An Overlapping K-means Clustering Algorithm for Large Datasets

Ph.D. Research proposal

by

José Antonio Sánchez Tiro

Doctoral Advisors:

Ph.D. José Francisco Martínez Trinidad, INAOE

Ph.D. Jesús Ariel Carrasco Ochoa, INAOE

Instituto Nacional de Astrofísica, Óptica y Electrónica

January 16, 2025

Santa María de Tonantzintla, Puebla, CP 72840

Contents

Abstract

Overlapping clustering algorithms allow objects to belong to multiple clusters simultaneously. One of the most widely used algorithms in overlapping clustering is overlapping k-means, which has motivated the development of several other algorithms. However, although these algorithms build good clusters, they are time-consuming for large datasets. This research proposal aims to develop an overlapping k-means clustering algorithm for large datasets.

Keywords: Overlapping clustering, k-means, large datasets.

1 Introduction

After facing data collection and storage challenges, the problem has shifted to handling enormous amounts of data today. Websites, social networks, cloud-connected electronic devices, and user activity, produce large datasets continuously. This has become increasingly crucial due to the diverse fields that demand analysis for large datasets, requiring more efficient algorithms. In this context, clustering analysis is essential for exploring and identifying structures for large datasets.

Clustering is a fundamental unsupervised learning techniques in which the label of the data is unknown, and learning is done through observation rather of examples. The main task is to group objects into clusters so that objects are most similar within each cluster while objects in other cluster are differ. These similarities are a function of a specific measurement that assesses the features of the objects.

Over the years, various clustering approaches have been developed with the resulting clusters being exclusive, fuzzy, and overlapping. Disjoint or exclusive clustering involves dividing the data into subsets where each object belongs to a single cluster. Fuzzy clustering, each object can belong to all groups with a membership value between $\{0, 1\}$. Overlapping clustering reflects multiple assignments; each object can be simultaneously more than one cluster [1]. In this research proposal, the focus is on this last approach.

In the literature, there has been a wide range of different clustering algorithms that have been applied in application areas, such as biology, security, health and web search, for different purposes, such as organizing or revealing patterns. In this sense, also many applications (for example, biology, documents or health) require assigning an object to several clusters and overlapping clustering algorithms become relevant because they can be naturally associated with real datasets since they contain innate overlaps.

Despite the progress in exclusive and fuzzy clustering algorithms, the overlapping clustering algorithms proposed so far have limitations that prevent them from efficiently

satisfying practical requirements. One of limitations is the ability of algorithms to process large datasets (in high dimensionality or in the number of objects) in a reasonable runtime, which reduces their usefulness in a wide variety of applications.

This research work aims to develop an overlapping clustering algorithm for large datasets. It is intended to combine the advantages of overlapping clustering and large datasets manipulation strategies.

2 Related work and State-of-the-art

In the current era, the amount of data generated and collected has reached enormous volumes, ao ability to manage, organize and extract information from data in its natural form has become essential. Overlapping clustering emerges as a fundamental tool for identifying patterns and extracting knowledge from data.

Various works have reported clustering algorithms handling large datasets for different problems [2, 3, 4]. However, this section will focus on works introducing overlapping clustering algorithms whose contribution is on the algorithmic aspect, i.e., those works reporting algorithms considering hardware architectures for handling large datasets are out of the scope of this thesis.

Over the years, many works have addressed the problem of overlapping clustering by extending classical algorithms. The [5] proposes a categorization of overlapping clustering algorithms. Overlapping clustering algorithms are divided into hierarchical [6, 7, 8], graphical [9, 10, 11, 12, 13, 14, 11, 15], generative [16, 17, 18], partitional [19, 20, 21, 22, 23, 24, 25, 23, 26, 27], correlation [28, 29, 30], and topological [31, 32]. Partitional overlapping clustering algorithms are divided into Uncertain Memberships and Hard Memberships (geometrical and additive). [33] showed the advantages and disadvantages of some of the clustering algorithms that are based on k-means. In this research proposal, we extend the categorization proposed in [5] as: graphical, correlation, hierarchical, generative, fuzzy, additive, three-way, topological, density, and k-means-based. We extend the category of partition-based clustering to fuzzy, additive, and k-means-based, as each new category shares unique characteristics. And we add the categories of density [34] and three-way [35, 36, 37] because their algorithms do not belong to any existing category.

In the algorithms based on graph theory, a network is used that is represented as a directed or undirected graph [5]. The algorithms based on graph theory are [10, 12, 13,

14, 11, 15, 9]. The main difference between them is the criteria used to sort and select the subgraphs. These algorithms generally suffer from high computation time, many clusters, and high overlap [5, 33].

The overlapping correlation clustering algorithms seek to map of the similarity between objects that coincide as closely as possible with the clusters. The overlap is generated through the relaxation of the function to the cluster assignment; the algorithms reported are [28, 29, 30]. The disadvantages of this category are that they tend to be sensitive to noise and are very expensive, which makes it difficult to handle large datasets.

The algorithms based on hierarchies, the clustering algorithm combine the advantages of hierarchies with overlap to have representations that allow visualizing similarity structures in datasets and better understanding the relationships between objects. The main disadvantage is that they do not analyze all possible combinations to select the clusters to overlap. The works based on this category are [6, 7, 8].

Additive algorithms model cluster overlap as the sum of centroids of the clusters [38][5]. These algorithms aim at minimizing the sum of squared residuals between each object and the sum of centroids to which the object in question belongs [38]. The disadvantage is the poor scalability for large datasets. Example of some works based on additive classification are [39, 40, 41, 42, 23, 43, 27].

On the other hand, generative-based overlapping clustering algorithms use a Bayesian perspective approach to represent the distribution of data and the probability of cluster membership. Mixtures of distributions can be additive as in [16, 17] or multiplicative as in [18]. Among their disadvantages, they are not parameterizable and some of the algorithms are very expensive.

A fuzzy-based category assigns the membership of objects to clusters, where all objects have a degree of membership in a cluster. This category generally uses a membership threshold to allow an object to belong to more than one cluster [19, 44, 45, 22]. However, knowing the ideal threshold is difficult as the number of clusters and data increases.

The algorithms based on three-way clustering have three ways of organizing the data: Inside, containing objects that belong to a cluster; Outside, containing objects that do not belong to any cluster; and Partial, containing objects that may belong to a single cluster or more than one cluster. These works [35, 36, 37] generally use a threshold to control overlap to allow multi-assignment. Among their disadvantages, these algorithms are expensive when evaluating the three-way organization.

In overlapping clustering algorithms that extend topological maps (such as SOM [46]), The general idea is to assign one or more neurons to an object, searching for a subset of winning neurons and thus updating the weights of the winning neurons' subset and those of their neighborhood [5]. Examples of theses algorithms [31, 32]. Their disadvantages are that they are highly expensive and not very scalable for large datasets.

Density-based algorithms are capable of generating overlapping clustering with arbitrary and non-spherical shapes. [34] is the only algorithm that relies on density and distances to detect highly dense regions and connected clusters. OC-DD does not need to reconfigure the number of clusters. The disadvantage of OC-DD is that it is not scalable for large datasets.

The main idea of k-means-based algorithms is to solve the clustering problem by considering overlapping observations resulting from the intersections of the cluster boundaries. The k-means-based clustering algorithms are represented by prototype clusters (centroids, medoids, etc.) with their own objective functions. Besides defining a distance measure to evaluate an object with one or more prototype clusters. One of the most used overlapping clustering algorithms is k-means, an algorithm widely studied in partitional clustering. K-means aim is to minimize the distance from the centroid to each object by dividing the space into k-clusters. K-means is considered an efficient algorithm because of its heuristics and convergence rate and its simplicity in scaling it to different areas. In this research proposal, we focus on the category of algorithms based on k-means, as k-means is one of the most popular algorithms for high efficiency [47, 48, 49] and allows the user

to specify the number of clusters. In this way, a description of the following algorithms is presented.

OKM. It is an algorithm proposed in [20] that extends the k-means algorithm [50] to produce overlapping clustering with three steps: choosing arbitrary k-centers and two iterative steps, updating the clusters and assigning objects to one or more clusters. OKM's multi-assignment heuristic explores possible assignments for each object from the nearest centroids clusters to the farthest ones; for this, OKM denotes the image of an object as a combination of centroids of the clusters where the object appears, so the image is the average of the centroids of each cluster to which the object belongs. The objective function minimizes the distance of each object to its image. The heuristic starts by assigning each object to the centroid of the nearest cluster and calculating the image; then, it assigns the object to two clusters, the one of the nearest and the one of the second nearest centroid, and calculates a new image considering both clusters. If the distance from the object to the new image is less than the distance from the object to the initial image with only one cluster, the object is assigned to the two clusters. The process continues assigning the object to more clusters only if the distance to the new image with more clusters can be reduced. While updating the centroids is done locally in the clusters, unlike k-means, OKM considers multi-assignment of objects to the clusters. Among its limitations are a) like k-means the resulting clustering depends on initial centroids; this is commonly done randomly b) OKM is an expensive algorithm because it evaluates which clusters are assigned to each object. Therefore, OKM is not scalable for large data sets.

OKMED. This overlapping clustering algorithm [21] extends to K-Medoid algorithm [51] and OKM, that chooses centroids (medoids) from the data itself. In OKMED, the image is defined using medoids; for each object, its image is the object closest to the medoids to which it belongs. The objective function of OKMED minimizes the distance of each object to its image. The multi-assignment uses the heuristic of OKM adapted to medoids, searching for the nearest to the farthest medoid and assigning an object to the cluster if the distance between the object and its image decreases. For the update in

OKMED, each cluster tests each object until it finds the one that improves the objective function, which becomes the new medoid. The OKMED algorithm is expensive as the dataset grows, which is its main limitation.

WOKM. This algorithm [21] generalizes OKM and weighted k-means [52]. This algorithm adds a vector of local feature weighting for each cluster. This allows objects to be assigned to clusters according to their most important features. WOKM defines the concept of image as the weighted average of the centroids of the clusters to which each object belongs. The objective function minimizes the distance between each object and its image by combining the weight vector of the clusters to which the object belongs. The multi-assignment step, similar to the OKM algorithm, assigning each object to clusters from closest to farthest, assigns the object to the cluster if the distance of each object from its image by the weight vector of each cluster decreases. Unlike OKM, WOKM adds a weight update, which calculates a new weight for each cluster, estimates the variance in each feature, and updates it only if the objective function improves. The limitations of WOKM are a) it is sensitive to initial centroids and b) it is expensive for large datasets.

R-OKM. R-OKM [53] is an extension of OKM, but unlike OKM, it adds the cardinality of the assignments to objective function and multi-assignment, which allows it to regulate overlapping. Also, the authors develop another algorithm called Parametrized-ROKM that carries out the R-OKM strategy; however, it adds a parameter to the cardinality of the assignments to have greater control over overlapping, which improves performance compared to R-OKM. Among the limitations of R-OKM and Parameterized R-OKM are a) the value that regulates the overlap depends on the user b) random selection of the initial centroids, and d) it is expensive for large datasets.

