Density-Based Clustering Based on Hierarchical Density Estimates

Ricardo J.G.B. Campello*, Davoud Moulavi, and Joerg Sander**

Dept. of Computing Science, University of Alberta, Edmonton, AB, Canada
{rcampell,moulavi,jsander}@ualberta.ca

Abstract. We propose a theoretically and practically improved density-based, hierarchical clustering method, providing a clustering hierarchy from which a simplified tree of significant clusters can be constructed. For obtaining a "flat" partition consisting of only the most significant clusters (possibly corresponding to different density thresholds), we propose a novel cluster stability measure, formalize the problem of maximizing the overall stability of selected clusters, and formulate an algorithm that computes an optimal solution to this problem. We demonstrate that our approach outperforms the current, state-of-the-art, density-based clustering methods on a wide variety of real world data.

1 Introduction

Density-based clustering [1,2] is a popular clustering paradigm. However, the existing methods have a number of limitations: *(i)* Some methods (e.g., DBSCAN [3] and DENCLUE [4]) can *only* provide a "flat" (i.e. non-hierarchical) labeling of the data objects, based on a global density threshold. Using a single density threshold can often not properly characterize common data sets with clusters of very different densities and/or nested clusters. *(ii)* Among the methods that provide a clustering hierarchy, some (e.g., gSkeletonClu [5]) are not able to automatically simplify the hierarchy into an easily interpretable representation involving only the most significant clusters. *(iii)* Many hierarchical methods, including OPTICS [6] and gSkeletonClu, suggest only how to extract a flat partition by using a global cut/density threshold, which may not result in the most significant clusters if these clusters are characterized by *different* density levels. *(iv)* Some methods are limited to specific classes of problems, such as networks (gSkeletonClu), and point sets in the real coordinate space (e.g., DECODE [7], and Generalized Single-Linkage [8]). *(v)* Most methods depend on multiple, often critical input parameters (e.g., [3], [4], [7], [8], [9]).

In this paper, we propose a clustering approach that, to the best of our knowledge, is unique in that it does not suffer from any of these drawbacks. In detail, we make the following contributions: *(i)* We introduce a hierarchical clustering

* Currently on a sabbatical leave, he is originally from SCC/ICMC/USP, University of São Paulo at São Carlos, Brazil. He acknowledges FAPESP and CNPq.
** This work has been partly supported by NSERC.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 160–172, 2013.

method, called HDBSCAN, which generates a complete density-based clustering hierarchy from which a simplified hierarchy composed only of the most significant clusters can be easily extracted. *(ii)* We propose a new measure of cluster stability for the purpose of extracting a set of significant clusters from possibly different levels of a simplified cluster tree produced by HDBSCAN. *(iii)* We formulate the task of extracting a set of significant clusters as an optimization problem in which the overall stability of the composing clusters is maximized. *(iv)* We propose an algorithm that finds the globally optimal solution to this problem. *(v)* We demonstrate the advancement in density-based clustering that our approach represents on a variety of real world data sets.

The remainder of this paper is organized as follows. We discuss related work in Section 2. In Section 3, we redefine DBSCAN, and we propose the algorithm HDBSCAN in Section 4. In Section 5, we introduce a new measure of cluster stability, propose the problem of extracting an optimal set of clusters from a cluster tree, and give an algorithm to solve this problem. Section 6 presents an extensive experimental evaluation, and Section 7 concludes the paper.

2 Related Work

Apart from methods aimed at getting approximate estimates of level sets and density-contour trees for continuous-valued p.d.f. — e.g., see [8] and references therein — not much attention has been given to hierarchical density-based clustering in more general data spaces. The works most related to ours are those in [6,9,5,10]. In [6], a post-processing procedure to extract a simplified cluster tree from the reachability plot produced by the OPTICS algorithm was proposed. This procedure did not become as popular as OPTICS itself, probably because it is very sensitive to the choice of a critical parameter that cannot easily be determined or understood. Moreover, no automatic method to extract a flat clustering solution based on local cuts in the obtained tree was described. In [9], an improved method to extract trees of significant clusters from reachability plots was proposed that is less sensitive to the user settings than the original method in [6]. However, this method is based on heuristics with embedded threshold values that can strongly affect the results, and the problem of extracting a flat solution from local cuts in the cluster tree was practically untouched; the only mentioned (ad-hoc) approach was to arbitrarily take all the leaf clusters and discard the others. In [5], the original findings from [6,9,11] were recompiled in the particular context of community discovery in complex networks. However, no mechanism to extract a simplified cluster tree from the resulting (single-linkage-like) clustering dendrogram was adopted, and only a method producing a global cut through the dendrogram was described. The algorithm AUTO-HDS [10] is, like our method, based on a principle used to simplify clustering hierarchies, which in part refers back to the work of [12]. The clustering hierarchy obtained by AUTO-HDS is typically a subset of the one obtained by our method HDBSCAN. Conceptually, it is equivalent to a sampling of the HDBSCAN hierarchical levels, from top to bottom, at a geometric rate controlled by a user-defined parameter, r_{shave}. Such

