DenLAC: Density Levels Aggregation Clustering – A Flexible Clustering Method –

Iulia-Maria Rădulescu(✉)[0000−0002−4502−970X],
Alexandru Boicea[0000−0001−9739−0161],
Ciprian-Octavian Truică[0000−0001−7292−4462],
Elena-Simona Apostol[0000−0001−6397−4951], and
Mariana Mocanu[0000−0002−8305−2652], Florin Rădulescu[0000−0001−6425−7567]

University Politehnica of Bucharest, Bucharest, Romania
{iulia.m.radulescu, alexandru.boicea, ciprian.truica, elena.apostol, mariana.mocanu, florin.radulescu}@upb.ro

Abstract. This paper introduces DenLAC (Density Levels Aggregation Clustering), an adaptable clustering algorithm which achieves high accuracy independent of the input's shape and distribution. While most clustering algorithms are specialized on particular input types, DenLAC obtains correct results for spherical, elongated and different density clusters. We also incorporate a simple procedure for outlier identification and displacement. Our method relies on defining clusters as density intervals comprised of connected components which we call density bins, through assembling several popular notions in data mining and statistics such as Kernel Density Estimation, the density attraction and density levels theoretical concepts. To build the final clusters, we extract the connected components from each density bin and we merge adjacent connected components using a slightly modified agglomerative clustering algorithm.

Keywords: Clustering, Kernel Density Estimation, Probability Density Function, Hierarchical Clustering

1 Introduction

Clustering is the process of grouping a set of points into clusters such that the points in the same cluster have high similarity but are significantly dissimilar from the points in other clusters. Even though a large number of clustering algorithms have been developed over time, most of them perform well only on certain datasets, as we show in Section 2 and Section 6. Therefore, we propose DenLAC, which achieves high accuracy for various types of datasets regardless of the input's shape or distribution.

Relying on the cluster definitions proposed in [14] and [4, 13], we demonstrate that clusters consist of density intervals containing connected components, which we call density bins, the key concept of our method. We then extract the connected components employing a nearest neighbour approach: we iterate through each density bin's objects, trying to expand clusters by recurrently merging the nearest neighbors for each object. The DenLAC algorithm sequentially applies the following steps: i) evaluate the input's probability density function; ii) remove the outliers with the help of the probability density function; iii) assign the input objects to the corresponding density bins; iv) split the density bins into adjacent connected components; v) aggregate the connected components from the previous step in order to build the final clusters.

We evaluate DenLAC alongside several well-known clustering algorithms on eight synthetic bi-dimensional datasets and on the Iris and Tetragonula real datasets in order to demonstrate its flexibility. The employed quality functions reveal that DenLAC obtains higher accuracy as compared to the other considered algorithms.

2 Related Work

Clustering algorithms fall into five broad categories: partitional (representative based), hierarchical, density-based, probabilistic and spectral.

Representative based algorithms rely on the intuitive notion of distance to cluster the data points. The objects within the dataset are partitioned around a subset of chosen representatives. K-means [18] and CLARANS (Clustering Large Applications based on Randomized Search) [22] are two examples of these clustering methods. K-means is sensitive to outliers because it relies on the mean computation. It is also a randomized algorithm, therefore it considerably depends on the parameters initialization. Also, K-means uses the square-error to measure the quality of the clustering results and therefore works well when clusters are compact clouds that are rather well separated from one another [12]. CLARANS is also randomized, thus, it is sensitive to the parameters initialization; moreover it only relies on the distances between the objects to be clustered.

Hierarchical algorithms rely on computing the pairwise distances between the input dataset's instances. There are two main types of hierarchical clustering methods: agglomerative (bottom-up) clustering and divisive (top-down) clustering. Hierarchical algorithms do not scale well with large data sets due to the quadratic computational complexity of calculating all the pairwise distances and suffer from problems such as chaining. BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [31] is a more efficient hierarchical clustering algorithm used for large datasets. Because BIRCH structures the data as a tree, the clustering result depends on the input objects order. In addition to this, BIRCH does not obtain accurate results for arbitrarily shaped clusters as it uses the diameter of the cluster to determine its boundaries. CURE (Clustering Using Representatives) [12] is an agglomerative hierarchical algorithm which can identify non-spherical clusters. CURE fails to take into account special characteristics of individual clusters, thus making incorrect merging decisions when the underlying data does not follow the assumed model [16].

Density based clustering methods rely on the consideration that the typical density of points within a cluster is considerably higher than outside of the cluster [7]. DBSCAN (Density Based Spatial Clustering of Applications with

Noise) [7] measures density as the number of points within the radius ε of a point within the analyzed dataset and is accurate for arbitrarily shaped groups of objects. However, it fails to cluster datasets with varying density accurately.

