OMCOKE: A Machine Learning Outlier-based Overlapping Clustering Technique for Multi-Label Data Analysis

Said Baadel[1], Fadi Thabtah[2], Joan Lu[3], Saida Harguem[4]
[1]Mt. Royal University, Calgary, Canada, [2]ASDTests, Auckland, New Zealand, [3]University of Huddersfield, Huddersfield, UK, [4]Canadian University Dubai, Dubai, UAE
Email: sbaadel@mtroyal.ca, fadi@asdtests.com, j.lu@hud.ac.uk, saida.harguem@cud.ac.ae

Keywords: K-means; Machine Learning; OMCOKE; Overlapping Clustering; Unsupervised Learning

Received: March 3, 2021

Clustering is one of the challenging machine learning techniques due to its unsupervised learning nature. While many clustering algorithms constrain objects to single clusters, K-means overlapping partitioning clustering methods assign objects to multiple clusters by relaxing the constraints and allowing objects to belong to more than one cluster to better fit hidden structures in the data. However, when datasets contain outliers, they can significantly influence the mean distance of the data objects to their respective clusters, which is a drawback. Therefore, most researchers address this problem by simply removing the outliers. This can be problematic especially in applications such as fraud detection or cybersecurity attacks risk analysis. In this study, an alternative solution to this problem is proposed that captures outliers and stores them on-the-fly within a new cluster, instead of discarding. The new algorithm is named Outlier-based Multi-Cluster Overlapping K-Means Extension (OMCOKE). Empirical results on real-life multi-label datasets were derived to compare OMCOKE's performance with other common overlapping clustering techniques. The results show that OMCOKE produced a better precision rate compared to the considered clustering algorithms. This method can benefit various stakeholders as these outliers could have real-life applications in cybersecurity, fraud detection, and the anti-phishing of websites.

Povzetek: V tej študiji je predlagana alternativna rešitev (OMCOKE), ki zajame izstope in jih sproti shrani v novo gručo, namesto da bi jih odstranila.

1 Introduction

Clustering is an unsupervised learning process that involves grouping a set of data objects into subsets, each of which has its own label based on a predefined similarity metric [2] [5]. Prior to learning, each resulting subset will contain data objects usually exhibiting similar traits but dissimilar from data objects in the other subsets [25]. In clustering, some structural characteristics are not known a priori unless some sort of domain knowledge is presented in advance (i.e. there are no labels attached to the data patterns as in the case of supervised classification), thus deeming clustering a difficult problem due to this unsupervised nature [23] [32]. Various clustering techniques such as probabilistic, distance-based, and grid-based have been explored in machine learning with the distance-based proving to be popular [1] [24].

Undoubtedly, the K-means [30], and its generic extensions and adaptations, is one of the most widely used distance-based partition-clustering algorithms [23] [26] [28]. There are many reasons attributed to this, such as it is easy to implement, its versatility allows any part to be easily modified, and its guaranteed nature to converge at a quadratic rate [16]. Thus, the K-means algorithm has been primarily utilized to deal with non-overlapping clustering problems that limit each data object to a single cluster. However, one of the main challenges of K-means and its successors is sensitivity to exceptional data (outliers). K-means often derives clusters by optimizing the mean Sum of Squared Error (SSE) (Equation 1) by calculating the Euclidean distance between the data objects and the clusters 'computed centroids.

$$SSE = \sum_{k=1}^{K} \sum_{xi \in Ck} ||xi - ck||^2 \quad (1)$$

Where Ck is the kth cluster, xi is a point in Ck, and ck is the mean of the kth cluster.

In cases when the input dataset contains few outliers, this may significantly influence the mean distance (the outlier will skew the mean and variance) of the data objects to their respective clusters, and thus K-means tends to discard outliers [6] [15] [34]. Existing algorithms that extend the K-means and allow objects to overlap include Kernel Overlapping K-means (KOKM) [9][11], Overlapping K-means (OKM) [17-18], Parametrized R-OKM [9] and Multi-Cluster Overlapping K-Means Extension (MCOKE) [3][4]. Detecting these outliers is advantageous for decision makers as these outliers could be used for fraudulent activities such as in the case of cybersecurity or a fraud insurance claim. Therefore, it will

be more useful to store these outliers (as opposed to discarding them) in a separate cluster for potential usage as they represent exceptional patterns.