a sampling can lead to an underestimation of the stability of clusters or even to missed clusters, and these side effects can only be prevented if $r_{shave} \to 0$. In this case, however, the asymptotic running time of AUTO-HDS is $O(n^3)$ [13] (in contrast to $O(n^2 \log n)$ for "sufficiently large" values of r_{shave}). In addition, the stability measure used in AUTO-HDS has the undesirable property that the stability value for a cluster in one branch of the hierarchy can be affected by the density and cardinality of other clusters lying on different branches. AUTO-HDS also attempts to perform local cuts through the hierarchy in order to extract a flat clustering solution, but it uses a greedy heuristic for selecting clusters that may give suboptimal results in terms of an overall stability.

3 DBSCAN Revisited — The Algorithm DBSCAN*

Let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ be a data set of n objects, and let D be an $n \times n$ matrix containing the pairwise distances $d(\mathbf{x}_p, \mathbf{x}_q)$, $\mathbf{x}_p, \mathbf{x}_q \in \mathbf{X}$, for a metric distance $d(\cdot, \cdot)$.[1] We define density-based clusters based on *core objects* alone:

Definition 1. (Core Object): An object $\mathbf{x}_p$ is called a *core object* w.r.t. ε and m_{pts} if its ε-neighborhood contains at least m_{pts} many objects, i.e., if $|\mathbf{N}_\varepsilon(\mathbf{x}_p)| \geq m_{pts}$, where $\mathbf{N}_\varepsilon(\mathbf{x}_p) = \{\mathbf{x} \in \mathbf{X} \mid d(\mathbf{x}, \mathbf{x}_p) \leq \varepsilon\}$ and $|\cdot|$ denotes cardinality. An object is called *noise* if it is not a core object.

Definition 2. (ε-Reachable): Two *core* objects $\mathbf{x}_p$ and $\mathbf{x}_q$ are *ε-reachable* w.r.t. ε and m_{pts} if $\mathbf{x}_p \in \mathbf{N}_\varepsilon(\mathbf{x}_q)$ and $\mathbf{x}_q \in \mathbf{N}_\varepsilon(\mathbf{x}_p)$.

Definition 3. (Density-Connected): Two *core* objects $\mathbf{x}_p$ and $\mathbf{x}_q$ are *density-connected* w.r.t. ε and m_{pts} if they are directly or transitively ε-reachable.

Definition 4. (Cluster): A *cluster* $\mathbf{C}$ w.r.t. ε and m_{pts} is a non-empty maximal subset of $\mathbf{X}$ such that every pair of objects in $\mathbf{C}$ is density-connected.

Based on these definitions, we can devise an algorithm DBSCAN* (similar to DBSCAN) that conceptually finds clusters as the connected components of a graph in which the objects of $\mathbf{X}$ are vertices and every pair of vertices is adjacent if and only if the corresponding objects are ε-reachable w.r.t. user-defined parameters ε and m_{pts}. Non-core objects are labeled as noise.

Note that the original definitions of DBSCAN also include the concept of *border* objects, i.e., non-core objects that are within the ε-neighborhood of a core object. Our new definitions are more consistent with a statistical interpretation of clusters as connected components of a level set of a density (as defined, e.g., in [14]), since border objects do not technically belong to the level set (their estimated density is below the threshold). The new definitions also allow a precise relationship between DBSCAN* and its hierarchical version, to be discussed next. This was only approximately possible between DBSCAN and OPTICS.

[1] The matrix D is not required if distances $d(\cdot, \cdot)$ can be computed from $\mathbf{X}$ on demand.

4 Hierarchical DBSCAN* — HDBSCAN

In this section, we introduce a hierarchical clustering method, HDBSCAN, which can be seen as a conceptual and algorithmic improvement over OPTICS. Our method has as its single input parameter a value for m_{pts}, which is a classic smoothing factor in density estimates whose behavior is well understood (methods with a corresponding parameter, e.g., [6,10,7,8], are quite robust to it). Different density levels in the resulting density-based cluster hierarchy will then correspond to different values of the radius ε.