MCOKE. This algorithm proposed by [24], extended k-means. MCOKE starts by using the k-means algorithm to assign each data point to a cluster. The maximum distance in the k-means assignment of an object to any centroid is saved as a global threshold to allow overlapping clustering. Then, MCOKE calculates the distance of each object to

each centroid of the clusters to which it does not belong. If the distance of each object to each centroid is less than the global threshold, the object is assigned to that centroid cluster. The limitations of MCOKE are a) the random initialization of centroids gives different results for the overlap threshold and b) MCOKE is expensive for large datasets since it re-evaluates the distances of the centroids to the objects to compare them with the global threshold. Other algorithms have been developed that extend MCOKE to detect outliers, such as [54, 55, 56]. In [57], a similar algorithm was proposed using the k-median clustering. However, the extensions and MCOKE do not address the problem of large datasets.

KOKM. KOKM algorithm generalizes kernel k-means and the multi-assignment heuristic of OKM. This algorithm maps input objects to a higher feature space [58]. Then, KOKM defines the concept of an object's image as the average of the clusters to which the object belongs in the feature space. The objective function minimizes the distance using the Mercer Kernel similarity measure between each object and its image in the feature space. In the multi-assignment step, each object is first assigned to the nearest cluster, and the object's image is updated. Then, the distance between the object and its image is evaluated. Next, the algorithm searches for the next nearest centroid, updates the object's image, and assigns the object to that cluster if the distance between the object and its image decreases. The process is repeated for each cluster centroid to which the object does not belong. The process ends when the distance of the image with the most clusters cannot be reduced. Centroid updates are defined as the average of the objects assigned to each cluster according to the number of assignments each object has. is highly expensive when evaluating the objective function [25], making it unsuitable for large datasets. On the other hand, KOKMII algorithm was proposed that improves the efficiency of KOKM with medoids [25]. The main difference between KOKMII and KOKM is the update step. KOKMII defines each medoid as minimizing the sum of distances over all objects assigned to the cluster using Mercer Kernel. Among its disadvantages, it is susceptible to choosing the kernel function for each dataset, and the data mapping to a feature space requires a

different parameter for each kernel function; a poor parameter affects the final clustering. Furthermore, KOKMII is only more efficient than KOKM to handle larger datasets, as KOKMII maintains the problem of being expensive when the number of objects increases.

NEO-KMeans. This algorithm [26] is a k-means variant, which allows overlapping through thresholds. Threshold represents the number of assignments that must be made in the overlapping clustering. NEO-KMeans proposes two strategies to estimate the threshold, for both first apply k-means algorithm. The first strategy uses the clustering obtained, and in each cluster, it calculates the distance of the objects to centroids and obtains the mean and standard deviation of the distances. The threshold is the number of times objects' distance to clusters centroids they do not belong to is less than the mean plus standard deviation multiplied by a fixed parameter. In the second strategy, the distances of each object to the clusters centroids it belongs to are normalized. The threshold is the number of times that objects whose distance to the clusters centroids to which they do not belong is less than 1/(k+1). Afterward, NEO-KMeans calculates the distances of each object to cluster centroids and sorts the objects in ascending order by their distance to the cluster centroids. Then, in the multi-assignment step, look for the object with the shortest distance to a centroid and assign the object to the cluster. This process is repeated until the assignments specified by the threshold are completed. Centroids are updated similarly to k-means by averaging the objects of each cluster. The limitation of NEO-KMeans is determining the threshold, since it requires an initial clustering and the calculation of distances of each object with the clusters to which it does not belong. The threshold and parameter are fixed for all clusters, which can generate errors in the overlap. Therefore, NEO-KMeans is expensive for large datasets. In [59], a variant of NEO-KMeans was made to avoid using fixed parameters; this algorithm is based on the cluster's radius and the distance between the clusters for overlapping clustering. However, the variant is more expensive than NEO-KMeans since it performs more distance calculations and thus does not solve large dataset problems.

KHM-OKM. This variant proposed by [60] combines K-Harmonic Means (KHM)

[61] and OKM. The idea of using KHM is to initialize the centroids and then apply OKM. This allows OKM not to have random initialization and to converge faster. The limitations of this variant are the same of the OKM algorithm. On the other hand, an initial clustering might not be beneficial for large datasets. In [62], a similar algorithm was proposed, but it uses WOKM to build the overlapping clustering.

OCCW. This work [63] proposed a correlation-weighted overlapping clustering algorithm similar to WOKM and OKM. OCCW defines the concept of membership weight to indicate the degree of correlation between each object and the clusters to which it belongs; the image of each object is the average weight of the clusters to which it belongs. The OCCW objective function minimizes the distance between each object and its image. Multi-assignment uses the OKM heuristic; for each object, it searches for the cluster centroid from closest to furthest but adds an intermediate step, calculating the correlation weight between each object and the multiple assignments to the clusters to which the object belongs. OCCW assigns an object to a new cluster if the distance between the object and its image decreases. Updating centroids is similar to OKM, but OCCW uses correlation weight. The disadvantages of this algorithm are that it is sensitive to initial centroids and expensive because it evaluates which cluster is best to assign. Therefore, it is expensive for large datasets.

RTKM. This algorithm [64], focuses on outlier detection and data overlap. The objective of the algorithm is outliers and data overlap control using parameters. Then, objective function is given by k-means but adds weights to each object that are associated with each cluster to determine the extent to which the object belongs to clusters. Multi-assignment depends on a threshold regulating the minimum number of clusters associated with an object. RTKM adds a step to update the weights of each object. Updating centroids of RTKM is similar to k-means. The limitations of this work are a) overlapping threshold depends on knowing the overlap of the dataset, and b) it maintains the disadvantages of the based k-means overlapping clustering algorithms, so RTKM is expensive for large datasets.

In [65], the authors propose applying the k-means clustering algorithm. Afterwards, the multi-assignment is computed for every two clusters by calculating the distance of each object of a cluster to each object of the other cluster. Then, the closest objects between the two clusters are assigned to both clusters.

2.1 Discussion

Overlapping clustering allows objects to belong to more than one cluster. As it has been shown, there are algorithms to solve overlapping clustering [1, 5, 33, 66]. Among them are the k-means-based clustering algorithms, which are the most studied due to their good results. Most of the k-means-based clustering algorithms are based on three steps: selection of centroids, multi-assignment and updating of representatives. However, these algorithms share a common limitation: They are inefficient when applied to large datasets.

The related work showed that no k-means-based overlapping clustering algorithm handles large datasets (see Table 1). Additionally, the above review shows that the multi-assignment process of all these algorithms is time-consuming. Typically, after an object is assigned to a cluster, it is required to re-evaluate whether it can be overlapped with another cluster. Therefore, these algorithms are computationally expensive when applied to large datasets, highlighting the practical need for an overlapping clustering algorithm suitable for large datasets.

The above highlights a gap in the literature on overlapping clusters. Thus, this doctoral research will focus on developing an overlapping clustering algorithm for large datasets.

Algorithms	Contribution	Type of data	Large datasets
OKM	Centroid combination and a multi-assignment heuristic	Numeric	No
OKMED	Medoid-based	Any type	No
WOKM	Local weighting of the clusters	Numeric	No
R-OKM	Overlap regulation	Numeric	No
MCOKE	Threshold with maximum distance	Numeric	No
KOKM	Kernel methods	Any type	No
KOKMII	Kernel methods and medoid-based	Any type	No
NEO-KMeans	Overlap regulation	Numeric	No
KHM-OKM	K-Harmonic-Means-OKM	Numeric	No
OCCW	Correlation weighting of the clusters	Numeric	No
RTKM	Weights per object in each cluster	Numeric	No
Algorithm proposal	**Handle large datasets**	**Numeric**	**Yes**

Table 1: Summary of k-means-based overlapping algorithms.

3 Research Proposal

First, the proposal's motivation and justification are discussed. Then, the problem is defined. The research questions, hypotheses, and objectives are presented. Finally, the methodology is described.

3.1 Motivation

Cluster analysis is an area focused on the analysis of unlabeled data. With the growth of technology and the availability of large amounts of data generated by different information systems, the need arises to develop new clustering algorithms to process large datasets for analysis and interpretation. Several techniques and strategies for clustering large datasets have been presented in the literature, ranging from parallel architectures to optimization and dataset reduction [67, 68, 69]. The challenge arises when there are overlapping clustering algorithms because they become slow for large datasets. The need to develop scalable clustering algorithms opens new opportunities in areas not explored as the overlapping clustering approach. Developing a strategy without using a parallel or distributed infrastructure to manage large datasets allows it to require a small amount of computational resources.

Currently, some works help solve partitional clustering for large datasets with different techniques, while in overlapping clustering, there is a lack of research to handle large datasets. Therefore, it is essential to continue this line of research and develop overlapping clustering algorithms that can handle large datasets.

3.2 Justification

Clustering algorithms based on k-means are the most used due to their simplicity and high efficiency, as well as because they allow the specification of the number of clusters to

build. As seen in Section 2, the most successful overlapping clustering algorithms modify the objective function of classical clustering algorithms to decide whether to overlap clusters, while others use thresholds. However, the problem with these overlapping clustering algorithms is that they are expensive and cannot handle large datasets since they need more calculations to perform the overlap.

For example, the most commonly used algorithm for solving data overlap in the literature is OKM, which has the disadvantage of its runtime, which can be extremely long for large datasets. In OKM, the multi-assignment step is the most demanding computational task (in time). Since it is necessary to calculate the distances of the n objects with the k centroids to be created, evaluating each object to determine whether it can belong to more than one cluster is also time-consuming. On the other hand, other algorithms that depend on thresholds need to run more times to obtain better results. Thus, handling large datasets in overlapping clustering based on k-means is still a problem due to the nonexistence of overlapping clustering algorithms that can handle large datasets. Thus it would be important to develop an algorithm of this type that aims to reduce the clustering time without losing much quality and that can process large datasets on computers without specialized hardware.

3.3 Problem Statement

As mentioned above, k-means-based overlapping algorithms are widely used because they achieve good results in the quality of overlapping clustering and are simple. However, handling large datasets in overlapping clustering has been a little studied area, and those algorithms are expensive for large datasets as they require more computations than k-means. Therefore, this PhD research addresses the problem of developing an overlapping clustering algorithm for large datasets.

3.4 Research Questions

1. What is the appropriate strategy to develop an efficient overlapping clustering algorithm for large datasets (high dimensionality and large numbers of objects)?
2. How can we guarantee the response time and the quality of the overlapping clustering when the dataset to be processed is very large?
3. Is it possible to develop a fast overlapping clustering algorithm for large datasets?