This research addresses the above issue by detecting outliers during the clustering process and then storing them, making it different from the other overlapping clustering algorithms. Since the user identifies the number of k clusters a priori when running clustering algorithms, our algorithm is able to adjust to this and accommodate the outliers by adding a new cluster during the learning process called an Outlier cluster (k+1). The proposed algorithm is called Outlier-based Multi-Cluster Overlapping K-Means Extension (OMCOKE); uses the outlier cluster for later analysis by decision makers.

The current study approach adds immense value to the learning process as we save these data objects to investigate and understand their characteristics. These data objects could potentially be a result of an imbalanced data set with high cardinality (i.e. natural overlaps) and perhaps the k number of clusters, which is defined a priori, can be revised to accommodate the data and allow the algorithm to better fit the clusters.

Outliers could also indicate suspicious data objects with malicious intent. Therefore, an outlier cluster that can be investigated has profound real-life implications such as in e-banking, website phishing, cyber security, or medical screening. For example, in cyber security, historical data can reveal statistical acceptable trends through the data patterns and how they are clustered together. Any outlier objects outside the regular clustered trends will automatically raise red flags. Such red flags can be used in data analytics to alert the user of a potential security threat or an intrusion attempt.

Experimental results using real datasets indicate that OMCOKE is able to detect outliers and to produce clusters with higher precision and accuracy when compared to existing algorithms such as OKM, KOKM, and R-OKM among others.

The rest of the paper is structured as follows: Section 2 reviews the literature concerning overlapping clustering. Section 3 discusses OMCOKE and datasets used in the empirical experiments. Section 4 provides the results and analysis with a comparison of different ML clustering techniques. Lastly, we provide conclusions and further research in Section 5.

2 Literature review

Outliers are data objects or points that do not conform to the normal behaviour or model of the dataset, hence are deemed inconsistent or grossly different [12]. This data can be erroneous, but could also be classified as suspicious data in fraudulent activity; that could be useful for fraud detection, intrusion detection marketing, website phishing sites, etc.

Outlier detection is considered a task in itself; research in the data mining domain has focused on an efficient and optimal way to detect distance-based outliers. Outlier detection surveys such as by Chandola, et al. [15], Bay and Schwabacher [7], and Kadam and Pund [27] discussed several approaches used to tackle anomalies and noise data. In [8] and [31] the authors provide methods that would efficiently mine outliers in large datasets. Other recent studies have devised methods in clustering analysis that will prune or screen out outliers from the dataset such as Liu, et al. [29], Barai and Dey[6], Gan and Nk [21] , Danganan et al. [19] and Chagas et al.[14] . For example, Yu, et al. [35] proposed an outlier detection method to identify and eliminate outliers in the dataset forming an outlier-eliminated dataset (OED). The authors then applied the K-means algorithm on the OED, thereby improving the accuracy of the clustering.

Similarly, the Barai & Dey [6] approach is to divide their algorithm into two steps. The first step calculates the threshold value used in detecting outliers by taking the average of the maximum and minimum values of pairwise distance of all data. Each data point is then reiterated and compared to the threshold. Those that have a distance value greater than the threshold are deemed as outliers and are subsequently tossed out of the dataset. The second step then runs the K-means algorithm without outliers, thus improving the clustering process.

Liu et al., [29] also propose a two-phased approach for their clustering with the outlier removal (COR) algorithm. In the first phase, their method runs the K-means algorithm to generate basic partitions and discover outliers. The outliers here are identified as objects with large distances to their nearest centroid. The second phase removes the identified outlier objects and the remainder are partitioned into k clusters.

Similarly, Danganan et al [19] proposed a modification of MCOKE [3] by incorporating a median absolute deviation (MAD) that measures any potential outliers in the dataset. The authors proposed a three-phased approach in which the objects are ranked in ascending order and the distance of each object is calculated against MAD which is multiplied to a certain constant number determined by the user to obtain a decision value. If the distance of an object is greater than the decision value, that object is deemed an outlier and is pruned from the dataset.