For a proper formulation of the density-based hierarchy w.r.t. a value of m_{pts}, we define the notions of core distance and a symmetric reachability distance (following the definition used in [11]), a new notion of *ε-core objects*, as well as the notion of a *conceptual*, transformed proximity graph, which will help us to explain a density-based clustering hierarchy.

Definition 5. (Core Distance): The *core distance* of an object $\mathbf{x}_p \in \mathbf{X}$ w.r.t. m_{pts}, $d_{core}(\mathbf{x}_p)$, is the distance from $\mathbf{x}_p$ to its m_{pts}-nearest neighbor (incl. $\mathbf{x}_p$).

Definition 6. (ε-Core Object): An object $\mathbf{x}_p \in \mathbf{X}$ is called an *ε-core* object for every value of ε that is greater than or equal to the core distance of $\mathbf{x}_p$ w.r.t. m_{pts}, *i.e.*, if $d_{core}(\mathbf{x}_p) \leq \varepsilon$.

Definition 7. (Mutual Reachability Distance): The *mutual reachability distance* between two objects $\mathbf{x}_p$ and $\mathbf{x}_q$ in $\mathbf{X}$ w.r.t. m_{pts} is defined as $d_{mreach}(\mathbf{x}_p, \mathbf{x}_q) = \max\{d_{core}(\mathbf{x}_p), d_{core}(\mathbf{x}_q), d(\mathbf{x}_p, \mathbf{x}_q)\}$.

Definition 8. (Mutual Reachability Graph): It is a complete graph, $G_{m_{pts}}$, in which the objects of $\mathbf{X}$ are vertices and the weight of each edge is the mutual reachability distance (w.r.t. m_{pts}) between the respective pair of objects.

Let $G_{m_{pts},\varepsilon} \subseteq G_{m_{pts}}$ be the graph obtained by removing all edges from $G_{m_{pts}}$ having weights greater than ε. From Definitions 4, 6, and 8, it is straightforward to infer that clusters according to DBSCAN* w.r.t. m_{pts} and ε are the connected components of ε-core objects in $G_{m_{pts},\varepsilon}$; the remaining objects are noise. Consequently, all DBSCAN* partitions for $\varepsilon \in [0, \infty)$ can be produced in a nested, *hierarchical* way by removing edges in decreasing order of weight from $G_{m_{pts}}$.

Proposition 1. *Let $\mathbf{X}$ be a set of n objects described in a metric space by $n \times n$ pairwise distances. The partition of this data obtained by DBSCAN* w.r.t m_{pts} and ε is identical to the one obtained by first running Single-Linkage over the transformed space of mutual reachability distances, then, cutting the resulting dendrogram at level ε of its scale, and treating all resulting singletons with $d_{core}(\mathbf{x}_p) > \varepsilon$ as a single class representing "Noise".*

Proof. Proof sketch as per discussion above, after Definition 8. □

Proposition 1 states that we could implement a hierarchical version of DBSCAN* by an algorithm that first computes a Single-Linkage hierarchy on the space of

Algorithm 1. HDBSCAN main steps

1. Compute the core distance w.r.t. m_{pts} for all data objects in $\mathbf{X}$.
2. Compute an MST of $G_{m_{pts}}$, the Mutual Reachability Graph.
3. Extend the MST to obtain MST_{ext}, by adding for each vertex a "self edge" with the core distance of the corresponding object as weight.
4. Extract the HDBSCAN hierarchy as a dendrogram from MST_{ext}:
   4.1 For the root of the tree assign all objects the same label (single "cluster").
   4.2 Iteratively remove all edges from MST_{ext} in decreasing order of weights (in case of ties, edges must be removed simultaneously):
      4.2.1 Before each removal, set the dendrogram scale value of the current hierarchical level as the weight of the edge(s) to be removed.
      4.2.2 After each removal, assign labels to the connected component(s) that contain(s) the end vertex(-ices) of the removed edge(s), to obtain the next hierarchical level: assign a new cluster label to a component if it still has at least one edge, else assign it a null label ("noise").

transformed distances (i.e., mutual reachability distances) and, then, processes this hierarchy to identify connected components and noise objects at each level. Here, we propose a more efficient and elegant equivalent solution.

