MAFIA: An Adaptive Grid-Based Subspace Clustering Approach

Henrik Daniel Christensen[*hench13@student.sdu.dk*]

University of Southern Denmark, SDU
Department of Mathematics and Computer Science

Abstract. This paper presents a comprehensive analysis of three bottom-up subspace clustering algorithms: *MAFIA*, *CLIQUE*, and *SUBCLU*. MAFIA extends the grid-based approach of CLIQUE by introducing adaptive grid sizes, offering improved scalability and clustering quality. SUBCLU, in contrast, utilizes a density-connectivity method that allows better identification of arbitrarily shaped clusters, overcoming some of the limitations inherent in grid-based algorithms. The paper explores the relationships and distinctions between these algorithms, evaluating their performance in terms of scalability of dataset size, data- and cluster-dimensionality, as well as their clustering quality. Through a series of experiments with both synthetic and real-world datasets, we demonstrate the strengths and limitations of each approach. The findings reveal that while MAFIA is the most scalable, SUBCLU excels at detecting clusters with irregular shapes. However, all algorithms require proper parameter tuning for optimal performance.

Keywords: High-Dimensional Subspace Clustering · Grid-Based- and Density-Connectivity approach · Comparative Study.

1 Introduction

Clustering is one of the main techniques within data mining. The main goal is to discover unknown patterns within a data set, by partitioning the data objects into *clusters* in reasonable amount of time. Here, each object in a cluster is similar to one another, but different from objects in other clusters. Clustering is widely used in many applications, such as advertising, biology, web search and business intelligence [5, p. 444].

As the data sets, today, often are large in size, the clustering algorithm must be *scalable*. Additionally, data sets often contains numerous features/dimensions (*attributes*), which introduces the problem of *curse of dimensionality*, which refers to a several challenges related to high-dimensional data spaces:

First, the issue of *concentration of distances*, where distances between objects in high-dimensional spaces become increasingly similar as dimensionality increases. This means that data points tend to become nearly equidistant from one another, making it difficult for traditional distance-based clustering algorithms to discover clusters.

Secondly, the problem of *local feature relevance* and *local feature correlation*, where only a subset of features or different combinations of feature correlations may be relevant for clustering. Consequently, feature reduction techniques like *Principal Component Analysis* (PCA), which project the original space onto a lower-dimensional subspace, are inadequate because they typically identify only one global subspace that best fits the entire dataset. Also, algorithms that evaluate the entire feature space does not address this issue effectively. [8, p. 43–46]

Instead of relying on a global approach to feature selection, a local approach that addresses the second problem is necessary. However, then two separate problems needs to be dealt with, which both needs to be solved simultaneously. First of all, the clustering algorithm must be able to find the relevant subspaces for each cluster. Secondly, it must be able to find the clusters in each relevant subspace. To solve them efficiently, heuristics needs to be employed into the clustering algorithms. [8, p. 6–7]

For many applications, it is reasonable to focus only on clusters in axis-parallel subspaces, thus restricting the search space to $O(2^d)$ attributes. These algorithms are called *subspace clustering* (or *projected clustering*) algorithms. These can be further divided into: *top-down*- or *bottom-up* approach. In top-down approach, the relevant subspaces for the clusters are determined by gradually reducing the subspaces, starting from the entire space. In contrast, bottom-up approaches, finds the relevant subspaces for the clusters from the original space starting from 1-dimensional using the *monotonicity property* (or *downward closure property*), see Lemma 1 [2]. [8, p. 8, 11]

For the rest of this paper, the following notation will be adopted: Let $\mathcal{A} = \{A_1, \ldots, A_d\}$ represent the set of attributes in a d-dimensional numerical space, $\mathcal{S} = A_1 \times A_2 \times \cdots \times A_d$. A subspace is then a subset of $\mathcal{A}$. Let $\mathcal{D} = \{p_1, \ldots, p_n\}$ be the dataset, where $p_i \in \mathcal{S}$ represents a data point. A cluster $\mathcal{C}$ is then a subset of $\mathcal{D}$.

Lemma 1. *If $\mathcal{C}$ is a cluster in a k-dimensional subspace, then $\mathcal{C}$ must also form a cluster in any $(k-1)$-dimensional projection of the same space.*

A proof is provided in [2].

1.1 Contributions

The primary focus of this paper is to analyze the grid-based bottom-up subspace clustering algorithm *MAFIA* [11], which builds upon the first grid-based approach *CLIQUE* [2]. The relationship between these two is examined. Additionally, the density-connected-based algorithm *SUBCLU* [9] is included, as it offers a contrasting approach to the grid-based approach. All three algorithms are also evaluated experimentally in terms of scalability and clustering quality. Note that, a slightly different naming convention is used in this paper, for clarity.