While many studies focus on pruning and discarding the outliers to improve the classification process, rarely do we find algorithms that detect outliers simultaneously while performing clustering [19]. The K-means with outlier removal (KMOR) algorithm is similar to the standard K-means algorithm but introduces an outlier cluster (k+1) that takes into account objects that don't fit in the k defined clusters. The algorithm identifies outliers as objects that are above a calculated threshold which is defined by the average distance multiplied by a certain parameter greater or equal to 0. The average distance is calculated during the clustering phase. The KMOR algorithm requires three parameters such as the k number of clusters, the maximum number of outliers n0 (to control the number of objects being assigned as outliers), and finally, a third parameter to classify outliers and those that

are not. Two additional parameters are used to help terminate the algorithm.

All the studies mentioned above utilize the K-means partition algorithm that eventually constrains objects to single clusters. Overlapping partitioning clustering methods tends to relax or remove the constraints allowing overlaps between clusters; this better fits any hidden structures in the data and assign data objects to one or more clusters building a non-disjoint partition of the data [4] [5].

The present study focuses on overlapping partitioning methods which have several applications in real-life such as dynamic system identification, document categorization (a document belonging to different clusters), data compression, bioinformatics, image recognition, model construction, etc. [1] [21].

Extensions of the K-means that allow overlaps include Kernel Overlapping K-means (KOKM) [9,11], Overlapping K-means (OKM) [17,18], Parametrized R-OKM [10] and Multi-Cluster Overlapping K-Means Extension (MCOKE) [3][4].

The OKM algorithm is an extension of K-means that allows overlaps by using a heuristic that discovers a combinatorial set of possible assignments of the data points. For each observation, the heuristic sorts the clusters from the closest to the farthest; it then assigns the objects to those centroids in the defined order while minimizing the distance between the centroid and the observed object.

The KOKM algorithm is a variant of OKM that utilizes the use of kernel methods for overlapping clustering. The authors use two variants in their method; one is a kernelization of the Euclidean metric, similar to the one used in OKM, that calculates the distances between the objects and the clusters in a high dimensional mapping space; the second variant performs all the clustering steps where data is implicitly mapped.

The Parameterized R-OKM algorithm is another variant of OKM that lets users regulate the overlaps via a parameter. As the size of the parameter increases, the algorithm builds clusters with reduced overlaps, and vice-versa when the size of the parameter approaches zero. The PR-OKM algorithm is reduced to OKM when this parameter is set to exactly zero.

Unlike other algorithms that prune the outliers and discard them, the proposed algorithm saves them on a newly created outlier cluster during the iteration process. The present study considers the same idea as the KMOR algorithm and introduces an outlier cluster k+1 that stores the anomalies or outlier objects separately from the normal instances. As noted above, the KMOR algorithm requires users to define the maximum number of outliers, including a parameter to classify the outliers and those that are not. This is impractical in real-life scenarios in unsupervised datasets where no prior knowledge of the data is given. Also, their method requires additional parameters to help terminate the algorithm. This is not an easy feat to be determined by novice users. However, in this study we do not require users to enter parameters to terminate the algorithm or to identify the maximum number of outliers in the dataset; this makes it more practical in machine learning. None of the overlapping K-means algorithms above have the capability to detect outliers and store them for additional scrutiny. Thus, we provide additional value to the literature by introducing this new overlapping clustering method.

This study considers the key classification evaluation measures of Precision and F-measure. We evaluate and compare the results to highlight the significance of excluding the outliers in the dataset when clustering and how that improves the precision of the algorithm.

The following section discusses the proposed clustering algorithm and the dataset used for evaluation.

3 The proposed OMCOKE algorithm and experimental dataset

The proposed method is an enhancement of the MCOKE algorithm [3] that allows objects to overlap and belong to more than one cluster based on their distance comparison to the maxdis variable. Maxdist calculates the largest distance of any object assigned to any centroid during the partitioning phase for it to belong to a particular cluster. That distance is used as an outer radius of similarity threshold and as the benchmark to allow objects to belong to other clusters that were not initially assigned to them, allowing them to overlap. However, K-means, being a greedy algorithm, guarantees all objects to be assigned to a cluster including any outliers, hence the maxdist radius benchmark could easily be influenced by outliers.