3.5 Hypothesis

It is possible to develop an overlapping k-means clustering algorithm for large datasets, which significantly improves the runtime and maintains the quality of the results compared to the k-means-based overlapping clustering algorithms reported in the literature.

3.6 Objectives

3.6.1 General objective

Develop an overlapping k-means clustering algorithm for large datasets that allows a trade-off between the fast and quality of the solution obtained. The algorithm is oriented to problems of large datasets in the overlapping clustering and partitional clustering areas.

3.6.2 Specific objectives

1. Develop an efficient strategy to manage large datasets by dividing them into small subsets.
2. Develop a strategy to obtain representative objects from small subsets and merge them to perform an overlapping clustering of the representative objects.

3. Developing a fast strategy to map objects to the overlapping clustering of representative objects.

4. Develop an overlapping clustering algorithm for large datasets using 1), 2), and 3).

3.7 Scope and Limitations

Large datasets will be handled algorithmically. It will not be based on hardware architectures or distributed or parallel frameworks. This research will define large datasets based on the literature on overlapping clustering and partitional clustering algorithms.

3.8 Expected Contributions

The main contribution expected at the end of this Ph.D. research is:

1. An overlapping k-means clustering algorithm for large datasets.

3.9 Methodology

The following methodology is presented to achieve the objectives and validate the hypothesis raised in this research proposal.

1. Literature review of overlapping clustering algorithms:

   (a) Identify work that addresses overlapping clustering, including those focusing on solving fast overlapping clustering or with large datasets, if any.

   (b) Identify datasets used in clustering overlapping with real data where timing issues exist. The closest works will be used in the evaluation to measure the performance of this proposal.

(c) Identify evaluation metrics used for overlapping clustering and evaluation methods when using large datasets.

2. To propose a solution to manage large datasets by dividing them into small subsets:

   (a) Identify the strategies used to handle large datasets

      i. Sampling strategy.

      ii. Projection strategy.

      iii. Approximation strategy.

      iv. Divide and conquer strategy.

   (b) Select the best strategy regarding overlapping clustering quality, performance, and scalability.

   (c) Propose strategies for small subsets that maintain a cost-benefit balance of computational resources and are suitable for overlapping clustering.

      i. Define the best strategy to handle large datasets with high dimensionality and with a large number of objects.

      ii. Define an efficient indexing structure for objects of each subset.

      iii. Define a stopping criterion to generate subsets of the dataset.

3. Develop a strategy to obtain representative objects

   (a) Identify the strategies used to obtain the representative objects of a subset.

      i. Clustering algorithms based on distances.

      ii. Clustering algorithms based on density.

      iii. Clustering algorithms based on models.

   (b) Select the fastest and most scalable strategy for selecting representative objects.

   (c) Develop a fast and noise-robust algorithm to select the best representative objects of each small subset.

4. Developing a fast strategy to map objects to the overlapping clustering of representative objects:

   (a) Develop an efficient strategy to build a set of representative objects for the whole dataset.

   (b) Define a method to index each subset and its representative objects.

   (c) Define a recursive criterion if the set of representative objects is large.

5. Develop an overlapping k-means clustering algorithm for large datasets using 2) 3) and 4).

   (a) Define an overlapping clustering algorithm to group representative objects efficiently.

      i. Develop the fast initialization algorithm.

      ii. Define the objective function with overlap.

      iii. Define the multi-assignment strategy.

      iv. Define updating of centroids.

   (b) Extrapolate the results of the overlapping clustering of the representative objects to the subsets.

6. Evaluate the proposed algorithm

   (a) The real multi-label datasets using for evaluating from the overlapping clustering algorithms reported in the literature.

   (b) The multi-label datasets reported in the literature that are considered large and those that cause problems. Additionally, they will create multi-label synthetic datasets.

   (c) Perform experimental analyses to evaluate clustering quality and overlap quality with metrics reported in the literature for evaluating overlapping clustering. For example, the Fbcubed metric evaluates overlapping clusters.

(d) Perform a runtime analysis with datasets with high dimensionality and many objects.

(e) Compare the proposed algorithm with overlapping clustering algorithms reported in the literature regarding clustering quality and runtime.

(f) Determine the computational complexity of the developed algorithm.

(g) Analysis of the results obtained and determine the limitations that affect the proposed algorithm. If these limitations exist, modify the algorithm to reduce them.

3.10 Publications Plan

Three papers are expected to be published from the research in this PhD research, of which two should be in journals and one in a conference. The publication plan is as follows:

1. The first paper will be submitted to a journal in the first quarter of 2025. This paper will publish the algorithm presented as a preliminary result of this proposal.

2. The second paper will also be submitted to a journal in the first quarter of 2026 and will present the final algorithm developed as a result of this PhD research.

3. The third paper will be submitted at a conference in the first quarter of January 2027. In it, we will report on the proposed algorithm's application to a real data from specific application.

3.11 Work Plan

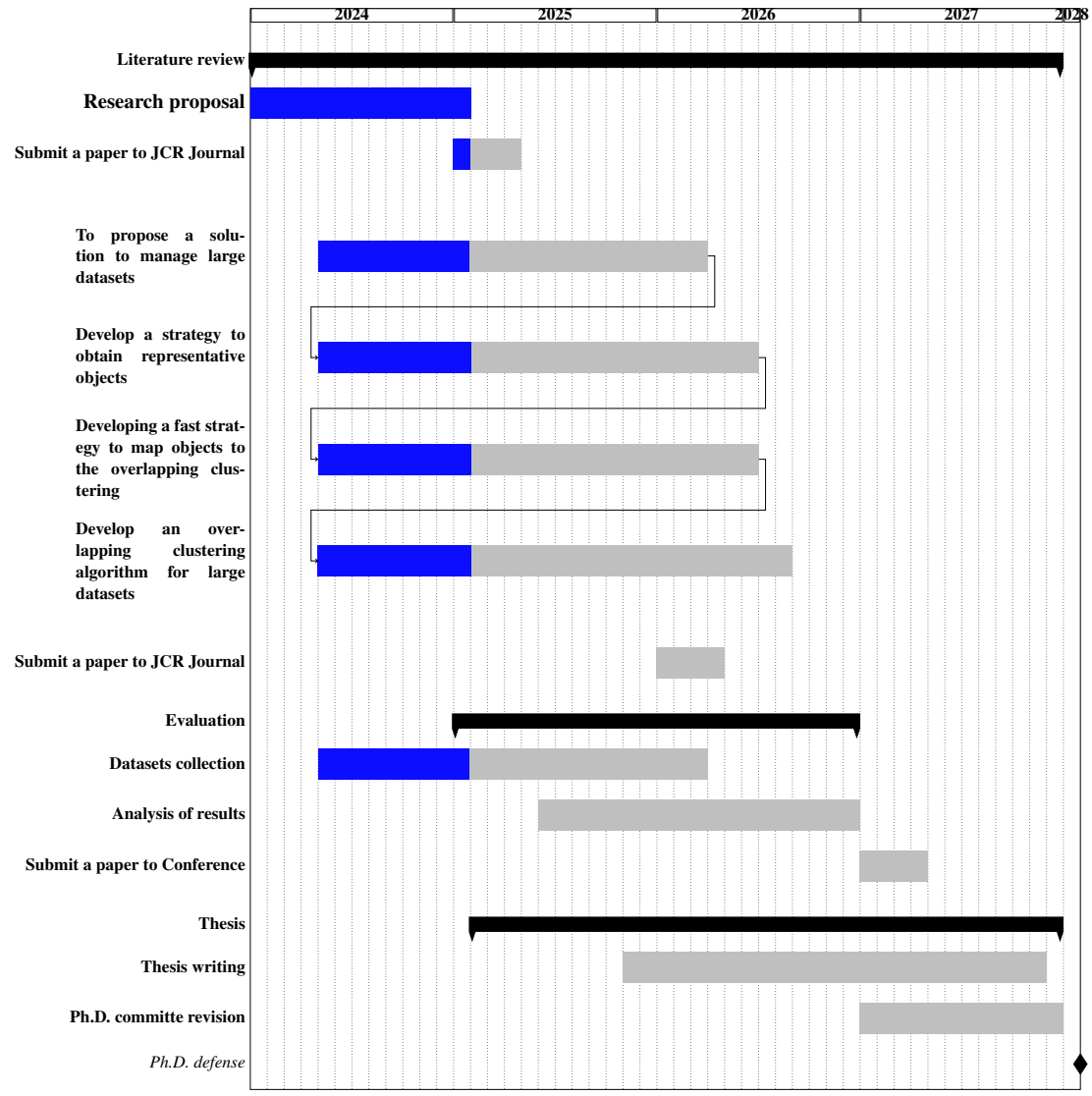


Figure 1: Research proposal schedule.

4 Preliminary Results

In this section, as preliminary results of this Ph.D. research, the first version of a k-means-based overlapping clustering algorithm for large datasets (OKCLD) has been developed (Section 4.1). Section 4.2 shows the experiments performed that asses our proposed algorithm are shown.

4.1 Proposed algorithm

The proposed algorithm uses the ”divide and conquer” heuristic, which is widely used for processing large datasets. The idea is to split a large datasets into small and manageable subsets. Then, a few objects are chosen from each subset as representatives. The selected objects from all the subsets represent the large datasets; thus, instead of building an overlapping clustering over the large dataset, we propose building an overlapping clustering over the set of representative objects. It is important to highlight that if the set of representative objects is still too large, OKCLD can be recursively applied. Once the overlapping clustering of the representative objects has been built; the remaining objects are assigned to the same clusters to which their representative objects belong to.

4.1.1 Split of the large dataset

Using the ”divide-and-conquer” heuristic for splitting small subsets and directly applying an overlapping clustering algorithm to each subset would not reduce the computational time due to the cost of these algorithms, and the cost of combining the results. Therefore, splitting the large dataset X aims to work with small and manageable subsets of objects for selecting a smaller set of representatie objects, which will be then overlapping clustered. The manageable size depends on the available computer; thus, our proposed algorithm OKCLD allows the final users, through a parameter p, to define the size of a manageable

subset of objects in their context. Thus, we randomly split a dataset X into d subsets $S = \{S_1, S_2, S_3, ..., S_d\}$ of size p, where $d = \left\lceil \frac{|X|}{p} \right\rceil$, and $X = \bigcup_{i=1}^{d} S_i$ with $S_i \cap S_j = \emptyset$, for $i \neq j$, and $\forall\, i = 1, ..., d$, $S_i \neq \emptyset$. If $|X|$ is not divisible by d, the objects in X are distributed evenly across the subsets to ensure balance. Consequently, the size of each subset is define as $|S_i| \in \{\lfloor |X|/d \rfloor, \lceil |X|/d \rceil\}$.