The remainder of the paper is structured as follows: Section 2 describes and analyzes the three algorithms in detail. Section 3 evaluates their performance in

terms of scalability, considering dataset size, data- and cluster-dimensionality, as well as their clustering quality. Section 5 discusses the findings and explores the contributions of each algorithm to the field of subspace clustering. Finally, Section 6 concludes the main findings and suggests some future work.

2 Subspace Clustering

The overall idea of subspace clustering is to identify the subspaces that allows better clustering than the entire space. Two different approaches of subspace clustering will be discussed in this section including three algorithms. All three is bottom-up approaches meaning they must implement the monotonicity property [8, p. 1:11], see Lemma 1.

2.1 Grid-Based approach

In a grid-based approach, $\mathcal{S}$ is partitioned into an axis-parallel grid structure. Each grid cell forms a axis-parallel, hyper-rectangular, *unit*, and the number of points within each unit is counted. This process is initially applied in the 1-dimensional space for each A_i. Units that contain a number of points exceeding a predefined threshold are retained as *dense units* (DUs). These DUs are then merged with adjacent ones in the next level (2-dimensional space) to form *candidate dense units* (CDUs), those of which are actually dense are kept. The final DUs are then the clusters found by the algorithm which will be described using minimal representations in the form of *Disjunctive Normal Form* (DNF) expressions using intervals of the attributes. Two algorithms that follow this approach are CLIQUE and MAFIA.

CLIQUE is one of the first proposed subspace clustering methods. It starts, by partitioning $\mathcal{S}$ into equal-sized *windows* (or *units*) of width ε (input parameter). Then, DUs are identified in for the 1-dimensional case, by counting the number of points in each window, for example, using a histogram, as shown in Figure 1, where each window has a certain *bin* count. For example, if $\varepsilon = 0.2$, then 3 DUs are identified in A_1 as they exceed the *density threshold* τ (input parameter), while 4 DUs are identified in A_2. The total of 7 DUs from the 1-dimensional space, will then be used to form CDUs in the 2-dimensional space. Specially, CLIQUE uses the following definition to form CDUs in k-dimensional space:

Definition 1. *CDUs in k-dimensional space are formed by merging DUs from the $(k-1)$-dimensional space that share the first $(k-2)$ attributes.*

We denote Definition 1 as the *first-approach*. Having formed the CDUs, then as mentioned, only the CDUs that are actually dense are kept for further processing.

To optimize performance, CLIQUE applies a pruning technique to the subspaces by calculating their *coverage*. Subspaces with low coverage (i.e., those containing few points) are pruned to reduce computation time. This approach

helps manage the potentially large number of CDUs that CLIQUE can generate. However, this pruning strategy carries the risk of removing subspaces that might still contain valuable clusters.

Once the final DUs have been identified, CLIQUE seeks to group these DUs into clusters. A *region* is formed when adjacent DUs are connected within a subspace. A *maximal region* is the largest set of connected DUs that cannot be expanded further by adding more adjacent DUs. These maximal regions represent the extent of a cluster in the given subspace. For instance, in Figure 1, two distinct maximal regions, labeled A and B, are shown. The minimal cover for the cluster has the following DNF expression: $((0.2 \leq A_1 < 0.6) \wedge (0.4 \leq A_2 < 0.8)) \vee ((0.4 \leq A_1 < 0.8) \wedge (0.2 \leq A_2 < 0.6))$.

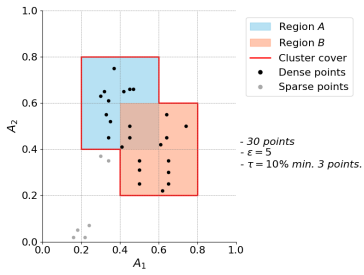


Fig. 1. 2-dimensional space containing 8 DUs and despicting two overlapping dense regions A and B.

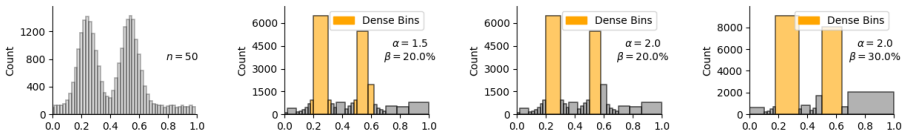
MAFIA can be seen as an extension to CLIQUE and follows mostly the same procedure. However, MAFIA has the following changes:

1. Grid sizes are adapted and automatically determined based on the data distribution.
2. CDUs are formed using all combinations of the dimensions of the DUs.
3. Do not use the pruning technique as it could result in lost information, as already noted in [2].
4. Tries to parallelize the clustering process.

