Fuzzy Density Based Clustering with Generalized Centroids

Christian Braune *Member, IEEE*
Institute for Intelligent Cooperating Systems
Department for Computer Science,
Otto von Guericke University Magdeburg
Universitätsplatz 2, 39106 Magdeburg, Germany
Email: christian.braune@ovgu.de

Rudolf Kruse *Fellow, IEEE*
Institute for Intelligent Cooperating Systems
Department for Computer Science,
Otto von Guericke University Magdeburg
Universitätsplatz 2, 39106 Magdeburg, Germany
Email: rudolf.kruse@ovgu.de

***Abstract*—Fuzzy density-based clustering has been a challenge. Research has been focused on fuzzyfying the DBSCAN algorithm. Different methods have been proposed that use a fuzzy definition of core points within the DBSCAN algorithm. Our approach adapts the membership degree calculation known from fuzzy c-means by replacing the need for a distinguished centroid point by a more general cluster skeleton. These skeletons represent the clusters' shapes more accurately than a single point. We show how membership degrees can be calculated and that the resulting partitioning matrices allow the selection of more favorable parameters in the clustering process.**

I. Introduction

A common task in data analysis is clustering [1]. It is the process of grouping data points into a number of previously unknown and possibly overlapping classes. Data points are usually represented as vectors $\vec{x}_i \in \mathbb{R}^{\mathrm{d}}$ in a d-dimensional vector space. A data set $\mathcal{X} \subseteq \mathbb{R}^{\mathrm{d}}$ is a collection of data points. A clustering of the data as the result of a clustering algorithm is a partitioning of the data set $\mathcal{X}$ into a finite number k of subsets $\mathcal{C}_i$ such that $\bigcup_{i=1}^{k} \mathcal{C}_i = \mathcal{X}$. This is also called a crisp clustering. For non-overlapping clusters we also require that $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset, \forall i \neq j$.

Clustering algorithms can be broadly distinguished into three categories [2]: (a) hierarchical clustering, where each point starts as a singleton cluster and clusters are merged pairwise until only one cluster remains; (b) centroid or model-based algorithms, which cover well-known algorithms such as k-means [3] or fuzzy c-means [4]; (c) density-based clustering such as DBSCAN [5]. The fundamental differences between these classes are, how clusters are found and which characteristics the resulting clusters may have. In the case of hierarchical clustering a complete hierarchy of structures is built depending on the chosen linkage criterion. Centroid-based methods usually assign points to the cluster center that they are closest to. Such algorithm tend to prefer spherical or globular clusters (w.r.t. the chosen metric), while density-based methods assign points to clusters by finding areas of similarly high density.

All these methods mentioned so far have in common that a point is always only assigned to exactly one cluster. DBSCAN might be seen as the one exemption here, since it also might assign points to no cluster at all, if they are considered noise. Yet, from an implementation's point of view these points are all assigned to a *noise cluster*. So the argument might still hold. However, if clusters are actually overlapping or the spatial structure of the data does not justify an unambiguous cluster assignment, we can use fuzzy clustering. Here the points are not assigned to a single cluster but rather belong to every cluster found at the same time but to a different degree. This membership degree is in inverse proportion the the distance of the clusters' centers, i.e. if a point has a small distance to a cluster center then it has a high degree of membership to that cluster and vice versa.

978-1-4799-7560-0/15/$31

For centroid-based clustering algorithms often fuzzy version exist. The best-known algorithm of this family appears to be the fuzzy c-means algorithm [4]. One advantage of fuzzy clustering algorithms is, that they a more robust than their crisp counterparts. They do not end in local minima of their respective objective function as often because of the smoother structure of this function.

Like its original, crisp version, the fuzzy c-means algorithm tends to find globular cluster shapes for which the cluster center acts as a natural representative. Because of that the membership degree can be calculated easily. Density-based clustering algorithms lack such a distinguished point. Fuzzifying a density-based clustering algorithm therefore comes much less naturally than in the centroid-based case. We try to solve this problem with our approach.

We propose that for fuzzy clustering of such non-convex clusters – as they may result from density-based clustering algorithms – another kind of centroid should be used. Single points cannot capture the shape of the clusters anymore once they contain too many or too strong concavities. Instead we want to generate a skeleton that represents the cluster structure and that can be used to assign fuzzy membership degrees.