The present study introduces another variable that calculates the average distance (averdist) between the object and the centroid for all clusters. Averdist acts as a new threshold for the inner radius between the object and the centroid.

$$averdist = \frac{1}{n_i}\sum_{x_i \in C_k} ||x_i - C_k||^2 \quad i = 1, 2, \dots K \qquad (2)$$

Where Ck is the kth cluster, xi is a point in Ck.

It is assumed that most objects being clustered will fall close to the inner radius threshold (i.e. close to their cluster centroid) that is based on the average distance of all objects belonging to the cluster centroids. Anomalies or outliers therefore tend to be further away from their closest cluster centroid. Objects that have a distance greater than the inner radius but less or equal to the outer radius (maxdist) are subject to further scrutiny and are flagged to ensure they are not outliers on the border of the clusters. Therefore, the maxdistThreshold defines the radius distance to be considered from the outer boundary, for example, 0.98 will mean the area covered inside the

outer boundary for objects is not to be considered an anomaly. This logic is based on the assumptions that:

a) Anomalies tend to be in sparse clusters, whereas normal instances usually belong to dense clusters

b) Anomalies tend to be far from the closest cluster centroid, whereas normal instances tend to be near their closest cluster centroid.

In cases where some knowledge of the data is known beforehand, this value can also be adjusted by the user prior to running the algorithm.

This modification logic is summarized in the pseudocode provided below.

Outlier Detection Pseudocode

```
1. For each x_i ϵ C_k
2. Do
3. If (dist (x_i, centroid C_k) ≤ averdist)
4.         Cluster ← x_i
5. Else
6.     If (dist (x_i, centroid C_k) ≥ maxdist * maxdistThreshold)
7.         Outlier_Cluster ← x_i
8.     Else
9.         Cluster ← x_i
10.    End if
11. End if
```

In Step 6 of the code above, the area covered by the maxdistThreshold is multiplied by the maxdist, calculated as a percentage of the overall maximum distance for any object belonging. This acts as the cut-off point and any object that has a distance value greater than the upper percentile of this value is deemed an outlier. Upon identification of at least one outlier, the k number of clusters entered by the user prior to running the method is incremented by 1 on the fly; the outlier object is assigned to this newly created cluster. All other identified outliers, a subset S from the initial population, are assigned to belong to this newly created cluster. Once an outlier is detected, the algorithm adds *k+1* clusters as the new output vector with the outlier cluster indexes listed as part of the output. This allows for further investigation of those data points as opposed to discarding them as is usual. When no outliers are detected, the algorithm will simply cluster with overlaps without incrementing the number of *k* clusters.

3.1 Experimental dataset

Different datasets from the Mulan: A Java Library for Multi-Label Learning repository [34] are used to evaluate the proposed algorithm's performance. The data repository hosts more than 25 different datasets in the domains of text, audio, video, music, images, and biology to mention only a few. Items of multi-label datasets can be members of multi-groups which are true for real world problems and, as a result, ideal for the study of overlapping clustering. In our empirical experiment, three different domain datasets that have been used, along with their specifications and descriptive statistics, are displayed in Table 3 and Table 4 respectively.

Data Set	Instances	# of Labels	Attribute	Cardinality
Emotions	593	6	72	1.869
Yeast	2417	14	103	4.237
Scene	2407	6	294	1.074

Table 3: Statistics of used Benchmarks

Data Set	Min	Max	Mean	StdDev
Emotions	0.01	0.195	0.069	0.031
Yeast	0.371	0.52	0.001	0.097
Scene	0.0	1.0	0.659	0.214

Table 4: Descriptive Statistics of used Benchmarks

3.2 Description of the Overlapping Datasets

This study conducted experiments on real-life overlapping datasets to measure the effectiveness of the methods used to identify such overlapping groups. The three datasets have a wide diversity in their dataset making them a suitable combination for use as benchmarks. For example, their sizes vary from 593 (Emotions) to 2417 (Yeast), their dimension (attributes) from 72 (Emotions) to 294 (Scene), cardinality (i.e. overlap rates) from 1.074 (Scene) to 4.237 (Yeast). Their application domain also varies considerably i.e. music, biology, and images.