Point 1 and 2 will in the following be explained in more detail.

The Adaptive Grid Algorithm starts by divide each A_i into high n-number of equal-sized windows (default $n = 1000$). Then, from left to right, two windows are merged together if their bin count are within a percentage of difference β (input parameter). That means, a high β value result in many merged windows, and vice-versa. If two windows are merged together, the highest bin count are assigned to the window, meaning the bin count that should be used to be compared with the next window. In other words, the algorithm does not use the total bin count of the two for the next comparison. Figure 2 shows an example before and after running the Adaptive algorithm using different values of α and β. Having merged the windows, the algorithm finally determines a *dense-level* for each of the windows, that determines how many points a window must contain

for a unit to be dense, thus determines whether it should be a CDU for higher dimensional spaces. The dense-level is given by: dense-level $= \frac{\alpha \cdot a \cdot |\mathcal{D}|}{D_i}$, where a is the *size* of the window, α is the *cluster dominance factor* (input parameter) and D_i is the *total range* A_i covers. Thus, a higher α results in a higher dense-level, and vice-versa, as seen in Figure 2.



(a) Before Adaptive. **(b)** After Adaptive, $\alpha = 1.5$, $\beta = 20\%$. **(c)** After Adaptive, $\alpha = 2$, $\beta = 20\%$. **(d)** After Adaptive, $\alpha = 2$, $\beta = 30\%$.

Fig. 2. Illustration of the effect of adaptive grids.

The CDU Generation of MAFIA is slightly different compared to CLIQUE. MAFIA uses the following definition when forming CDUs in k-dimensional space:

Definition 2. *CDUs in k-dimensional space are formed by merging DUs from the $(k-1)$-dimensional space that share any $(k-2)$ attributes.*

We denote Definition 2 as the *any-approach*. The difference between CLIQUE's *first-approach* and MAFIA's *any-approach* is that MAFIA explores any combination that the DUs may share. Indeed, the *any-approach* could result in many combinations, and in fact the worst case running time is $O(Ndu^2)$, where Ndu is the number of DUs. MAFIA can afford this high running time as it uses adaptive grids which reduces the number of generated DUs. Actually, CLIQUE could also use the *any-approach*, however, big grid sizes may then be required to ensure few DUs are generated. Also note that, the *any-approach* can produce repeated CDUs, which therefore needs to be eliminated before proceeding to higher dimensions. Table 1 illustrates the advantage of the *any-approach*. Here, 3 DUs are provided to the a 4-dimensional space. Both CLIQUE and MAFIA would form a 4-dimensional CDU based on DU1 and DU2, as they share the *first* two dimensions. However, only MAFIA would form a CDU based on DU1 and DU3.

	DU1	DU2	DU3
Dim. used by DU	$\{1,2,3\}$	$\{1,2,4\}$	$\{2,3,4\}$

Table 1. Illustration showing the difference of CDU generation of CLIQUE and MAFIA. The resulting CDUs are used in the 4-dimensional space.

2.2 Density-Connected approach

A drawback of grid-based methods is that the quality of clustering depends on the positiong of the grids. Figure 1 shows cluster points closely to the dense units which are missed due to the rigid grid structure and the threshold for

being dense. The problem especially occurs for clusters that are not of a shape of a hyper-rectangle. One way to address this issue is to use a density-connected algorithm, which uses the neighborhood of points to determine clusters. One such algorithm is SUBCLU.

SUBCLU extends the principles of the well-known DBSCAN algorithm [3] to higher dimensions by leveraging Lemma 1, which guides the search for clusters across subspaces.

The algorithm begins by applying DBSCAN to each attribute A_i separately in 1-dimensional subspaces. It identifies dense regions using two input parameters: ε which defines the radius of points to be considered neighbors, and $minPts$, the minimum number of points required for a point to qualify as a *core point*. If a point is not a core point but lies within ε of a core point, it is classified as a *border point*. Points that are neither core nor border points are treated as *noise*.

A *density-connected set* consists of core points that are linked through other core points, and border points that extend the cluster but do not meet the density criteria themselves. These border points are connected to the cluster via core points within their ε-neighborhood. A cluster is thus defined as the union of all density-connected sets, including both core and border points.