The remaining paper will be organized as follows: In Section II we will briefly explain the DBSCAN algorithm and in Section III how the k-means algorithm has been fuzzified. Section IV will give an overview over existing techniques for fuzzifying density-based clustering algorithms, mainly the existing DBSCAN algorithm. In Section V we will present our approach. We evaluate our approach in Section VI and conclude in Section VII.

II. DBSCAN

DBSCAN is a well-known clustering algorithm [5] whose properties have been extensively studies over the past years. It is able to find nearly arbitrarily shaped clusters – as long as the connection between two different clusters is weak enough. Its main parameters are ε and $minPts$. The first is used to describe the size of a hypersphere centered around each data point while the second parameter is used as a threshold to identify cluster cores. If the number of data points inside of its neighborhood exceeeds this threshhold it becomes the core of a cluster.

Points that both fulfill the core condition and are mutually contained in each others neighborhoods are said to be *directly density reachable*. Points are *density reachable* if they fulfill the core condition and can be linked to each other by a series of points that are each pairwise directly density reachable. At last, two points are density connected if there exists a series of points where the first and last point of the series are density reachable and the two points in question are contained in the first's and second's neighborhood respectively. A cluster is then identified as a subseet of the dataset where each pair of points is density connected.

DBSCAN is very sensitive to the choice of parameters. Improper values might lead to either (almost) all points points becoming core points or none at all [6]. This problem can, however, be overcome by trying to estimate the parameters [7] or a completely unsupervised alternating optimization approach [8].

III. Fuzzy Clustering

Sometimes points cannot be unambiguously assigned to one cluster alone [9]. In the case of the k-means algorithm if a point is just halfway between two cluster centers, a race condition occurs and the point usually is assigned to the cluster center that is handled first by the algorithms implementation. However, even in less strict cases where points lie roughly in the middle between two clusters one could be hesitant to assign a point to a single cluster. In such cases fuzzy clustering becomes a proper tool to handle these situations.

In the fuzzy c-means algorithm the membership of a point to a single cluster is 1.0 if and only if its distance to the cluster center is 0. In all other cases the membership is inversely proportional to the distance of the point to the cluster center. It can be easily calculated by the formula

$$\mu(\vec{x_j}, \vec{c_i}) = \frac{d_{ij}^{\frac{2}{1-\omega}}}{\sum_{k=1}^{c} d_{kj}^{\frac{2}{1-\omega}}} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}}\right)^{\frac{2}{\omega-1}}} \tag{1}$$

where d_{ij} is the distance between the point $\vec{x_j}$ and the cluster center $\vec{c_i}$. ω is called *fuzzifier* and influences how crisp the clustering around the centroids is. Higher values for ω produce crisp clustering only in the closest vicinity of the clusters' centroids, see Figure 1.

Fuzzy clustering can also be used when there is no ambiguity since it tends to produce more robust results than crisp clustering [11].

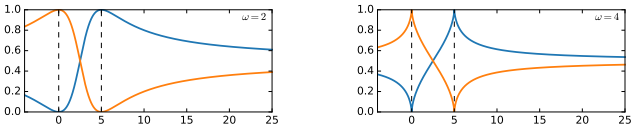

Fig. 1: Membership degrees for two cluster centers (one-dimensional case). Left with lower, right with higher fuzzifier [10].

IV. Related Work

To obtain fuzzy results from a clustering algorithm's result one can use the crisp clustering algorithm and simply apply a fuzzification formula to obtain membership degrees. This works well in the case of centroid-based clustering algorithms – the fuzzy c-means algorithm [4] as extension of the k-means algorithm [3] is a prime example. In density-based clustering the only cluster representatives available are the core points. This is where most of the previous work aims at when fuzzifying the clustering process.

First attempts to creat fuzzy density-based algorithm have been based on the DBSCAN algorithm in [12]–[14]. Here the fuzziness is introduced into the core condition. Points that are within the ε-neighborhood of a point contribute differently to the cardinality of the neighborhood. Theoretically DBSCAN does not differentiate between points that are close to the center of the neighborhood and points that are close to its border. Central points should however contribute more to the centers qualification as cluster core. Thus the cardinality of the neighborhood set is fuzzified. However, the cluster assignment remains crisp in the end. Clusters

