Density-Based Clustering of Uncertain Data

Hans-Peter Kriegel
University of Munich, Germany
Institute for Computer Science
kriegel@dbs.ifi.lmu.de

Martin Pfeifle
University of Munich, Germany
Institute for Computer Science
pfeifle@dbs.ifi.lmu.de

ABSTRACT

In many different application areas, e.g. sensor databases, location based services or face recognition systems, distances between objects have to be computed based on vague and uncertain data. Commonly, the distances between these uncertain object descriptions are expressed by one numerical distance value. Based on such single-valued distance functions standard data mining algorithms can work without any changes. In this paper, we propose to express the similarity between two fuzzy objects by distance probability functions. These fuzzy distance functions assign a probability value to each possible distance value. By integrating these fuzzy distance functions directly into data mining algorithms, the full information provided by these functions is exploited. In order to demonstrate the benefits of this general approach, we enhance the density-based clustering algorithm DBSCAN so that it can work directly on these fuzzy distance functions. In a detailed experimental evaluation based on artificial and real-world data sets, we show the characteristics and benefits of our new approach.

Categories and Subject Descriptors

G.3 [Probability and Statistics]: Probabilistic algorithms (including Monte Carlo).

General Terms

Algorithms

Keywords

density-based clustering, uncertain data, fuzzy distance functions.

1. INTRODUCTION

In many modern application ranges, e.g. the clustering of moving objects [12] or sensor databases [2], only uncertain data is available. For instance, in the area of mobile services, the objects continuously change their positions so that exact positional information is often not available. In other application areas such as the clustering of distributed feature vectors [6, 9], due to security aspects or to limited bandwidth, only approximated information is transmitted to a central server site.

KDD'05, August 21-24, 2005, Chicago, Illinois, USA.

In order to extract knowledge from these fuzzy object descriptions by means of standard data mining algorithms, the similarity between the objects has to be measured by one numerical value, i.e. the complete fuzzy distance information is aggregated by only one distance value. Obviously, aggregation goes hand in hand with information loss. For instance, we have no information about the degree of uncertainty of such a single distance value. Even if we had one, it would be of no use because traditional data mining algorithms, e.g. clustering algorithms, cannot handle this additional information.

In this paper, we propose to use fuzzy distance functions to measure the similarity between fuzzy objects. Contrary to the traditional approaches, we do not extract aggregated values from the fuzzy distance functions but propose to enhance the data mining algorithms so that they can exploit the full information provided by these functions. As for many important application ranges where fuzzy distance functions naturally occur, e.g. the clustering of moving objects, density-based clustering algorithms seem to be the method of choice [13], we demonstrate in this paper how fuzzy distance functions can be integrated into the density-based clustering algorithm DBSCAN [3]. We call the resulting clustering algorithms $\mathcal{F}$DBSCAN indicating that it is applicable to cluster fuzzy objects.

The remainder of this paper is organized as follows. In Section 2, we present the related work in the area of density-based clustering of uncertain data. In Section 3, we introduce fuzzy distance functions. In Section 4, we show how we can integrate these functions into the density-based clustering algorithm DBSCAN. In Section 5, we will experimentally show the benefit of our new fuzzy clustering algorithm $\mathcal{F}$DBSCAN. We will close this paper, in Section 6, with a short summary and a few remarks on future work.

2. RELATED WORK

Given a set of objects with a distance function on them, an interesting data mining question is, whether these objects naturally form groups (called clusters) and what these groups look like. Data mining algorithms that try to answer this question are called *clustering algorithms*. In Section 2.1, we shortly classify clustering algorithms according to different categorization schemes. Then, in Section 2.2, we present the basic concepts of fuzzy clustering algorithms, and describe how the approach of this paper differs from the fuzzy clustering approaches presented in the literature. In Section 2.3, we present the density-based clustering algorithm DBSCAN in a level of detail which is indispensable to understand the remainder of this paper. As fuzzy objects can also be regarded as multi-represented objects, we will finally present a density-based clustering approach which is suitable for clustering multi-represented objects.