The following is a brief description of the three datasets (Emotion, Yeast, and Scene).

3.2.1 Emotion dataset

Analyzing music signals is used in the detection of emotion in music. In this case, music can be classified into several categories at the same time since they are not usually disjointed i.e. it can make you feel both "sad" and "angry". The dataset contains sound clips that can be described by 72 attributes which were annotated by three male music experts into six emotional clusters. Only the songs that had all three experts unanimously agree on their label were kept, resulting in a total of 593 songs being selected for the dataset.

3.2.2 Yeast dataset

The Yeast dataset is classified into 14 gene groups or classes. A gene can belong to several different classes at the same time thus making this a multilabel dataset. For example, the gene YAL014W may belong in the following four groups: {Cell Growth, Cell Division}, {Cellular Organization}, {Cellular Communication, Signal

Transduction} and {Transposable elements, Viral and Plasmid Proteins}.

3.2.3 Scene dataset

The dataset contains 2407 natural scene images. The images were classed into six categories. In this case, the images can be classified into different categories at the same time since they are not usually disjointed i.e. they become multilabelled and can belong to more than one category such as field + mountain or fall foliage + mountain.

4 Experimental results

4.1 Experimental settings

To exhibit the performance of our algorithm, with respect to different measures when contrasted with a wide range of ML, the current study selected clustering algorithms using the following criteria:

a) Algorithms that utilize the partitioning method that extends the K-means algorithm
b) The algorithms use the Euclidian distance to calculate the similarities between the sets of observations
c) All algorithms work on numeric attributes only
d) All are known algorithms that have been evaluated by previous researchers in ML.

All experiments have been run on an Intel Core i7 computer with a 3.4 GHz processor and 8.0 GB RAM running on a 64-bit, Windows 10 Operating System.

We used the pair-based Precision-Recall measure that is calculated over pairs of observations. The precision-recall is computed as follows:

Where TP is a true positive decision, FP is a false positive decision (two dissimilar objects assigned to the same cluster), and FN is a false negative (two similar objects assigned to different clusters).

$$\Pr ecision = \frac{|TP|}{|TP + FP|} \tag{3}$$

$$\Re call = \frac{|TP|}{|TP + FN|} \tag{4}$$

$$F - measure = \frac{|2 * \Pr ecision * \Re call|}{|\Pr ecision + \Re call|} \tag{5}$$

Where TP is a true positive decision, FP is a false positive decision (two dissimilar objects assigned to the same cluster), and FN is a false negative (two similar objects assigned to different clusters).

4.2 Empirical results and analysis

For fair comparisons, datasets with different sizes and from different domains have been chosen and are compared to well-known algorithms that have been evaluated by previous researchers. Through experimental study, we evaluated and compared the performance of OMCOKE with three existing methods namely: Kernel Overlapping K-means (KOKM), Overlapping K-means (OKM), and Parametrized R-OKM as shown in Table 5 below.

For each experiment, we set the parameters for KOKM, OKM, and P-ROKM as follows:

- Maximum iterations = 10
- Number of clusters = 3
- Number of labels = Emotions (6), Yeast (14), and Scene (6).
- Minimal improvement = 0.01
- Alpha = 1 and 0.1 for P-ROKM algorithms.

In addition to the number of iterations and clusters set as above, the following parameters were also set in OMCOKE:

- maxdistThreshold = 0.99
- useMeasures = True

Overlapping methods will have an overlap that is greater than 1 since the objects belong to more than one cluster. The size of the overlaps affects the value of Precision i.e., there will be low value of Precision because the observations are assigned to more than one cluster.