The Gaussian Mixture [2] is a probabilistic clustering algorithm. It assumes that the input objects were drawn from k multivariate normal distributions, where k is the number of clusters and relies on Expectation Maximization to determine each distribution's specific parameters. The algorithm works well only on datasets which follow the normal distribution (elliptical clusters) and since it uses Expectation Maximization for the mixture model's parameters computation, it is sensitive to the parameters initialization.

Spectral Clustering [21] relies on computing the eigenvectors associated with the Laplacian matrix derived from the dataset's similarity graph and then clusters them using any algorithm, for example K-means. Spectral clustering cannot successfully cluster datasets that contain structures at different scales of size and density [20].

3 Methodology

In this section we present the theoretical concepts on which DenLAC is based on. We first introduce the notion of density attraction, defined in [14]. Then, we reproduce Chaudhuri et al.'s density levels based cluster definition [4]. Finally we explain how our method utilizes the aforementioned concepts.

3.1 DENCLUE (DENsity-based CLUstEring) Algorithm

The DENCLUE (DENsity-based CLUstEring) algorithm [14] identifies clusters using a hill climbing procedure which is guided by the probability density's function gradient, assigning the objects approaching the local maximas of the probability density function to the same cluster. [14] characterize center-defined clusters and arbitrary-shaped clusters in terms of the local maximas of the probability density function, called "density attractors": i) the center-defined clusters consist of the objects which are "density attracted" by a density attractor whose density is greater than a given threshold ξ and ii) the arbitrary shaped clusters consist of objects which are "density attracted" by a set of density attractors, provided that there exists a path between the density attractors with density above a given threshold ξ.

We display the formal density attractor-based cluster definitions proposed in [14] below, using the following notations: i) D represents the d-dimensional input dataset, ii) $x \in D$ denotes an object in the input dataset iii) we note the probability density function with f iv) $x*$ represents a density attractor, more specifically a local maxima of the probability density function f

Definition 1 (Center-Defined Clusters). *A center-defined cluster given a density attractor $x*$ is a subset C of the input dataset D with $x \in C$ being attracted by $x*$ and $f(x*) \geq \xi$. Points $x \in D$ are called outliers if they are density attracted by a local maximum $x*_0$ with $f(x_0*) < \xi$.*

Definition 2 (Arbitrary-Shape Clusters). *An arbitrary-shape cluster given the set X of density attractors is a subset C of the input dataset D where: i) $\forall x \in C \exists x* \in X : f(x*) \geq \xi$, x is density attracted to $x*$ and ii) $\forall x*_1, x*_2 \in X : \exists$ a path $P \subset F^d$ from $x*_1$ to $x*_2$ with $\forall p \in P : f(p) \geq \xi$, where F^d is a d-dimensional feature space.*

3.2 Clusters as Connected Components At Different Density Levels

[4, 13] define clusters as connected components at each density level λ of the dataset D. The density levels are regions for which the density value is equal to or above a particular level and can be estimated using Kernel Density Estimation [5]. The formal definition is:

Definition 3 (Density Levels). *Given a level λ, the λ-density level is: $L_\lambda = \{x : P(x) \geq \lambda\}$.*

The connected components at a lower density level incorporate the connected components at higher density levels. Thus, a cluster of density f is any connected component of $x : f(x) \geq \lambda$, for any $\lambda > 0$. The collection of all such clusters forms an (infinite) hierarchy called the cluster tree [4]. Finding consistent and reliable approximations of the cluster tree has been studied extensively in [13, 4].

3.3 Clusters as Adjacent Density Bins

From Definition 1 we infer that the members of a center-defined cluster tend to the local maxima of the probability density function within the cluster: the estimated probability density values of the cluster members decrease linearly as they move away from the density attractor. From Definition 2 we infer that the probability density values for the members of an arbitrary shaped cluster are greater than a given threshold ξ. Assuming that the probability density function is estimated using KDE employing a smooth kernel function, such as the Gaussian kernel, we infer that the estimate's values do not rise or drop suddenly.

Considering the deductions above, we view clusters as sets of adjacent density intervals consisting of connected components, which we call density bins. We formally define density bins with the help of Definition 3 as the set differences between adjacent density levels described in Definition 4.

Definition 4 (Density Bins). *For a given dataset D with the probability density function $f(\overline{D})$ and n density levels L_{λ_i}, where $i \in (0, n)$, we define density bins as: $B_i = L_{\lambda_i} \setminus L_{\lambda_{i-1}}$.*

For example, bn bins, each consisting of cm_n connected components, are structured as follows:

$$\begin{aligned}
B_1 &: L_{\lambda_1} \approx C_{1,1} \cup C_{1,2} \cup ... \cup C_{1,m_1} \\
B_2 &: L_{\lambda_2} \setminus L_{\lambda_1} \approx C_{2,1} \cup C_{2,2} \cup ... \cup C_{2,m_2} \\
&\cdots \\
B_{bn} &: L_{\lambda_{bn}} \setminus L_{\lambda_{bn-1}} \approx C_{bn,1} \cup C_{bn,2} \cup ... \cup C_{bn,m_{bn}}
\end{aligned}$$

Where C_{i,m_i} is a connected component for the i-th bin, with $i = \overline{1, bn}$.