2.1 Clustering Algorithms

Clustering algorithms can be classified along different, independent dimensions. One well-known dimension categorizes clustering methods according to the *result* they produce. Here, we can distinguish between *hierarchical* and *partitioning clustering* algorithms [7]. Partitioning algorithms construct a flat (single level) partition of a database D of n objects into a set of k clusters such that the objects in a cluster are more similar to each other than to objects in different clusters. Another dimension according to which we can classify clustering algorithms is from an *algorithmic* point of view. Here we can distinguish between *optimization-based* or *distance-based* algorithms and *density-based* algorithms. Density-based algorithms apply a local cluster criterion. Clusters are regarded as regions in the data space in which the objects are dense, and which are separated by regions of low object density (noise). In this paper, we will present an extension for the partitioning density-based clustering algorithm DBSCAN [3]. For a more detailed general overview on clustering algorithms, we refer the interested reader to [7].

2.2 Fuzzy Clustering

In real applications there is very often no sharp boundary between clusters so that fuzzy clustering is often better suited for the data. Membership degrees between zero and one are used in fuzzy clustering instead of crisp assignments of the data to clusters. In contrast to fuzzy clustering algorithms where objects are assigned to different clusters, in this paper, we cluster fuzzy object representations and assign each fuzzy object to exactly one cluster. For more details about fuzzy clustering algorithms, we refer the reader to [5].

2.3 Density-based Clustering

The key idea of density-based clustering is that for each object of a cluster the neighborhood of a given radius ε has to contain at least a minimum number of μ objects, i.e. the cardinality of the neighborhood has to exceed a given threshold. In the following, we will present the basic definitions of density-based clustering.

Definition 1 *Core Object*

Object o is called a *core object* w.r.t. ε and μ in a set of objects D, if $|N_\varepsilon(o)| \geq \mu$, where $N_\varepsilon(o)$ denotes the subset of D contained in the ε-neighborhood of o.

Definition 2 *Directly Density-Reachable*

Object p is *directly density-reachable* from object o w.r.t. ε and μ in a set of objects D, if o is a core object and $p \in N_\varepsilon(o)$, where again $N_\varepsilon(o)$ denotes the subset of D contained in the ε-neighborhood of o.

Note that objects can be directly density-reachable only from core objects.

Definition 3 *Density-Reachable*, *Density-Connected*

An object p is *density-reachable* from an object o w.r.t. ε and μ in the set of objects D, if there is a chain of objects $p_1, \ldots, p_n$, $p_1 = o$, $p_n = p$ such that $p_i \in D$ and p_{i+1} is directly density-reachable from p_i w.r.t. ε and μ. Object p is *density-connected* to object q w.r.t. ε and μ in the set of objects D, if there is an object $o \in D$ such that both p and q are density-reachable from o w.r.t. ε and μ in D.

Density-reachability is the transitive closure of direct density-reachability and does not have to be symmetric. On the other hand, density-connectivity is a symmetric relation.

DBSCAN. A flat density-based cluster is defined as a set of density-connected objects which is maximal w.r.t. density-reachability. Then the noise is the set of objects not contained in any cluster. Thus a cluster contains not only core objects but also objects that do not satisfy the core object condition. These border objects are directly density-reachable from at least one core object of the cluster.

The algorithm DBSCAN [3], which discovers the clusters and the noise in a database, is based on the fact that a cluster is equivalent to the set of all objects in D which are density-reachable from an arbitrary core object in the cluster (cf. lemma 1 and 2 in [3]). The retrieval of density-reachable objects is performed by iteratively collecting directly density-reachable objects. DBSCAN checks the ε-neighborhood of each point in the database. If the ε-neighborhood $N_\varepsilon(o)$ of a point o has more than μ elements, o is a so-called *core point*, and a new cluster C containing the objects in $N_\varepsilon(o)$ is created. Then, the ε-neighborhood of all points p in C which have not yet been processed is checked. If $N_\varepsilon(p)$ contains more than μ points, the neighbors of p which are not already contained in C are added to the cluster and their ε-neighborhood is checked in the next step. This procedure is repeated until no new point can be added to the current cluster C. Then the algorithm continues with a point which has not yet been processed trying to expand a new cluster.

2.4 Clustering of Multi-Represented Objects

In many different application ranges, several representations for each object exist, e.g. molecules are characterized by an amino acid sequence, a secondary structure and a 3D representation. Fuzzy objects (cf. Definition 4) can also be regarded as multi-represented objects. In [10] a density-based approach for clustering such multi-represented objects was proposed which is based on DBSCAN. To determine a clustering which takes all representations into account, the basic definitions of DBSCAN, i.e. the core object definition and the reachability definition, are extended. Thereby, the ε-neighborhoods of each representation are combined to a global neighborhood. For sparse data sets, the *union* method was proposed which assumes that an object is a core object. if μ objects are found within the union of all ε-neighborhoods of all representations. Furthermore, the *intersection* method was introduced where an object is a core object, if at least μ objects are within the intersection of all ε-neighborhoods of all representations. In our experimental evaluation, we will use the approach presented in [10] as comparison partner. As a side effect of this paper, it becomes clear that a slight adaption of the $\mathcal{F}$DBSCAN algorithm would be much more suitable for clustering multi-represented objects than the approaches introduced in [10].