After detecting clusters in 1-dimensional subspaces, it uses the following definition to combine k-dimensional subspaces:

Definition 3. *Candidate k-dimensional subspaces are formed by $(k-1)$-dimensional subspaces that shares any $(k-2)$ dimensions.*

DBSCAN is reapplied to these candidate subspaces using the same ε and $minPts$ values to verify whether the clusters found in lower-dimensional subspaces persist as new dimensions are added. Lemma 1 helps to prune the search space efficiently, allowing SUBCLU to prune subspaces where no clusters are detected, thus reducing unnecessary computations.

An example is shown in Figure 3, where SUBCLU identifies a cluster in a 2-dimensional subspace consisting of 6 points. Note that if $minPts = 2$ instead of 3 it identifies two core points, and if $minPts > 3$ all points are classified as noise.

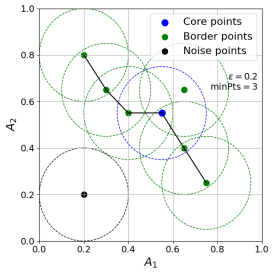


Fig. 3. Illustration of SUBCLU identifying a cluster in a 2-dimensional subspace.

3 Evaluation

The evaluation of the clustering algorithms was performed on a Intel i7 1.70 GHz processor (12th gen.) with 16 GB of RAM running Windows 11.

The evaluation of MAFIA was performed using *GPUMAFIA* [1], which was installed on a virtual machine running Ubuntu, configured with 4 CPUs and 4 GB of RAM.

CLIQUE and SUBCLU were evaluated on the main machine using *ELKI* [13]. Thus, one should be careful to compare the results of the three algorithms directly, as the execution environment and the implementation may affect the results. However, the growth rate and their clustering of the data sets can be compared.

Throughout the different experiments, a range of input parameters for the three algorithms were tested. The best found were selected to show their best performance on the corresponding data set. The complete evaluation project can be found in the GitHub repository: https://github.com/henrikdchristensen/SDU-Data-Mining-Exam. Here, additional experiments as well as detailed descriptions of how to generate the data sets and how to install and use GPUMAFIA is described.

3.1 Data set generation

The aim for the synthetic data generation was to be able to produce similar data sets as discussed in both [2,11]. That is, axis-parallel, hyper-rectangles, non-overlapping clusters in different subspaces. This was achieved using *MDCGen* [6], where different sizes and different densities of the clusters could be set, as well as which attributes for each cluster that acts as noise for which values at random is selected from a uniform distribution over the entire range of the attribute.

However, to see the one of the main advantages of SUBCLU compared to CLIQUE and MAFIA, a data set containing a Bezier-shaped cluster was created using *Artificial Cluster* [10]. In addition, a self-populated data set containing a plus-shaped cluster was created as discussed in [11].

Finally, two real-world data sets were selected to evaluate the algorithms in a more realistic setting.

All data sets were normalized so that each attribute are in the range [0, 1].

3.2 Experimental Results

Scalability with Data Set Size To evaluate the scalability of the algorithms a 20-dimensional data set containing 5 clusters in 5 different subspaces with 10% noise records was used. The data set size ranges from 10k to 1mio records. The results can be seen in Figure 4. As can be seen, MAFIA is the most scalable algorithm of the three. CLIQUE could handle up to 500k records, while SUBCLU was only able to handle up to 100k records.

CLIQUE and MAFIA achieves a linear growth rate, and SUBCLU have a quadratic growth. The linear behavior of CLIQUE and MAFIA comes from the

fact that the number of passes over the data set depends only on the cluster dimensionality. An increase in the size of the data set just means that more data has to be scanned on every pass over the data set while finding DUs resulting in a linear growth rate. SUBCLU, on the other hand, relies on DBSCAN which needs partial range queries, which are costly in terms of running time.

Similar growth rates is achieved for all algorithms as their original papers [11,2,9], however, the achieved running time in this experiment is better, which is probably due to better hardware or slightly different data sets.

Note that, the *minPts* in SUBCLU were scaled linearly with the data set size, as the clusters in the dateset were of fixed sizes, which means, by increasing the data set size, the density of the clusters increases linearly. However, for the other two algorithms, there was no need to scale any of input parameters, as they relies on the density of the units for the total amount of records.