4 Algorithm Pipeline

The proposed algorithm pipeline, which is graphically displayed in Figure 1, consists of the following modules: i) *Density estimation*: we estimate the density of the input dataset using Kernel Density Estimation ii) *Outlier identification and displacement*: we eliminate the outliers using the probability density function; iii) *Density based partitioning*: we identify the density levels in the input dataset and extract compact sets of objects that are positioned at large distances one to another which represent the differences between adjacent density levels; iv) *Distance based splitting*: we split the compact sets of object using the distance between them; v) *Merging the final partitions*: we merge the closest partitions and return the final clusters in order to minimize the distance function [31].

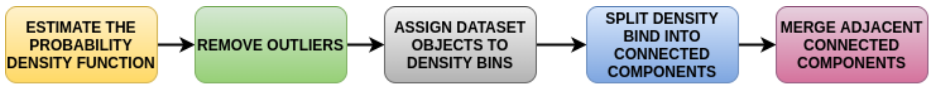

ESTIMATE THE PROBABILITY DENSITY FUNCTION
REMOVE OUTLIERS
ASSIGN DATASET OBJECTS TO DENSITY BINS
SPLIT DENSITY BIND INTO CONNECTED COMPONENTS
MERGE ADJACENT CONNECTED COMPONENTS

Fig. 1: DenLAC Algorithm Pipeline

In the following sections we describe each step in detail, using the following notations: i) $D = d_1, ..., d_n$ represents the d-dimensional, input dataset of size n ii) $B = \bigcup_{i=1}^{bn} B_i$ represents the density bins set of size bn, defined in the previous section.

4.1 Density Estimation. Outlier Identification and Displacement

We identify the dense regions of the input set D by estimating the probability density function using Kernel Density Estimation (also known as KDE). We choose KDE due to its independence of the input dataset distribution's underlying form. Out of the variety of available kernel functions (cosine, exponential, linear, epanechnikov, gaussian, etc.) we prefer the gaussian kernel function since it produces the smoothest estimate, thus being the most suitable for the density levels partitioning. We employ Scott's rule of thumb [25] for determining the optimal bandwidth, as it is a classic, well-established method.

DenLAC utilizes the probability density function to partition the input dataset according to its density. However, the probability density function is also helpful for removing potential outliers by detecting probability density function's unusually low values.

To determine these values we used the interquartile range, noted IQR [26], defined as the difference between the 25^{th} and the 75^{th} percentiles. This method is more robust than Z-score for the reason that Z-score relies on the mean and the standard deviation, which are affected by outliers. The k^{th} percentile can be defined as a value x_j from a given data set $X = x_1, x_2, ..., x_n$ having the property that k percent of the values are less or equal than x_j. The probability density function values of the outliers, associated with regions with extremely low density, are more than $1.5 \times IQR$ below the 25th percentile. Therefore, we remove the objects with probability density function values in this interval.

4.2 Density Based Partitioning

We partition the input dataset D into bn density bins by allocating each object d_i to a density bin B_i according to its estimated probability density function, as follows:

Given a dataset instance $d_i \in D$ and a density bin B_i delimited by the boundaries B_i^{min} and B_i^{max}, we assign d_i to B_i if $f(d_i) \geq B_i^{min}$ and $f(d_i) < B_i^{max}$, where $f : D \to \mathbb{Z}$ is the estimated probability density function. The number of density intervals, bn, must be supplied at runtime as parameter. Section 5, Subsection 5.1 offers several guidelines on choosing the algorithm's parameters. The density bins set and the density based split result for the Aggregation [11] synthetic dataset can be visualized in Figure 2. The density bins before and after allocating the input dataset's objects are graphically represented in Figure 2b (the regions colored with the same shade blue) and Figure 2c (the groups of points with the same colour) respectively.