4.1.2 Selecting Representative Objects

As already mentioned, the idea is to process a large dataset using a smaller set that represents the whole set. Therefore, once the large dataset is randomly divided, we propose building clusters of similar objects into each subset, and from each of these clusters select an object that represents it. In this way, the representative objects will allow us to reduce the number of objects to work with and simultaneously represent the original large dataset through them. In this sense, a clustering algorithm, like the well known k-means, would allow us to build these groups and to select a representative for each cluster (the centroid).

The k-means clustering algorithm requires specifying the number of clusters to be built (the parameter k). Hence, defining the value of k is critical to selecting a good set of objects representing the large dataset. Thus, finding a balance of the size of k without being too large to produce too many representative objects but not too small to make large clusters that the centroids cannot adequately be represented by. Then, given that $d * k$ represents the total number of representative objects, which must be less or equal to p, the manageable size. Therefore, k should be such that $d * k \leq p$ and since $d = \left\lceil \frac{|X|}{p} \right\rceil$; then we compute $k = \left\lceil \frac{p^2}{|X|} \right\rceil$. To ensure that k depends on the subset size, we calculate $k = \left\lceil \frac{|S_i|^2}{|X|} \right\rceil$. When $|S_i|^2 < |X|$, we can obtain $\frac{|S_i|^2}{|X|} \leq 1$; it implies $k = 1$ according to the proposed formula. In this case, we use $k = 2$ as the number of representative objects per subset. This value was obtained experimentally since it allows OKCLD to be faster and obtain quality results similar to state-of-the-art algorithms.

4.1.3 Overlapping clustering of the Representative Objects

Once the representative objects of each subset have been computed, the union of all representative objects might be an unmanageable set (when p is too small compared to $|X|$), which occurs when $d * k > p$, resulting in a set of representative objects larger than p. In this case, when the set is manageable we can apply our algorithm recursively to obtain an overlapping clustering of the representative objects. Otherwise, we apply a k-means-based overlapping clustering algorithm on the set of all representative objects.

4.1.4 Multiasigment to the whole dataset

In the final step, the multi-assignment performed by the overlapping clustering algorithm on the set of representative objects is spread to the whole dataset. To this end, each object in the large dataset is assigned to the clusters to which its representative object belongs. This way, OKCLD indirectly finds the overlapping relationships between the original objects through the representative objects.

Algorithm 1 OKCLD Algorithm

1: **Input:** A set of n data points $X = \{x_1, x_2, \ldots, x_n\}$, the number of clusters K and p the size of each subset.
2: **Output:** An overlapping clustering of X.
3: Randomly split X without replacement to build $d = \left\lceil \frac{|X|}{p} \right\rceil$ subsets Si of size less than or equal to p and the objects are evenly distributed among the subsets.
4: **for** each subset S_i **do**
5: Set $k = \left\lceil \frac{|S_i|^2}{|X|} \right\rceil$ if $k \geq 2$, otherwise $k = 2$.
6: Apply k-means on S_i with k
7: Save the set centroids from k-means in R.
8: **if** $|R| > p$ **then**
9: OKCLD(R,K,p).
10: **else**
11: Apply a k-means-based overlapping clustering algorithm on R with K clusters.
12: Extends the multi-assignment of R to X.

Algorithm 1 shows the pseudocode of the OKCLD algorithm.

4.2 Results

Four experiments were considered to evaluate the performance of the proposed overlapping clustering algorithm: one varying the parameter p, a second comparing OKCLD and state-of-the-art algorithms, a third showing the behavior of OKCLD regarding quality and runtime on large datasets, and the last one showing the scalability of the OKCLD algorithm.

For our experiments, we used state-of-the-art overlapping clustering algorithms following the k-means approach, which, as we have already commented, according to the literature review, are commonly used. We chose the algorithms that have reported the best clustering quality, OCCW [63], and HWOKM [21]; additionally, we chose OCKMEX [57] since it is the most recent version of MCOKE [24]. We also used the alternative NEO-KMeans [59] because it reported better results than RTKM [70]. Also, we added OKM [20] since, according to [60], it is one of the most commonly used algorithms in the literature. The algorithm reported in [65] was not chosen because it is oriented to solve elliptic problems. For the HWOKM, the parameter β was set to 2.0, as suggested by its authors [62]. For the OCKMEX, the distance from the centroid to the farthest object is used as a threshold for multi-assignment for each cluster, as suggested by its authors [57].

Since our proposed algorithm allows the use of any k-means-based overlapping clustering algorithm, we will evaluate our algorithm using the following: HWOKM (OKCLD-HWOKM), NEO-KMeans (OKCLD-NEOKMEANS), OCCW (OKCLD-OCCW), OCKMEX (OKCLD-OCKMEX), and OKM (OKCLD-OKM).

In our experiment, each overlapping clustering algorithm was tested 20 times as in [5], employing different centroid initialization in each run.

In Table 2, we show the small datasets used to assess the clustering quality of our algorithm and compare it with the state-of-the-art algorithms. These datasets were chosen since they are commonly used to assess overlapping clustering algorithms, and they con-

tain few objects and few features. In Table 3, we show the large datasets used to assess the clustering quality of our proposed algorithm. These datasets were chosen since they are large in objects and features, and some have already been tested [62].

To evaluate the scalability of OKCLD in our last experiment, we generated synthetic datasets using the sci-kit-learn library's make_multilabel_classification [1] function, following the setup described in [71]. We created fifteen synthetic datasets, dividing them into three groups: scalability on the number of objects, scalability on the number of features, and scalability on the number of classes. To generate overlap in the synthetic datasets, we applied the suggested values of the make_multilabel_classification function. The values used to generate overlap were: `n_labels=2` (the average number of labels per object), and `allow_unlabeled=False` (each object belongs to at least one class). Table 4 shows the synthetic datasets.

All the overlapping clustering algorithms were implemented in the C++ programming language. The experiments were performed on a computer with 8 GB RAM, an Intel (R) core (TM) i5 – 7200 CPU at 2.50 GHz, and the Ubuntu 22.04.1 operating system.

We used FBcubed metric to compare the clustering quality of the algorithms because this metric evaluates overlapping .

FBcubed is calculated using $Bcubed_{Precision}$ and $Bcubed_{Recall}$. $Bcubed_{Precision}$ metric of x_i is defined as [60, 72]:

$$Bcubed_{precision}(x_i) = \frac{\sum_{x_j \in D(x_i)} \frac{Min(|\pi(x_i) \cap \pi(x_j)|, |C(x_i) \cap C(x_j)|)}{|\pi(x_i) \cap \pi(x_j)|}}{|D(x_i)|} \quad (1)$$

where $D(x_i)$ is the set of objects that share at least one cluster with x_i, $C(x_i)$ are the class labels associated to x_i, $\pi(x_i)$ are the cluster labels associated to x_i. $Becubed_{Recall}$ is defined

[1] https://scikit-learn.org/1.5/modules/generated/sklearn.datasets.make_multilabel_classification.html

Name	Objects	Features	Labels
Birds	645	260	19
CHD_49	555	49	6
Emotions	593	72	6
Enron	1702	1001	53
GnegativeGO	1392	1717	8
GnegativePseAAC	1392	440	8
GpositivePseAAC	519	440	4
HumanPseAAC	3106	440	14
Image	2000	294	5
Medical	978	1449	45
PlantPseAAC	978	440	12
Scene	2407	294	6
Water-quality	1060	16	14
Yeast	2417	103	14
Yelp	10810	668	5

Table 2: Multi-label small datasets to assess the clustering quality.

Name	Objects	Features	Labels
Arts	7485	23146	26
Bookmarks	87856	2150	208
Business	11214	21924	30
Computers	12444	34096	33
Education	12030	27534	33
Entertainmet	12730	32001	21
Health	9205	30605	32
IMDB	120919	1001	28
Mediamill	43907	120	101
Nus-Wide Bow	269648	500	81
Nus-Wide cVLADPlus	269600	129	81
Recreation	12830	30320	22
Science	6428	37190	40
Social	12110	52350	39
Society	14510	31802	27
Tmc2007	28696	49060	22

Table 3: Multi-label large datasets to assess the clustering quality.

Name	Objects	Features	Labels
SynO1	1,000	20	4
SynO2	10,000	20	4
SynO3	100,000	20	4
SynO4	1,000,000	20	4
SynO5	10,000,000	20	4
SynC1	10,000	20	10
SynC2	10,000	20	20
SynC3	10,000	20	30
SynC4	10,000	20	40
SynC5	10,000	20	50
SynF1	10,000	10	4
SynF2	10,000	50	4
SynF3	10,000	100	4
SynF4	10,000	1,000	4
SynF5	10,000	10,000	4

Table 4: Synthetic datasets.

as [60, 72]:

$$Bcubed_{Recall}(x_i) = \frac{\sum_{x_j \in H(x_i)} \frac{Min(|\pi(x_i) \cap \pi(x_j)|, |C(x_i) \cap C(x_j)|)}{|C(x_i) \cap C(x_j)|}}{|H(x_i)|} \tag{2}$$

where $H(x_i)$ is the set of objects that share at least one class with x_i including x_i. Then, the FBcubed metric [60, 72] is defined as:

$$FBcubed = \frac{2(\frac{1}{n}\sum_i^n Bcubed_{precision}(x_i))(\frac{1}{n}\sum_i^n Bcubed_{Recall}(x_i))}{(\frac{1}{n}\sum_i^n Bcubed_{precision}(x_i)) + (\frac{1}{n}\sum_i^n Bcubed_{Recall}(x_i))} \tag{3}$$

The FBcubed metric evaluates the consistency of an overlapping clustering by assessing the precision and recall relative to the ground truth. However, because it computes cluster and class intersections for each pair of related objects in its clusters or classes, computing FBcubed is expensive when the number of objects and classes is large. Therefore, we introduce an alternative to compute FBcubed (AproxFBcubed) to evaluate the quality of overlapping clustering for large datasets. AproxFBcubed randomly split the dataset X into g subsets of size l, where l is a parameter (the manageable size for each subset). If $g = \left\lceil \frac{|X|}{l} \right\rceil$ is not an integer g, the objects are distributed equally as possible between the

subsets to ensure balance. Once the subsets are built, the FBcubed metric is calculated for each subset using the overlapping clustering labels and the ground truth labels corresponding only to the objects in that subset. Finally, the FBcubed values obtained for each subset are averaged; we call this average the AproxFBcubed value. AproxFBcubed approximates FBcubed but reduces the overhead when processing a large dataset.