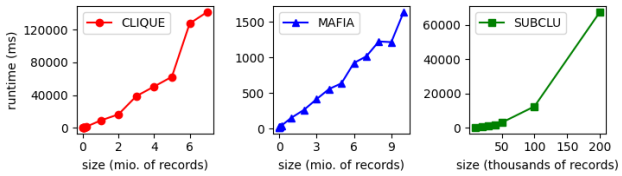
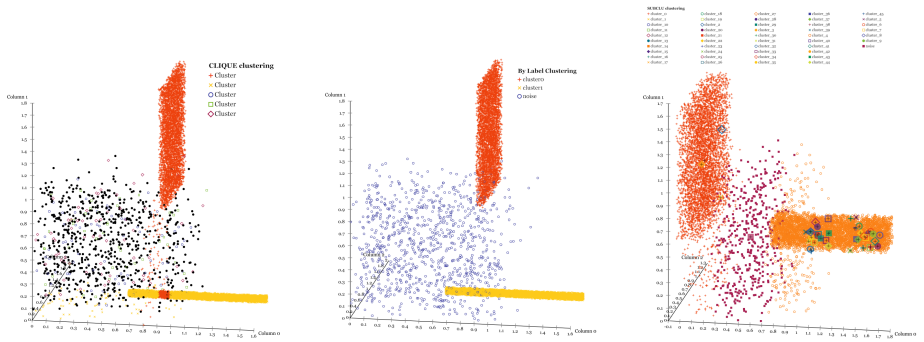


Fig. 4. Scalability with increasing data set size.

Clustering Accuracy Many different synthetic data sets were generated to test the clustering accuracy of the algorithms.

The first dataset was a 2-dimensional set containing a single cluster, shaped like a skewed plus-sign, with 10% noise added. Results similar to those reported in [11] were achieved. MAFIA successfully identified a single cluster that accurately captured the cluster's boundaries. In contrast, CLIQUE was unable to report a single cluster that correctly detected the borders. However, MAFIA's accuracy comes at a cost, as it also identifies many lower-dimensional clusters.

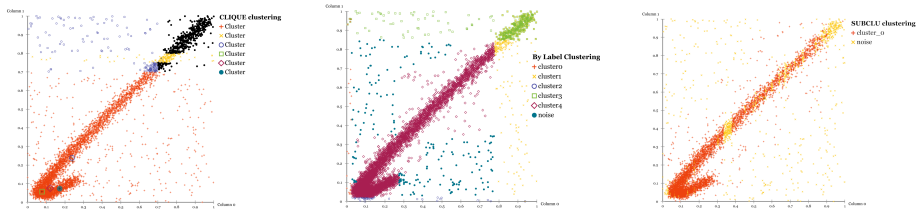
The second data was a 10-dimensional set containing 2 clusters embedded in a different 4 dimensional subspace. 10% of the data was added as noise records. Similar data set as the one used in [11]. The results can be seen in Figure 5. Here, MAFIA was able to detect the clusters without any additional lower-dimensional clusters. In contrast, CLIQUE reports some overlapping clusters and some of the noise records as clusters. SUBCLU detects also the two clusters, but reports many lower-dimensional clusters and noise records as clusters as well.



(a) CLIQUE (b) MAFIA (c) SUBCLU

Fig. 5. Two clusters in 4 different subspaces.

The third data set were a 2-dimensional set containing a single cluster formed as a Bezier curve with 10% noise added to the data set. As expected, SUBCLU outperforms the other two algorithms as can be seen in Figure 6. MAFIA and CLIQUE only partly detects the cluster and reports additional clusters and noise records as clusters. The result of SUBCLU demonstrates its use of neighborhood of points.



(a) CLIQUE (b) MAFIA (c) SUBCLU

Fig. 6. A single bezier-shaped cluster.

Scalability with Data Dimensionality and Cluster Dimensionality Figure 7 shows the scalability of the CLIQUE and MAFIA with increasing data set dimensionality. The data set contains 1 mio. records with 3 clusters in 5 different subspaces and 10% noise records. The data set dimensionality ranges from 10 to 100 dimensions. However, CLIQUE was only able to handle half of the dimensions, as the PC constantly freezes – probably due to the high memory consumption. In [2] it is noted that CLIQUE has a quadratic growth rate in terms of data set dimensionality. To investigate this even further, one could try to reduce the memory consumption by having a smaller amount of records in the data set. Nevertheless, the experiment clearly shows that MAFIA is the most scalable algorithm of the two.

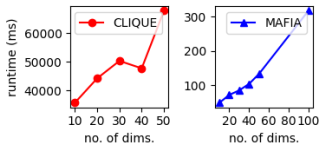


Fig. 7. Scalability with increasing data set dimensionality.

Figure 8 shows the scalability of the CLIQUE and MAFIA with increasing cluster dimensionality. A 20-dimensional data set, containing 500k records with a single cluster were used. The cluster was embedded in increasing number of dimensions starting from 10 to 100 dimensions. 10% of the records was added as noise. The data set is similar to the one used in [11]. Both algorithms heavily suffers from the increasing cluster dimensionality, however, MAFIA is able to handle a higher dimensional cluster than CLIQUE. The reason why both algorithms depends on the cluster dimensionality is due to the fact that a higher cluster dimensionality results in a large subspace coverage and a large number of CDUs. In other words, each pass on the data needs to populate a large number of CDUs, and increase in cluster dimensionality also increases the number of passes over the data set.

