in Proc. 4th IEEE Int. Conf. on Data Mining (ICDM 04), Brighton, UK, 2004

Density Connected Clustering with Local Subspace Preferences

Christian Böhm, Karin Kailing, Hans-Peter Kriegel, Peer Kröger
Institute for Computer Science, University of Munich, Germany
{boehm,kailing,kriegel,kroegerp}@dbs.ifi.lmu.de

Abstract

Many clustering algorithms tend to break down in high-dimensional feature spaces, because the clusters often exist only in specific subspaces (attribute subsets) of the original feature space. Therefore, the task of projected clustering (or subspace clustering) has been defined recently. As a novel solution to tackle this problem, we propose the concept of local subspace preferences, which captures the main directions of high point density. Using this concept we adopt density-based clustering to cope with high-dimensional data. In particular, we achieve the following advantages over existing approaches: Our proposed method has a determinate result, does not depend on the order of processing, is robust against noise, performs only one single scan over the database, and is linear in the number of dimensions. A broad experimental evaluation shows that our approach yields results of significantly better quality than recent work on clustering high-dimensional data.

1. Introduction

Clustering is one of the major data mining tasks. Many useful clustering methods proposed in the last decade (see e.g. [6] for an overview) compute flat or hierarchical partitions of the data points in a complete feature space, i.e. each dimension is equally weighted when computing the distance between points. These approaches are successful for low-dimensional data sets. However, in higher dimensional feature spaces, their accuracy and efficiency deteriorates significantly. The major reason for this behavior is the so-called curse of dimensionality: In high dimensional feature spaces, a full-dimensional distance is often no longer meaningful, since the nearest neighbor of a point is expected to be almost as far as its farthest neighbor [7].

A common approach to cope with high dimensional feature spaces is the application of a global dimensionality reduction technique such as Principal Component Analysis (PCA). A standard clustering method can then be used to compute clusters in this subspace. But if different subsets of the points cluster well on different subspaces of the feature space, a global dimensionality reduction will fail.

To overcome these problems of global dimensionality reduction, recent research proposed to compute *subspace clusters*. Subspace clustering aims at computing pairs (C, S) where C is a set of objects representing a cluster and S is a set of attributes spanning the subspace in which C exists. Mapping each cluster to an associated subspace allows more flexibility than global methods projecting the entire data set onto a single subspace. In the example given in Figure 1, a subspace clustering algorithm will find the two clusters (C_1, A_1) and (C_2, A_2) (see Figure 1(a)). As a d-dimensional data set has 2^d subspaces which may contain clusters, the output of subspace clustering algorithms is usually very large. However, a lot of application domains require that the data set is divided into one single partitioning, where each point belongs exclusively to one cluster. For this case, projected clustering algorithms have been introduced, where each point is assigned to a unique cluster. One of the problems of this approach is shown in Figure 1(a): the points in the black circle can only be assigned to one of the two clusters. In this case, it is not clear to which cluster they should be assigned.

In this paper, we introduce the concept of *subspace preferences* to avoid this ambiguity. Our new approach PreDeCon is founded on the concept of density connected sets proposed in density-based clustering (DBSCAN) [5]. In the example it will generate the two clusters (C_1, A_1) and (C_2, A_2) visualized in Figure 1(b). Of course, in this example DBSCAN itself could have found the two clusters. But in high-dimensional spaces the parameter ε specifying the density threshold must be chosen very large, because a lot of dimensions contribute to the distance values. Thus, using the Euclidian distance measure and a large ε will result in large and unspecific clusters, while a small ε will yield only noise. To ensure the quality of the clusters in high-

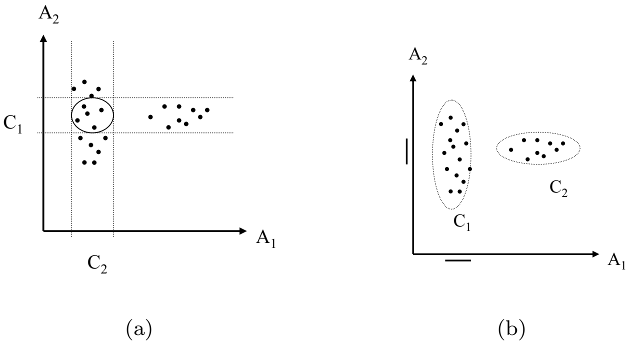


Figure 1: Clusters according to projected/subpace clustering (a) and according to subspace preference clustering (b).

dimensional spaces we suggest to use a weighted Euclidean distance measure to compute smaller but more specific clusters instead of trying to cluster all available points, resulting in large and unspecific clusters. Thus, we build for each point a so-called subspace preference vector based on the variance in each attribute and use a weighted Euclidean distance measure based on this subspace preference vector. Using this more flexible model, we propose the algorithm *PreDeCon* (subspace PREference weighted DEnsity CONnected clustering) to efficiently compute exact solutions of the subspace preference clustering problem. PreDeCon performs a single scan over the database, and is linear in the number of dimensions. The user can select a parameter λ indicating the dimensionality threshold of the searched clusters. Only those clusters with a subspace dimensionality of no more than λ are determined.