3. FUZZY OBJECT SIMILARITY

In this section, we introduce the concept of fuzzy object representations along with suitable similarity measures, i.e. distance functions, between these vague object representations. Based on these distance functions, we present in the following section the clustering algorithm $\mathcal{F}$DBSCAN.

3.1 Fuzzy Object Representations

In many different application areas, e.g. the clustering of moving objects [12] or distributed clustering [6], an object is described not only by one single feature vector, but by a region in which all points within the region equally likely represent the object. In [12], for in-

stance, an object is located somewhere within a moving micro-cluster represented by a rectangle, and in [6] an object is located somewhere in a hyper-sphere. In [2] 1-dimensional probability density functions (pdf) are used to describe attributes of uncertain sensor data. We extend this approach and propose to describe an object no longer by one single feature vector, but by a probability density function indicating the likelihood that an object is located at a certain position.

Definition 4 *Fuzzy Object Representation*
Let $o \in D \subseteq IR^d$ be an object from a database. A *fuzzy object representation* is a function o_{fuzzy}: $IR^d \rightarrow IR_0^+ \cup \infty$, for which the following condition holds:

$$\int_{IR^d} o_{fuzzy}(v)dv = 1$$

3.2 Distance Functions between Fuzzy Objects

Traditional data mining algorithms require distance functions which express the similarity between two objects by exactly one numerical value. In this section, we introduce distance functions which do not express the similarity between two objects by a single numerical value. Instead, we propose to use *fuzzy distance functions,* where the similarity between two fuzzy objects is expressed by means of a probability function which assigns a numerical value to each distance value. Two fuzzy distance functions are the *distance density function* and the *distance distribution function.*

Definition 5 *Distance Density Function*
Let d: $D \times D \rightarrow IR_0^+$ be a distance function, and let $P(a \le d(o, o') \le b)$ denote the probability that $d(o,o')$ is between a and b. Then a probability density function p_d: $D \times D \rightarrow (IR_0^+ \rightarrow IR_0^+ \cup \infty)$ is called a *distance density function* if the following condition holds:

$$P(a \le d(o, o') \le b) = \int_a^b p_d(o, o')(x)dx$$

If the distance $\tau = d(o,o')$ between two objects can exactly be determined, the probability density function p_d is equal to the dirac-delta function δ, i.e. $p_d(o, o')(x) = \delta(x-\tau)$. For arbitrary functions f, e.g. $f(x) = 1$, the dirac-delta function has the following important property:

$$\int_a^b f(x)\delta(x-\tau)dx = \begin{cases} f(\tau), & if(a \le \tau \le b) \\ 0, & otherwise \end{cases} \quad [1]$$

Similar to distance density functions, we can define distance distribution functions.

Definition 6 *Distance Distribution Function*
Let d: $D \times D \rightarrow IR_0^+$ be a distance function, and let $P(d(o, o') \le b)$ denote the probability that $d(o, o')$ is smaller than b. Then a probability distribution function P_d: $O \times O \rightarrow (IR_0^+ \rightarrow [0..1])$ is called a *distance distribution function* if the following condition holds:

$$P_d(o, o')(b) = P(d(o, o') \le b)$$

Let us note that $P_d(o, o')(b) = \int_{-\infty}^b p_d(o, o')(x)dx$ holds, and that therefore p_d and P_d contain basically the same information.

As already mentioned, traditional algorithms can only handle distance functions which yield a unique distance value. In order to make our fuzzy distance functions useful for standard (clustering) algorithms, we could extract an aggregated value of them. For instance, we could use the distance expectation value E_d: $O \times O \rightarrow IR_0^+$ which can be regarded as the average distance between the fuzzy objects, i.e. $E_d(o, o') = \int_{-\infty}^{\infty} x \cdot p_d(o, o')(x)dx$. Note that clustering based on the distance expectation values E_d seems to be more appropriate than using the centroids of the fuzzy object representations for clustering [9]. For instance, think of situations where the centroids are close to each other, but due to a rather high fuzziness of the objects, the distance expectation values indicate a rather high distance between the objects. In this case, where it is not very likely that the objects form a cluster, the centroid approach would wrongly detect clusters and the expectation approach would correctly detect no clusters.