To show that AproxFBcubed results are similar to FBcubed, we applied the OKM algorithm on three datasets: emotion, scene, and yeast, which are widely used for assessing overlapping clustering algorithms. The reported results correspond to the best outcome from twenty OKM runs on each dataset, following the procedure described in [5]. The value of k was set to the number of classes in the dataset. AproxFBcubed was tested with the following manageable size values: 50, 100, 300, 500, and 1000; AproxFBcubed with each manageable size was run five times. The results of the OKM algorithm, assessed using the AproxFBcubed and FBcubed metrics for each dataset, were compared through the Wilcoxon signed rank test with a significance level of $\alpha = 0.05$. Table 5 shows the dataset, FBcubed, l, the average of calculating 5 times AproxFBcubed, the difference between FBcubed and AproxFBcubed, the FBcubed computation time, the AproxFBcubed computation time, the percentage improvement in computation time for AproxFBcubed over FBcubed, and the statistical significance of the results. The dash in table indicates that the parameter l cannot be applied to the dataset since the dataset size is smaller than l. The quality of the clustering when measured with FBcubed and with AproxFBcubed according to the statistical test there is no significant difference. In time, AproxFBcubed is faster than FBcubed in computation.

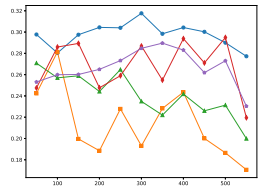
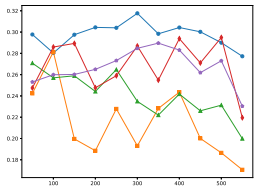
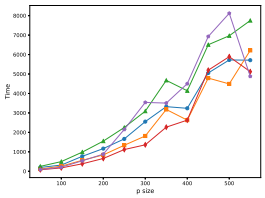
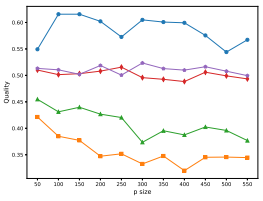
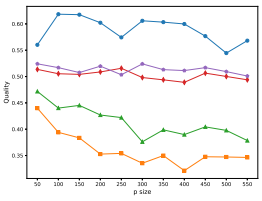
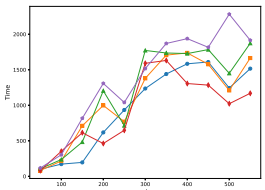
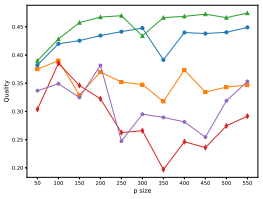
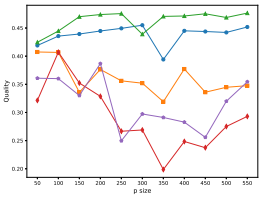
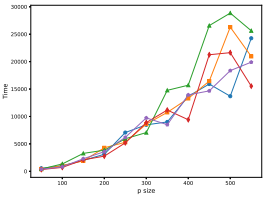
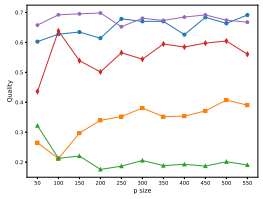
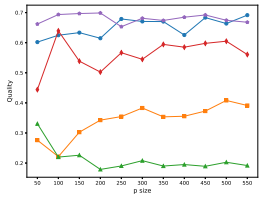
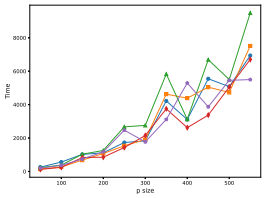
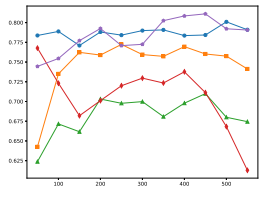
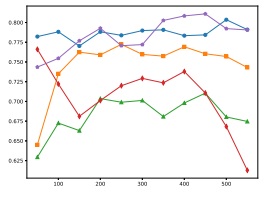
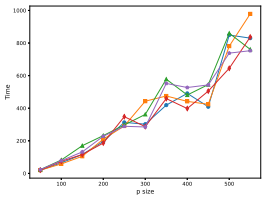
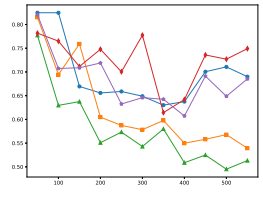
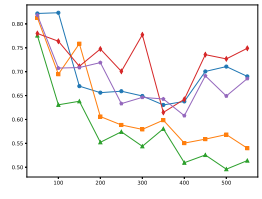
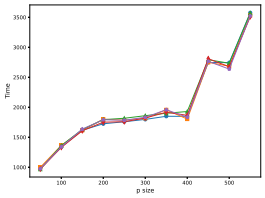
4.2.1 Experiment 1: Evaluate OKCLD with different values for the parameter p.

Since our proposed algorithm uses the parameter p (the value of manageable size) as a first experiment, we tested OKCLD with varying p values to study its behavior. This experiment tested the most used datasets in the state-of-the-art and synthetic datasets.

Dataset	FBcubed	l	AproxFBcubed	FBcubed-AproxFBcubed	FBcubed time	AproxFBcubed time	% time improvement	Significant
emotions	0.5282	50	0.538178	0.00997	27	2	92.59%	No
		100	0.531472	0.0032712		4	85.19%	No
		300	0.528678	0.00086492		12	55.56%	No
		500	0.528358	0.0001573		13	51.85%	No
		1000	-	-		-	-	-
scene	0.32321	50	0.338971	0.015764	358	9	97.49%	No
		100	0.330406	0.0071987		17	95.25%	No
		300	0.325182	0.001975		46	87.15%	No
		500	0.324593	0.0013858		83	76.82%	No
		1000	0.323879	0.00067207		137	61.73%	No
yeast	0.68439	50	0.686264	0.0018691	853	26	96.95%	No
		100	0.685683	0.0012882		43	94.96%	No
		300	0.684533	0.00013864		114	86.64%	No
		500	0.684671	0.00027657		172	79.84%	No
		1000	0.684451	0.000056803		301	64.71%	No

Table 5: Quality results of OKM with the FBcubed and AproxFBcubed metrics on Emotions, Scene, and Yest are shown.

Synthetic datasets of different sizes were used: one small (1,000 objects) and one large (100,000 objects). Other datasets with varying numbers of classes: one with 10 classes and one with 30 classes, and the other with different numbers of features: one with 10 features and one with 1000 features. These were selected as they are where FBcubed can be computed. The reported results correspond to the best result of twenty runs of each algorithm in each dataset, following the procedure described in [5]. The value of k was set to the number of classes in the dataset. The parameter l for AproxFBcubed was $l = 50$ as this value was shown to be faster statistically significant. Table 6 shows the datasets tested with various values of p (50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550), the results of each algorithm of OKCLD with FBcubed and AproxFBcubed metrics, and the time (in miliseconds) for each dataset with each value of p. The OKCLD-HWOKM algorithm showed the best clustering quality results on most of the evaluated datasets. In table 6, the values for the parameter p with the best results in terms of FBcubed are in the range from 50 to 300. Within this range, lower values of the parameter of p were associated with faster runtimes for each algorithm. Finally, a consistent runtime behavior was observed for all algorithms: as the value of p increases, the runtime also increases.

Dataset	FBcubed	AproxFBcubed	Time
Birds			
Emotions			
Scene			
Yeast			
SynO1			
SynO3			

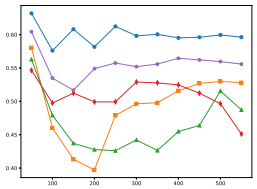
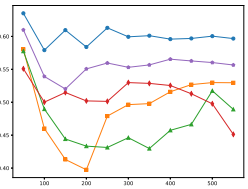
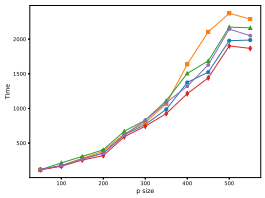
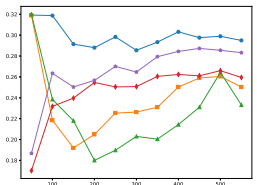
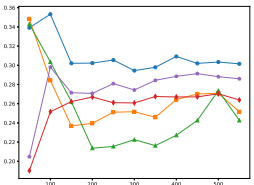
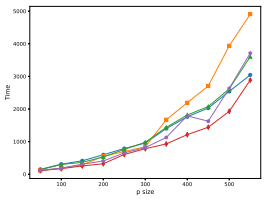
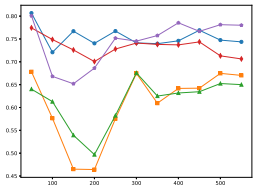
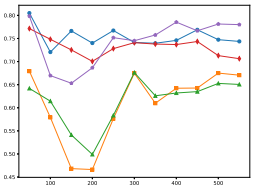
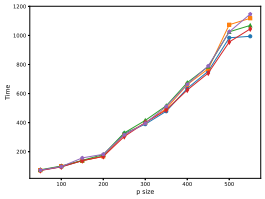
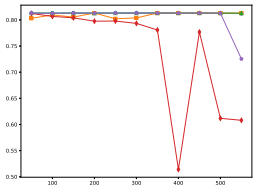
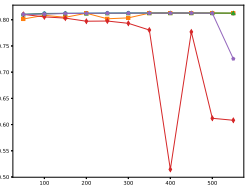
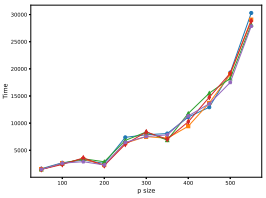
Dataset	FBcubed	AproxFBcubed	Time
SynC1			
SynC3			
SynF1			
SynF4			

Table 6: Results of varying the values for the parameter p for each algorithm on each dataset.

4.2.2 Experiment 2: Compare the proposed algorithm with state-of-the-art algorithms

In our second experiment, we compared OKCLD with state-of-the-art algorithms to evaluate whether our algorithm achieves similar results regarding FBcubed on the small datasets

listed in Table 2. For all algorithms, the parameter k was set to the number of classes in each dataset. The parameter value p of OKCLD was $p = 50$; since it was the fastest value in the first experiment and showed good results, and the best clustering result obtained of the 20 runs of each algorithm with FBcubed metric was reported.

Table 7 shows the clustering results regarding FBcubed for all algorithms tested on each small dataset. It highlights in bold the best results in the row regarding FBcubed obtained for the state-of-the-art algorithms and our proposed algorithm using them. Table 7 shows that the HWOKM algorithm achieved the best results in most data sets among the state-of-the-art overlapping clustering algorithms evaluated. The proposed OKCLD-HWOKM algorithm produced similar results and, in four datasets, outperformed HWOKM. Meanwhile, the OKCLD-NEOKMEANS, OKCLD-OCCW, OKCLD-OCKMEX, and OKCLD-OKM algorithms achieved better clustering quality in most datasets compared to NEO-KMeans, OCCW, OCKMEX, and OKM, respectively. Furthermore, in the Enron, HumanPseAAC, and Yelp datasets, all versions of OKCLD obtained the best clustering quality. This experiment shows that OKCLD achieves comparable results to the evaluated overlapping clustering algorithms based on k-means.