Another approach is described in [15]. In contrast to the fuzzy neighborhood approach mentioned above, cluster assignments can become fuzzy here. Core points are identified in the same way as in the classical DBSCAN algorithm with one distinction: There are two thresholds for $minPts$. If the lower threshold is exceeded the point starts to become a cluster core. The membership to being a cluster core rises linearly until the upper threshold is exceeded. Here the point fully becomes a core point. For non-core points the membership degree to the respective cluster is simply the maximum of all the membership degrees of those core points in whose neighborhoods the point lies. This may however lead to membership assignments that do not sum up to 1.0 if the point is part of several overlapping neighborhoods from different clusters but not a core point itself. Also, points that were (possibly incorrectly) identified as noise do not receive any membership to any cluster at all. Lastly, the clustering is not truly *fuzzy*: due to the linear interpolation of membership degrees (a) only a discrete amount of fuzzy membership values can be obtained, and (b) the distance between a point and its associated core points does not matter. The obtained membership degrees resemble more or less the α-cuts of a fuzzy membership degree distribution of a more general cluster representation. The advantage of this approach however is, that clusters of varying density can be found in a more robust way.

A completely different approach is given in [16] in which DBSCAN is used to find an initialization for the fuzzy c-means algorithm. By choosing the parameters ε and $minPts$

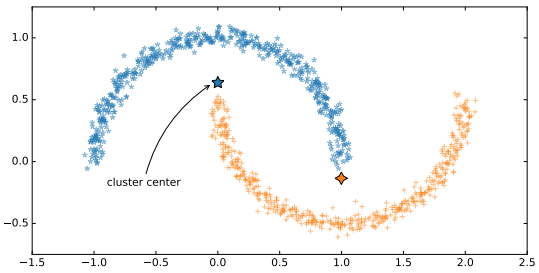

Fig. 2: Two non-convex clusters. The centroids are no proper representatives for each cluster.

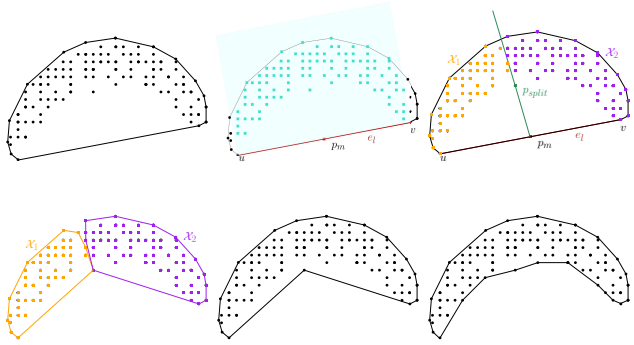

Fig. 3: Iteraritve refinement of a cluster's hull. Starting from the convex hull (top left) edges on the hull are replaced one by one until the resulting hull fits the data points better [18].

sufficiently small or high respectively, a lot of small and very dense clusters are retrieved. These act as seeds for the fuzzy c-means algorithm. Instead of euclidean distance the authors use the Mahalanobis distance [17]. However, instead of adjusting the centroid calculation accordingly, the authors use the arithmetic mean (weighted by the membership degrees of the associated points) thus using an inappropriate estimator for the centroids given the distance metric used.

V. Generalized Centroids

Essentially the idea behind assigning fuzzy memberships is sound and good as it is. It just cannot be easily applied to density-based clustering. If we recall Equation 1 then we see that the membership degree of a point to a cluster is always calculated by its distance to the cluster's centroid. In the previous section we reviewed some ways that try to overcome the need for this calculation by either introducing fuzziness into the neighborhood definition or the core points themselves.

Our approach differs from these in a sense that we consider a closer approach to the fuzzy c-means approach. Instead of trying to find a way to assign fuzzy membership degrees to a data point by changing the neighborhood or core point definitions of DBSCAN we aim the centroid definition itself. From Figure 2 can be seen, that the centroids of density-based clusters do not always lie in the set of points forming the clusters anymore. Instead they are closer to the points of the other cluster.

Additionally our new algorithm allows to calculate interpretable fuzzy membership degrees to arbitrary points inside the data space. This may be useful if not all points are known during the modeling phase or if we are interested in regions where there is maximal ambiguity between the different clusters (e.g. for an active learning approach, see Figure 6).

A better way to describe the clusters' shape would be cluster skeletons. There are several ways to calculate a cluster's skeleton. By first calculating the concave hull of each cluster (see Figure 3 for a depiction of the generation process, following [18]) we can obtain the medial axis [19] which we use for simplicity here. Another way of obtaining the concave hull would be by using α-shapes [20] or χ-shapes [21]. Our proposed method has the slight advantage that the resulting hulls are more stable over a broader parameter range than the other methods. As such we do not need to put too much effort into the parametrization of the skeletonization process.