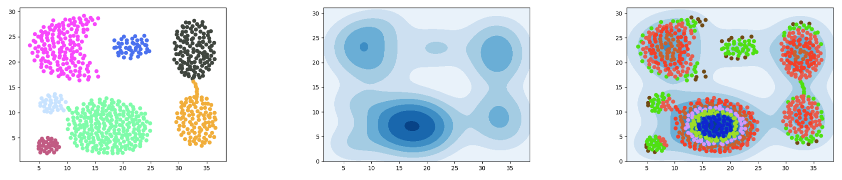

(a) Clusters (b) Density levels (c) Density split

Fig. 2: Density based splitting

4.3 Splitting the Density Bins Into Distinct Connected Components

The resulting density bins consist of distinct connected components (Definition 4) which must be separated. For example, all the points marked with red in Figure 2c are assigned to the same density interval according to their estimated probability density. Therefore, the set of points must be divided into distinct connected components. We split the connected components using a nearest-neighbour approach. We examine each object within a density bin and start extending a cluster from it. We extend a cluster by adding the object's nearest neighbours whose distances to the selected object are smaller than a specific threshold.

We determine the correct threshold for each bin using the same approach as [17] and [10]. The above-mentioned papers address the problem of DBSCAN's radius threshold ϵ automatic computation by laying out a k-distance plot representing the density levels as interconnected smooth curves. Similar to [17], we compute the distances between each point and its k-th neighbour, where k depends on the input dataset's D dimension d and is automatically determined as follows: $k = 2 \cdot d - 1$. In order to deduce k we used the same rule of thumb for deducing the nearest neighbour as the one mentioned in [24]. Then, we use these values to draw the k-distance plot; the threshold value, $threshold_{kDistance}$, corresponds to the sharp change point of the outlined curve. We formally define our k-distance plot as $kDistance \colon B_{i_{sortedK}} \rightarrow D_k$, where $B_{i_{sortedK}}$ represents the set of points belonging to bin B_i sorted by their distance to their k-th neighbour and D_k represents the set of the actual distances. We note that because a bin consists of points belonging to the same density interval, the k-distance plot will usually incorporate a single sharp change point.

If the clusters are not well separated either because of noise or chains, we need to over partition the bins in order to better separate and emphasise the high density regions. This is why we introduce the expansion factor ef, a custom parameter utilized to weight the threshold. We cover ef's value selection in detail in Section 5, Subsection 5.1. Therefore, we formally define the threshold using the inflection point of the k-distance plot, $threshold_{kDistance}$ and the expansion factor ef: $threshold = ef \cdot threshold_{kDistance}$.

4.4 Merging the Adjacent Connected Components

The resulting adjacent connected components must ultimately be merged in order to build the final clusters. For this purpose we use an agglomerative approach: we compute the pairwise distances between all partitions, we merge the closest partitions into a cluster, we recompute the distances that have changed due to the merge and continue with the following closest partitions. In the majority of cases, where no over-partitioning is needed, the process is considerably faster than plain hierarchical agglomerative clustering, because it starts with already clustered large groups of points instead of just points. The linkage method is configurable trough the input parameters. We offer two options: smallest pairwise distance (the default option) and centroid linkage. For elongated clusters or well separated gaussian clusters, we use the smallest pairwise distance linkage. This is the most suitable linkage criterion because regardless the input dataset's shape, the continuous regions previously obtained, which we want to join, are elongated. For noisy datasets or gaussian clusters connected by chains, we use centroid linkage - so the noise or the chains do not affect the quality of the result.

5 Algorithm

5.1 Choosing the Hyperparameters

The number of clusters, the bn number of bins, the expansion factor ef and the method to compute the inter cluster linkage must be provided as parameters.

Determining the optimal number of clusters is a comprehensively documented problem; the elbow and silhouette methods represent popular solutions. The number of bins bn is generally equal to the number of density levels of the optimal cluster tree (as defined in [4]). However, if the input dataset contains chains (see Aggregation [11]), we must choose a greater value for bn in order to force the separation of the chained bins. Therefore, $bn \in [3, 9]$. The expansion factor ef is generally equal to one. For elongated and well separated spherical clusters there is no need to apply any weight on the closest mean (defined in Section 4, Subsection 4.3). However, if the input dataset is noisy, ef should be lower than 1 in order to "prune" the noisy points. Therefore, $ef \in [0.1, 2]$. We support two linkage methods: single linkage (smallest pairwise distances), which is the default option, and centroid linkage. For elongated and well separated spherical clusters single linkage performs best. This is because the intermediary connected components which we join in the aggregation step are usually elongated, regardless the input dataset's structure. For noisy or chained datasets, centroid linkage obtains best results.

5.2 Implementation

We start by estimating the probability density function. Then, we group the input objects into partitions representing bn density bins. Afterwards, we iterate trough each partition and split its connected components. Lastly, we merge the adjacent connect components in order to build the final clusters. The density bins splitting procedure consists of the following steps: i) we start with an arbitrary object and retrieve its $k = d + 1$ nearest neighbours computed as all the objects whose distance to the specified object is smaller than the threshold defined in Subsection 4.3; ii) we start a new cluster from the current object, add the object's neighbors to the cluster and try to expand it; iii) when we cannot further expand the current cluster we return to Step 1 and repeat the process.