4. $\mathcal{F}$DBSCAN

In this section, we will describe our extended clustering algorithm $\mathcal{F}$DBSCAN which does not rely on lossy aggregated information but exploits the complete information provided by the fuzzy distance functions. We first present the formal definitions underlying the $\mathcal{F}$DBSCAN algorithm (cf. Section 4.1) before we look at computational aspects (cf. Section 4.2).

4.1 Theoretical Foundations

The algorithm $\mathcal{F}$DBSCAN is based on an enhanced version of the core object definition (cf. Definition 1). The *core object probability* of an object o indicates the likelihood that o is a core object.

Definition 7 *Core Object Probability*
Let D be a database, and let P_d: $D \times D \rightarrow (IR_0^+ \rightarrow [0..1])$ be a distance distribution function. Then, the *core object probability* of an object o is defined as:

$$P^{core}_{\varepsilon, \mu, d, D}(o) = \sum_{\substack{A \subseteq D \\ |A| \ge \mu}} \prod_{p \in A} P_d(p, o)(\varepsilon) \prod_{p' \in D \setminus A} (1 - P_d(p', o)(\varepsilon))$$

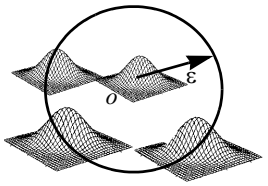
Lemma 1. The core object probability $P^{core}_{\varepsilon, \mu, d, D}(o)$ is equal to the probability value $P(|N_\varepsilon(o)| \ge \mu)$ indicating the likelihood that o is a core object.

Proof. In Definition 7, we determine for each subset A of D having a cardinality higher than μ the probability that only the points of A are within an ε-range of o but no points of $D \setminus A$. The sum of all these probability values indicates the probability that o is a core object, i.e. $P^{core}_{\varepsilon, \mu, d, D}(o) = P(|N_\varepsilon(o)| \ge \mu)$. □

Note that the traditional definition of a core object can also be regarded as a function which assigns to each object o a value equal to 1 iff o is a core object, and 0 otherwise. If the distance distribution function P_d yields only values 0 and 1 at position ε, the traditional and the probability definition of a core object coincide.

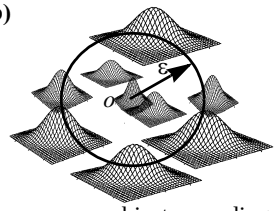
Figure 1 shows how our probability definition of a core object differs from the "traditional" approach where the similarity between fuzzy objects is measured by their distance expectation values. Although, the object o in Figure 1a does not seem to be located in a very dense area it is a core object according to the traditional approach as the distance expectation value between o and $\mu = 4$ other objects is smaller than ε. On the other hand, it is very unlikely that all μ objects are indeed located in $N_\varepsilon(o)$. Therefore, the probability that o is a core object is very small. In Figure 1b the reverse situation

a)



- core object according to the traditional approach based on the distance expectation values E_d.
- very *unlikely* a core object according to the probability approach of Definition 7.

b)



- no core object according to the traditional approach based on the distance expectation values E_d.
- very *likely* a core object according to the probability approach of Definition 7.

Figure 1: Determination of core-point property (μ = 4).

is sketched. Object *o* is located in a very dense area but there do not exist μ objects *p* for which $E_d(o, p) \leq \mu$ holds. Therefore, *o* is no core object according to the traditional approach, although it is very likely that there exist μ elements *p* for which $d(o, p) \leq \varepsilon$ holds[1].

Based on the core object probability definition, we can define how likely it is that an object *p* is directly density reachable from an object *o*. In the traditional approach, two conditions have to hold. First, *o* has to be a core object, and, second, the distance between *p* and *o* has to be smaller than ε. In the context of this paper, both of these conditions are fuzzy holding only with a certain probability.

Definition 8 *Reachability Probability*

Let *D* be a database, and let P_d: $D \times D \rightarrow (IR_0^+ \rightarrow [0..1])$ be a distance distribution function. Then, the reachability probability of *p* w.r.t. *o* is defined as follows:

$$P^{reach}_{\varepsilon, \mu, d, D}(p, o) = P^{core}_{\varepsilon, \mu-1, d, D\setminus\{p\}}(o) \cdot P_d(p, o)(\varepsilon)$$

Lemma 2. $P^{reach}_{\varepsilon, \mu, d, D}(p, o)$ reflects the probability that *p* is directly density-reachable from *o*.