For a polygon (such as the concave hull) in the plane the medial axis is defined as the set of all points which do not have a unique nearest neighbor on the polygon. In the case of convex polygons these points would all lie inside of the polygon. Since we are only interested in the part of the medial axis which spans the data set or cluster, we filter those parts of the medial axis which are not contained inside the polygon. The resulting skeleton undergoes an additional filtering step. For each edge of the skeleton at least two of the three following conditions must hold for it to be contained in the final, generalized centroid:

a) **The edge must not end on the border polygon.**
Necessarily this happens when calculating the medial axis, but these parts of the axis can be seen as an overfitting of the cluster's skeleton to the data. This condition must be always fulfilled.

b) **The edge must be contained in the longest path along the medial axis.**
If we consider the data set shown in Figure 4 then the medial axis will contain an elongated series of edges that span a the complete cluster. Additionally several shorter edges will be part of the medial axis that run from the main axis towards indentations of the border polygon.

c) **The edge must lie on any path that exists between each pair of degree 3 or more.**
These edges that run toward the indentations of the border polygon branch off of the main axis (thus creating nodes of degree 3 or possibly higher). In the case of the data set shown in Figure 5 these edges bear additional information about the clusters' shapes, i.e. where clusters are broader or where clusters deviate from a single axis structure.

By pruning the branches we can avoid overfitting of the skeleton to the data while still getting a proper representation of the cluster structure.

With these skeletons we are now able to calculate fuzzy membership degrees in a very similar fashion to the fuzzy c-means membership degree calculation. The membership calculation (see Equation 1) requires the only the distance from a point to the cluster's centroid. By replacing the previous centroid by the generalized centroid (or the cluster's skeleton) we need to calculate the distance from the point to the skeleton instead.

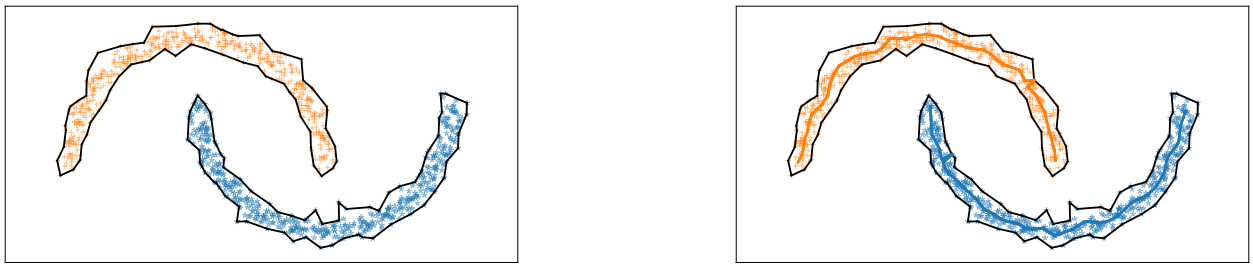

Fig. 4: Cluster hulls ($p = 0.2$, according to [18], left) and the resulting generalized centroid for each cluster (right).

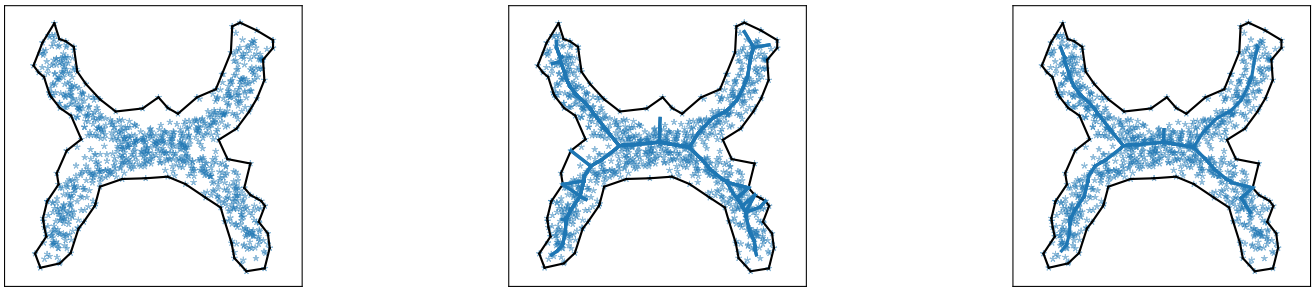

Fig. 5: Cluster hulls ($p = 0.2$, according to [18], left) with the original, unpruned skeleton (middle, all edges that belong to the medial axis except those that run onto the border) and the resulting, pruned generalized centroid for each cluster (right).