A density-based cluster hierarchy has to represent the fact that an object o is noise below the level l that corresponds to o's core distance. To represent this in a dendrogram, we propose to include an additional dendrogram node for o at level l representing the cluster containing o at that level and higher. To directly construct such a hierarchy, we propose an extension of a Minimum Spanning Tree (MST) of the Mutual Reachability Graph $G_{m_{pts}}$, from which we then can construct the extended dendrogram by removing edges in decreasing order of weights. More precisely, we extend the MST with edges connecting each vertex o to itself (self-loops), where the edge weight is set to the core distance of o. These "self edges" will then be considered when removing edges.

Algorithm 1 shows the pseudo-code for HDBSCAN, which has as inputs a value for m_{pts} and the data set $\mathbf{X}$. It produces a clustering tree that contains all partitions obtainable by DBSCAN* (w.r.t. m_{pts}) in a hierarchical, nested way. It also contains nodes that indicate when an isolated object changes from core (i.e., dense) to noise. The result is called the "HDBSCAN hierarchy". Using an implementation of Prim's algorithm based on an ordinary list search (instead of a heap) to construct the MST, the method can be fully implemented in $O(dn^2)$ running time, where d is the number of data attributes. Also, by noticing that only the currently processed hierarchical level is needed at any point in time, the algorithm needs to keep in main memory essentially the data set $\mathbf{X}$ and the extended MST that can be constructed directly from it, which requires $O(dn)$ space. If a data matrix D is provided as input in lieu of $\mathbf{X}$, the algorithm requires $O(n^2)$ space instead, but its time complexity reduces to $O(n^2)$.

Algorithm 2. HDBSCAN step 4.2.2 with (optional) parameter $m_{clSize} \geq 1$

4.2.2 After each removal (to obtain the next hierarchical level), process one at a time each cluster that contained the edge(s) just removed, by relabeling its resulting connected subcomponent(s):

Label *spurious* subcomponents as noise by assigning them the null label. If all subcomponents of a cluster are *spurious*, then the **cluster has disappeared**.

Else, if a single subcomponent of a cluster is *not spurious*, keep its original cluster label (**cluster has just shrunk**).

Else, if two or more subcomponents of a cluster are *not spurious*, assign new cluster labels to each of them (**"true" cluster split**).

4.1 Hierarchy Simplification

The HDBSCAN hierarchy can easily be visualized as a traditional dendrogram or related representations. However, these plots are not easy to interpret or process for large and "noisy" data sets, so it is a fundamental problem to extract from a dendrogram a summarized tree of only "significant" clusters. We propose a simplification of the HDBSCAN hierarchy based on a fundamental observation about estimates of the level sets of continuous-valued probability density functions (p.d.f.), which refers back to Hartigan's concept of *rigid clusters* [14], and which has also been employed similarly by Gupta *et al.* in [10]. For a given p.d.f., there are only three possibilities for the evolution of the connected components of a continuous density level set when increasing the density level (decreasing ε in our context) [12]: *(i)* the component shrinks but remains connected, up to a density threshold at which either *(ii)* the component is divided into smaller ones, or *(iii)* it disappears. This observation can be applied to the HDBSCAN hierarchy to select only those hierarchical levels in which new clusters arise by a "true" split of a cluster, or in which clusters disappear; these are the levels in which the most significant changes in the clustering structure occur. When decreasing ε, the ordinary removal of noise objects from a cluster is not a "true" split; a cluster only shrinks in this case, so it should keep the same label.

We can generalize this idea by setting a minimum cluster size, a commonly used practice in real cluster analysis (see, e.g., the notion of a *particle* in AUTOHDS [10]). With a minimum cluster size, $m_{clSize} \geq 1$, components with fewer than m_{clSize} objects are disregarded, and their disconnection from a cluster does not establish a "true" split. We can adapt HDBSCAN accordingly by changing Step 4.2.2 of Algorithm 1 as shown in Algorithm 2: a connected component is deemed *spurious* if it has fewer than m_{clSize} objects or, for $m_{clSize} = 1$, if it is an isolated, non-dense object (no edges). Any spurious component is labeled as noise and its removal from a bigger component is not considered as a cluster split. In practice, this can reduce the size of the hierarchy dramatically.

The optional parameter m_{clSize} represents an independent control for the smoothing of the resulting cluster tree, in addition to m_{pts}. To make HDBSCAN more similar to previous density-based approaches and to simplify its use, we can set $m_{clSize} = m_{pts}$, which turns m_{pts} into a single parameter that acts as both a smoothing factor and a threshold for the cluster size.

5 Optimal Non-hierarchical Clustering

In many applications a user is interested in extracting a "flat" partition of the data, consisting of the prominent clusters. Those clusters, however, may have very different local densities and may not be detectable by a single, global density threshold, i.e., global cut through a hierarchical cluster representation. In this section, we describe an algorithm that provides the optimal global solution to the formal optimization problem of maximizing the overall stability of the set of clusters extracted from the HDBSCAN hierarchy.

