

# Density-based Cluster Algorithms in Low-dimensional and High-dimensional Applications

Benno Stein<sup>1</sup> and Michael Busch<sup>2</sup>

<sup>1</sup> Faculty of Media, Media Systems  
Bauhaus University Weimar, 99421 Weimar, Germany  
benno.stein@medien.uni-weimar.de

<sup>2</sup> IBM Silicon Valley Laboratory  
WebSphere II OmniFind Edition Project

**Abstract** Cluster analysis is the art of detecting groups of similar objects in large data sets—without having specified these groups by means of explicit features. Among the various cluster algorithms that have been developed so far the density-based algorithms count to the most advanced and robust approaches.

However, this paper shows that density-based cluster analysis embodies no principle with clearly defined algorithmic properties. We contrast the density-based cluster algorithms DBSCAN and MajorClust, which have been developed having different clustering tasks in mind, and whose strengths and weaknesses can be explained against the background of the dimensionality of the data to be clustered.

Our motivation for this analysis comes from the field of information retrieval, where cluster analysis plays a key role in solving the document categorization problem. The paper is organized as follows: Section 1 recapitulates the important principles of cluster algorithms, Section 2 discusses the density-based algorithms DBSCAN and MajorClust, and Section 3 illustrates the strengths and weaknesses of both algorithms on the basis of geometric data analysis and document categorization problems.

**Key words:** density-based cluster analysis, high-dimensional data, document categorization

## 1 Cluster Algorithms

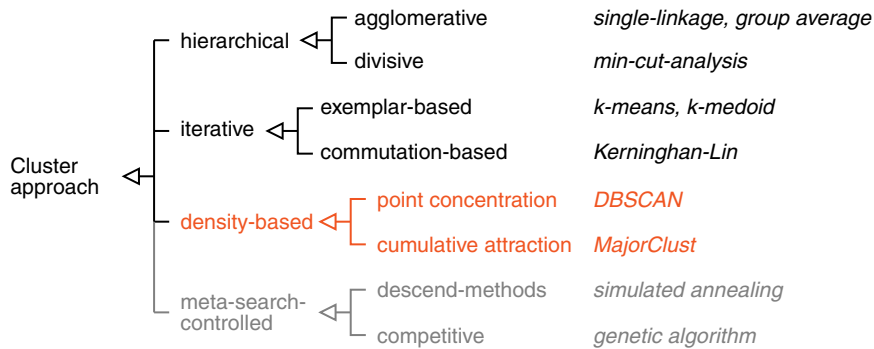
This section gives a short introduction to the problem of cluster analysis and outlines existing cluster approaches along the classical taxonomy. Moreover, it presents an alternative view to cluster algorithms, which is suited to explain different characteristics of their behavior.

### 1.1 The Classical Taxonomy of Cluster Algorithms

**Definition 1 (Clustering).** Let  $D$  be a set of objects. A clustering  $\mathcal{C} \subseteq \{C \mid C \subseteq D\}$  of  $D$  is a division of  $D$  into sets for which the following conditions hold:  $\bigcup_{C_i \in \mathcal{C}} C_i = D$ , and  $\forall C_i, C_j \in \mathcal{C} : C_i \cap C_j \neq i = \emptyset$ . The sets  $C_i$  are called clusters.

With respect to the set of objects  $D$  the following shall be stipulated:

- $|D| = n$
- The objects in  $D$  represent points in the Euclidean space of dimension  $m$ .
- Based on a metric  $d : D \times D \rightarrow \mathbf{R}$ , the similarity or the dissimilarity between any two points in  $D$  can be stated.



**Figure 1.** The classical taxonomy of cluster algorithms.

A cluster algorithm takes a set  $D$  of objects as input and operationalizes a strategy to generate a clustering  $\mathcal{C}$ . Informally stated, the overall objective of a cluster algorithm is to maximize the inner-cluster similarity and to minimize the intra-cluster similarity. The fulfillment of this objective can be quantified by a statistical measure like the Dunn index, the Davies-Bouldin index, or the  $\Delta$ -measure [3, 5, 35].

Various cluster algorithms have been devised so far; usually they are classified with respect to their underlying algorithmic principle, as shown in Figure 1. Note that among these principles only the meta-search-controlled algorithms pursue a strategy of global optimization; hierarchical, iterative, as well as density-based algorithms are efficient implementations of particular heuristics. In the following we shortly outline the working principles.