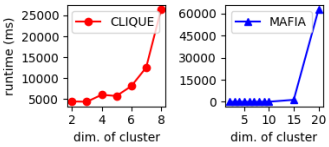


Fig. 8. Scalability with increasing cluster dimensionality.

The results reported for SUBCLU in [9] were also tried to be replicated, however, the PC constantly freezes – probably due to the high memory consumption. In future work, it would be interesting to investigate this further maybe starting with a very small data set.

Real Data Sets Two real-world data sets are used to evaluate the algorithms in a more realistic setting. Small data sets were selected for both performance and visualization reasons. Both data sets were normalized such that each attribute was in the range [0, 1].

The first data set is the well-known *Iris* data set [4], which contains 150 records of 3 different iris types (e.g. Setosa, Versicolor, Virginica). The data set contains 4 features (sepal length, sepal width, petal length and petal width). All three algorithms successfully generated clusters using their corresponding approach. However, the overall cluster quality was not optimal.

The second data set is the *Date Fruit* data set [7], which contains 898 records of 7 different date fruit types (e.g. Barhee, Deglet Nour, Sukkary, Rotab). These were obtained via a computer vision, where 34 features (e.g shape and color) was extracted. Only SUBCLU were able to produce clusters somewhat close to the true clusters but also finds many lower-dimensional clusters and small clusters.

CLIQUE and MAFIA either produced way too many clusters or close to one single cluster.

4 Discussion

CLIQUE, MAFIA, and SUBCLU are three prominent subspace clustering algorithms, each offering unique strengths and limitations. Despite the rigid grid structure CLIQUE uses it is very simple and intuitive. However, its reliance on fixed grid boundaries can lead to issues when clusters do not align with these grids, resulting in missed or fragmented clusters. Moreover, the algorithm is sensitive to grid size and density thresholds, which may not be optimal for all datasets.

MAFIA builds upon CLIQUE by introducing adaptive grid sizes, allowing it to dynamically adjust grid boundaries based on data density. This modification enables MAFIA to detect clusters more precisely, providing better resolution than CLIQUE, especially in complex datasets. Additionally, experiments indicate that MAFIA is the most scalable algorithm when it comes to handling large data sizes. However, even though the original paper states that it is not sensitive to input parameters, several experiments have shown that proper tuning is needed.

SUBCLU, on the other hand, are able to detect clusters of arbitrary shapes, which both were demonstrated using an arbitrarily shaped synthetic cluster and by two real datasets. However, SUBCLU is not as efficient as MAFIA in terms of data size scalability, particularly for very large datasets and high dimensional data sets. Also, the input parameters are critical for good clustering quality. If the parameters are not well-tuned, SUBCLU may either miss important clusters or over-partition the data into smaller, less meaningful clusters.

A critical observation across these algorithms is that their performance can vary significantly depending on the data characteristics. Experiments using synthetic data show that clustering quality is highly use-case dependent, and allowing or disallowing lower-dimensional clusters can affect outcomes. Furthermore, while these synthetic experiments highlight each algorithm's strengths, they also reveal their limitations. It is possible to generate datasets where none of these methods can identify the correct clusters, showing that subspace clustering remains a challenging problem.

All three algorithms struggle with closely clustered data points, often requiring fine-tuning of the input parameters to effectively separate clusters. This need for parameter optimization underscores the complexity of subspace clustering and the necessity for tailored solutions depending on the specific data distribution and clustering objectives.

"*The current paradigm of subspace clustering requires the user to set parameters, and clusters that satisfy these parameters are returned. We can broadly classify the parameters into cluster parameters and algorithm parameters, where cluster parameters are used in defining the clusters and algorithm parameters are used in guiding the clustering process.*" [14, p. 342]

"There are some weaknesses in using intervals. First, as the intervals are non-overlapping, and wrong positioning of the grids may lead to 'truth' subspace clusters being overlooked. Second, setting a fixed size on the intervals using ε may result in poor clustering quality, as the distribution of the objects in each attribute is different."" [14, p. 351]

"Setting a fixed τ may degrade the clustering quality, as a high threshold leads to a small number of subspace clusters, while a low threshold leads to a large number of clusters. To circumvent this problem" [14, p. 352]

"The cluster is sensitive to the tuning parameters. If wrong τ and ε are set, actual dense units may be overlooked." [14, p. 352]

"The parameters are difficult to set as they are non-meaningful and non-intuitive. A possible remedy is to try a range of parameter settings, and check for results which are stable in a particular range of parameter settings. Another possible option is to adjust the parameter setting until a suitable number of clusters is obtained." [14, p. 352]

Both can have overlapping clusters meaning that the same data point can belong to multiple clusters.