The remainder of the paper is organized as follows: In Section 2, we discuss related work and point out our contributions. In Section 3, we formalize our notion of subspace preference clusters. We present the algorithm PreDeCon to efficiently compute such subspace preference clusters in Section 4. Section 5 contains an extensive experimental evaluation and Section 6 concludes the paper.

2. Related Work and Contributions

2.1. Density-Based Clustering

The density-based notion is a common approach for clustering used by various clustering algorithms such as DBSCAN [5], DENCLUE [8], and OPTICS [4]. All these methods search for regions of high density in a feature space that are separated by regions of lower density. A typical density-based clustering algorithm needs two parameters to define the notion of density: First, a parameter μ specifying the minimum number of points, and second, a parameter ε specifying a volume. These two parameters determine a density threshold for clustering. Our approach follows the formal definitions of density connected clusters underlying the algorithm DBSCAN [5]. In this model, a cluster is defined as a maximal set of density connected points.

2.2. Projected and Subspace Clustering

The pioneering approach to subspace clustering is CLIQUE [3], a grid-based algorithm using an *Apriori*-like method to recursively navigate through the set of possible subspaces in a bottom-up way. A density connected version of subspace clustering is SUBCLU [9]. As the number of subspaces which possibly contain clusters is 2^d, the output of those clustering algorithms is usually very large, because points may be assigned to multiple clusters. However, for a lot of application domains a single partitioning of the data is mandatory. Thus, we focus on projected clustering algorithms.

PROCLUS (PROjected CLUStering) [2] is a projected clustering algorithm, which picks up the concepts of k-medoid clustering. The number of clusters k and the average subspace dimension l are input parameters. PROCLUS iteratively computes good medoids for each cluster. The Manhattan Distance divided by the subspace dimension is used as normalized metric for trading between subspaces of different dimensionality. For performance reasons, the iterative medoid-searching phase is performed on a sample using a greedy hill-climbing technique. After this iterative search, an additional pass over the data is performed for refinement of clusters, medoids and associated subspaces. An extension to PROCLUS is the algorithm ORCLUS [1] which computes arbitrarily oriented (i.e. not axis-parallel) projected clusters which are usually hard to interpret and thus, are not suitable for many applications.

In [11] a mathematical definition of an "optimal projected cluster" is presented along with a Monte Carlo algorithm called DOC to compute approximations of such optimal projected clusters. Using the user-specified input parameters w and α, an optimal projected cluster is defined as a set of points C associated with a subspace of dimensions D such that C contains more than α% points of the database and the projection of C onto the subspace spanned by D must be contained in a hyper-cube of width w whereas in all other dimensions $d \notin D$ the points in C are not contained in a hyper-cube of width w. The proposed algorithm DOC only finds approximations because it generates projected clusters of width $2w$. In addition, no assumption on the distribution of points inside such a hyper-cube is made. The reported projected clusters may contain additional noise objects (especially when the size of the projected cluster is considerably smaller than $2w$) and/or may miss some points that naturally belong to the projected cluster (especially when the size of the projected cluster is considerably larger than $2w$).

Both methods are limited to computing approximations using sampling techniques and therefore, the assignment of points to clusters is no longer determinate and may vary for different runs of the algorithm.

2.3. Our Contributions

In this paper, we make the following contributions: Analogously to projected clustering which was introduced to enhance the quality of k-means like clustering algorithms in high-dimensional space, we extend the well-founded notion of density connected clusters to ensure high quality results even in high-dimensional spaces. We do not use any sampling or approximation techniques, thus the result of our clustering algorithm is determinate. We propose an efficient method called PreDeCon which is able to compute all subspace preference clusters of a certain dimensionality in a single scan over the database and is linear in the number of dimensions. And finally, we successfully apply our algorithm PreDeCon to several real-world data sets, showing its superior performance over existing approaches.

3. The Notion of Subspace Preference Clusters

In this section, we formalize the notion of subspace preference clusters. Let $\mathcal{D}$ be a database of d-dimensional points ($\mathcal{D} \subseteq \mathbb{R}^d$), where the set of attributes is denoted by $\mathcal{A} = \{A_1, \ldots A_d\}$. The projection of a point p onto an attribute $A_i \in \mathcal{A}$ is denoted by $\pi_{A_i}(p)$. Let $dist : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a metric distance function between points in $\mathcal{D}$, e.g. one of the L_p-norms. Let $\mathcal{N}_\varepsilon(p)$ denote the ε-neighborhood of $p \in \mathcal{D}$, i.e. $\mathcal{N}_\varepsilon(p)$ contains all points q where $dist(p, q) \le \varepsilon$.