*Hierarchical Algorithms.* Hierarchical algorithms create a tree of node subsets by successively subdividing or merging the objects in  $D$ . In order to obtain a unique clustering, a second step is necessary that prunes this tree at adequate places. Agglomerative hierarchical algorithms start with each vertex being its own cluster and union clusters iteratively. For divisive algorithms on the other hand, the entire graph initially forms one single cluster which is successively subdivided. Representatives are  $k$ -nearest-neighbor, linkage, Ward, minimum-spanning-tree, or min-cut methods. Usually, these methods construct a complete similarity matrix, which results in  $\mathcal{O}(n^2)$  runtime [8, 10, 32, 17, 25, 38, 40].

*Iterative Algorithms.* Iterative algorithms strive for a successive improvement of an existing clustering and can be further classified into exemplar-based and commutation-based approaches. The former assume for each cluster a representative, i. e. a centroid (for interval-scaled features) or a medoid (otherwise), to which the objects become assigned according to their similarity. Iterative algorithms need information with regard to the expected cluster number,  $k$ . Well-known representatives are  $k$ -Means,  $k$ -Medoid, Kohonen, Fuzzy- $k$ -Means. The runtime of these methods is  $\mathcal{O}(nkl)$ , where  $l$  designates the number of iterations to achieve convergence [16, 27, 20, 23, 39, 13, 14, 24, 15, 36].

Commutation-based approaches take a random clustering or the outcome of another cluster algorithm as starting point and successively exchange nodes between the clusters until some cluster quality criterion is fulfilled [9, 21, 19].

*Density-based Algorithms.* Density-based algorithms try to separate the set  $D$  into subsets of similar densities. In the ideal case they can determine the cluster number  $k$  automatically and detect clusters of arbitrary shape and size. Representatives are DBSCAN, MajorClust, or Chameleon. The runtime of these algorithms is in magnitude of hierarchical algorithms, i. e.,  $\mathcal{O}(n^2)$ , or even  $\mathcal{O}(n \log(n))$  for low-dimensional data if efficient data structures are employed [34, 7, 18].

*Meta-Search Algorithms.* Meta-search algorithms treat clustering as an optimization problem where a global goal criterion is to be minimized or maximized. Though this approach offers maximum flexibility, there runtime is typically unacceptably high. Meta-search driven cluster detection may

	Cluster approach			
	hierarchical	iterative	density-based	meta-search controlled
<b>Analysis strategy</b>	relative	absolute	relative	absolute
	comparison	comparison	comparison	comparison
	based on two items	based on $k$ items	based on $k$ items	based on all items
<b>Recovery characteristics</b>	irrevocable	revocable	revocable	revocable

**Table 1.** Characterization of cluster algorithms with respect to the number of investigated items (points, clusters) and their ability to recover from suboptimum decisions.

be operationalized by genetic algorithms, simulated annealing, or a two-phase greedy strategy [2, 30, 31, 30, 29, 11, 22, 41].

## 1.2 An Alternative View to Cluster Algorithms

Experience has shown that—with respect to the clustering quality—density-based cluster algorithms outperform hierarchical as well as iterative approaches.<sup>3</sup> By definition, globally optimizing cluster algorithms will produce better clustering results; however, they play only an inferior role in practice:

- (1) their runtime performance is usually unacceptable compared to the other approaches,
- (2) for most clustering problems it is difficult to formally specify a global goal criterion.

Hierarchical cluster algorithms are well suited to detect complex structures in the data; however, they are susceptible to noise. Both properties result from the fact that—based on a pairwise similarity comparison of any two items—nearest neighbors are always fused. Iterative algorithms behave robustly with respect to noise but preferably detect spherical clusters. Density-based cluster algorithms provide a high flexibility with respect to cluster forms and address the problem of noise detection by simultaneously examining several items. Table 1 summarizes the properties.

The following section presents two density-based cluster algorithms in greater detail and discusses their properties.

## 2 Density-based Cluster Analysis with DBSCAN and MajorClust

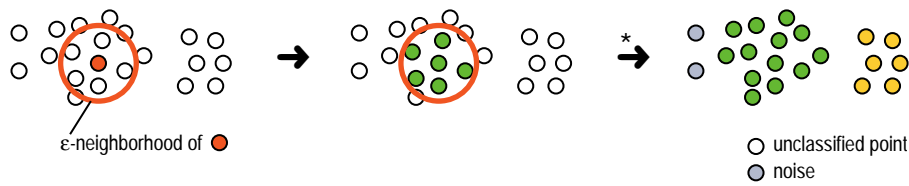
A density-based cluster algorithm operationalizes two mechanisms:

- (1) one to define a region  $R \subseteq D$ , which forms the basis for density analyses;
- (2) another to propagate density information (the provisional cluster label) of  $R$ .