4.2.3 Experiment 3: OKCLD in large datasets

We present a third experiment evaluating our proposed algorithm to study its clustering quality and runtime performance on the large datasets of Table 3. The overlapping clustering algorithms tested on our proposed algorithm (OKCLD) for large datasets were: HWOKM, NEO-KMeans, OCCW, OCKMEX, and OKM. For all algorithms, the parameter k was set to the number of classes in each dataset. The parameter p of OKCLD, was $p = 50$; since it was the fastest value in the first experiment and showed good results. The parameter value l for AproxFBcubed was $l = 50$; as this value was shown to be faster and statistically significant. Table 8 reports the best clustering result regarding FBcubed from the twenty runs for each algorithm. The best results in the rows concerning FBcubed and

Dataset	HWOKM	NEO-KMeans	OCCW	OCKMEX	OKM	OKCLD-HWOKM	OKCLD-NEOKMEANS	OKCLD-OCCW	OKCLD-OCKMEX	OKCLD-OKM
Birds	**0.293088**	0.21172	0.222936	0.267353	**0.268072**	0.289599	**0.273613**	**0.245515**	**0.275016**	0.248133
CHD_49	**0.774668**	0.583091	0.488153	**0.699665**	**0.737558**	0.739679	**0.657629**	**0.540359**	0.696913	0.732747
Emotions	0.605184	0.348892	0.432705	**0.511942**	0.528201	**0.615631**	**0.446979**	**0.454789**	0.510104	**0.530439**
Enron	0.656738	0.413735	0.426414	0.539854	0.586151	**0.673042**	**0.673221**	**0.672085**	**0.672189**	**0.664363**
GnegativeGO	**0.663198**	**0.637966**	**0.796632**	**0.61506**	**0.681878**	0.657754	0.486207	0.634413	0.41569	0.636457
GnegativePseAAC	**0.508416**	0.507522	0.542914	**0.427689**	0.415722	0.467264	**0.538041**	**0.558704**	0.420954	**0.424815**
GpositivePseAAC	**0.504483**	**0.513976**	**0.539203**	0.396745	**0.501458**	0.469322	0.476347	0.517356	**0.42725**	0.457758
HumanPseAAC	0.360963	0.292801	0.33469	0.300605	0.291112	**0.363938**	**0.338805**	**0.36285**	**0.337723**	**0.355577**
Image	**0.464319**	0.396551	0.429133	0.30402	0.413324	0.462949	**0.45366**	**0.462747**	**0.434171**	**0.449472**
Medical	**0.434894**	**0.410493**	**0.424625**	0.239778	**0.390913**	0.351334	0.32924	0.372116	**0.303278**	0.193316
PlantPseAAC	**0.31988**	0.296972	0.314098	0.267966	**0.257156**	0.310793	**0.313491**	**0.321485**	**0.307794**	0.209364
Scene	**0.472611**	**0.462385**	**0.493064**	0.176984	0.335602	0.422528	0.400489	0.423614	**0.336966**	**0.349296**
Water-quality	**0.714227**	0.415795	0.358587	0.479398	0.618968	0.704476	**0.512193**	**0.606486**	**0.527272**	**0.65554**
Yeast	**0.696485**	**0.395175**	0.211607	**0.586727**	0.684395	0.634799	0.265882	**0.322052**	0.533927	**0.700979**
Yelp	0.660158	0.660158	0.628669	0.292535	0.64607	**0.718568**	**0.71862**	**0.718353**	**0.718296**	**0.717921**

Table 7: Clustering results in terms of FBcubed for state-of-the-art algorithms and their OKCLD-applied versions on small datasets. The best results from comparing each state-of-the-art algorithm with its OKCLD version are marked in bold.

AproxFBcubed obtained are highlighted in bold; the dash in table represents that FBcubed was not be computed as it is expensive with the dataset in question either due to the number of objects or the number of classes. The table shows that OKCLD-OCCW achieves the best results on eight of the large datasets. However, our proposed algorithm, which includes variants such as OKCLD-HWOKM, OKCLD-NEOKMEANS, OKCLD-OCCW, OKCLD-OCKMEX, and OKCLD-OKM, shows similar results across these variants. In particular, in the Bookmarks and Mediamill datasets, the OKCLD-OKM variant shows superior quality in terms of ApproxFBcubed compared to the other algorithms.

Figure 2, illustrates the runtime of the OKCLD versions on large datasets, with the x-axis representing the datasets and the y-axis representing the runtime. OKCLD-HWOKM was the algorithm with the longest runtime, appearing at the top of the graph in most cases. Although the Mediamill dataset had the shortest runtime among the algorithms, OKCLD-HWOKM still had the highest runtime, as the dataset contains 120 classes. This could be attributed to HWOKM using KHM to initialize the centroids, which

Dataset	OKCLD-HWOKM		OKCLD-NEOKMEANS		OKCLD-OCCW		OKCLD-OCKMEX		OKCLD-OKM	
	FBcubed	**AproxFBcubed**	**FBcubed**	**AproxFBcubed**	**FBcubed**	**AproxFBcubed**	**FBcubed**	**AproxFBcubed**	**FBcubed**	**AproxFBcubed**
Arts	0.344683	0.356642	0.342795	0.363191	0.347268	0.36713	0.343951	0.359395	**0.346568**	**0.368261**
Bookmarks	-	0.124192	-	0.125939	-	0.125495	-	0.109093	-	**0.412554**
Business	**0.846574**	**0.845249**	0.832929	0.832986	0.844846	0.843679	0.58335	0.606128	0.580501	0.585516
Computers	0.519883	0.527479	0.457874	0.489679	**0.520823**	**0.527916**	0.519289	0.529379	0.431692	0.478862
Education	0.359622	0.369731	0.327678	**0.382984**	**0.361282**	0.37144	0.35886	0.374342	0.360435	0.376683
Entertainment	0.343061	0.363042	0.339453	0.359033	**0.344068**	**0.366353**	0.340454	0.36722	0.343874	0.35847
Health	0.539755	**0.54955**	0.464743	0.478569	**0.540449**	0.54789	0.532122	0.548865	0.539875	0.546182
IMDB	-	0.246539	-	0.225538	-	**0.247316**	-	0.247834	-	0.236446
Mediamill	-	0.132833	-	0.449645	-	0.469248	-	0.422031	-	**0.605072**
Nus-Wide Bow	-	**0.364149**	-	0.364295	-	0.364204	-	0.364127	-	0.258085
Nus-Wide cVLADPlus	-	**0.373993**	-	0.366141	-	0.364438	-	0.300324	-	0.21406
Recreation	0.264306	0.28825	0.264919	0.289657	**0.266272**	**0.277917**	0.262944	0.281803	0.265083	0.28993
Science	0.25553	0.278301	0.257915	**0.285558**	**0.261781**	0.277352	0.252846	0.296137	0.259722	0.275341
Social	0.437484	0.452293	0.421133	0.430082	0.437484	0.546545	0.532122	0.545691	**0.539875**	**0.546182**
Society	0.487851	0.497173	0.456718	0.46838	0.483925	0.495196	**0.488231**	**0.498838**	0.487904	0.497509
Tmc2007	**0.631933**	0.637539	0.461578	0.493524	0.630227	0.637216	0.629371	0.631512	0.630554	**0.637648**

Table 8: Results of the FBcubed and AproxFBcubed metrics for each algorithm on different datasets. The best results are shown in bold.

increases runtime when the data includes many classes or attributes. In contrast, the other versions of OKCLD show similar runtimes across all datasets. In contrast, OKCLD-OCKMEX and OKCLD-NEOKMEANS variants showed a lower runtime on five and four large datasets respectively. The datasets that required the longest runtime were Arts, Bookmarks, Business, Computers, Education, Science, Social, and Tmc2007, as these datasets have more features and classes. This could explain the observed increase in runtime.

In this experiment, the different versions of our algorithm, including HWOKM, NEOKMEANS, OCCW, OCKMEX, and OKM, show similar clustering quality in terms of FBcubed and AproxFBcubed, achieving good results across all versions. Furthermore, the OKCLD variants strike a balance between quality and runtime.

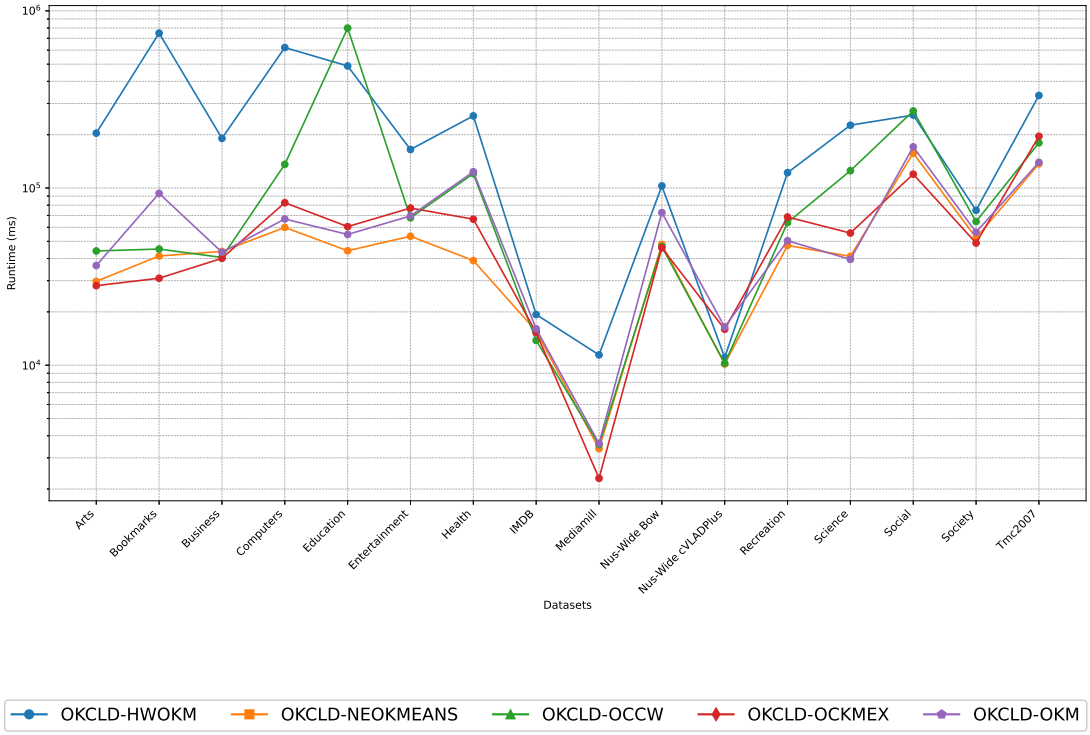


Figure 2: The runtime of each algorithm on each large dataset.