Could be interesting to investigate other algorithms like top-down approaches, amd other approaches which do not use the monotonicity property.

"It has been observed that a global density threshold, as used by SUBCLU and the grid-based approaches, leads to a bias towards a certain dimensionality: A tighter threshold which is able to separate clusters from noise well in low dimensions tends to lose clusters in higher dimensions, whereas a more relaxed threshold which is able to detect high-dimensional clusters will produce an excessive amount of low-dimensional clusters. Therefore, the dimensionality" [8, p.1:16]

"we find most bottom-up approaches free from the locality assumption. Instead, they pursue a complete enumeration approach facilitated by an APRIORI-like search. Thus, they remain in $O(2^d)$ in the worst case." [8, p.1:41]

"In order to apply an efficient bottom-up subspace search approach similar to frequent itemset mining, the cluster criterion must implement the downward closure property. Existing bottom-up approaches usually rely on a density-based cluster criterion. A limitation of most of these approaches is that the cluster criterion must use a fixed density threshold for all subspaces in order to implement the downward closure property, although the use of a global density threshold is obviously contraindicated by the observations made before concerning the curse of dimensionality. As a consequence, the same globally defined density threshold applies for subspaces of considerably different dimensionality, although a significant cluster in a higher-dimensional subspace will most likely be less dense (in an absolute sense) than a significant cluster in a lower-dimensional subspace. In order to find higher-dimensional subspaces, the user has to define a less strict density threshold. This, however, would produce a lot of meaningless lower-dimensional clusters. On the other hand, choosing a more strict density threshold, the reported lower-dimensional clusters will probably be more meaningful but higher-dimensional subspace clusters will most likely be

lost. The problem of applying a global density threshold obviously also affects the detection of meaningful subspace cluster hierarchies where lower-dimensional clusters are embedded in higher-dimensional ones. If the dimensionality of these" [8, p.1:48]

"The different heuristics and assumptions (refer to Section 5.1) and the different problems tackled (refer to Section 5.2) should always be kept in sight. In most cases, whether the trade-off between efficiency and effectiveness is tolerable will depend on the application." [8, p.1:50]

"The user must use their domain knowledge to help select and tune these settings." [12]

The main parameter for the density threshold can be hard to set across all dimensions. Adaptive grids try to solve this problem by adjusting the grid size based on the density of the data.

A big drawback of using SUBCLU is its use of DBSCAN, which takes logaritmic time for each range query if spatial index is used [5, p. 473]. SUBCLU runs DBSCAN on each d subspace initially, and then recursively run DBSCAN on each candidate subspace. This can be very time-consuming for high-dimensional data.

"application to synthetic data that has been designed with knowledge about the internal structure" [15]

Internal vs. External Evaluation - "internal" evaluation (unsupervised) - how well does the result fit to the data, assuming certain properties of “good” clusters - cohesion/separation (e.g., TD2, Silhouette) - similarity matrix (correlation, visualization) - basic assumption: the clustering algorithm is suitable for the given problem - "external" evaluation (supervised) - validation of the result independent of the algorithm, given some ground truth e.g., how well could the clustering algorithm identify known classes of objects - assumption: application to new data from the same domain and a similar problem set-up might perform similarly well - problem: the algorithm is “punished” if it actually detects “novel” patterns

Internal Evaluation - fundamental problem: is the algorithm suitable to tackle the given task? - decision (needs to precede the application) w.r.t. basic properties of the algorithm and expected characteristics of the data and clusters

Cohesion and Separation - cohesion: how strong are the cluster objects connected (how similar, pairwise, to each other)? - separation: how well is a cluster separated from other clusters? - validation index: suitable combination of cohesion and separation

Relationship Between Validity Index and Type of Clustering - cohesion and separation are suitable for convex cluster models, not for arbitrarily shaped clusters as possibly found, e.g., by density-based approaches

Problems of Internal Evaluation - suitability of the clustering model for the given dataset? - determinism? - determine k? - comparison of different cluster models? - connection between objective function of the clustering model and the validation index?