In DBSCAN a region is defined as the set of points that lie in the  $\varepsilon$ -neighborhood of some point  $p$ . Cluster label propagation from  $p$  to the other points in  $R$  happens if  $|R|$  exceeds a given *MinPts*-threshold (cf. Figure 2). The following advantages (+) and disadvantages (–) are bound up with this concept:

- + good clustering results for geometrical and low dimensional data, if cluster distances can be inferred unambiguously from the density information in  $D$ ,

<sup>3</sup> With respect to runtime performance density-based algorithms can be considered being in between hierarchical and iterative approaches.

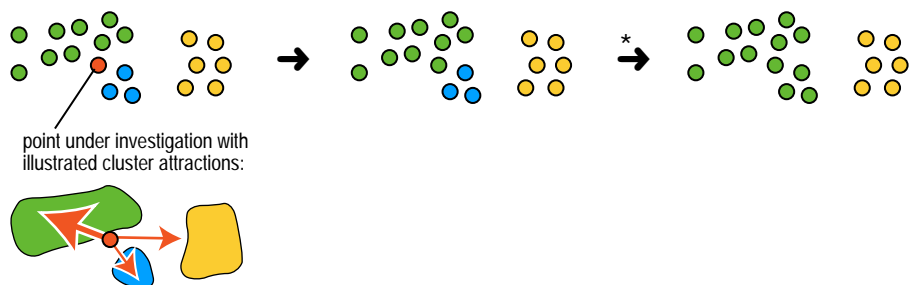


**Figure 2.** Illustration of DBSCAN's cluster process; to obtain the shown clustering result a *MinPts*-value from  $\{3, 4, 5\}$  is required.

- + efficient runtime behavior of  $O(n \log(n))$ , if the *R*-tree data structure is employed to answer region queries for *D*,
- parameters that characterize the density ( $\epsilon$ , *MinPts*) are problem-specific and must be chosen manually,
- if great variations in the point density or unreliable similarity values require a large  $\epsilon$ -neighborhood, the density analysis becomes questionable: DBSCAN implements no intra-region-distance concept, and all points in the  $\epsilon$ -neighborhood of some point are treated equally.
- in high dimensions ( $> 10-20$ ), the underlying *R*-tree data structure degenerates to a linear search and makes an MDS-embedding of *D* necessary, which affects both runtime and classification performance.

In MajorClust a region is not of a fixed size but implicitly defined by the current clustering  $\mathcal{C}$ . While DBSCAN uses the point concentration in  $\epsilon$ -neighborhoods to estimate densities, MajorClust derives its density information from the attraction a cluster  $C$  exerts on some point  $q$ , which is computed as the sum of all similarity values  $\varphi(q, p)$ ,  $p \in C$ . Cluster label propagation from  $C$  to  $q$  happens if the attraction of  $C$  with respect to  $q$  is maximum among the attraction values of all clusters in  $\mathcal{C}$  (cf. Figure 3). Observe that the “true” density values for some data set *D* evolve during the clustering process. The following advantages (+) and disadvantages (–) are bound up with this concept:

- + robust for narrow data sets, for high-dimensional data, and for unreliable similarity values since the density computation does not rely on a fixed number of points and does consider distance information as well,
- + adapts automatically to different problems: no parameters must chosen manually,
- with respect to runtime not so efficient as DBSCAN, since points are re-labeled several times,
- clusters that are extended in one dimension are not reliably identified, since during re-labeling a tie-situation may occur (“tie-effect”).



**Figure 3.** Illustration of MajorClust's clustering process; each cluster  $C$  exerts attraction to some point  $q$  depending on both its size,  $|C|$ , and distance to  $q$ .

The above discussion reveals the strong and weak points of both algorithms, and the experiments presented in Section 3 will illustrate this behavior at realistic cluster problems. The next two subsections give a pseudo-code specification of both algorithms.

## 2.1 DBSCAN

DBSCAN operationalizes density propagation according to the principle “accessibility from a core point”: Each point whose  $\varepsilon$ -neighborhood contains more points than  $MinPts$  is called a core point. Each point which lies in an  $\varepsilon$ -neighborhood of a core point  $p$  adopts the same cluster label as  $p$ . The DBSCAN-algorithm propagates this relation through the set  $D$ . The algorithm terminates if each point is either assigned to a certain cluster or classified as noise.

---

Algorithm DBSCAN.

Input: object set  $D$ , region radius  $\varepsilon$ , density threshold  $MinPts$ .

Output: function  $\gamma : D \rightarrow \mathbf{N}$ , which assigns a cluster label to each point.