Intuitively, a subspace preference cluster is a density connected set of points associated with a certain subspace preference vector. In order to identify subspace preference clusters, we are interested in all sets of points having a small variance along one or more attributes, i.e. a variance smaller than a given $\delta \in \mathbb{R}$.

Definition 1 (variance along an attribute) *Let $p \in \mathcal{D}$ and $\varepsilon \in \mathbb{R}$. The* variance of $\mathcal{N}_\varepsilon(p)$ along an attribute $A_i \in \mathcal{A}$, *denoted by* $\textsc{Var}_{A_i}(\mathcal{N}_\varepsilon(p))$, *is defined as follows:*

$$\textsc{Var}_{A_i}(\mathcal{N}_\varepsilon(p)) = \frac{\sum_{q \in \mathcal{N}_\varepsilon(p)} (dist(\pi_{A_i}(p), \pi_{A_i}(q)))^2}{|\mathcal{N}_\varepsilon(p)|}$$

Definition 2 (subspace preference dimensionality)

Let $p \in \mathcal{D}$ and $\delta \in \mathbb{R}$. The number of attributes A_i with $\textsc{Var}_{A_i} \le \delta$ is called the subspace preference dimensionality *of $\mathcal{N}_\varepsilon(p)$, denoted by* $\textsc{PDim}(\mathcal{N}_\varepsilon(p))$.

The intuition of our formalization is to consider those points as core points of a cluster which have enough dimensions with a low variance in their neighborhood. Therefore, we associate each point p with a subspace preference vector $\bar{\mathbf{w}}_p$ which reflects the variance of the points in the ε-neighborhood of p along each attribute in $\mathcal{A}$.

Definition 3 (preference weighted similarity measure)

Let $p \in \mathcal{D}$, $\delta \in \mathbb{R}$ and $\kappa \in \mathbb{R}$ be a constant with $\kappa \gg 1$. Let $\bar{\mathbf{w}}_p = (w_1, w_2, \ldots w_d)$ be the so-called subspace preference vector *of p, where*

$$w_i = \begin{cases} 1 & if \quad \textsc{Var}_{A_i}(\mathcal{N}_\varepsilon(p)) > \delta \\ \kappa & if \quad \textsc{Var}_{A_i}(\mathcal{N}_\varepsilon(p)) \le \delta \end{cases}$$

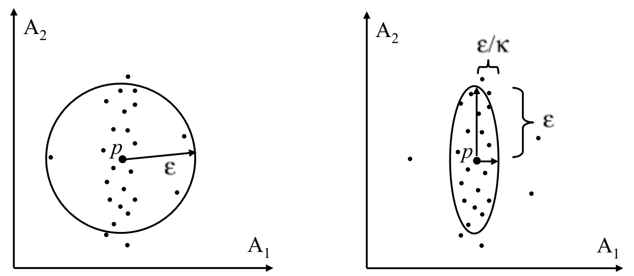
The preference weighted similarity measure *associated with a point p is denoted by*

$$dist_p(p, q) = \sqrt{\sum_{i=1}^{d} w_i \cdot (\pi_{A_i}(p) - \pi_{A_i}(q))^2}$$

where w_i is the i-th component of $\bar{\mathbf{w}}_p$.

Let us note, that the preference weighted similarity measure $dist_p(p, q)$ is simply a weighted Euclidean distance. The parameter δ specifies the threshold for a low variance. As we are only interested in distinguishing between dimensions with low variance and all other dimensions, weighting the dimensions inversely proportional to their variance is not useful. Thus, our weight vector has only two possible values.

The preference weighted similarity measure is visualized in Figure 2. The ε-neighborhood of a 2-dimensional point p exhibits low variance along attribute A_1 and high variance along attribute A_2. The similarity measure $dist_p$ weights attributes with low variance considerably lower (by the factor κ) than attributes with a high variance. However, we face the problem that the similarity measure in Definition 3 is not symmetric, because $dist_p(p, q) = dist_q(q, p)$



(a) (b)

Figure 2: ε-neighborhood of p according to (a) simple Euclidean and (b) preference weighted Euclidean distance.

does obviously not hold in general. If an asymmetric similarity measure is used in DBSCAN, a different clustering result can be obtained depending on the order of processing (e.g. which point is selected as the starting point). Although the result is typically not seriously affected by this ambiguity effect, we avoid this problem easily by an extension of our similarity measure which makes it symmetric. We simply combine both similarity measures $dist_p(p,q)$ and $dist_q(p,q)$ by a suitable arithmetic operation such as the maximum of the two.