4.2.4 Experiment 4: OKCLD scalability

To assess the scalability of our proposed algorithm, we evaluated our OKCLD algorithm with the different k-means-based clustering algorithms used in previous experiments on the synthetic data shown in Table 4, varying the number of objects, features, and classes. Figure 3a shows the behavior of OKCLD's runtime when the number of objects increases, using a fixed value of $k = 4$ and 20 features. In this figure, we can see that the proposed algorithm's runtime increases as the number of objects increases in a similar way, regardless of the overlapping clustering employed. Figure 3b also shows the behavior of OKCLD's runtime as the number of classes increases, with a fixed number of objects and features (10,000 objects and 20 features). In this experiment, as the number of classes increases, our proposed algorithm employing OKCLD-OCKMEX was the fastest, showing a runtime that decreases as the number of classes grows. On the other hand, when

HWOKM was employed in OKCLD, our proposed algorithm became the most computationally expensive. Meanwhile, when the other overlapping clustering algorithms were used in OKCLD they had a similar behavior. Finally, figure 3c shows the behavior of OKCLD's runtime when the features are increased while keeping the number of objects and classes constant (10,000 objects and $k = 4$). We can see that the proposed algorithm's runtime increases similarly as the number of objects increases, regardless of the overlapping clustering employed.

In this experiment, our proposed algorithm showed that regardless of the k-means-based overlapping clustering algorithms used, the runtimes were similar for any synthetic dataset when the objects and features were increased. In contrast, when the classes increase, our proposed algorithm with the overlapping clustering algorithm OCKMEX performed best.

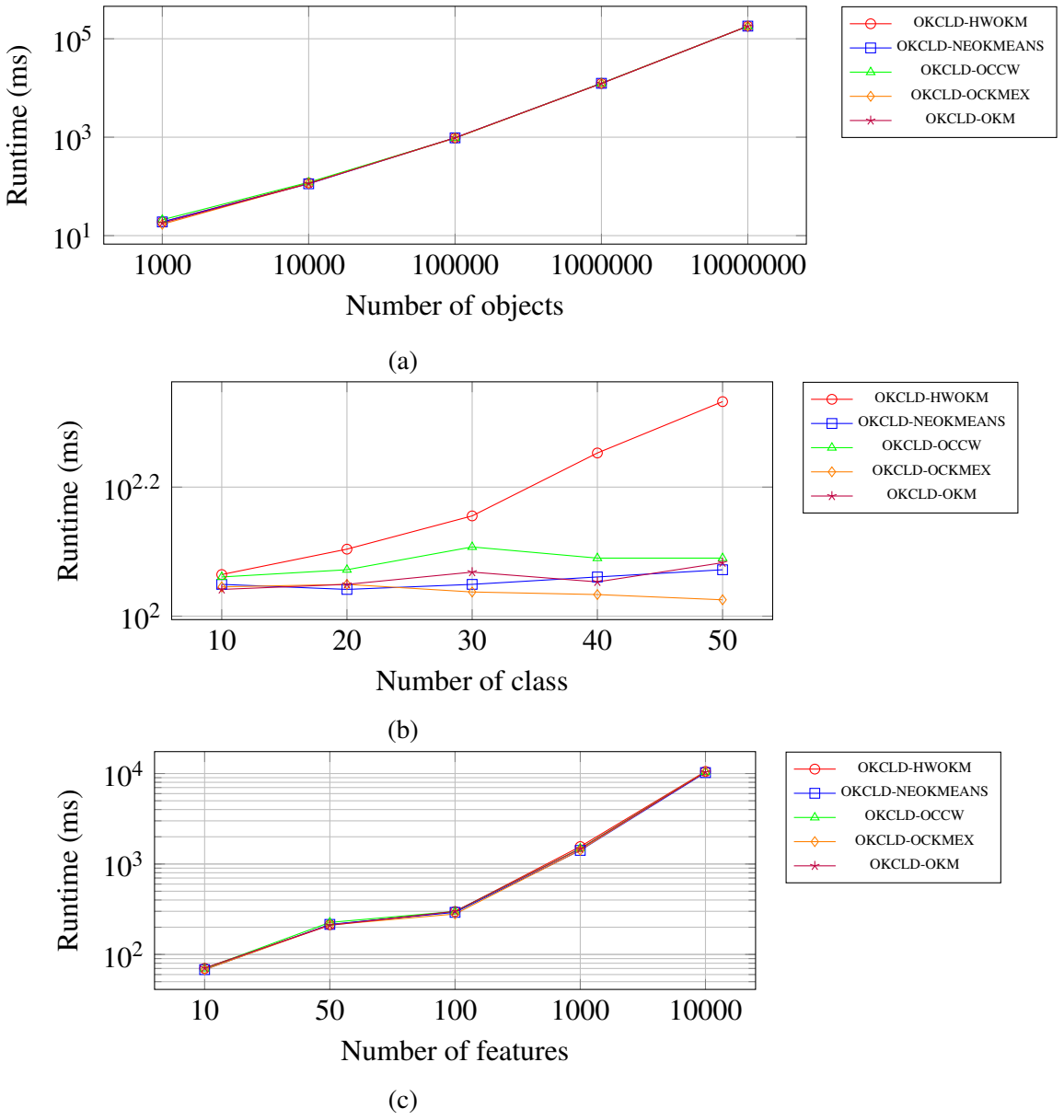


Figure 3: Runtime for our proposed algorithm varying the number of (a) objects, (b) classes, and (c) features.

5 Final Remarks

In this document, the PhD research proposal has been presented, including the motivation and justification, the problem statement, the research questions, the general and specific objectives, the expected contributions, the methodology, the work plan and the preliminary results.

From our experiment, we can conclude that regardless of the k-means-based overlapping clustering algorithm used, the proposed algorithm OKCLD gets a balance between clustering quality and runtime on large datasets, where conventional algorithms would require a lot of time, sometimes impractical.

The preliminary results in this research proposal show that improving the clustering quality and the runtime of the overlapping clustering algorithms for large datasets is still possible. Although some state-of-the-art overlapping clustering algorithms show good results in clustering quality, the proposed algorithm is competitive, achieves similar results in most cases, and still has the advantage that the proposed algorithm can be applied to large datasets. Furthermore, the results presented above guide research to develop new, faster strategies for each of the stages of the proposed algorithm. Finally, the objectives can be achieved within the stipulated time with everything presented in this research proposal.

References

[1] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[2] V. Melnykov and S. Michael, "Clustering large datasets by merging k-means solutions," *Journal of Classification*, vol. 37, no. 1, pp. 97–123, 2020.

[3] H. Cardot, P. Cénac, and J.-M. Monnez, "A fast and recursive algorithm for clustering large datasets with k-medians," *Computational Statistics & Data Analysis*, vol. 56, no. 6, pp. 1434–1449, 2012.

[4] C. Fraley, A. Raftery, and R. Wehrens, "Incremental model-based clustering for large datasets with small clusters," *Journal of Computational and Graphical Statistics*, vol. 14, no. 3, pp. 529–546, 2005.

[5] C.-E. B. N'Cir, G. Cleuziou, and N. Essoussi, "Overview of overlapping partitional clustering methods," *Partitional Clustering Algorithms*, pp. 245–275, 2015.

[6] P. Bertrand and M. F. Janowitz, "The k-weak hierarchical representations: an extension of the indexed closed weak hierarchies," *Discrete applied mathematics*, vol. 127, no. 2, pp. 199–220, 2003.

[7] E. Diday, *Orders and overlapping clusters by pyramids*. PhD thesis, INRIA, 1987.

[8] I. Jeantet, Z. Miklós, and D. Gross-Amblard, "Overlapping hierarchical clustering (ohc)," in *Advances in Intelligent Data Analysis XVIII: 18th International Symposium on Intelligent Data Analysis, IDA 2020, Konstanz, Germany, April 27–29, 2020, Proceedings 18*, pp. 261–273, Springer, 2020.

[9] A. Pérez-Suárez, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and J. E. Medina-Pagola, "Oclustr: A new graph-based algorithm for overlapping clustering," *Neurocomputing*, vol. 121, pp. 234–247, 2013.

[10] J. A. Aslam, E. Pelekhov, and D. Rus, "The star clustering algorithm for static and dynamic information organization.," *J. Graph Algorithms Appl.*, vol. 8, no. 1, pp. 95–129, 2004.

[11] M. Goldberg, S. Kelley, M. Magdon-Ismail, K. Mertsalov, and A. Wallace, "Finding overlapping communities in social networks," in *2010 IEEE Second International Conference on Social Computing*, pp. 104–113, IEEE, 2010.

[12] O. Zamir and O. Etzioni, "Web document clustering: A feasibility demonstration," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 46–54, 1998.

[13] R. J. Gil-García, J. M. Badía-Contelles, and A. Pons-Porrata, "Extended star clustering algorithm," in *Progress in Pattern Recognition, Speech and Image Analysis: 8th Iberoamerican Congress on Pattern Recognition, CIARP 2003, Havana, Cuba, November 26-29, 2003 Proceedings 8*, pp. 480–487, Springer, 2003.

[14] A. P. Suárez and J. E. M. Pagola, "A clustering algorithm based on generalized stars," in *Machine Learning and Data Mining in Pattern Recognition: 5th International Conference, MLDM 2007, Leipzig, Germany, July 18-20, 2007. Proceedings 5*, pp. 248–262, Springer, 2007.

[15] A. Pérez-Suárez, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and J. E. Medina-Pagola, "A dynamic clustering algorithm for building overlapping clusters," *Intelligent Data Analysis*, vol. 16, no. 2, pp. 211–232, 2012.

[16] A. Banerjee, C. Krumpelman, J. Ghosh, S. Basu, and R. J. Mooney, "Model-based overlapping clustering," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pp. 532–537, 2005.

[17] K. A. Heller and Z. Ghahramani, "A nonparametric bayesian approach to modeling overlapping clusters," in *Artificial Intelligence and Statistics*, pp. 187–194, PMLR, 2007.

[18] Q. Fu and A. Banerjee, "Multiplicative mixture models for overlapping clustering," in *2008 Eighth IEEE International Conference on Data Mining*, pp. 791–796, IEEE, 2008.

[19] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, "A possibilistic fuzzy c-means clustering algorithm," *IEEE transactions on fuzzy systems*, vol. 13, no. 4, pp. 517–530, 2005.

[20] G. Cleuziou, "An extended version of the k-means method for overlapping clustering," in *2008 19th International Conference on Pattern Recognition*, pp. 1–4, Dec. 2008. ISSN: 1051-4651.

[21] G. Cleuziou, "Two variants of the okm for overlapping clustering," *Advances in Knowledge Discovery and Management*, pp. 149–166, 2010.