---

```
(01) Label := 1
(02)  $\forall p \in D$  do  $\gamma(p) := 'UNCLASSIFIED'$  enddo
(03)  $\forall p \in D$  do
(04)   if  $\gamma(p) = 'UNCLASSIFIED'$  then
(05)     if  $ExpandCluster(D, p, Label, \varepsilon, MinPts) = true$  then Label := Label + 1
(06)   endif
(07) enddo
```

---

Function ExpandCluster.

Input: object set  $D$ ,  $CurrentPoint$ ,  $Label$ ,  $\varepsilon$ ,  $MinPts$ .

Output:  $true$  or  $false$ .

---

```
(01) seeds := regionQuery(D, CurrentPoint,  $\varepsilon$ )
(02) if |seeds| < MinPts then
(03)    $\gamma(CurrentPoint) := 'NOISE'$ 
(04)   return false
(05) else
(06)    $\forall p \in P$  do  $\gamma(p) := Label$  enddo
(07)   seeds := seeds \ {CurrentPoint}
(08)   while seeds  $\neq \emptyset$  do
(09)     p := seeds.first()
(10)     result := regionQuery(D, p,  $\varepsilon$ )
(11)     if |result|  $\geq MinPts$  then
(12)        $\forall ResPoint \in P$  do
(13)         if  $\gamma(ResPoint) \in \{'UNCLASSIFIED', 'NOISE'\}$  then
(14)           if  $\gamma(ResPoint) = 'UNCLASSIFIED'$  then seeds := seeds  $\cup \{ResPoint\}$ 
(15)            $\gamma(ResPoint) := Label$ 
(16)         endif
(17)       enddo
(18)     endif
(19)     seeds := seeds \ {p}
(20)   enddo
(21)   return true
(22) endif
```

---

*Remarks.* The function  $regionQuery(D, p, \varepsilon)$  returns all points in the  $\varepsilon$ -neighborhood of some point  $p$ .

## 2.2 MajorClust

MajorClust operationalizes density propagation according to the principle “maximum attraction wins”: The algorithm starts by assigning each point in  $D$  its own cluster. Within the following re-labeling steps, a point adopts the same cluster label as the majority of its weighted neighbors. If several such clusters exist, one of them is chosen randomly. The algorithm terminates if no point changes its cluster membership.

---

Algorithm MajorClust.

Input: object set  $D$ , similarity measure  $\varphi : D \times D \rightarrow [0; 1]$ , similarity threshold  $t$ .

Output: function  $\gamma : D \rightarrow \mathbf{N}$ , which assigns a cluster label to each point.

---

```

(01)  $i := 0, ready := false$ 
(02)  $\forall p \in D$  do  $i := i + 1, \gamma(p) := i$  enddo
(03) while  $ready = false$  do
(04)    $ready := true$ 
(05)    $\forall q \in D$  do
(06)      $\gamma^* := i$  if  $\sum \{\varphi(p, q) \mid \varphi(p, q) \geq t \text{ and } \gamma(p) = i\}$  is maximum.
(07)     if  $\gamma(q) \neq \gamma^*$  then  $\gamma(q) := \gamma^*, ready := false$ 
(08)   enddo
(09) enddo

```

---

*Remarks.* The similarity threshold  $t$  is not a problem-specific parameter but a constant that serves for noise filtering purposes. Its typical value is 0.3.

## 3 Illustrative Analysis

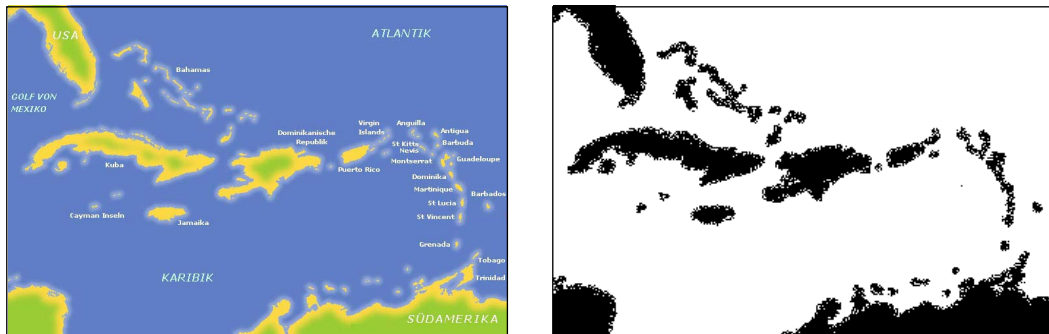
This section presents selected results from an experimental analyses of the algorithms DBSCAN and MajorClust. Further details and additional background information can be found in [4].