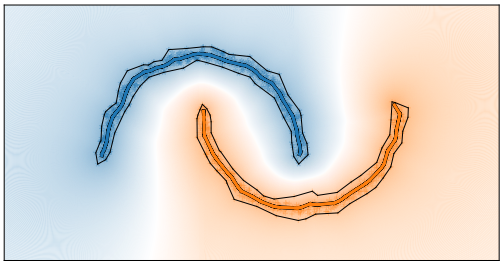

Fig. 6: Two non-convex clusters. The centroids are no proper representatives for each cluster.

Let $\mathcal{H}^*_{\mathcal{C}}$ be the concave hull of a cluster $\mathcal{C}$. This can be either the shape descriptor obtained by [18], the α-shape [20] or χ-shape [21]. $\mathcal{H}^*_{\mathcal{C}}$ can be represented by a set of edges $e_{\mathcal{H}} = (x_i, x_j)$ between the points x_i and x_j on the border polygon in counter-clockwise order (CCW). The points on the border polygon are necessarily points of the data set $\mathcal{X}$.

Let $\mathcal{MA}_{\mathcal{C}} = \{p \in \mathbb{R}^{\mathrm{d}} | \mathrm{p}$ has no unique nearest neighbor on $\mathcal{H}^*_{\mathcal{C}}\}$ be the cluster's medial axis. This can be calculated as the Voronoi diagram of the border polygon's edges (not the points, as one would do in k-means clustering). Since our polygon has only straight line segments as border the medial axis will also mostly consist of straight line segments $e_{\mathcal{MA}} = (p_i, p_j)$. Under certain circumstances the edges may be arched. For simplicity in the following calculations we assume direct connections between these edges' points. Note that the p_i forming the edges now do not need to be actual points of the data set. Let $E(\mathcal{MA}_{\mathcal{C}})$ denote the set of edges (straight line segments) that represent the medial axis.

Let $\mathcal{S}_{\mathcal{C}} = \{e \in E(\mathcal{MA}_{\mathcal{C}})|$ the conditions a, b and c hold for $e\}$ be the cluster's skeleton. With this skeleton given we are now able to compute the fuzzy membership of a data point to any given cluster (see Figure 6):

$$\mu(\vec{x_j}, \mathcal{S}_{\mathcal{C}_\rangle}) = \frac{{d_{ij}^{\mathcal{S}}}^{\frac{2}{1-\omega}}}{\sum_{k=1}^{c} {d_{kj}^{\mathcal{S}}}^{\frac{2}{1-\omega}}} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}^{\mathcal{S}}}{d_{kj}^{\mathcal{S}}}\right)^{\frac{2}{\omega-1}}} \tag{2}$$

The only thing that has changed in the calculation is that we do not compute the membership to the centroid of the cluster but to its skeleton. This – of course – changes how the $d_{ij}^{\mathcal{S}}$ are computed. Instead of simply taking the euclidean distance between the points x_j and c_i we now have to deal with several line segments. The distance between a point and a straight line

$$l(\lambda) = p_i + \lambda(p_j - p_i) \tag{3}$$

can be calculated by projecting the vector between the point and the base point of the line onto the directional vector of the line. By this we find the closest point on the line and can

calculate the length between this point and the point we are examining.

For line *segments* however the closest point on the line carrying the line segment may lie outside of the segment itself. In this case we need to know whether the point lies before the line segment's start or behind its end point. In that case the smallest distance between a point and the line segment would be the distance between the point and the starting or ending point respectively.

$$\vec{v} = \vec{b} - \vec{a} \tag{4}$$
$$\vec{w} = x - \vec{a} \tag{5}$$
$$c_1 = \vec{v} \cdot \vec{w} \tag{6}$$
$$c_2 = \vec{v} \cdot \vec{v} \tag{7}$$

$$d(\vec{x}_j, e) = \begin{cases} d(\vec{x}_j, \vec{a}) & \text{if } c_1 < 0 \\ d(\vec{x}_j, \vec{b}) & \text{if } c_2 < c_1 \\ d(\vec{x}_j, l(c_1/c_2)) & \end{cases}, \tag{8}$$

with $e = (\vec{a}, \vec{b})$.