The full algorithm is presented in pseudocode 1, whereas the distance based split is displayed in pseudocode 2. The implementation is freely available online[1], alongside run instructions and graphical results for all synthetic bidimensional datasets.

6 Experimental Results

6.1 Clustering Evaluation Methods

In order to evaluate DenLAC's accuracy we use two different classes of evaluation methods: i) evaluation methods based on the mutual information (specifically the Entropy [29] and ii) evaluation methods based on counting pairs of objects distributed either to the same cluster or to different clusters by a given clustering algorithm versus the ground truth [29] (specifically the Rand Index [23] and the Adjusted Rand Index [27]).

[1] https://github.com/IuliaRadulescu/DENLAC

Algorithm 1 DenLAC algorithm

```
1: procedure DenLac($inputDataset$, $numberOfClusters$, $bn$, $ef$, $linkageMethod$)
2:     $pdf \leftarrow$ the probability density estimate of $inputDataset$;
3:     $outliers \leftarrow$ the outliers, identified employing the IQR method on the $pdf$;
4:     $inputDataset = inputDataset \setminus outliers$;
5:     Recompute $pdf$ for the updated $inputDataset$;
6:     Split $pdf$ into $bn$ intervals: $B_1$, ..., $B_{bn}$;
7:     for $B_k$ in $B_1$, ..., $B_{bn}$ do
8:         for $d_i$ in $inputDataset$ do
9:             if $d_i >= B_k^{min}$ and $d_i < B_k^{max}$ then
10:                $B_k = B_k \cup d_i$;
11:            end if
12:        end for
13:    end for
14:    $connectedComponents \leftarrow \emptyset$;
15:    for $B_k$ in $B_1$, ..., $B_{bn}$ do
16:        $connectedComponentsBk$ = separateConnectedComponents($B_k$, $ef$);
17:        $connectedComponents = connectedComponents \cup connectedComponentsBk$;
18:    end for
19:    clusters $\leftarrow$ the connectedComponents aggregation result using $linkageMethod$ and $numberOfClusters$;
20:    return $clusters$;
21: end procedure
```

The *Entropy* [29] measures the uncertainty regarding the cluster membership of a randomly chosen object, assuming that all elements of the input dataset have the same probability of being picked. Therefore, a small entropy indicates a good clustering.

The *Rand Index* [23] represents the level of agreement between the grouping produced by the evaluated clustering algorithm and the ground truth. It is computed as a fraction where the sum between the number of pairs assigned to the same cluster and the number of pairs assigned to different clusters is the numerator and the total number of pairs is the denominator. The value of *Rand Index* is a number between 0 (the groupings are completely different) and 1 (the groupings are the same).

However, the value of the *Rand Index* converges to 1 as the number of clusters increases. For this reason the *Adjusted Rand Index* quality measure was introduced as the normalized difference of the *Rand Index* and its expected value under the null hypothesis [27].

6.2 Evaluation Results

We evaluate DenLAC's performances on eight synthetic bi-dimensional datasets and two real datasets: Iris [6] and Tetragonula bee species [8] (retrieved from [1]). Each of the eight bi-dimensional datasets holds its particularities: Aggre-

Algorithm 2 Splitting the density bins into connected components

```
    procedure SEPARATECONNECTEDCOMPONENTS($objectsInDensityBin$, $ef$)
 2:     $splitPartitions \leftarrow \emptyset$;
        $clustId \leftarrow 1$;
 4:     for $obj$ in $objectsInDensityBin$ do
            Mark $obj$ as already parsed;
 6:         $splitPartitions[clusterId] = splitPartitions[clusterId] \cup obj$;
            $threshold_{kDistance}$ ← the inflection point of the k-distance plot of $objectsInDensityBin$
 8:         $kNeighbours \leftarrow$ $obj$'s nearest neighbors within $threshold_{kDistance} \cdot ef$
            for $kNeighbour$ in $kNeigbours$ do
10:             if $kNeighbour$ not already parsed then
                    Mark $kNeigbour$ as already parsed;
12:                 $splitPartitions[clustId] = splitPartitions[clustId] \cup kNeighbour$;
                    Continue expansion by parsing each of the neighbors neighbor;
14:             end if
            end for
16:         Increment $clustId$;    ▷ if we can't expand the current cluster anymore, create a new cluster and start from a new not already parsed point;
        end for
18:     return $splitPartitions$;
    end procedure
```

gation [11] is comprised of spherical clusters with different dimensions, Compound [30] contains clusters with varying densities, Pathbased [3], Spiral[3], Jain [15] and Flame [9] consist of arbitrary shaped clusters, Flame also incorporating outliers, R15 [28] and D31 [28] encompass spherical clusters of approximately the same size.