### 3.1 A Low-Dimensional Application: Analysis of Geometrical Data

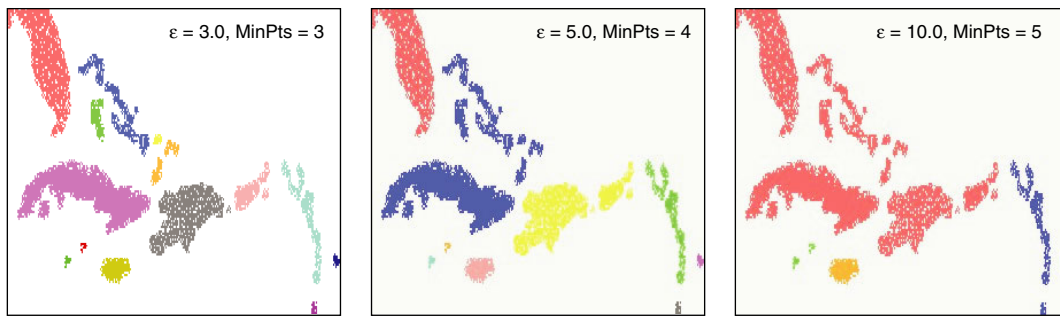
The left-hand side of Figure 4 shows a map of the Caribbean Islands, the right-hand side shows a monochrome and dithered version (approx. 20,000 points) of this map, which forms the basis of the following cluster experiments.

A cluster analysis with DBSCAN requires useful settings for  $\varepsilon$  and  $MinPts$ . Figure 5 shows the clustering results with selected values for these parameters. Note that in this application the quality of the resulting clustering was more sensitive with respect to  $\varepsilon$  than to  $MinPts$ .<sup>4</sup>

<sup>4</sup> Ester et al. propose a heuristic to determine adequate settings for  $\varepsilon$  and  $MinPts$ ; however, the heuristic is feasible only for two-dimensional data [7].



**Figure 4.** Map of the Carribeian Islands (left) and its dithered version (right).



**Figure 5.** DBSCAN-clusterings of the Caribbean Islands for selected parameter settings.

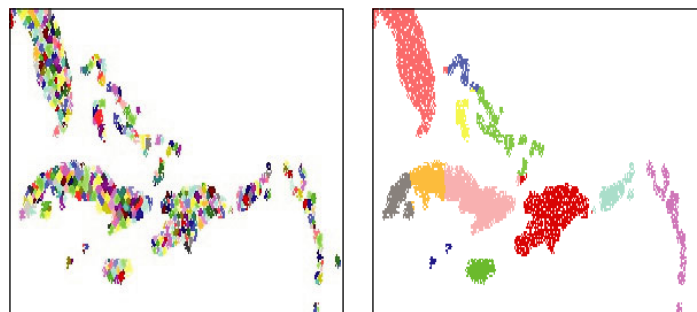
A cluster analysis with MajorClust does not require the adjustment of special parameters. However, to alleviate noise effects the algorithm should apply a similarity threshold of (about) 0.3, i. e., discard all similarity values that are below this threshold. Figure 6 shows an intermediate clustering (left) and the resulting final clustering (right); obviously not all islands were correctly identified—a fact for which the formerly explained “tie-effect” is responsible.

### 3.2 A High-Dimensional Application: Document Categorization

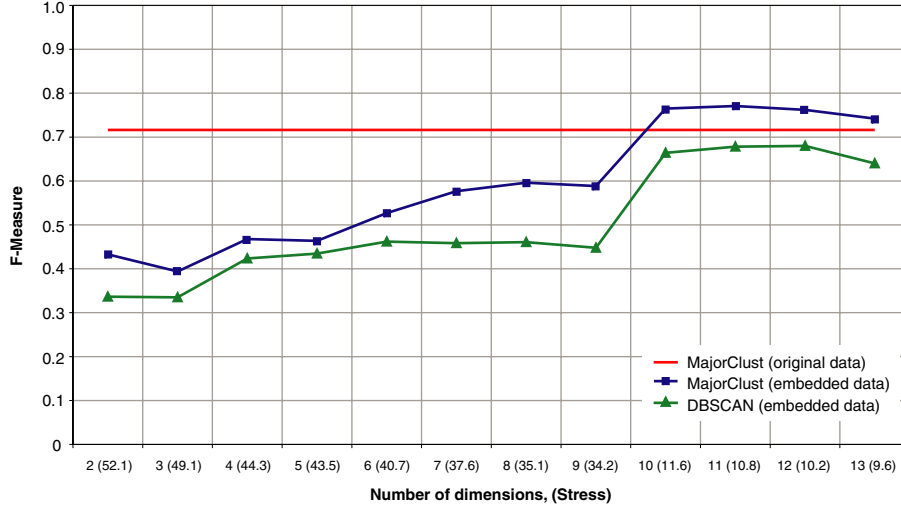
Cluster technology has come into focus in recent years, because it forms the backbone of most document categorization applications [33]. At the moment it is hard to say which of the approaches shown in Figure 1 will do this job best:  $k$ -means and bisecting  $k$ -means are used because of their robustness, the group-average approach produces similar or even better clustering results but is much less efficient, and from the density-based approaches the MajorClust-algorithm has repeatedly proved its high classification performance especially for short documents [1].