$\vec{v}$ is the directional vector of the carrier line, $\vec{w}$ the vector between the base point and the point for which we want to calculate the distance. c_1 gives us the value for λ such that the vector from $l(\lambda)$ towards x is minimal. If c_1 is smaller than 0 then the point lies *before* the starting point of the line segment and we need to calculate the distance between x and p_i. If it is larger than c_2 then x lies *behind* the end point of the line segment and we need to calculate the distance between x and p_j. In all other cases we calculate the projection of $\vec{w}$ onto $\vec{v}$ to get the point closest to x and calculate their distance.

If now we want to calculate the distance between a point x and a cluster's skeleton $\mathcal{S}_{\mathcal{C}_i}$, we calculate the distance between x and every line segment building $\mathcal{S}_{\mathcal{C}_i}$. The minimum over all these distances is the distance between x and $\mathcal{S}_{\mathcal{C}_i}$:

$$d_{ij}^{\mathcal{S}} = \min_{e \in \mathcal{S}_{\mathcal{C}_i}} d(x_j, e) \tag{9}$$

Because of the possibly non-convex structure of the skeleton and the varying lengths of the segments it is necessary to actually check every line segment. Heuristics proposing which edges are better candidates may fail even for the simplest skeletons.

VI. Evaluation

Internal validation measures allow to compare two different clusterings of the same data set and choose the better clustering. This can be either used to find the most appropriate number of clusters in a data set (if used with centroid or model-based methods) or to choose proper parameters for a density based clustering algorithm (cf. [8]). Here we will show that our algorithm yields appropriate partitioning matrices that can be used to evaluate clustering results obtained by density based algorithms.

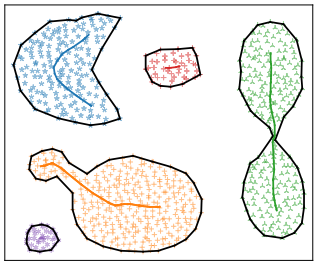

Fig. 7: `Aggregation` data set. With clusters found by DBSCAN ($\varepsilon = 2, minPts = 10$) and cluster skeletons.

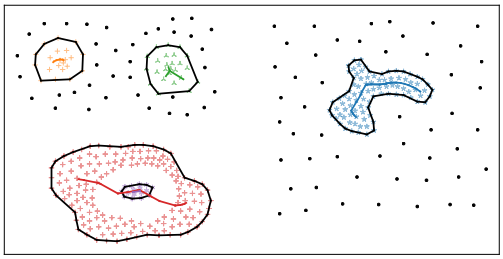

Fig. 8: `Compound` data set. With clusters found by DBSCAN ($\varepsilon = 1.4, minPts = 10$) and cluster skeletons. Black points indicate noise.

TABLE I: Data sets

Name	# points	# clusters
`aggregation`	788	7
`compound`	399	5
`r15`	600	15
`t4.8k`	8000	6

The data sets we use for evaluation show several interesting properties. In Figure 2 we already see intersecting convex hulls, which leads to common centroids not being representative anymore. Figure 7 shows the `aggregation` data set [22]. Clusters in this data set are not properly seperated (lower left) or connected by short bridges. Figure 8 shows the `compound` data set [23]. Clusters in this data set vary in density or are completely contained in each others convex hulls (lower left). Figure 9 shows the `r15` data set [24]. Clusters are very close to each other in the data set's central region. Figure 10 shows the `t4.8k` data set [25]. Clusters in this data set are overlapping and connected by a thin bridge of points following a sinusoidal curve. This data set contains – in addition – a high amount of noise.

To choose whether to use one specific partitioning of the data or another internal cluster validaten measures can be used. In contrast to *external* validation measures that rely on knowledge about the ground truth of the data (or can be used to calculate to which extent two different clusterings agree), internal validation measures are solely based on the clustering

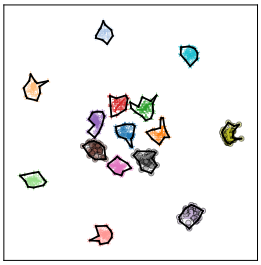

Fig. 9: `r15` data set. With clusters found by DBSCAN ($\varepsilon = 1.4, minPts = 10$) and cluster skeletons. Black points indicate noise.

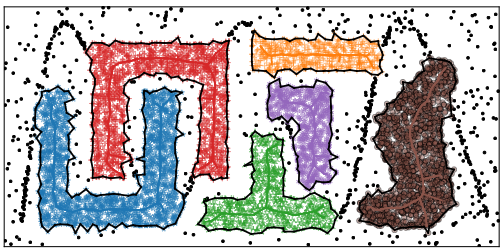

Fig. 10: `t4.8k` data set. With clusters found by DBSCAN ($\varepsilon = 10, minPts = 20$) and cluster skeletons. Black points indicate noise.

result and maybe some information about the data's structure.