Method	Emotion		Yeast		Scene	
	P.	***F.***	***P.***	***F.***	***P.***	***F.***
KOKMII	0.471	**0.641**	0.785	**0.878**	0.193	0.324
OKM	0.467	0.586	0.234	0.376	0.234	0.376
P-ROKM (α=1)	0.474	0.524	0.919	0.565	0.379	**0.506**
P-ROKM (α=0.1)	0.468	0.578	0.802	0.654	0.288	0.439
OMCOKE	**0.565**	0.419	**0.972**	0.496	**0.706**	0.453

Table 5: Comparison of Performace

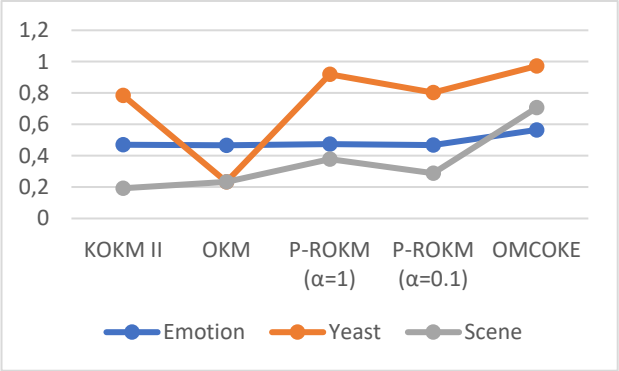

Fig. 3: Precision Accuracy of the Benchmark Datasets

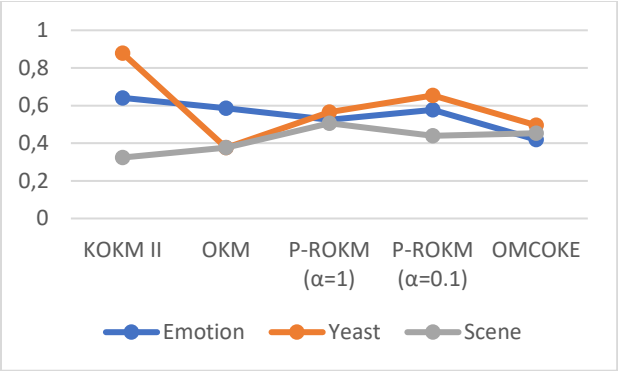

Fig. 4: F-Measure of the Benchmark Datasets

The pair-based Precision-Recall method used in the empirical results is calculated over pairs of observations. This allows for the evaluations of clusters independently and compares their partitions with different numbers of clusters in the dataset. It measures whether the predicted pair is correctly assigned in the same cluster as indicated in the true class datasets. However, the Recall measure uses a binary function to compute the relationship between pairs of observations, and not considering that those pairs of observations could also feature in multiple clusters in the overlap. This results in a biased Recall measure, especially when the cardinality in the dataset is large. Thus, we chose not to use the Recall in our experiment as a measure of OMCOKE.

It is evident from the above empirical results that the OMCOKE algorithm has a high precision rate and outperforms all the other overlapping algorithms in the study as shown in Figure 3 above. This can be attributed to the algorithm's ability to separate outliers from the rest of the data objects when assigning them to clusters. For the Emotion, Yeast, and Scene datasets, OMCOKE precision was 0.565, 0.972, and 0.706 followed by P-ROKM (α=1) at 0.474, 0.919, and 0.379 respectively.

High values of F-Measures are generally induced by the high values of Recalls as opposed to non-overlapping algorithms whose high values of F-measures are generally as a result of the Precision. When compared to the other algorithms, OMCOKE performs relatively well in the F-Measure as shown in Figure 4 below, scoring second behind P-ROKM (with α=1) in the Scene dataset; the P-ROKM method with the alpha value of 1 yielded an overlap of exactly 1 and dataset had a cardinality of 1.07.

The F-Measure values are higher for clustering methods whose overlap rates are closer to the actual cardinality of the dataset. The cardinality shown in Table 3 is the natural overlaps in the dataset i.e., the average number of categories each observation can belong to. The analysis shows that the F-Measures and Precision are significantly affected by the overlap rate in the actual dataset. Algorithms that have partitions with smaller overlaps fared well in their F-Measure meaning that they produced non-disjointed partitions that fit the data better compared to others. OMCOKE performed reasonably well in the Scene and Emotion datasets since the cardinalities of the datasets are low (1.074 and 1.869 respectively) nearing 1 but did poorly in the Yeast dataset that had an overlap of over 4. Our algorithm detected several outliers in the dataset. These are listed in Table 6 below.

