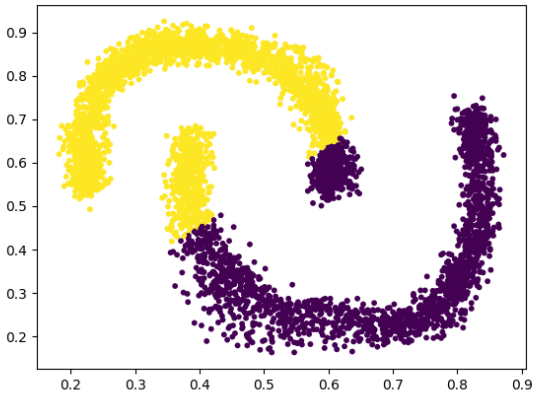


**Figure 1:** Banana using KMeans (partition-based)

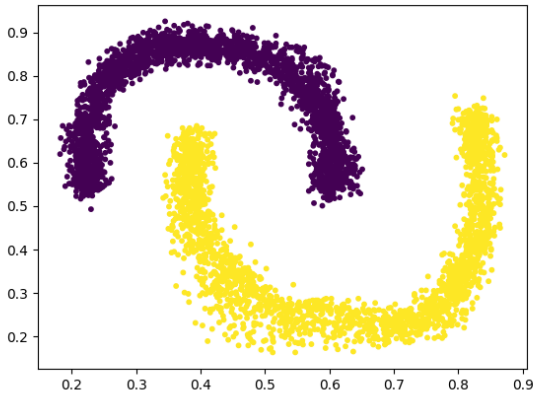


**Figure 2:** Banana using DBScan (density-based)

The cluster fusion technique was inspired by the Cluster Self-Ensemble technique proposed by [2], but differs from it by applying the process on the detected bridge points instead.
The noise robustness improvements were conducted by integrating a noise detection method through an extensive testing campaign aiming to evaluate the noise robustness over different levels of noise. The noise detection method is implemented by labeling bridge points as noise points depending on their neighborhood. A classifier was trained to identify noisy datasets and conditionally activate the noise detection component, thus making the method semi-supervised.

The Distributed version of Bridge Aware Clustering was tested on Open Street Map and GeoLife datasets both of Gigabyte order size. The results have shown some difficulties. Specifically, a bottleneck was identified during the outlier identification step, which requires the computation of an approximated quantile.

The Bridge Aware Clustering version with Cluster Fusion and the one with Noise Detection was tested on 25 synthethic bi-dimensional benchmark datasets used in the Bridge Aware Clustering work, which were selected from the public GitHub repository https://github.com/deric/clustering-benchmark.
For the Noise Detection version further experiments were conducted by synthetizing for each dataset a noisy version with different levels of noise.

The Cluster Fusion showed to be too aggressive by aggregating separated clusters together often ignoring fragmented datasets as shown in Figure 3 and Figure 4.

The Noise Detection on the other hand showed a 0.2% of improvement on the Mean ARI score over the 25 datasets for each different level of noise as shown in results reported in Table 1 and Table 2.

| Algorithm | Mean ARI |
|---|---|
| DBSCAN | 0.6874 |
| OPTICS | 0.5035 |
| BorderPeeling | 0.4627 |
| BAC | 0.7933 |
| BAC with noise detection | 0.7951 |

**Table 1:** Algorithms comparisons with 10% of noise

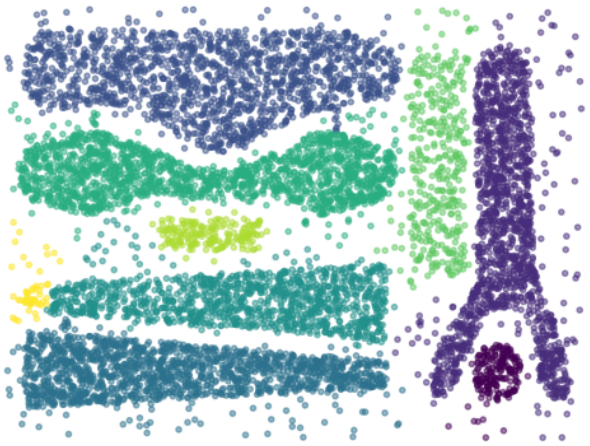

**Figure 3:** Cluto-t8-8k BAC

**Figure 4:** Cluto-t8-8k BAC Ensembled

| Algorithm | Mean ARI |
|---|---|
| DBSCAN | 0.6190 |
| OPTICS | 0.4518 |
| BorderPeeling | 0.4120 |
| BAC | 0.7128 |
| BAC with noise detection | 0.7146 |

**Table 2:** Algorithms comparisons with 15% of noise

This work showed an attempt to improve an already extremely good algorithm and bringing it into the Big Data world. The work proposed a new way of detecting noise in starting from a newly defined type of points called "Bridge Points" which are likely to belong to cluster frontiers. Such proposed helped to improve performances of Bridge Aware Clustering on noisy datasets. Despite its very simple approach, the experiments conducted on the same datasets used by the original version of Bridge Aware Clustering showed an improvement of 0.2%.

Future development of this work would be:

1. Investigate new strategies in computing the threshold for marking points as bridge points which may overcome current scalability limitations.

2. Investigate new strategies that may futher enhance the noise detections capabilities, creating a noisy-metric that tells for each point the likeliness to be a noise point.

# References


[1] Yan Y. Cao L. Kuhlman C. Rundensteiner E. “Distributed Local Outlier Detection in Big Data”. In: *KDD 2017 Research Paper* (2017), pp. 1226–1231 (cit. on p. 1).

[2] Chen J. and Yu P. “A domain adaptive density clustering algorithm for data with varying density distribution”. In: *IEEE Transactions on Knowledge and Data Engineering* (2019), p. 4 (cit. on p. 2).