5.1 Cluster Stability

Without loss of generality, let us initially consider that the data objects are described by a single continuous-valued attribute x. Following Hartigan's model [14], the density-contour clusters of a given density $f(x)$ on $\Re$ at a given density level λ are the maximal connected subsets of the level set defined as $\{x \,|\, f(x) \geq \lambda\}$. Most density-based clustering algorithms are to some extent based on this concept. The differences lie in the way the density $f(x)$ and the maximal connected subsets are estimated, e.g., DBSCAN* estimates the density-contour clusters for a density threshold $\lambda = 1/\varepsilon$ and a (non-normalized) K-NN estimate (for $K = m_{pts}$) of the density $f(x)$, given by $1/d_{core}(x)$.

HDBSCAN produces all possible DBSCAN* solutions w.r.t. a given value of m_{pts} and all thresholds $\lambda = 1/\varepsilon$ in $[0, \infty)$. Intuitively, when increasing λ (i.e., decreasing ε), clusters get smaller and smaller, until they disappear or break into sub-clusters; more prominent clusters will "survive" longer after they appear. To formalize this concept, we adapt the notion of *excess of mass* [15]: Imagine increasing the density level λ, and assume that a density-contour cluster $\mathbf{C}_i$ appears at level $\lambda_{min}(\mathbf{C}_i)$. The excess of mass of $\mathbf{C}_i$ is defined in Equation (1), and illustrated in Figure 1, where the darker shaded areas represent the excesses of mass of three clusters, $\mathbf{C}_3$, $\mathbf{C}_4$, and $\mathbf{C}_5$. The excess of mass of $\mathbf{C}_2$ (not highlighted in the figure) encompasses those of its descendants $\mathbf{C}_4$ and $\mathbf{C}_5$.

$$E(\mathbf{C}_i) = \int_{x \in \mathbf{C}_i} \Big(f(x) - \lambda_{min}(\mathbf{C}_i) \Big) \, dx \tag{1}$$

The excess of mass exhibits a monotonic behavior along any branch of the hierarchical cluster tree. As a consequence, this measure cannot be used to compare the stabilities of nested clusters, such as $\mathbf{C}_2$ against $\mathbf{C}_4$ and $\mathbf{C}_5$. To be able to do so, we introduce here the notion of *Relative Excess of Mass* of a cluster $\mathbf{C}_i$, which appears at level $\lambda_{min}(\mathbf{C}_i)$, as:

$$E_R(\mathbf{C}_i) = \int_{x \in \mathbf{C}_i} \Big(\lambda_{max}(x, \mathbf{C}_i) - \lambda_{min}(\mathbf{C}_i) \Big) \, dx \tag{2}$$

where $\lambda_{max}(x, \mathbf{C}_i) = \min\{f(x), \lambda_{max}(\mathbf{C}_i)\}$, and $\lambda_{max}(\mathbf{C}_i)$ is the density level at which $\mathbf{C}_i$ is split or disappears. For example, for cluster $\mathbf{C}_2$ in Figure 1 it follows that $\lambda_{max}(\mathbf{C}_2) = \lambda_{min}(\mathbf{C}_4) = \lambda_{min}(\mathbf{C}_5)$. The corresponding relative excess of mass is represented by the lighter shaded area in Figure 1.

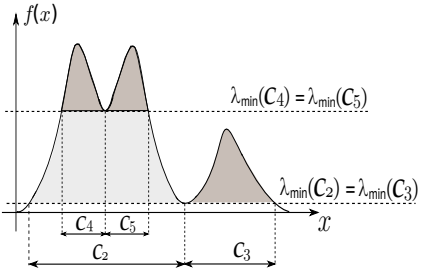

Fig. 1. Illustration of a density function, clusters, and excesses of mass

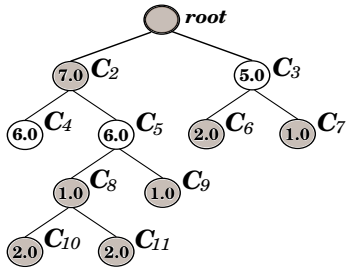

Fig. 2. Illustration of the optimal selection of clusters from a given cluster tree