Definition 4 (general preference weighted similarity)

The general preference weighted similarity *of two arbitrary points $p, q \in \mathcal{D}$, denoted by $dist_{pref}(p,q)$, is defined as the maximum of the corresponding preference weighted similarity measures of p ($dist_p$) and q ($dist_q$), formally:*

$$dist_{pref}(p,q) = \max\{dist_p(p,q), dist_q(q,p)\}.$$

Based on these considerations, we define the preference weighted ε-neighborhood as a symmetric concept:

Definition 5 (preference weighted ε-neighborhood)

Let $\varepsilon \in \mathbb{R}$. The preference weighted ε-neighborhood *of a point $o \in \mathcal{D}$, denoted by $\mathcal{N}_\varepsilon^{\bar{\mathbf{w}}_o}(o)$, is defined by:*

$$\mathcal{N}_\varepsilon^{\bar{\mathbf{w}}_o}(o) = \{x \in \mathcal{D} \mid dist_{pref}(o,x) \le \varepsilon\}.$$

Preference weighted core points can now be defined as follows.

Definition 6 (preference weighted core point)
Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A point $o \in DB$ is called preference weighted core point *w.r.t. ε, μ, δ, and λ (denoted by $\textsc{Core}^{\text{pref}}_{\text{den}}(o)$), if the preference dimensionality of its ε-neighborhood is at most λ and its preference weighted ε-neighborhood contains at least μ points, formally:*

$$\textsc{Core}^{\text{pref}}_{\text{den}}(o) \Leftrightarrow \textsc{PDim}(\mathcal{N}_\varepsilon(o)) \le \lambda \wedge |\,\mathcal{N}_\varepsilon^{\bar{\mathbf{w}}_o}(o)\,| \ \ge \mu.$$

Let us note that in $\textsc{Core}^{\text{pref}}_{\text{den}}$ the acronym "pref" refers to the parameters δ and λ which are responsible for preference weighting. In the following, we omit the parameters ε, μ, δ, and λ wherever the context is clear and use "den" and "pref" instead.

Definition 7 (direct preference weighted reachability)

Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A point $p \in \mathcal{D}$ is directly preference weighted reachable *from a point $q \in \mathcal{D}$ w.r.t. ε, μ, δ, and λ (denoted by $\textsc{DirReach}^{\text{pref}}_{\text{den}}(q,p)$), if q is a preference weighted core point, the subspace preference dimensionality of $\mathcal{N}_\varepsilon(p)$ is at most λ, and $p \in \mathcal{N}_\varepsilon^{\bar{\mathbf{w}}_q}(q)$, formally:*

$$\begin{aligned}
&\textsc{DirReach}^{\text{pref}}_{\text{den}}(q,p) \Leftrightarrow\\
&(1)\quad \textsc{Core}^{\text{pref}}_{\text{den}}(q)\\
&(2)\quad \textsc{PDim}(\mathcal{N}_\varepsilon(p)) \le \lambda\\
&(3)\quad p \in \mathcal{N}_\varepsilon^{\bar{\mathbf{w}}_q}(q).
\end{aligned}$$

Direct preference weighted reachability is symmetric for preference weighted core points. Both $dist_p(p,q) \le \varepsilon$ and $dist_q(q,p) \le \varepsilon$ must hold.

Definition 8 (preference weighted reachability)

Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A point $p \in \mathcal{D}$ is preference weighted reachable *from a point $q \in \mathcal{D}$ w.r.t. ε, μ, δ, and λ (denoted by $\textsc{Reach}^{\text{pref}}_{\text{den}}(q,p)$), if there is a chain of points $p_1, \cdots p_n$ such that $p_1 = q, p_n = p$ and p_{i+1} is directly preference weighted reachable from p_i, formally:*

$$\begin{aligned}
&\textsc{Reach}^{\text{pref}}_{\text{den}}(q,p) \Leftrightarrow\\
&\quad \exists p_1, \ldots, p_n \in \mathcal{D} : p_1 = q \ \wedge \ p_n = p \ \wedge\\
&\quad \forall i \in \{1, \ldots, n-1\} : \textsc{DirReach}^{\text{pref}}_{\text{den}}(p_i, p_{i+1}).
\end{aligned}$$

It is easy to see, that preference weighted reachability is the transitive closure of direct preference weighted reachability.

Definition 9 (preference weighted connectivity)

Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A point $p \in \mathcal{D}$ is preference weighted connected *to a point $q \in \mathcal{D}$, if there is a point $o \in \mathcal{D}$ such that both p and q are preference weighted reachable from o, formally:*

$$\begin{aligned}
&\textsc{Connect}^{\text{pref}}_{\text{den}}(q,p) \Leftrightarrow\\
&\exists o \in \mathcal{D} : \textsc{Reach}^{\text{pref}}_{\text{den}}(o,q) \ \wedge \ \textsc{Reach}^{\text{pref}}_{\text{den}}(o,p).
\end{aligned}$$

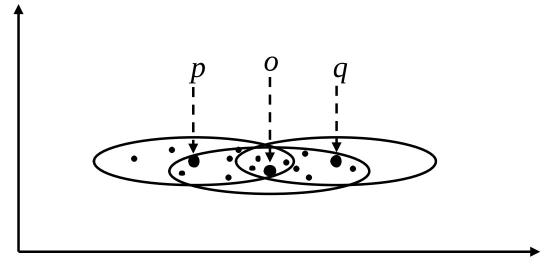


Figure 3: p and q are preference weighted connected via o.