External Evaluation - based on a mapping between clustering-result and given clusters (classes), i.e., some ground truth or gold standard why not simply confusion matrix? - two basic approaches: - mapping of sets of objects - comparison of assignments for pairs of objects ("pair counting") - assessment of agreement between given and found partitions - mapping of sets: mainly information theoretic measures - pair counting: many measures available, e.g. also measures known from classification (such as F-measure etc.)

The advantage of entropy-based measures is that they do not require a one-to-one matching of sets (clusters/classes).

Problems of External Evaluation - It might establish a wrong bias on the development of clustering algorithms if they are designed with a bias to discover known classes again [Färber et al., 2010]. - Class structure and cluster structure in a dataset need not be the same [Färber et al., 2010]. - Data can contain different "truths", so different clustering results might be equally meaningful and interesting [Zimek and Vreeken, 2015, Bailey, 2013].

5 Conclusion

The analysis of CLIQUE, MAFIA, and SUBCLU highlights the strengths and limitations of each algorithm in subspace clustering. CLIQUE offers a straightforward, grid-based approach suitable for detecting axis-aligned clusters, but its reliance on fixed grids and sensitivity to parameters can limit performance. MAFIA improves upon CLIQUE by adapting grid sizes, providing better resolution and scalability for large datasets, though it still remains somewhat dependent on parameter tuning. SUBCLU, using a density-based method, excels in identifying arbitrarily shaped clusters, especially in real-world data, but it may not scale as efficiently as CLIQUE and MAFIA.

Overall, the choice of algorithm depends on the use case, data size, and the shape of clusters. The experiments demonstrated that these algorithms perform well under controlled conditions, but in highly complex data sets, none of the algorithms detects clusters accurately. Hence, while each method has its advantages, they all require careful adjustment of parameters and may need further refinement for optimal performance in diverse datasets.

References

1. Adinetz, A., Kraus, J., Meinke, J., Pleiter, D.: Gpumafia: Efficient subspace clustering with mafia on gpus pp. 838–849 (08 2013). https://doi.org/10.1007/978-3-642-40047-6_83
2. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications p. 94–105 (1998). https://doi.org/10.1145/276304.276314, https://doi.org/10.1145/276304.276314
3. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. p. 226–231. KDD'96, AAAI Press (1996)

4. Fisher, R.A.: Iris. UCI Machine Learning Repository (1936). https://doi.org/https://doi.org/10.24432/C56C76
5. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and techniques. Elsevier (6 2011)
6. Iglesias Vázquez, F., Zseby, T., Ferreira, D., Zimek, A.: Mdcgen: Multidimensional dataset generator for clustering. Journal of Classification **36** (04 2019). https://doi.org/10.1007/s00357-019-9312-3
7. Koklu, M., Kursun, R., Taspinar, Y.S., Cinar, I.: Classification of date fruits into genetic varieties using image analysis. Mathematical Problems in Engineering **2021**(1), 4793293 (2021). https://doi.org/https://doi.org/10.1155/2021/4793293, https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/4793293
8. Kriegel, H.P., Kröger, P., Zimek, A.: Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans. Knowl. Discov. Data **3**(1) (Mar 2009). https://doi.org/10.1145/1497577.1497578, https://doi.org/10.1145/1497577.1497578
9. Kröger, P., Kriegel, H.P., Kailing, K.: Density-connected subspace clustering for high-dimensional data **246-257** (04 2004). https://doi.org/10.1137/1.9781611972740.23
10. Li, W., Zhou, Z.: Ac: A data generator for evaluation of clustering (02 2022). https://doi.org/10.36227/techrxiv.19091330
11. Nagesh, H., Goil, S., Choud, A., Choudhary, P.: Adaptive grids for clustering massive data sets (01 2002). https://doi.org/10.1137/1.9781611972719.7
12. Parsons, L., Haque, E., Liu, H.: Subspace clustering for high dimensional data: A review. SIGKDD Explor. Newsl. **6**(1), 90–105 (Jun 2004). https://doi.org/10.1145/1007730.1007731, https://doi.org/10.1145/1007730.1007731
13. Schubert, E.: Automatic indexing for similarity search in ELKI **13590**, 205–213 (2022). https://doi.org/10.1007/978-3-031-17849-8_16, https://doi.org/10.1007/978-3-031-17849-8_16
14. Sim, K., Gopalkrishnan, V., Zimek, A., Cong, G.: A survey on enhanced subspace clustering. Data Mining and Knowledge Discovery **26**(2), 332–397 (2 2012). https://doi.org/10.1007/s10618-012-0258-x, https://doi.org/10.1007/s10618-012-0258-x
15. Zimek, A.: Advanced Data Mining (9 2024)