Proof. According to Lemma 1, the probability that at least μ-1 objects from $D\setminus p$ are located in $N_\varepsilon(o)$ is equal to $P^{core}_{\varepsilon, \mu-1, d, D\setminus\{p\}}(o)$. Second, the probability that the distance between *p* and *o* is smaller than ε is equal to $P_d(p, o)(\varepsilon)$. As these two conditions are independent from each other, their product corresponds to the probability that at least μ objects from *D* are located in $N_\varepsilon(o)$ and that *p* is one of them. Note that this value reflects the probability that *p* is directly density-reachable from *o*. □

Definition 8 can be regarded as an extension of the traditional approach. It coincides with the traditional approach, if we assume that the core object probability is always 0 or 1, and the distance distribution function P_d yields only values 0 and 1 at position ε.

4.2 Computational Aspects

The traditional DBSCAN algorithm clusters a data set by always adding objects to the current cluster which are directly density reachable from the current query object *o*. The fuzzy version FDBSCAN works very similar to the traditional approach. An object *p* is added to the current cluster, if the value $P^{reach}_{\varepsilon, \mu, d, D}(p, o)$ exceeds 0.5 where *o* is the current query object. Note that if $P^{core}_{\varepsilon, \mu-1, d, D\setminus\{p\}}(o) < 0.5$ holds, for no object *p* the value $P^{reach}_{\varepsilon, \mu, d, D}(p, o)$ can exceed 0.5. Therefore, *p* will not be added to the current cluster. Again, this is a generalization of the traditional approach.

The remaining question is how to compute the values $P^{reach}_{\varepsilon, \mu, d, D}(p, o)$ efficiently. Although there might exist situations where we can compute these values directly based on the fuzzy object representations (cf. Definition 4), in this paper, we propose a generally applicable approach based on monte-carlo sampling. In many applications the fuzzy objects might already be described by a discrete probability density function, i.e. we have the sample set already. If the fuzzy object is described by a continuos probability density function, we can easily sample according to this function and derive thus a sequence of samples. In the following, we assume that each object *x* is represented by a sequence of *s* sample points, i.e. *x* is represented by *s* different representations $\langle x_1, ..., x_s \rangle$.

Based on the sample sequences, we could now compute discrete distance density functions consisting of s^2 many discrete distance values. Based on these functions we could then compute the reachability probabilities according to Definition 8. The big problem is that we have to compute for each query object *o*, O(|*DB*|) many different core-object versions leaving out always one element from the database. Furthermore, the computation of each of these core-object values has to consider (in |*DB*|) exponentially many sets $A \subseteq DB$ (cf. Definition 7). Obviously, this is impracticable.

The idea of our approach is to determine the core-object probabilities based on s^2 meaningful samples. Then, we compute the reachability values according to Definition 8.

We first compute for all objects *x* the minimum bounding rectangle MBR(*x*) of the sample points $\langle x_1, ..., x_s \rangle$ (cf. Figure 2). If we now carry out a range query around *o*, we create a sample matrix *M*(*o*) which contains for each object instance o_i *s* different values $m_{i,j} = |N_{\varepsilon, D_j}(o_i)|$, where D_j denotes the *j*th database instance $\{x_j | \langle x_1, ..., x_j, ..., x_s \rangle \in D \wedge x_j \neq o_j\} \cup o_i$ and $N_{\varepsilon, D_j}(o_i)$ denotes the set $\{x_j | d(o_i, x_j) \leq \varepsilon \wedge x_j \in D_j\}$ (cf. Figure 2). We test for each object *x* in the database whether there exist sample instances x_j for which $d(o_i, x_j) \leq \varepsilon$ holds. If this is true we increase the current value of $m_{i,j}$. Note, that often we do not have to compute the s^2 distances $d(o_i, x_j)$, but we can decide based on the boxes MBR(*o*) and MBR(*x*) whether we have to increase all values of the sample matrix *M*(*o*) or none of them. If for the maximum distance $d_{max}(o, x)$ between the two boxes MBR(*o*) and MBR(*x*) $d_{max}(o, x) \leq \varepsilon$ holds, we can increase all values of the sample matrix *M*(*o*) by 1 (cf. object *c* in Figure 2). If for the minimum distance $d_{min}(o, x)$ between the two boxes $d_{min}(o, x) \geq \varepsilon$ holds, we do not have to increase any of these values (cf. object *d* in Figure 2). Only if the value of ε is somewhere in between the two values $d_{min}(o, x)$ and $d_{max}(o, x)$, we have to compute the distances between the samples to decide which values $m_{i,j}$ of the sample matrix have to be increased (cf. object *a* in Figure 2). Finally, we would like to mention that we can compute this sample matrix by only one range scan.