For a HDBSCAN hierarchy, where we have a finite data set $\mathbf{X}$, cluster labels, and density thresholds associated with each hierarchical level, we can adapt Equation (2) to define the *stability* of a cluster $\mathbf{C}_i$ as:

$$S(\mathbf{C}_i) = \sum_{\mathbf{x}_j \in \mathbf{C}_i} \Big(\lambda_{max}(\mathbf{x}_j, \mathbf{C}_i) - \lambda_{min}(\mathbf{C}_i) \Big) = \sum_{\mathbf{x}_j \in \mathbf{C}_i} \Big(\frac{1}{\varepsilon_{min}(\mathbf{x}_j, \mathbf{C}_i)} - \frac{1}{\varepsilon_{max}(\mathbf{C}_i)} \Big) \quad (3)$$

where $\lambda_{min}(\mathbf{C}_i)$ is the minimum density level at which $\mathbf{C}_i$ exists, $\lambda_{max}(\mathbf{x}_j, \mathbf{C}_i)$ is the density level beyond which object $\mathbf{x}_j$ no longer belongs to cluster $\mathbf{C}_i$, and $\varepsilon_{max}(\mathbf{C}_i)$ and $\varepsilon_{min}(\mathbf{x}_j, \mathbf{C}_i)$ are the corresponding values for the threshold ε.

5.2 Optimization Algorithm

Let $\{\mathbf{C}_2, \cdots, \mathbf{C}_\kappa\}$ be the collection of all clusters in the simplified cluster hierarchy (tree) generated by HDBSCAN, except the root $\mathbf{C}_1$, and let $S(\mathbf{C}_i)$ denote the stability value of each cluster. The goal is to extract the most "prominent" clusters (plus possibly noise) as a *"flat"*, non-overlapping partition. This task can be formulated as an optimization problem with the objective of maximizing the sum of stabilities of the extracted clusters in the following way:

$$\max_{\delta_2, \ldots, \delta_\kappa} \quad J = \sum_{i=2}^{\kappa} \delta_i \, S(\mathbf{C}_i)$$
$$\text{subject to} \begin{cases} \delta_i \in \{0, 1\}, \;\; i = 2, \cdots, \kappa \\ \sum_{j \in \mathbf{I}_h} \delta_j = 1, \forall h \; \in \; \mathbf{L} \end{cases} \quad (4)$$

where δ_i $(i = 2, \cdots, \kappa)$ indicates whether cluster $\mathbf{C}_i$ is included in the flat solution ($\delta_i = 1$) or not ($\delta_i = 0$), $\mathbf{L} = \{h \mid \mathbf{C}_h$ is a leaf cluster$\}$ is the set of indexes of leaf clusters, and $\mathbf{I}_h = \{j \mid j \neq 1$ and $\mathbf{C}_j$ is ascendant of $\mathbf{C}_h$ (h included)$\}$ is the set of indexes of all clusters on the path from $\mathbf{C}_h$ to the root (excluded). The constraints prevent nested clusters on the same path to be selected.

Algorithm 3. Solution to Problem (4)

1. Initialize $\delta_2 = \cdots = \delta_\kappa = 1$, and, for all leaf nodes, set $\hat{S}(\mathbf{C}_h) = S(\mathbf{C}_h)$.
2. Starting from the deepest levels, do bottom-up (except for the root):
   2.1 If $S(\mathbf{C}_i) < \hat{S}(\mathbf{C}_{i_l}) + \hat{S}(\mathbf{C}_{i_r})$, set $\hat{S}(\mathbf{C}_i) = \hat{S}(\mathbf{C}_{i_l}) + \hat{S}(\mathbf{C}_{i_r})$ and set $\delta_i = 0$.
   2.2 Else: set $\hat{S}(\mathbf{C}_i) = S(\mathbf{C}_i)$ and set $\delta_{(\cdot)} = 0$ for all clusters in $\mathbf{C}_i$'s subtrees.

To solve Problem (4), we process every node except the root, starting from the leaves (bottom-up), deciding at each node $\mathbf{C}_i$ whether $\mathbf{C}_i$ or the best-so-far selection of clusters in $\mathbf{C}_i$'s subtrees should be selected. To be able to make this decision locally at $\mathbf{C}_i$, we propagate and update the total stability $\hat{S}(\mathbf{C}_i)$ of clusters selected in the subtree rooted at $\mathbf{C}_i$ in the following, recursive way:

$$\hat{S}(\mathbf{C}_i) \;=\; \begin{cases} S(\mathbf{C}_i), & \text{if } \mathbf{C}_i \text{ is a leaf node} \\ \max\{S(\mathbf{C}_i), \hat{S}(\mathbf{C}_{i_l}) + \hat{S}(\mathbf{C}_{i_r})\} & \text{if } \mathbf{C}_i \text{ is an internal node} \end{cases} \tag{5}$$

where $\mathbf{C}_{i_l}$ and $\mathbf{C}_{i_r}$ are the left and right children of $\mathbf{C}_i$ (for the sake of simplicity, we discuss the case of binary trees; the generalization to n-ary trees is trivial).