Preference weighted connectivity is a symmetric relation. The concept is visualized in Figure 3. A subspace preference cluster can now be defined as a maximal preference weighted connected set:

Definition 10 (subspace preference cluster)
Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A non-empty subset $\mathcal{C} \subseteq \mathcal{D}$ is called a subspace preference cluster *w.r.t. ε, μ, δ, and λ, if all points in $\mathcal{C}$ are preference weighted connected and $\mathcal{C}$ is maximal w.r.t. preference weighted reachability, formally:*

$\textsc{ConSet}^{\text{pref}}_{\text{den}}(\mathcal{C}) \Leftrightarrow$

Connectivity: $\forall o, q \in \mathcal{C} : \textsc{Connect}^{\text{pref}}_{\text{den}}(o, q)$

Maximality: $\forall p, q \in \mathcal{D} : q \in \mathcal{C} \wedge \textsc{Reach}^{\text{pref}}_{\text{den}}(q, p) \Rightarrow p \in \mathcal{C}$.

The following two lemmata are important for validating the correctness of our clustering algorithm. Intuitively, they state that we can discover a subspace preference cluster for a given parameter setting in a two-step approach: First, choose an arbitrary preference weighted core point o from the database. Second, retrieve all points that are preference weighted reachable from o. This approach yields the subspace preference cluster containing o.

Lemma 1
Let $p \in \mathcal{D}$. If p is a preference weighted core point, then the set of points, which are preference weighted reachable from p is a subspace preference cluster, formally:

$$\textsc{Core}^{\text{pref}}_{\text{den}}(p) \wedge \mathcal{C} = \{o \in \mathcal{D} \,|\, \textsc{Reach}^{\text{pref}}_{\text{den}}(p, o)\} \Rightarrow \textsc{ConSet}^{\text{pref}}_{\text{den}}(\mathcal{C}).$$

Proof.
(1) $\mathcal{C} \neq \emptyset$:
By assumption, $\textsc{Core}^{\text{pref}}_{\text{den}}(p)$ *and thus,* $\textsc{PDim}(\mathcal{N}_\varepsilon(p)) \leq \lambda$
$\Rightarrow \textsc{DirReach}^{\text{pref}}_{\text{den}}(p, p)$
$\Rightarrow \textsc{Reach}^{\text{pref}}_{\text{den}}(p, p)$
$\Rightarrow p \in \mathcal{C}$.

(2) Maximality:
Let $x \in \mathcal{C}$ and $y \in \mathcal{D}$ and $\textsc{Reach}^{\text{pref}}_{\text{den}}(x, y)$
$\Rightarrow \textsc{Reach}^{\text{pref}}_{\text{den}}(p, x) \wedge \textsc{Reach}^{\text{pref}}_{\text{den}}(x, y)$
$\Rightarrow \textsc{Reach}^{\text{pref}}_{\text{den}}(p, y)$ *(since preference weighted reachability is a transitive relation)*
$\Rightarrow y \in \mathcal{C}$.

(3) Connectivity:
$\forall x, y \in \mathcal{C} : \textsc{Reach}^{\text{pref}}_{\text{den}}(p, x) \wedge \textsc{Reach}^{\text{pref}}_{\text{den}}(p, y)$
$\Rightarrow \textsc{Connect}^{\text{pref}}_{\text{den}}(x, y)$ *(via p).* □

Lemma 2
Let $\mathcal{C} \subseteq \mathcal{D}$ be a subspace preference cluster. Let $p \in \mathcal{C}$ be a preference weighted core point. Then $\mathcal{C}$ equals the set of points which are preference weighted reachable from p, formally:

$$\textsc{ConSet}^{\text{pref}}_{\text{den}}(\mathcal{C}) \wedge p \in \mathcal{C} \wedge \textsc{Core}^{\text{pref}}_{\text{den}}(p) \Rightarrow \mathcal{C} = \{o \in \mathcal{D} \,|\, \textsc{Reach}^{\text{pref}}_{\text{den}}(p, o)\}.$$

Proof. *Let $\bar{\mathcal{C}} = \{o \in \mathcal{D} \,|\, \textsc{Reach}^{\text{pref}}_{\text{den}}(p, o)\}$. We have to show that $\bar{\mathcal{C}} = \mathcal{C}$:*

(1) $\bar{\mathcal{C}} \subseteq \mathcal{C}$: obvious from the definition of $\bar{\mathcal{C}}$.