[22] Z.-G. Liu, J. Dezert, G. Mercier, and Q. Pan, "Belief c-means: An extension of fuzzy c-means algorithm in belief functions framework," *Pattern Recognition Letters*, vol. 33, no. 3, pp. 291–300, 2012.

[23] T. F. Wilderjans, D. Depril, and I. Van Mechelen, "Additive biclustering: A comparison of one new and two existing als algorithms," *Journal of Classification*, vol. 30, no. 1, pp. 56–74, 2013.

[24] S. Baadel, F. Thabtah, and J. Lu, "Mcoke: Multi-cluster overlapping k-means extension algorithm," *International Journal of Computer and Information Engineering*, vol. 9, no. 2, pp. 427–430, 2015.

[25] C.-E. B. N'Cir, N. Essoussi, and M. Limam, "Kernel-based methods to identify overlapping clusters with linear and nonlinear boundaries," *Journal of classification*, vol. 32, pp. 176–211, 2015.

[26] J. J. Whang, I. S. Dhillon, and D. F. Gleich, "Non-exhaustive, Overlapping k-means," in *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM)*,

Proceedings, pp. 936–944, Society for Industrial and Applied Mathematics, June 2015.

[27] S. Yokoyama, "Improving algorithm for overlapping cluster analysis," *Advanced Studies in Behaviormetrics and Data Science: Essays in Honor of Akinori Okada*, pp. 329–338, 2020.

[28] F. Bonchi, A. Gionis, and A. Ukkonen, "Overlapping correlation clustering," *Knowledge and information systems*, vol. 35, pp. 1–32, 2013.

[29] P. Li, H. Dau, G. Puleo, and O. Milenkovic, "Motif clustering and overlapping clustering for social network analysis," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pp. 1–9, IEEE, 2017.

[30] B. I. Mashiach and R. Sharan, "Integer programming based algorithms for overlapping correlation clustering," in *From Computational Logic to Computational Biology: Essays Dedicated to Alfredo Ferro to Celebrate His Scientific Career*, pp. 115–127, Springer, 2024.

[31] G. Cleuziou, "Osom: A method for building overlapping topological maps," *Pattern Recognition Letters*, vol. 34, pp. 239–246, Feb. 2013.

[32] A. Khazaei, H. Khaleghzadeh, and M. Ghasemzadeh, "Foct: Fast overlapping clustering for textual data," *IEEE Access*, vol. 9, pp. 157670–157680, 2021.

[33] S. Baadel, F. Thabtah, and J. Lu, "Overlapping clustering: A review," in *2016 SAI Computing Conference (SAI)*, pp. 233–237, IEEE, 2016.

[34] C. E. B. Ncir, "A density-based method for the identification of non-disjoint clusters with arbitrary and non-spherical shapes," *Computer Science*, vol. 22, no. 2, 2021.

[35] H. Yu and Y. Wang, "Three-way decisions method for overlapping clustering," in *Rough Sets and Current Trends in Computing: 8th International Conference, RSCTC*

2012, Chengdu, China, August 17-20, 2012. Proceedings 8, pp. 277–286, Springer, 2012.

[36] M. K. Afridi, N. Azam, and J. Yao, "Variance based three-way clustering approaches for handling overlapping clustering," *International Journal of Approximate Reasoning*, vol. 118, pp. 47–63, 2020.

[37] A. Shah, B. Ali, F. Wahab, I. Ullah, F. Alqahtani, and A. Tolba, "A three-way clustering mechanism to handle overlapping regions," *IEEE Access*, 2024.

[38] J. Rossbroich, J. Durieux, and T. F. Wilderjans, "Model selection strategies for determining the optimal number of overlapping clusters in additive overlapping partitional clustering," *Journal of Classification*, vol. 39, no. 2, pp. 264–301, 2022.

[39] B. Mirkin, "The method of principal clusters," *Automation and remote control*, vol. 48, no. 10, pp. 1379–1388, 1987.

[40] B. G. Mirkin, "A sequential fitting procedure for linear data analysis models," *Journal of Classification*, vol. 7, no. 2, pp. 167–195, 1990.

[41] D. Depril, I. Van Mechelen, and B. Mirkin, "Algorithms for additive clustering of rectangular data tables," *Computational Statistics & Data Analysis*, vol. 52, no. 11, pp. 4923–4938, 2008.

[42] D. Depril, I. Van Mechelen, and T. F. Wilderjans, "Lowdimensional additive overlapping clustering," *Journal of classification*, vol. 29, no. 3, pp. 297–320, 2012.

[43] M. I. Maiza, C.-E. B. N'cir, and N. Essoussi, "Overlap regulation for additive overlapping clustering methods," in *2016 IEEE Tenth international conference on research challenges in information science (RCIS)*, pp. 1–6, IEEE, 2016.

[44] O. Ammor, A. Lachkar, K. Slaoui, and N. Rais, "Optimal fuzzy clustering in overlapping clusters.," *International Arab Journal of Information Technology (IAJIT)*, vol. 5, no. 4, 2008.

[45] M.-H. Masson and T. Denoeux, "Ecm: An evidential version of the fuzzy c-means algorithm," *Pattern Recognition*, vol. 41, no. 4, pp. 1384–1397, 2008.

[46] T. Kohonen, "Essentials of the self-organizing map," *Neural networks*, vol. 37, pp. 52–65, 2013.

[47] M. A. Mahdi, K. M. Hosny, and I. Elhenawy, "Scalable clustering algorithms for big data: A review," *IEEE Access*, vol. 9, pp. 80015–80027, 2021.

[48] P. O. Olukanmi, F. Nelwamondo, and T. Marwala, "k-means-lite: Real time clustering for large datasets," in *2018 5th International Conference on Soft Computing & Machine Intelligence (ISCMI)*, pp. 54–59, IEEE, 2018.

[49] M. Capó, A. Pérez, and J. A. Lozano, "An efficient approximation to the k-means clustering for massive data," *Knowledge-Based Systems*, vol. 117, pp. 56–69, 2017.

[50] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.

[51] D. Cao and B. Yang, "An improved k-medoids clustering algorithm," in *2010 the 2nd international conference on computer and automation engineering (ICCAE)*, vol. 3, pp. 132–135, IEEE, 2010.

[52] E. Y. Chan, W. K. Ching, M. K. Ng, and J. Z. Huang, "An optimization algorithm for clustering using weighted dissimilarity measures," *Pattern recognition*, vol. 37, no. 5, pp. 943–952, 2004.

[53] C.-E. ben N'Cir, G. Cleuziou, and N. Essoussi, "Identification of non-disjoint clusters with small and parameterizable overlaps," in *2013 International Conference on Computer Applications Technology (ICCAT)*, pp. 1–6, Jan. 2013.

[54] A. E. Danganan, A. M. Sison, and R. P. Medina, "An improved overlapping clustering algorithm to detect outlier," *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, vol. 6, no. 4, pp. 401–409, 2018.

[55] A. E. Danganan and E. M. De Los Reyes, "ehmcoke: an enhanced overlapping clustering algorithm for data analysis," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 4, pp. 2212–2222, 2021.

[56] S. Baadel, F. Thabtah, J. Lu, and S. Harguem, "Omcoke: A machine learning outlier-based overlapping clustering technique for multi-label data analysis," *Informatica*, vol. 46, no. 4, 2022.

[57] A. E. Danganan and R. P. Arceo, "Overlapping clustering with k-median extension algorithm: An effective approach for overlapping clustering," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 26, no. 3, pp. 1607–1615, 2022.

[58] C.-E. B. N'Cir, N. Essoussi, and P. Bertrand, "Kernel overlapping k-means for clustering in feature space," in *International Conference on Knowledge Discovery and Information Retrieval*, vol. 2, pp. 250–256, SCITEPRESS, 2010.

[59] T. Limungkura and P. Vateekul, "Partition-based overlapping clustering using cluster's parameters and relations," in *2017 9th International Conference on Knowledge and Smart Technology (KST)*, pp. 144–149, IEEE, 2017.

[60] S. Khanmohammadi, N. Adibeig, and S. Shanehbandy, "An improved overlapping k-means clustering method for medical applications," *Expert Systems with Applications*, vol. 67, pp. 12–18, Jan. 2017.

[61] B. Zhang, M. Hsu, and U. Dayal, "K-harmonic means-a data clustering algorithm," *Hewlett-Packard Labs Technical Report HPL-1999-124*, vol. 55, 1999.

[62] B. Beltrán, D. Vilariño, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and D. Pinto, "K-means based method for overlapping document clustering," *Journal of Intelligent & Fuzzy Systems*, vol. 39, no. 2, pp. 2127–2135, 2020.

[63] Y. Xu, Y. Yang, H. Wang, and J. Hu, "An overlapping clustering approach with correlation weight," in *International Joint Conference on Rough Sets*, pp. 611–619, Springer, 2017.

[64] O. Dorabiala, J. N. Kutz, and A. Y. Aravkin, "Robust trimmed k-means," *Pattern Recognition Letters*, vol. 161, pp. 9–16, Sept. 2022.

[65] F. Soleymani, S. Zhu, and X. Hu, "An unsupervised k-means machine learning algorithm via overlapping to improve the nodes selection for solving elliptic problems," *Engineering Analysis with Boundary Elements*, vol. 168, p. 105919, 2024.

[66] B. Beltrán and D. Vilariño, "Survey of overlapping clustering algorithms," *Computación y Sistemas*, vol. 24, no. 2, pp. 575–581, 2020.

[67] D. Pandove, S. Goel, and R. Rani, "Systematic review of clustering high-dimensional and large datasets," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 12, no. 2, pp. 1–68, 2018.

[68] Z. Dafir, Y. Lamari, and S. C. Slaoui, "A survey on parallel clustering algorithms for big data," *Artificial Intelligence Review*, vol. 54, no. 4, pp. 2411–2443, 2021.

[69] A. Mohebi, S. Aghabozorgi, T. Ying Wah, T. Herawan, and R. Yahyapour, "Iterative big data clustering algorithms: a review," *Software: Practice and Experience*, vol. 46, no. 1, pp. 107–129, 2016.

[70] O. Dorabiala, J. N. Kutz, and A. Y. Aravkin, "Robust trimmed k-means," *Pattern Recognition Letters*, vol. 161, pp. 9–16, 2022.

[71] M. Jain and C. Verma, "Adapting k-means for clustering in big data," *International Journal of Computer Applications*, vol. 101, no. 1, pp. 19–24, 2014.

[72] A. A. Aroche-Villarruel, J. A. Carrasco-Ochoa, J. F. Martínez-Trinidad, J. A. Olvera-López, and A. Pérez-Suárez, "Study of overlapping clustering algorithms based

on kmeans through fbcubed metric," in *Pattern Recognition: 6th Mexican Conference, MCPR 2014, Cancun, Mexico, June 25-28, 2014. Proceedings 6*, pp. 112–121, Springer, 2014.