The partition coefficition (PC, see Equation 10) [26] and the partition entropy (PE, see Equation 11) [27] can be used to compute scores solely based on the fuzzy partition matrix itself. While the PE should be minimized, the optimal value for the PC is 1.0. Both optima are actually reached for a crisp clustering where all membership degrees are either 1.0 or 0.0.

Partition Coefficient [26]

$$PC = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij}^2 \tag{10}$$

Partition Entropy [27]

$$PE = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij} \cdot \log u_{ij} \tag{11}$$

An index that also uses structural information about the data set is the Xie-Beni-Index (XB_ω, see Equation 12) [28]. It is based on the Fukuyama-Index [29] and differs from this only by the term for the clusters' separation. The separation is usually calculated as the minimal distance between two clusters' centers. Here we applied the same logic and calculated the minimal distance between all pairs of line segments forming either clusters' skeletons. In the case of the `compound` data set this leads to an intersection of the two skeletons. In this case we fall back to the Fukuyama-Index to prevent division by zero (cf. Figure 8, lower left).

TABLE II: Comparison of index results for good clustering and bad clusterings.
*Fukuyama-Index used, since skeletons were intersecting in the lower left.

data set	good clustering			bad clustering		
	PC	PE	XB_2	PC	PE	XB_2
`aggregation`	**0.838**	**0.363**	**0.083**	0.748	0.588	0.500
`compound`	**0.766**	**0.466**	4.779*	0.711	0.610	**4.574***
`r15`	0.892	0.296	**0.082**	**0.965**	**0.109**	0.844
`t4.8k`	**0.890**	**0.246**	**0.206**	0.778	0.550	0.659

$$XB_\omega = \frac{\sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij}^{\omega} d_{ij}^{\mathcal{S}}}{n \cdot \min\limits_{\mathcal{C}_j \neq \mathcal{C}_k} d(\mathcal{S}_{\mathcal{C}_j}, \mathcal{S}_{\mathcal{C}_k})} \tag{12}$$

Table II shows the results for the test cases chosen for this evaluation. Compared to bad clusterings (i.e. low agreement with the ground truth) of the data set, our algorithm always yields the better or similar results. In the case of the `r15` data set the *worse* clustering is produced by choosing the parameters in a way that the eight central clusters are placed under the same cluster label (i.e. only eight clusters in total were generated). This central structure seems to yield a better partitioning (w.r.t. the chosen validation indices) than the ground truth of the data set.

VII. Conclusion

In this paper we have presented a novel fuzzy density-based clustering algorithm. Our approach takes the crisp clustering result and calculates a cluster skeleton based on the concave hull of the cluster. The fuzzy membership degrees are then calculated in a way similar to the well-known fuzzy c-means algorithm. The distance calculations needed are not performed against the arithmetic mean of all points belonging to the cluster but instead against the cluster's skeleton which acts as a new, generalized centroid. Problems can occur if one cluster is completely contained in another. This may lead to intersections between the skeletons (and the skeletons are – again – not suitable as representations for the clusters). Another problem that might occur is overfitting. If the hulls representing the clusters are to serrated, the resulting skeleton may become too dendritic and adapt to the data too well.

However, experimental results based on artificial data sets and selected, internal evaluation measures show, that our algorithm yields membership degrees that can be used to choose the a better set of parameters w.r.t. a fuzzy cluster definition given implicitly by the used validation index. This can be used to automatically determine the set of optimal parameters and perform fuzzy density-based clustering with only a cluster definition and no explicit parametrization of the density-based clustering algorithm itself.

TABLE III: Data sets

Name	ε	$minPts$
`aggregation`	2	20
`compound`	1	10
`r15`	2.75	30
`t4.8k`	10	30

APPENDIX

Table III shows the parameters used to obtain the bad clusterings for the evaluation. We used the DBSCAN implementation from [30]. The data sets were obtained from `https://cs.joensuu.fi/sipu/datasets/` .

REFERENCES

[1] R. Kruse, C. Borgelt, C. Braune, S. Mostaghim, and M. Steinbrecher, *Computational intelligence: a methodological introduction*. Springer London, 2016.