We compare DenLAC's results with the ones obtained by eight popular algorithms: K-Means [18], Clarans [22], Hierarchical agglomerative clustering [19], Birch [31], CURE [12], Gaussian Mixture [2], DBSCAN [7] and Spectral Clustering [21]. The output for the Pathbased synthetic datatset is graphically displayed in Figure 3.

The evaluation results shown in Table 1a, 1b, 1c indicate that DenLAC obtains the highest overall accuracy: the ARI values are greater than 0.9 for all datasets except Tetragonula and the uncertainty regarding the cluster of a randomly extracted an object given its ground-truth class is smaller than 0.01 for 7 datasets and smaller than 0.3 for all of the datasets.

As expected, the classic algorithms are correct only on particular datasets.

K-Means obtains optimal results solely for the datasets which contain spherical clusters (Iris, Aggregation, D31 and R15 datasets). However, it computes inaccurate results for the Jain, Flame, Pathbased and Spiral datasets.

Because it relies on minimizing the sum of pairwise distances within the clusters formed around medoids, Clarans also favors non-convex clusters, performing best on the R15, D31 and Iris datasets.

Table 1: Evaluation measures values

(a) Entropy evaluation results

Dataset	**DenLAC**	**Birch**	**Clarans**	**Cure**	**Gaussian Mixture**	**Hierarchical**	**K-Means**	**DBSCAN**	**Spectral Custering**
Aggregation	0.0111	0.099	0.183	0.0	0.081	0.037	0.062	0.174	0.011
Compound	0.114	0.185	0.315	0.186	0.187	0.279	0.21	0.282	0.18
D31	0.011	0.062	0.184	0.166	0.056	0.05	0.033	0.136	0.035
Flame	0.166	0.682	0.842	0.932	0.606	0.932	0.557	0.94	0.932
Jain	0.0	0.373	0.616	0.726	0.651	0.373	0.49	0.0	0.0
Pathbased	0.098	0.475	0.579	0.459	0.522	0.534	0.487	0.292	0.243
R15	0.008	0.019	0.139	0.046	0.006	0.014	0.006	0.084	0.006
Spiral	0.0	0.98	0.985	0.918	0.999	0.0	0.999	0.0	0.0
Iris	0.0	0.174	0.460	0.115	0.0	0.333	0.0	0.0	0.246
Tetragonula	0.189	0.71	0.71	0.73	0.71	0.71	0.71	0.1	0.8

(b) Rand Index evaluation results

Dataset	**DenLAC**	**Birch**	**Clarans**	**Cure**	**Gaussian Mixture**	**Hierarchical**	**K-Means**	**DBSCAN**	**Spectral Custering**
Aggregation	0.997	0.917	0.868	1.0	0.944	0.938	0.927	0.927	0.997
Compound	0.966	0.924	0.816	0.922	0.858	0.89	0.843	0.89	0.834
D31	0.997	0.993	0.97	0.971	0.994	0.995	0.997	0.974	0.997
Flame	0.95	0.694	0.603	0.541	0.65	0.541	0.727	0.539	0.541
Jain	1.0	0.759	0.726	0.501	0.512	0.759	0.662	0.975	1.0
Pathbased	0.968	0.757	0.709	0.762	0.729	0.723	0.747	0.819	0.857
R15	0.99	0.997	0.96	0.989	0.999	0.998	0.999	0.971	0.999
Spiral	1.0	0.558	0.536	0.453	0.554	1.0	0.554	1.0	1.0
Iris	1.0	0.912	0.773	0.949	1.0	0.835	1.0	0.92	0.695
Tetragonula	0.888	0.4	0.4	0.36	0.4	0.4	0.4	0.8	0.08

(c) Adjusted Rand Index evaluation results

Dataset	**DenLAC**	**Birch**	**Clarans**	**Cure**	**Gaussian Mixture**	**Hierarchical**	**K-Means**	**DBSCAN**	**Spectral Custering**
Aggregation	0.991	0.733	0.586	1.0	0.827	0.8	0.762	0.808	0.99
Compound	0.911	0.809	0.481	0.805	0.596	0.742	0.538	0.743	0.531
D31	0.991	0.884	0.6	0.664	0.897	0.92	0.953	0.682	0.95
Flame	0.9	0.385	0.183	0.013	0.299	0.013	0.453	0.0	0.013
Jain	1.0	0.515	0.356	-0.084	-0.004	0.515	0.324	0.948	1.0
Pathbased	0.929	0.477	0.395	0.487	0.432	0.423	0.46	0.605	0.683
R15	0.989	0.972	0.713	0.914	0.993	0.982	0.993	0.802	0.993
Spiral	1.0	0.017	0.003	0.037	-0.005	1.0	-0.006	1.0	1.0
Iris	1.0	0.803	0.515	0.885	1.0	0.631	1.0	0.81	0.695
Tetragonula	0.658	-0.01	-0.01	0.36	0.004	-0.01	-0.01	0.2	0.001

Hierarchical agglomerative clustering with average linkage splits the elongated clusters and merges sections within neighbouring clusters for the Jain and Pathbased datasets. For the Flame dataset, the results are affected by the presence of the two outliers.