Algorithm 3 gives the pseudo-code for finding the optimal solution to Problem (4). Figure 2 illustrates the algorithm. Clusters $\mathbf{C}_{10}$ and $\mathbf{C}_{11}$ together are better than $\mathbf{C}_8$, which is then discarded. However, when the set $\{\mathbf{C}_{10}, \mathbf{C}_{11}, \mathbf{C}_9\}$ is compared to $\mathbf{C}_5$, they are discarded as $\mathbf{C}_5$ is better. Clusters $\{\mathbf{C}_4\}$ and $\{\mathbf{C}_5\}$ are better than $\mathbf{C}_2$, and $\mathbf{C}_3$ is better than $\{\mathbf{C}_6, \mathbf{C}_7\}$, so that in the end, only clusters $\mathbf{C}_3$, $\mathbf{C}_4$, and $\mathbf{C}_5$ remain, which is the optimal solution to (4) with $J = 17$.

Step 2.2 of Algorithm 3 can be implemented in a more efficient way by not setting $\delta_{(\cdot)}$ values to 0 for discarded clusters down in the subtrees (which could happen multiple times). Instead, in a simple post-processing procedure, the tree can be traversed top-down in order to find, for each branch, the shallowest cluster that has not been discarded ($\delta_{(\cdot)} = 1$). Thus, Algorithm 3 can be implemented with two traversals through the tree, one bottom-up and another one top-down. This results in an asymptotic complexity of $O(\kappa)$, both in terms of running time and memory space, where κ is the number of clusters in the simplified tree (which is typically much smaller than the number of data objects).

6 Experiments and Discussion

Data Sets. We report the performance on 9 individual data sets plus the average performance on 2 collections of data sets, representing a large variety of application domains and data characteristics (no. of objects, dimensionality, no. of clusters, and distance function). The first three data sets ("CellCycle-237", "CellCycle-384", and "YeastGalactose") represent gene-expression data. CellCycle-237 and CellCycle-384 were made public by Yeung et al. [16]; they contain 237 resp. 384 objects (genes), 4 resp. 5 known classes, and have both 17 dimensions (conditions). YeastGalactose contains a subset of 205 objects

(genes) and 20 dimensions (conditions) used in [17], with 4 known classes. For these data sets we used Euclidean distance on the z-score normalized objects, which is equivalent to using Pearson correlation on the original data. The next three data sets are the Wine, Glass, and Iris from the UCI Repository [18], containing 178, 214, resp. 150 objects in 13, 9, resp. 4 dimensions, with 3, 7, resp. 3 classes. For these data sets we used Euclidean distance. The last three individual data sets consist of very high dimensional representations of text documents. In particular, "Articles-1442-5" and "Articles-1442-80", made available upon request by Naldi et al. [19], are formed by 253 articles represented by 4636 and 388 dimensions, respectively. "Cbrilpirivson" [20] is composed of 945 articles represented by 1431 dimensions and is available at http://infoserver.lcad.icmc.usp.br/infovis2/PExDownload. The number of classes in all three document data sets is 5, and we used the Cosine measure as dissimilarity function. In addition to individual data sets we also report *average* performance on two collections of data sets, which are based on the Amsterdam Library of Object Images (ALOI) [21]. Image sets were created as in [22] by randomly selecting k ALOI image categories as class labels 100 times for each $k = 2, 3, 4, 5$, then sampling (without replacement), each time, 25 images from each of the k selected categories, thus resulting in 400 sets, each of which contains 2, 3, 4, or 5 clusters and 50, 75, 100, or 125 images (objects). The images were represented using six different descriptors: color moments (144 attributes), texture statistics from the gray-level co-occurrence matrix (88 attributes), Sobel edge histogram (128 attributes), 1st order statistics from the gray-level histogram (5 attributes), gray-level run-length matrix features (44 attributes), and gray-level histogram (256 attributes). We report *average* clustering results for the texture statistics, denoted by "ALOI-TS88", which is typical for the individual descriptors. We also show results for a 6-dimensional representation combining the first principal component extracted from each of the 6 descriptors using PCA, denoted by "ALOI-PCA". We used Euclidean distance in both cases.