(2) $\mathcal{C} \subseteq \bar{\mathcal{C}}$: Let $q \in \mathcal{C}$. By assumption, $p \in \mathcal{C}$ and $\textsc{ConSet}^{\text{pref}}_{\text{den}}(\mathcal{C})$
$\Rightarrow \exists o \in \mathcal{C} : \textsc{Reach}^{\text{pref}}_{\text{den}}(o, p) \wedge \textsc{Reach}^{\text{pref}}_{\text{den}}(o, q)$
$\Rightarrow \textsc{Reach}^{\text{pref}}_{\text{den}}(p, o)$ *(since both o and p are preference weighted core points and preference weighted reachability is symmetric for preference weighted core points.*
$\Rightarrow \textsc{Reach}^{\text{pref}}_{\text{den}}(p, q)$ *(transitivity of preference weighted reachability)*
$\Rightarrow q \in \bar{\mathcal{C}}$. □

4. Efficiently Computing Subspace Preference Clusters

4.1. Algorithm PreDeCon

PreDeCon performs one pass over the database to find all subspace preference clusters for a given parameter setting. The pseudo code of the algorithm is given in Figure 4. At the beginning each point is marked as unclassified. During the run of PreDeCon all points are either assigned a certain cluster identifier or marked as noise. For each point which is not yet classified, PreDeCon checks whether this point is a preference weighted core point. If so, the algorithm expands the cluster belonging to this point. Otherwise the point is marked as noise. To find a new cluster, PreDeCon starts with an arbitrary preference weighted core point o and searches for all points that are preference weighted reachable from o. This is sufficient to find the whole cluster containing the point o, due to Lemma 2. When PreDeCon

```
algorithm PreDeCon($\mathcal{D}$, $\varepsilon$, $\mu$, $\lambda$, $\delta$)
  // assumption: each point in $\mathcal{D}$ is marked as unclassified
  for each unclassified $o \in \mathcal{D}$ do
    if $\textsc{Core}^{\text{pref}}_{\text{den}}(o)$ then        // expand a new cluster
      generate new clusterID;
      insert all $x \in \mathcal{N}^{\bar{\mathbf{w}}_o}_{\varepsilon}(o)$ into queue $\Phi$;
      while $\Phi \neq \emptyset$ do
        $q$ = first point in $\Phi$;
        compute $\mathcal{R} = \{x \in \mathcal{D} \mid \textsc{DirReach}^{\text{pref}}_{\text{den}}(q, x)\}$;
        for each $x \in \mathcal{R}$ do
          if $x$ is unclassified then
            insert $x$ into $\Phi$;
          if $x$ is unclassified or noise then
            assign current clusterID to $x$
        remove $q$ from $\Phi$;
    else        // $o$ is noise
      mark $o$ as noise;
end.
```

Figure 4: Pseudo code of the PreDeCon algorithm.

has found a preference weighted core point, a new cluster identifier "clusterID" is generated which will be assigned to all points found in the generation of the subspace preference cluster. PreDeCon begins by inserting all points in the preference weighted ε-neighborhood of point o into a queue. For each point in the queue, it computes all directly preference weighted reachable points and inserts those points into the queue which are still unclassified. This is repeated until the queue is empty and the entire cluster is computed.

As mentioned above, the results of PreDeCon do not depend on the order of processing, i.e. the resulting clustering (number of clusters and association of core points to clusters) is determinate.

4.2. Complexity Analysis

As the performance of most index structures deteriorates in high-dimensional spaces, we base our complexity analysis of PreDeCon on the assumption of no index structure.

Lemma 3 *The overall worst-case time complexity of our algorithm based on the sequential scan of the data set is $O(d \cdot n^2)$.*

Proof. *Our algorithm has to associate each point of the data set with a preference weighted similarity weight vector that is used for searching neighbors (cf. Definition 3). The corresponding vector must be computed once for each point. The computation of the preference weighted similarity weight vector $\bar{\mathbf{w}}$ is based on the result of a Euclidean range query which can be evaluated in $O(d \cdot n)$ time. Then the vector is built by checking the variance of the points in the Euclidean ε-neighborhood along each dimension which requires $O(d \cdot n)$ time. For all points together, this sums up to $O(d \cdot n^2)$.*
Checking the preference weighted core point property according to Definition 6, and expanding a preference weighted cluster, requires for each point in the Euclidean ε-neighborhood the evaluation of a weighted Euclidean distance which can be done in $O(d \cdot n)$ time. For all points together (including the above cost for the determination of the preference weighted similarity weight vector), we obtain a worst-case time complexity of $O(d \cdot n^2)$. □

Let us note, that the runtime of our algorithm does not depend on the dimensionality of the subspace preference clusters.

4.3. Input Parameters

PreDeCon has four input parameters, two density parameters ε and μ and two preference parameters λ and δ.