[1] Note that clustering on the centroids of the fuzzy object representations, would suffer from the same drawbacks as the approach based on the distance expectation values.

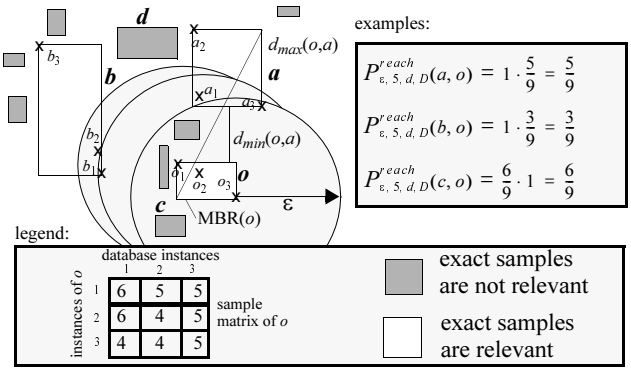


Figure 2: Computation of fuzzy reachability distances (s=3, μ=5).

After having computed the sample matrix $M(o)$, we can easily derive the reachability values for all objects x in the database D w.r.t. o. Thereto, $P^{core}_{\varepsilon, \mu-1, d, D\setminus x}(o) \cdot P_d(x, o)(\varepsilon)$ (cf. Definition 8) has to be computed. The first part can be derived from the matrix $M(o)$, if we decrease the values $m_{i,j}$ by 1 for which $d(o_i, x_j) \leq \varepsilon$ holds. Then, we can count the number of elements in the sample matrix $M(o)$ which contain values higher or equal to μ-1. Normalizing this number by s^2 yields the probability $P^{core}_{\varepsilon, \mu-1, d, D\setminus x}(o)$. The values $P_d(x, o)(\varepsilon)$ can be computed by counting the number of events $d(o_i, x_j) \leq \varepsilon$, and by normalizing this number again by s^2. For the computation of $P_d(x, o)(\varepsilon)$, the distances $d_{max}(o, x)$ and $d_{min}(o, x)$ can again be used for pruning.

If we assume n database objects which are not stored in any index structure, and a sample rate of s, we can summarize the characteristics of the $\mathcal{F}$DBSCAN implementation as follows.

- We need $O(n)$ range scans
- We require between $O(n^2)$ and $O(s^2 \cdot n^2)$ many distance computations between d-dimensional feature vectors.

Note that especially in the important case where the fuzzy objects are not too fuzzy, we only need around $O(n^2)$ many distance calculations. In this case, the introduced pruning distances d_{min} and d_{max} are very effective as they are rather close to each other. Therefore, it is very unlikely that the ε-value is in between them. Furthermore, in this case it is beneficial to organize the minimum bounding rectangles of the sample sets in R-tree [4] like index structures. As we can typically use rather small ε-values for DBSCAN, the number of distance computations can thus further be reduced to $O(n \cdot log n)$ which corresponds to the time complexity of the original DBSCAN algorithm based on index structures.

5. EXPERIMENTAL EVALUATION

In this section, we present the experimental results of the introduced clustering algorithm $\mathcal{F}$DBSCAN demonstrating the characteristics and benefits of our approach.

5.1 Setup

Data Sets. All experiments were based on two different test data sets, an *artificial* data set, and an *engineering* data set which are normalized in a data space $[0, 1]^d$. For each data set, we have exact object representations, i.e. an object is described by exactly one feature vector. Furthermore, each object is randomly surrounded by a box having a side length of $\rho < 1$ in each dimension. For our fuzzy clustering approaches, we assume that each position within the box is equally likely.

The *artificial* data set (ART) consists of 1000 2-dimensional objects which are normally distributed in $[0, 1]^2$.

The *engineering* data set (PLANE) consists of 5000 3D CAD objects provided by our industrial partner, an American airplane manufacturer. Each object is represented by a 42-dimensional feature vector which is derived from the cover sequence model as described in [8].