Algorithms. Our method, denoted here by "HDBSCAN(EOM)" (EOM refers to cluster extraction based on Excess Of Mass), is compared with: *(i)* AUTO-HDS [10]; and *(ii)* a method referred to here as "OPTICS(AutoCl)", which consists of first running OPTICS, and then using the method proposed by Sander *et al.* [9] to extract a flat partitioning. We set m_{pts} (n_ε in AUTO-HDS, *MinPts* in OPTICS) equal to 4 in all experiments. The speed-up control value ε in OPTICS was set to "infinity", thereby eliminating its effect. For AUTO-HDS, we set the additional parameters r_{shave} to 0.03 (following the authors' suggestion to use values between 0.01 and 0.05) and *particle size*, n_{part}, to $m_{pts} - 1$. The corresponding parameter m_{clSize} in HDBSCAN was set equivalently to m_{pts}.[2]

Quality Measures. The measures we report are the Overall F-measure [23] and Adjusted Rand Index [24], denoted by "FScore" resp. "ARI", which are measures commonly used in the literature. In addition, we also report the fraction of objects assigned to clusters (as opposed to noise), denoted by "%covered".

[2] We also tried other values of m_{pts}, r_{shave}, and n_{part}/m_{clSize}, with similar results.

Table 1. Results for all data sets

	OPTICS(AutoCl)			AUTO-HDS			HDBSCAN(EOM)		
Data Set	ARI	FScore	%covered	ARI	FScore	%covered	ARI	FScore	%covered
CellCycle-237	**0.62**	**0.71**	**0.80**	0.04	0.29	0.37	0.48	0.66	0.65
CellCycle-384	0	0.37	**1**	**0.35**	**0.50**	0.46	**0.35**	**0.50**	0.47
YeastGalactose	**0.96**	**0.97**	**0.96**	0.94	0.96	**0.96**	0.94	**0.97**	**0.96**
Wine	0.16	0.48	0.77	0.12	0.37	0.72	**0.29**	**0.62**	**0.97**
Glass	0.23	0.49	0.76	0.12	0.37	0.45	**0.24**	**0.51**	**0.79**
Iris	0.33	0.61	0.83	0.11	0.40	0.46	**0.57**	**0.78**	**1**
Articles-1442-80	0.91	0.96	0.96	0.66	0.74	0.76	**0.93**	**0.97**	**0.98**
Articles-1442-5	0.89	0.94	0.93	0.60	0.76	0.73	**0.90**	**0.95**	**0.94**
Cbrilpirivson	0.01	0.07	0.11	0.04	0.21	0.34	**0.19**	**0.47**	**0.48**
ALOI-TS88	0.45	0.67	0.74	0.50	0.70	0.78	**0.63**	**0.79**	**0.85**
ALOI-PCA	0.61	0.78	0.83	0.56	0.74	0.81	**0.72**	**0.85**	**0.91**

Clustering Results. The results obtained in our experiments are shown in Table 1. The highest obtained values for each data set are highlighted in bold. Note that HDBSCAN(EOM) outperforms the other two methods in a large majority of the data sets (in many cases by a large margin) and, in almost all cases, it covers a larger fraction of objects while having also high FScore and ARI values. A high fraction of clustered objects is only good when also the clustering quality is high. E.g., for CellCycle-384, OPTICS(AutoCl) covers 100% of the data, but with an ARI of 0, a meaningless clustering. In one of the only two cases where HDBSCAN(EOM) does not perform best, YeastGalactose, its ARI is very close to (and its FScore matches that of) the "winner", OPTICS(AutoCl).

The collections of image data sets, ALOI-TS88 and ALOI-PCA, allowed us to perform paired t-tests with respect to ARI and FScore, confirming that the observed differences in performance between all pairs of methods is statistically significant at the $\alpha = 0.01$ significance level. This means that the methods are doing indeed different things, and, in particular, that HDBSCAN(EOM) significantly outperforms the others on these data set collections.

7 Final Remarks

A novel density-based clustering approach has been introduced that provides: *(i)* a complete density-based clustering hierarchy representing all possible DBSCAN-like solutions for an infinite range of density thresholds and from which a simplified tree of significant clusters can be extracted; and *(ii)* a flat partition composed of clusters extracted from optimal local cuts through the cluster tree. An extensive experimental evaluation on a wide variety of real world data sets has shown that our method performs significantly better and more robust than state-of-the-art methods. Our work lends itself to a number of interesting challenges for future work, which includes integration of semi-supervision and the consideration of subspaces.