[2] M. R. Berthold, C. Borgelt, F. Höppner, and F. Klawonn, *Guide to intelligent data analysis: how to intelligently make sense of real data*. Springer Science & Business Media, 2010.

[3] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA., 1967, pp. 281–297.

[4] J. C. Bezdek, R. Ehrlich, and W. Full, "Fcm: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984.

[5] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[6] C. Braune, S. Besecke, and R. Kruse, "Density based clustering: Alternatives to dbscan," in *Partitional Clustering Algorithms*. Springer, 2015, pp. 193–213.

[7] R. J. Campello, D. Moulavi, A. Zimek, and J. Sander, "A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies," *Data Mining and Knowledge Discovery*, vol. 27, no. 3, pp. 344–371, 2013.

[8] A. Dockhorn, C. Braune, and R. Kruse, "An alternating optimization approach based on hierarchical adaptations of dbscan," in *Computational Intelligence, 2015 IEEE Symposium Series on*. IEEE, 2015, pp. 749–755.

[9] P. Held and R. Kruse, "Online Fuzzy Community Detection by Using Nearest Hubs," in *Int. Conf. Inf. Process. Manag. Uncertain. Knowledge-Based Syst.* Eindhoven: Springer, 2016, pp. 678–689.

[10] R. Winkler, "Prototype based clustering in high-dimensional feature spaces," Ph.D. dissertation, Otto von Guericke University Magdeburg, June 2015.

[11] F. Klawonn, "Fuzzy clustering: insights and a new approach," *Mathware & soft computing. 2004 Vol. 11 Núm. 3*, 2004.

[12] E. N. Nasibov and G. Ulutagay, "Robustness of density-based clustering methods with various neighborhood relations," *Fuzzy Sets and Systems*, vol. 160, no. 24, pp. 3601–3615, 2009.

[13] J. K. Parker, L. O. Hall, and A. Kandel, "Scalable fuzzy neighborhood dbscan," in *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1–8.

[14] H. L. S. O. M. Kurihara and H. Sato, "A fast density-based clustering algorithm using fuzzy neighborhood functions," *75th National Convention of the Information Processing Society*, vol. 1, p. 9, 2013.

[15] G. Bordogna and D. Ienco, "Fuzzy core dbscan clustering algorithm," in *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*. Springer, 2014, pp. 100–109.

[16] A. Smiti and Z. Eloudi, "Soft dbscan: Improving dbscan clustering method using fuzzy set theory," in *2013 6th International Conference on Human System Interactions (HSI)*. IEEE, 2013, pp. 380–385.

[17] P. C. Mahalanobis, "On the generalized distance in statistics," *Proceedings of the National Institute of Sciences (Calcutta)*, vol. 2, pp. 49–55, 1936.

[18] C. Braune and R. Kruse, "Obtaining shape descriptors from a concave hull-based clustering algorithm," in *Advances in Intelligent Data Analysis XV*. Springer, 2016.

[19] H. Blum, "A Transformation for Extracting New Descriptors of Shape," in *Models for the Perception of Speech and Visual Form*. Cambridge: MIT Press, 1967, pp. 362–380.

[20] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Transactions on information theory*, vol. 29, no. 4, pp. 551–559, 1983.

[21] M. Duckham, L. Kulik, M. Worboys, and A. Galton, "Efficient generation of simple polygons for characterizing the shape of a set of points in the plane," *Pattern Recognition*, vol. 41, no. 10, pp. 3224–3236, 2008.

[22] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 4, 2007.

[23] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters," *IEEE Transactions on computers*, vol. 100, no. 1, pp. 68–86, 1971.

[24] C. J. Veenman, M. J. T. Reinders, and E. Backer, "A maximum variance cluster algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1273–1280, 2002.

[25] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer*, vol. 32, no. 8, pp. 68–75, 1999.

[26] J. C. Bezdek, "Cluster Validity with Fuzzy Sets," *Journal of Cybernetics*, vol. 3, no. 3, 1973.

[27] ——, "Mathematical models for systematics and taxonomy," in *Proceedings of Eigth International Conference on Numerical Taxonomy, San Francisco*, 1975, pp. 143–66.

[28] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 13, no. 8, pp. 841–847, 1991.

[29] Y. Fukuyama and M. Sugeno, "A new method of choosing the number of clusters for the fuzzy c-means method," in *Proc. 5th Fuzzy Syst. Symp*, vol. 247, 1989, pp. 247–250.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.