Implementation. For clustering the fuzzy object representations, we have implemented the algorithm $\mathcal{F}$DBSCAN as described in Section 4.2. Furthermore, we implemented the two approaches UNION and INTERSECTION as described in [10], and the standard DBSCAN approach which carries out the fuzzy clustering based on the distance expectation values (referred to as EXPDBSCAN) and the clustering on the exact object representations.

All algorithms were implemented in Java 1.4. The experiments were run on a Windows laptop with a 730 MHz processor and 512 MB main memory. If not otherwise stated, we used a sample rate of s = 5.

Quality Measures. For comparing a given reference clustering to the clusterings resulting from clustering the fuzzy object representations, we used the approximating quality measure introduced in [11]. In [11] a quality measure for clusters based on the symmetric set difference was introduced and based on this distance measure between clusters a quality criteria for approximated partitioning clusterings Q_{APC} was introduced. This quality measure is based on the minimum weight perfect matching of sets.

Parameters. In all our tests, we set μ = 5 and used an ε-parameter for the various DBSCAN implementations such that between 30 and 50 clusters and between 10% and 40% noise objects for the reference clustering were created.

5.2 Experimental Results

Efficiency. First, we investigate the runtimes of our fuzzy DBSCAN clustering approaches. The following table depicts the absolute runtimes in seconds for the ART data set (ρ=0.01, s = 5).

$\mathcal{F}$DBSCAN	EXPDBSCAN	UNION	INTERSECTION
3,78	27,38	7,44	7,14

The good performance of the $\mathcal{F}$DBSCAN approach demonstrates the suitability of the filters introduced in Section 4.2 resulting in only $O(n^2)$, and not $O(s^2 \cdot n^2)$, many distance computations. Furthermore, we can see that all other fuzzy clustering approaches are slower which can be explained by the higher number of distance computations which have to be carried out, i.e. UNION and INTERSECTION require $O(s \cdot n^2)$ many distance computations, and EXPDBSCAN requires $O(s^2 \cdot n^2)$ many distance computations.

In the following sections, we will show that from an effectivity point of view our approaches also outperform the chosen comparison partners by far.

Effectivity. In a first set of experiments, we investigated the qualities of the different fuzzy clustering approaches w.r.t. a given reference clustering. Figure 3 shows for the ART and for the PLANE data set that for all fuzzy clustering algorithms the quality decreases with an increasing value of ρ, i.e. an increasing uncertainty area of the objects. Furthermore, we can see that the $\mathcal{F}$DBSCAN algorithm

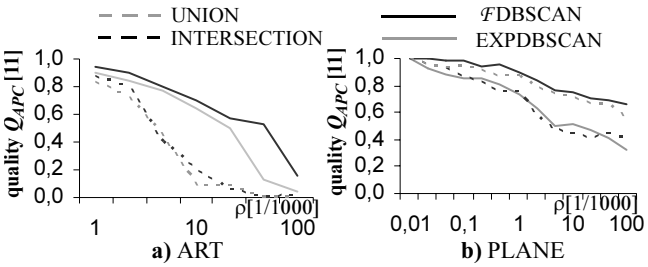


Figure 3: Quality of fuzzy DBSCAN clustering algorithms.

is for both data sets the most effective clustering algorithm over the complete range of ρ. For the ART data set the EXPDBSCAN performs also quite well but for higher dimensional data, e.g. for the PLANE data set, its quality is much worse than the quality of the FDBSCAN approach. On the other hand, the UNION approach performs for sparse high dimensional data well but unfortunately not for the low-dimensional ART data set.

An explanation for the superiority of the FDBSCAN algorithm can be found in Figure 4. In this figure, we investigate the accuracy of the core-point classification of the different algorithms[1]. For the EXPDBSCAN approach, the precision of the detected core objects is very high but unfortunately the recall is very low, i.e. the approach fails to detect many core objects. Thus we have very often the situation depicted in Figure 1b. Similar observations but much more pronounced can be made for the INTERSECTION approach. For the UNION approach the opposite observation can be made. The precision of this approach is very low, as the UNION approach classifies way to many objects as core objects which actually are no core objects. Thus, we often have situations similar to the one depicted in Figure 1a. On the other hand, for the FDBSCAN approach both precision and recall are rather high.