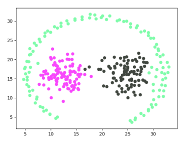
Birch obtains substandard results for non-spherical, elongated clusters such as Spiral, Jain, Flame and Pathbased.

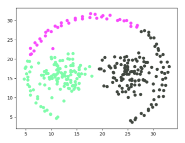
Cure performs well only on the Aggregation, R15 and Iris datasets, since their structure is adequately described by the computed set of representatives.

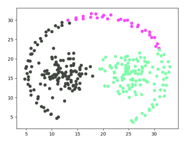
The Gaussian Mixture algorithm is imprecise on datasets containing non-normally distributed clusters such as Jain, Spiral and Pathbased.

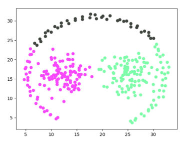
DBSCAN is sensitive to the clusters density discrepancies. For instance, it splits the Compound dataset in three different clusters ignoring the fine details and it overpartitions the Jain dataset into three clusters.

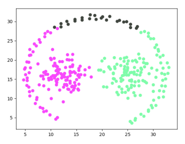
Although Spectral clustering is applicable on arbitrary shaped datasets, its ARI values for the Compound and the Pathbased datasets, 0.531 and 0.683 respectively, are significantly lower than the ones achieved by DenLAC (0.967 and 0.987 respectively).

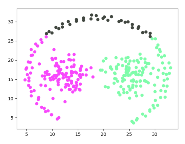

(a) DenLAC

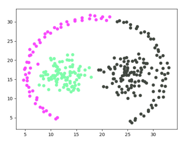

(b) Birch

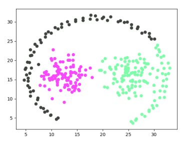

(c) Clarans


(d) Cure


(e) Gaussian Mixture


(f) K-Means


(g) DBSCAN


(h) Spectral

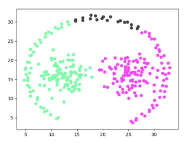

(i) Hierarchical

Fig. 3: Graphical representation of the results obtained by DenLAC versus other clustering algorithms for the Pathbased synthetic dataset

7 Conclusions

In this paper we present DenLAC, a flexible clustering algorithm which combines the benefits of density based and hierarchical clustering to produce accurate results regardless the input's shape and distribution. As we show in Section 6, our method surpasses the established clustering algorithms used for comparison in terms of accuracy, on both synthetic and real datasets. DenLac operates best on elongated, well separated clusters, obtaining perfect scores for the Jain and Spiral synthetic datasets. Aditionally, its accurracy is also maximum for the Iris real dataset and is significantly higher for the Tetragonula dataset as compared with the other methods. Our method requires choosing a number of hyperparameters: the number of clusters, the number of density bins, the expansion factor and the agglomerative clustering linkage method. We propose several heuristics for choosing these values: i) the elbow and silhouette methods are confirmed for determining a dataset's optimal number of clusters, ii) the number of density bins is normally equal to the optimal number of density levels of the cluster tree but should be larger for noisy datasets, iii) the expansion factor is normally equal to 1 but should be smaller for clusters with chains and iv) the preferred linkage method is single linkage; however, for elongated or noisy datasets centroid linkage must be used. We detail these approaches in Section 5, Subsection 5.1. We further intend to incorporate a procedure which computes the number of clusters automatically and improve the nearest neighbour expansion algorithm such that the computation of the expanding factor ef is automatic.