The parameter $\lambda \in \mathbb{N}$ specifies the preference dimension of the subspace preference clusters to be computed, i.e. the maximum number of attributes that have a low variance (cf. Definition 2). In our experiments, it turns out that the clusters computed by PreDeCon need not to have a preference dimension of λ. In fact, λ rather specifies a upper bound for the preference dimensions of the computed clusters. The parameter $\delta \in \mathbb{R}$ specifies the upper bound for the variance in an attribute. If the variance along an attribute is less than δ, this attribute is considered to yield a dense projection. The choice of δ depends on the maximum value MAX_{A_i} in each attribute A_i. It empirically turned out that PreDeCon is rather robust against different choices for δ. Our experiments showed that $\delta \leq 5$ is usually a good choice for data sets with $MAX_{A_i} = 100$. We suggest to normalize data sets that contain attributes not scaled within $[0, 100]$ accordingly.

The parameters $\varepsilon \in \mathbb{R}$ and $\mu \in \mathbb{N}$ specify the density threshold which clusters must exceed. They should be chosen as suggested in [5].

5. Evaluation

In this section, we present a broad evaluation of PreDeCon. We implemented PreDeCon as well as the three comparative methods DBSCAN, PROCLUS, and DOC in JAVA. All experiments were run on a Linux workstation with a 2.0 GHz CPU and 2.0 GB RAM.

We evaluated PreDeCon using several synthetic data sets generated using a self-implemented data generator.

We varied the dimension of the data sets from 2 to 50, the number of clusters from 2 to 5, the subspace dimensionality of the clusters from 2 to 10 and the amount of noise from 50% to 75%. The density of the clusters was chosen randomly. In all experiments, PreDeCon separated the generated clusters hidden in the data from each other and from noise.

Due to space limitations we focus on the following experiments using two different real world data sets: The first data set is derived from a gene expression experiment studying the yeast cell cycle by extracting the expression level of approximately 2800 genes at 17 time spots [12]. Since the genes have no class label, we have to judge the accuracy of the clustering by looking at the results. The aim is to find clusters of co-expressed genes that share similar functions. Biological criteria for similar functions of genes are direct interactions of genes, common complexes of gene products, or gene products participating in common pathways. We analyzed the clustering results according to these criteria using the publicly available Saccharomyces Genome Database (SGD) [1]. The second data set [10] is derived by a newborn screening and contains the concentrations of 43 metabolites in 2000 newborns. Each newborn has a class label attached indicating its genetic disease. All attribute values were normalized between 0 and 100.

Gene Expression Data. The clusters in the gene expression data set were generated using the following parameter setting: $\varepsilon = 80.0$, $\mu = 7$, $\delta = 4.0$, and $\lambda = 12$. PreDeCon found several clusters with varying size of around 10 to 40 genes. Each cluster contains functionally related genes according to the biological criteria mentioned above. For example, one cluster contains several genes that are involved in chromatin modelling and maintenance (NHP10, DPB4, IES3, and TAF9) where IES3 and NHP10 are even direct interaction partners. A second cluster contains more than 30 genes coding for structural components of the ribosome and 4 genes that are localized in the ribosome and build a larger complex (CDC33, TEF4, EFB1, and NHP2). A third cluster contains several genes involved in the glycolysis pathway (CDC19, TPI1, TDH2, FBA1, and GPM1). Let us note, that each cluster also contains a few genes of different or unknown function, which is no wonder since gene expression data is very noisy due to experimental impacts during data generation. Summing up, the clusters found by PreDeCon contained genes that interact with each other, build complexes with each other or participate in common pathways, and thus are functionally related. The detected groups of co-expressed genes are therefore biologically relevant and meaningful.

Table 1: Confusion matrix of clustering results on metabolome data.

cluster		class labels				
id	size	control	PKU	LCHAD	MCAD	others
1	269	264	3	2	0	0
2	29	0	29	0	0	0
3	38	0	38	0	0	0
4	10	0	10	0	0	0

Metabolome Data. The clusters in the metabolome data set were generated using the following parameter setting: $\varepsilon = 150.0$, $\mu = 10$, $\delta = 3.0$, and $\lambda = 4$. PreDeCon found 4 clusters. The contents of these clusters are visualized as confusion matrix in Table 1. As it can be seen, more than 98% of the points in the first cluster are healthy newborns (label "control"). The other three clusters contain 100% newborns suffering from the PKU (phenylketonuria), one of the prevalent inborn metabolic diseases. Newborns suffering from one of the other diseases were classified as noise, i.e. they were not separated by PreDeCon. However, a significant majority of healthy newborns and newborns suffering from PKU were separated by PreDeCon. In addition, the list of attributes (metabolites) exhibiting low variance in each cluster give useful hints for further medical research.

Comparison with DBSCAN. DBSCAN is a full-dimensional clustering algorithm, i.e. it computes clusters giving each dimension equal weights. For each run of DBSCAN on the biological data sets, we chose the parameters according to [5] using a k-nn-distance graph. Applied to the gene expression data, DBSCAN found 6 relatively large clusters where the fraction of genes with functional relationships was rather small. We made similar observations when we applied DBSCAN to the metabolome data: the computed clusters contained newborns with all sorts of class labels.