In this place we will report on experiments that rely on the Reuters-21578 corpus, which comprises texts from politics, economics, culture, etc. that have been carefully assigned to approx. 100 (sub-) categories by human editors [26]. For our experiments we constructed sub-corpora with 10 categories, each consisting of 100 documents that belong exclusively to a single category. The selected documents were transferred into the vector space model (VSM); the necessary pre-processing includes stop-word filtering, stemming, and the computation of term weights according to the  $tf$ - $idf$ -scheme. Note that the resulting word vectors represent points in a space with more than 10,000 dimensions.

Since DBSCAN relies on the  $R$ -tree data structure it cannot process high-dimensional data (see Subsection 3.3), and multi dimensional scaling (MDS) had to be employed to embed the data into a



**Figure 6.** Left: Clustering after the first iterations of MajorClust; right: the resulting clustering.



**Figure 7.** Classification performance of the algorithms in the document categorization task.

low-dimensional space. To account for effects that result from embedding distortions, MajorClust was applied to both the high-dimensional and the embedded data. The quality of the generated clusterings was quantified by the  $F$ -measure, which combines the achieved precision- and recall-values relative to the 10 classes.<sup>5</sup>

Figure 7 shows the achieved classification performance: The  $x$ -axis indicates the number of dimensions of the embedding space (from 2 to 13), the  $y$ -axis indicates the  $F$ -measure value. The horizontal line with an  $F$ -measure value of 0.72 belongs to MajorClust when applied to the original, high-dimensional data; with respect to the embedded data MajorClust dominates DBSCAN—independent of the dimension. In particular, it should be noted that the shown (high)  $F$ -measure values for DBSCAN are the result of extensive experimenting with various parameter settings.

Observe in Figure 7 that embedding can actually improve classification performance: If the number of dimensions in the embedding space equals the cluster number or is slightly higher, the noise-reduction effect compensates for the information loss due to the embedding error.<sup>6</sup> Put another way: Each dimension models a particular, hidden *concept*. This effect can be utilized for retrieval purposes and became known as *Latent Semantic Indexing* [6].

### 3.3 A Note on Runtime Performance

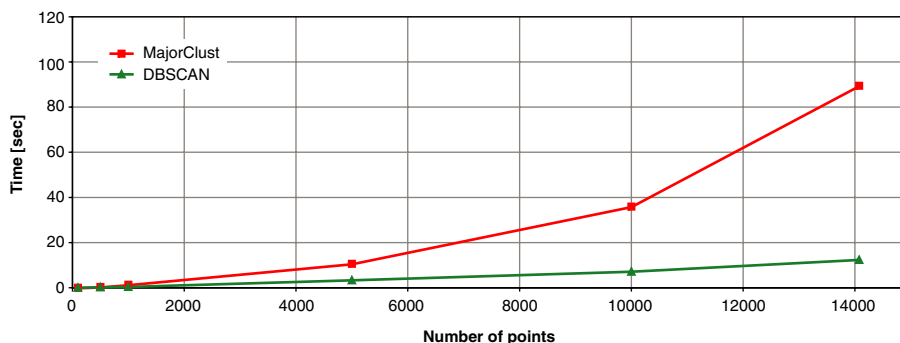
Both algorithms implement a density propagation heuristic that analyzes a region  $R$  with  $k$  contiguous points (recall Table 1). Based on  $R$ , DBSCAN performs a simple point count, while MajorClust evaluates the similarities between all points in  $R$  and some point  $q$ . Given an efficient means to construct the region  $R$  as  $\varepsilon$ -neighborhood of a point  $p$ , the runtime of both algorithms is in the same order of magnitude (cf. Figure 8).