References

1. BWorld Robot Control Software (2020), [Online; accessed 17-Feb-2020]
2. Basford, K.E., McLachlan, G.J.: Likelihood estimation with normal mixture models. Applied Statistics **34**(3), 282 (1985). https://doi.org/10.2307/2347474
3. Chang, H., Yeung, D.Y.: Robust path-based spectral clustering with application to image segmentation. In: International Conference on Computer Vision. IEEE (2005). https://doi.org/10.1109/ICCV.2005.210
4. Chaudhuri, K., Dasgupta, S.: Rates of convergence for the cluster tree. In: Advances in Neural Information Processing Systems. pp. 343–351 (2010)
5. Chen, Y.C.: A tutorial on kernel density estimation and recent advances. Biostatistics & Epidemiology **1**(1), 161–187 (jan 2017). https://doi.org/10.1080/24709360.2017.1396742
6. Dua, D., Graff, C.: UCI machine learning repository (2017)
7. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In: International Conference on Knowledge Discovery and Data Mining. pp. 226–231 (1996)
8. Franck, P., Cameron, E., Good, G., Rasplus, J.Y., Oldroyd, B.: Nest architecture and genetic differentiation in a species complex of australian stingless bees. Molecular Ecology **13**(8), 2317–2331 (2004)
9. Fu, L., Medico, E.: Flame, a novel fuzzy clustering method for the analysis of dna microarray data. BMC Bioinformatics **8**(1), 3 (2007). https://doi.org/10.1186/1471-2105-8-3
10. Gaonkar, M.N., Sawant, K.: Autoepsdbscan: Dbscan with eps automatic for large dataset. International Journal on Advanced Computer Theory and Engineering **2**(2), 11–16 (2013)
11. Gionis, A., Mannila, H., Tsaparas, P.: Clustering aggregation. ACM Transactions on Knowledge Discovery from Data **1**(1), 4–es (mar 2007). https://doi.org/10.1145/1217299.1217303
12. Guha, S., Rastogi, R., Shim, K.: Cure: An efficient clustering algorithm for large databases. ACM Sigmod Record **27**(2), 73–84 (1998). https://doi.org/10.1145/276305.276312
13. Hartigan, J.A.: Consistency of single linkage for high-density clusters. Journal of the American Statistical Association **76**(374), 388–394 (1981)
14. Hinneburg, A., Keim, D.A., et al.: An efficient approach to clustering in large multimedia databases with noise. In: KDD. vol. 98, pp. 58–65 (1998)
15. Jain, A.K., Law, M.H.C.: Data clustering: A user's dilemma. In: Pattern Recognition and Machine Intelligence. pp. 1–10. Springer Berlin Heidelberg (2005). https://doi.org/10.1007/11590316_1

16. Karypis, G., Han, E.H., Kumar, V.: Chameleon: hierarchical clustering using dynamic modeling. Computer **32**(8), 68–75 (1999). https://doi.org/10.1109/2.781637
17. Liu, P., Zhou, D., Wu, N.: Vdbscan: varied density based spatial clustering of applications with noise. In: 2007 International conference on service systems and service management. pp. 1–4. IEEE (2007)
18. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Symposium on Mathematical Statistics and Probability. pp. 281–297. University of California Press, Berkeley, Calif. (1967)
19. Murtagh, F.: A survey of recent advances in hierarchical clustering algorithms. The Computer Journal **26**(4), 354–359 (1983). https://doi.org/10.1093/comjnl/26.4.354
20. Nadler, B., Galun, M.: Fundamental limitations of spectral clustering. In: Proceedings of the 19th International Conference on Neural Information Processing Systems. pp. 1017–1024. NIPS'06, MIT Press, Cambridge, MA, USA (2006)
21. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems 14, pp. 849–856. MIT Press (2002)
22. Ng, R.T., Han, J.: Clarans: A method for clustering objects for spatial data mining. IEEE Transactions on Knowledge and Data Engineering **14**(5), 1003–1016 (2002). https://doi.org/10.1109/TKDE.2002.1033770
23. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association **66**(336), 846–850 (dec 1971). https://doi.org/10.2307/2284239
24. Schubert, E., Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Dbscan revisited, revisited: why and how you should (still) use dbscan. ACM Transactions on Database Systems (TODS) **42**(3), 1–21 (2017)
25. Scott, D.W.: Multivariate density estimation: theory, practice, and visualization. John Wiley & Sons (2015)
26. Seo, S.: A review and comparison of methods for detecting outliers in univariate data sets. Ph.D. thesis, University of Pittsburgh (2006)
27. Truică, C.O., Rădulescu, F., Boicea, A.: Comparing different term weighting schemas for topic modeling. In: 2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC). IEEE (sep 2016). https://doi.org/10.1109/SYNASC.2016.055
28. Veenman, C., Reinders, M., Backer, E.: A maximum variance cluster algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(9), 1273–1280 (sep 2002). https://doi.org/10.1109/TPAMI.2002.1033218
29. Wagner, S., Wagner, D.: Comparing clusterings: an overview. Tech. rep., ETH Zurich (2007)
30. Zahn, C.: Graph-theoretical methods for detecting and describing gestalt clusters. IEEE Transactions on Computers **C-20**(1), 68–86 (jan 1971). https://doi.org/10.1109/t-c.1971.223083
31. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: An efficient data clustering method for very large databases. ACM SIGMOD Record **25**(2), 103–114 (1996). https://doi.org/10.1145/235968.233324