Comparison with PROCLUS. We implemented PROCLUS [2] as an axis-parallel version of ORCLUS [1] as the authors of ORCLUS state that this version is slightly more stable and accurate than the original PROCLUS implementation. We tested PROCLUS on the metabolome data set using different values of k (from 5 to 13) and l (from 3 to 15). But in all cases, we did not get any cluster containing objects of less than 4 different classes. In all larger clusters almost all class labels appear. The clusters obtained for the gene expression data set seem

[1] http://www.yeastgenome.org/

equally unspecific — independent of the chosen parameters.

Comparison with DOC. DOC is a projected clustering algorithm proposed recently. Since the suggestions on how to choose the parameters for DOC presented in [11] are rather misleading, we had to choose valid parameters to run DOC on the biological data sets. In particular, we chose the following parameter settings for both data sets: the number m of random sets generated for each seed was set to 10 and the size r of these random sets was set to 20. In addition, we tested two density thresholds of the clusters, $\alpha = 0.05$ and $\alpha = 0.005$, where as the width w of the hypercubes was set to 30. The fraction of the cluster points that must be conserved, when an attribute is added to the cluster, was set to $\beta = 0.9$. Thus, for the computation of one cluster, DOC generated $\lfloor \frac{2}{\alpha} \rfloor = 40$ random points as seeds. We performed multiple runs of DOC on both biological data sets. The results of the runs on each data set were rather different. Applied to the gene expression data set, DOC found 2 to 7 clusters with varying dimensionality (ranging from 1 to 14 dimensions) in 10 runs. In all cases, the clusters contained more than 200 members. Each detected functional relationship was statistically meaningless. Applied to the metabolome data set, DOC found 4 to 24 clusters of a dimensionality varying from 26 to 38. In most clusters, the instances from different classes (newborns with different diseases) were rather equally distributed.

6. Conclusions

In this paper, we proposed PreDeCon, an algorithm for computing clusters of subspace preference weighted connected points. This algorithm searches for local subgroups of a set of feature vectors having a low variance along one or more (but not all) attributes. The attributes with low variance may vary between different clusters. PreDeCon is designed to find clusters in moderate-to-high dimensional feature spaces where traditional, “full-dimensional” clustering algorithms tend to break down. PreDeCon is determinate, robust against noise, and efficient with a worst case time complexity of $O(d \cdot n^2)$. Our extensive experimental evaluation shows a superior clustering accuracy of PreDeCon over relevant methods like DBSCAN, CLIQUE, ORCLUS, and DOC.

Acknowledgements

Parts of this work is supported by the German Ministry for Education, Science, Research and Technology (BMBF) (grant no. 031U112F).

References

[1] C. Aggarwal and P. Yu. ”Finding Generalized Projected Clusters in High Dimensional Space”. In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD’00), Dallas, TX*, 2000.

[2] C. C. Aggarwal and C. Procopiuc. ”Fast Algorithms for Projected Clustering”. In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD’99), Philadelphia, PA*, 1999.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. ”Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications”. In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD’98), Seattle, WA*, 1998.

[4] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. ”OPTICS: Ordering Points to Identify the Clustering Structure”. In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD’99), Philadelphia, PA*, pages 49–60, 1999.

[5] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. ”A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. In *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD’96), Portland, OR*, pages 291–316. AAAI Press, 1996.

[6] J. Han and M. Kamber. *”Data Mining: Concepts and Techniques”*. Morgan Kaufman, 2001.

[7] A. Hinneburg, C. C. Aggarwal, and D. A. Keim. ”What is the Nearest Neighbor in High Dimensional Spaces”. In *Proc. 26th Int. Conf. on Very Large Databases (VLDB’00), Cairo, Egypt*, 2000.

[8] A. Hinneburg and D. A. Keim. ”An Efficient Approach to Clustering in Large Multimedia Databases with Noise”. In *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining (KDD’98), New York, NY*, pages 224–228. AAAI Press, 1998.

[9] K. Kailing, H.-P. Kriegel, and P. Kröger. ”Density-Connected Subspace Clustering for High-Dimensional Data”. In *Proc. SIAM Int. Conf. on Data Mining (SDM’04), Lake Buena Vista, FL*, 2004.

[10] B. Liebl, U. Nennstiel-Ratzel, R. von Kries, R. Fingerhut, B. Olgemöller, A. Zapf, and A. A. Roscher. ”Very High Compliance in an Expanded MS-MS-Based Newborn Screening Program Despite Written Parental Consent”. *Preventive Medicine*, 34(2):127–131, 2002.

[11] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. ”A Monte Carlo Algorithm for Fast Projective Clustering”. In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD’02), Madison, Wisconsin*, pages 418–427, 2002.

[12] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. ”Systematic Determination of Genetic Network Architecture”. *Nature Genetics*, 22:281–285, 1999.