Furthermore, we would like to mention that our FDBSCAN algorithm outperforms the server-sided clustering approaches of state-of-the-art density-based distributed clustering algorithms [6, 9]. The server-sided approach presented in [6] corresponds to an FDBSCAN approach using a sample rate of 1. We noticed that if we use a sample rate *s* around 5 instead of 1, we can increase the average quality values considerably, e.g. for ρ = 0.01 on the PLANE dataset, we can increase the quality from 0.67 (*s* = 1) to 0.76 (s = 5). Furthermore, small sample rates always bear the risk of extreme values, e.g. for ρ = 0.01 on the PLANE dataset, we noticed quality values less than 0.60 when using a sample rate of 1. As sample rates higher than 5 do not much pay off, we suggest to use a sample rate of 5 to get a good trade off between accuracy and efficiency. Furthermore, the server-sided approach presented in [9] corresponds to the EXPDBSCAN approach. Figure 3 and 4 show that the FDBSCAN approach also clearly outperforms this comparison partner.

6. CONCLUSION

In this paper, we demonstrated how density-based clustering can be carried out based on vague and uncertain information which often occurs in modern application ranges like sensor databases, spatial-temporal applications, and biometric information systems. Be-

[1] In the FDBSCAN approach, we classified an object *o* as core object iff $P^{core}_{\varepsilon, \mu, d, D}(o) \geq 0.5$ holds.

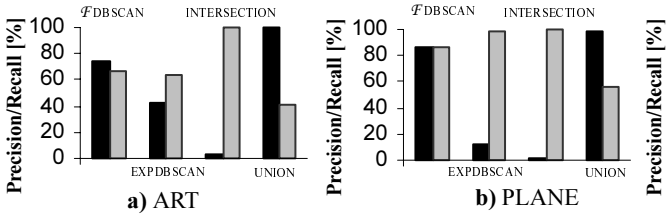


Figure 4: Core-object classification of fuzzy DBSCAN clustering algorithms (ρ = 0.01). recall precision

sides lying the theoretical foundations for density-based clustering of fuzzy data, we showed how to put these concepts into practice. The resulting partitioning density-based algorithm FDBSCAN can be used to cluster fuzzy data, e.g. moving objects, effectively and efficiently. The algorithm follows the new paradigm of integrating fuzzy distance functions directly into data mining algorithms instead of working on lossy aggregated information. In our experimental evaluation, we demonstrated that the newly introduced clustering algorithm FDBSCAN achieves much more accurate results than state-of-the-art comparison partners without sacrificing efficiency.

In our future work, we will show that also other data mining algorithms working on vague information can benefit from a direct integration of fuzzy distance functions.

7. REFERENCES

[1] Bracewell, R. *The Impulse Symbol.* Ch. 5 in The Fourier Transform and Its Applications, 3rd ed.: McGraw-Hill, 1999.

[2] Cheng R., Kalashnikov D.V., Prabhakar S.: *Evaluating probabilistic queries over imprecise data.* SIGMOD'03, pp. 551-562.

[3] Ester M., Kriegel H.-P., Sander J., Xu X.: *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* KDD'96, pp. 226-231.

[4] Guttman A.: *R-trees: A Dynamic Index Structure for Spatial Searching.* Proc. SIGMOD'84, pp. 47-57.

[5] Höppner F., Klawonn F., Kruse R., Runkler T: *Fuzzy Cluster Analysis*. Wiley (1999).

[6] Januzaj E., Kriegel H.-P., Pfeifle M.: *Scalable Density-Based Distributed Clustering*. PKDD'04, pp. 231-244.

[7] Jain A. K., Murty M. N., Flynn P. J.: *Data Clustering: A Review*. ACM Computing Surveys, Vol. 31, No. 3, Sep. 1999, pp. 265-323.

[8] Kriegel H.-P., Brecheisen S., Kröger P., Pfeifle M., Schubert M.: *Using Sets of Feature Vectors for Similarity Search on Voxelized CAD Objects*. SIGMOD'03, pp. 587-598.

[9] Kriegel H.-P., Kunath P., Pfeifle M., Renz M.: *Approximated Clustering of Distributed High Dimensional Data.* PAKDD'05, pp. 432-441.

[10] Kriegel H.-P., Kailing K., Pryakin A., Schubert M.: *Clustering Multi-Represented Objects with Noise*. PAKDD'04, pp. 394-403.

[11] Kriegel H.-P., Pfeifle M.: *Measuring the Quality of Approximated Clusterings*. BTW'05, pp. 415-424.

[12] Li Y., Han J., Yang J.: *Clustering Moving Objects*. KDD'04, pp. 617-622.

[13] Yiu M. L., N. Mamoulis N.: *Clustering Objects on a Spatial Network*. SIGMOD'04, pp. 443-454.