To answer region queries, the  $R$ -tree data structure was employed in the above experiments [12]. For low-dimensional data, say,  $m < 10$ , this data structure finds the  $\varepsilon$ -neighborhood for some  $p$  in  $O(\log(n))$ , with  $n = |D|$ . As a consequence, the runtime of both algorithms is in  $O(n \log(n))$ . Since

<sup>5</sup> An  $F$ -measure value of 1 indicates a perfect match; however,  $F$ -measure values  $> 0.7$  must be considered as certainly good since we are given a multi-class assignment situation where 10 classes are to be matched.

<sup>6</sup> The embedding error is called “stress” in the literature on the subject.





**Figure 8.** Runtime-behavior of the algorithms on two-dimensional data.

MajorClust evaluates the attraction of each point several times, its runtime is a factor above the runtime of DBSCAN.

Note that the  $R$ -tree data structure degenerates for higher dimensions ( $m > 100$ ) and definitely fails to handle the high-dimensional vector space model.<sup>7</sup> I. e., a high-dimensional cluster analysis task like document categorization cannot directly be tackled with DBSCAN but requires a preceding data embedding step. At the moment the fastest embedding technology is the MDS-variant described in [28], which has not been tested for high-dimensional document models yet.

## Summary and Current Work

This paper presented the classical and a new view to cluster technology and then delved into a comparison of the density-based cluster algorithms DBSCAN and MajorClust. This comparison is the first of its kind and our discussion as well as the experiments are interesting for the following reasons:

- (1) Density-based cluster analysis should be considered as a collective term for (heuristic) approaches that quantify and propagate density information. In particular, no uniform statements regarding runtime-behavior or suited problem classes can be made.
- (2) Since strengths and weaknesses of (density-based) cluster algorithms can be explained with the dimensionality of the data, a better mapping from algorithms to cluster problems may be developed.

Our current work concentrates on runtime issues of density-based cluster analysis for high-dimensional data: We investigate how the technology of Fuzzy fingerprints can be utilized to speed-up the region query task; the key challenge in this connection is to handle the inherent incompleteness of this retrieval technology.

## References

- [1] Mikhail Alexandrov, Alexander F. Gelbukh, and Paolo Rosso. An approach to clustering abstracts. In *NLDB*, pages 275–285, 2005.
- [2] Thomas Bailey and John Cowles. Cluster Definition by the Optimization of Simple Measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, September 1983.
- [3] J. C. Bezdek and N. R. Pal. Cluster Validation with Generalized Dunn’s Indices. In N. Kasabov and G. Coghill, editors, *Proceedings of the 2nd international two-stream conference on ANNES*, pages 190–193, Piscataway, NJ, 1995. IEEE Press.

<sup>7</sup> This relates also to other data-partitioning index trees such as Rf-tree or X-tree, as well as to space-partitioning methods like grid-files, KD-trees, or quad-trees [37].

- [4] Michael Busch. Analyse dichtebasierter Clusteralgorithmen am Beispiel von DBSCAN und MajorClust. Study work, Paderborn University, Institut for Computer Science, March 2005.
- [5] D.L. Davies and D.W. Bouldin. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Learning*, 1(2), 1979.
- [6] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.
- [7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD96)*, 1996.
- [8] B. S. Everitt. Cluster analysis. New York, Toronto, 1993.
- [9] C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *Proceedings of the nineteenth design automation conference*, pages 175–181. IEEE Press, 1982. ISBN 0-89791-020-6.
- [10] K. Florek, J. Lukaszewicz, J. Perkal, H. Steinhaus, and S. Zubrzycki. Sur la liaison et la division des points d'un ensemble fini. *Colloquium Mathematicum*, 2, 1951.
- [11] D. B. Fogel and L. J. Fogel. Special Issue on Evolutionary Computation. *IEEE Transaction of Neural Networks*, 1994.
- [12] Antonin Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In Beatrice Yormark, editor, *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984*, pages 47–57. ACM Press, 1984.
- [13] Eui-Hong Han and George Karypis. Centroid-Based Document Classification: Analysis and Experimental Results. Technical Report 00-017, University of Minnesota, Department of Computer Science / Army HPC Research Center, March 2000.
- [14] Eui-Hong Han, George Karypis, and Vipin Kumar. Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 53–65, Minneapolis, USA, 2001. University of Minnesota, Department of Computer Science / Army HPC Research Center.
- [15] Makoto Iwayama and Takenobu Tokunaga. Cluster-based text categorization: a comparison of category search strategies. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval*, pages 273–281, Seattle, USA, 1995. ACM Press, New York, US.
- [16] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering in Data*. Prentice Hall, Englewood Cliffs, NJ, 1990. ISBN 0-13-022278-X.
- [17] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32, 1967.
- [18] G. Karypis, E.-H. Han, and V. Kumar. Chameleon: A hierarchical clustering algorithm using dynamic modeling. Technical Report Paper No. 432, University of Minnesota, Minneapolis, 1999.
- [19] George Karypis and Vipin Kumar. Analysis of multilevel graph partitioning. In *Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM)*, page 29. ACM Press, 1995. ISBN 0-89791-816-9. doi: <http://doi.acm.org/10.1145/224170.224229>.
- [20] Leonard Kaufman and Peter J. Rousseuw. *Finding Groups in Data*. Wiley, 1990.
- [21] B.W. Kernighan and S. Lin. Partitioning Graphs. *Bell Laboratories Record*, January 1970.
- [22] R. W. Klein and R. C. Dubes. Experiments in Projection and Clustering by Simulated Annealing. *Pattern Recognition*, 22:213–220, 1989.
- [23] T. Kohonen. *Self Organization and Assoziative Memory*. Springer, 1990.
- [24] T. Kohonen, S. Kaski, K. Lagus, J. Salojrvi, J. Honkela, V. Paatero, and A. Saarela. Self organization of a massive document collection. In *IEEE Transactions on Neural Networks*, volume 11, may 2000.
- [25] Thomas Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout*. John Wiley & Sons, New York, 1990.

- [26] David D. Lewis. Reuters-21578 Text Categorization Test Collection. <http://www.research.att.com/~lewis>, 1994.
- [27] J. B. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.
- [28] Alistair Morrison, Greg Ross, and Matthew Chalmers. Fast Multidimensional Scaling through Sampling, Springs and Interpolation. *Information Visualization*, 2(1):68–77, 2003.
- [29] V. V. Raghavan and K. Birchand. A Clustering Strategy Based on a Formalism of the Reproduction Process in a Natural System. In *Proceedings of the Second International Conference on Information Storage and Retrieval*, pages 10–22, 1979.
- [30] Tom Roxborough and Arunabha. Graph Clustering using Multiway Ratio Cut. In Stephen North, editor, *Graph Drawing*, Lecture Notes in Computer Science, Springer, 1996.
- [31] Reinhard Sablowski and Arne Frick. Automatic Graph Clustering. In Stephan North, editor, *Graph Drawing*, Lecture Notes in Computer Science, Springer, 1996.
- [32] P. H. A. Sneath. The application of computers to taxonomy. *J. Gen. Microbiol.*, 17, 1957.
- [33] Benno Stein and Sven Meyer zu Eißén. Document Categorization with MAJORCLUST. In Amit Basu and Soumitra Dutta, editors, *Proceedings of the 12th Workshop on Information Technology and Systems (WITS 02), Barcelona Spain*, pages 91–96. Technical University of Barcelona, December 2002.
- [34] Benno Stein and Oliver Niggemann. On the Nature of Structure and its Identification. In Peter Widmayer, Gabriele Neyer, and Stefan Eidenbenz, editors, *Graph-Theoretic Concepts in Computer Science*, volume 1665 LNCS of *Lecture Notes in Computer Science*, pages 122–134. Springer, June 1999. ISBN 3-540-66731-8.
- [35] Benno Stein, Sven Meyer zu Eißén, and Frank Wißbrock. On Cluster Validity and the Information Need of Users. In M. H. Hanza, editor, *Proceedings of the 3rd IASTED International Conference on Artificial Intelligence and Applications (AIA 03), Benalmádena, Spain*, pages 216–221, Anaheim, Calgary, Zurich, September 2003. ACTA Press. ISBN 0-88986-390-3.
- [36] Michael Steinbach, George Karypis, and Vipin Kumar. A comparison of document clustering techniques. Technical Report 00-034, Department of Computer Science and Engineering, University of Minnesota, 2000.
- [37] Roger Weber, Hans-J. Schek, and Stephen Blott. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In *Proceedings of the 24th VLDB Conference New York, USA*, pages 194–205, 1998.
- [38] Zhenyu Wu and Richard Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, November 1993.
- [39] J. T. Yan and P. Y. Hsiao. A fuzzy clustering algorithm for graph bisection. *Information Processing Letters*, 52, 1994.
- [40] C. T. Zahn. Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters. *IEEE Transactions on computers*, C-20(1), 1971.
- [41] Ying Zaho and George Karypis. Criterion Functions for Document Clustering: Experiments and Analysis. Technical Report 01-40, University of Minnesota, Department of Computer Science / Army HPC Research Center, Feb 2002.