Dataset	Number of Outliers	Identified Outlier Instances
Emotion	1	27
Scene	2	304; 1502
Yeast	1	1819

Table 6: Outliers Detected in the Three Datasets

As indicated, an input dataset containing a few outliers significantly influences the mean distance (the outlier will skew the mean and variance) of the data objects to their respective clusters. This explains why OMCOKE outperformed the other methods in all datasets in terms of Precision rate. This also shows that by separating the outliers from the rest of the data, the OMCOKE was able to build its model relatively closer and more acceptable to the actual overlaps in each of the datasets; this is as compared to the other methods for the precision to be higher than the rest.

5 Conclusions and future work

In this paper, some different K-means variants of overlapping clustering methods were discussed.

The proposed algorithm, with the capability of detecting outliers and treating them as a separate cluster, was evaluated and compared with three existing overlapping clustering methods namely: Kernel Overlapping K-means (KOKM), Overlapping K-means (OKM), and Parametrized R-OKM. We used real-life multi-label datasets for our experiments. The empirical results showed that the F-Measures and Precision were significantly affected by the overlap rate in the actual dataset. OMCOKE did well in the Scene dataset since the cardinality of the dataset is very low and did poorly in the

Yeast dataset that had a significant high overlap rate of over 4. However, when it came to Precision, OMCOKE outperformed the other overlapping algorithms in all datasets indicating that our method had a better detection rate of clusters and for assigning observations with a better precision after it segregated the outliers in the dataset.

The proposed algorithm detects and stores outliers during the clustering process making it different from the other overlapping clustering algorithms, thus adding value in this domain. As opposed to discarding anomalies and outliers, our method can provide tremendous benefit to cyber security experts, medical practitioners, IT administrators, data mining researchers, and other stakeholders as these outliers could have real-life applications such as fraudulent activities as in the case of cybersecurity, fraud insurance claims in the banking domain, or to help raise flags in the medical field especially in the screening process.

In future, we plan to extend the method to increment k cluster to more than 1 to cater for other dispersed objects that may not necessarily be deemed anomalies but could form dispersed clusters that have common characteristics that are somehow dissimilar from the rest of the data objects. These newly created clusters can then be fused and merged based on their similarity weights to minimize the number of clusters produced in large datasets.

References

[1] Aggarwal, C., & Reddy, C. K. (2014). *Data clustering: Algorithms and applications*. CRC Press.

[2] Arabie, L. J., Hubert, G., & DeSoete, P. (1999*). Clustering and classification.* World Scientific.

[3] Baadel, S., Thabtah, F., & Lu, J. (2015). MCOKE: Multi-Cluster Overlapping K-Means Extension Algorithm. International Journal of Computer, Control, Quantum and Information Engineering 9(2). Pp. 374-377.

[4] Baadel, S., Thabtah, F., & Lu, J. (2016). *Overlapping clustering: A review*. IEEE SAI Computing Conference, London, UK. Pp 233-237. https://doi.org/10.1109/sai.2016.7555988

[5] Baadel, S. (2021). Big Data Analytics: A Tutorial of Some Clustering Techniques. International Journal of Management and Data Analytics, 1(2). Pp 38-46.

[6] Barai, A., & Dey, L. (2017). Outlier detection and removal algorithm in K-means and hierarchical clustering. *World Journal of Computer Application and Technology, 5*(2). 24-29. https://doi.org/10.13189/wjcat.2017.050202

[7] Bay, S., & Schwabacher, M. (2003). Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In KDD. https://doi.org/10.1145/956750.956758

[8] Beltran, B., Vilarino, D., Martinez-Trinidad, J., Carrasco-Ochoa, J.A. (2020). K-means based method for overlapping document clustering. Journal of Intelligent and Fuzzy Systems, 39 (2). Pp. 2127-2135. https://doi.org/10.3233/jifs-179878

[9] BenN'Cir, C., & Essoussi, N. (2012). Overlapping patterns recognition with linear and non-linear separations using positive definite kernels. International Journal of Computer Applications (IJCA), pp 1–8. https://doi.org/10.5120/8916-2981

[10] BenN'Cir, C., Cleuziou, G., & Essoussi, N. (2013). Identification of non-disjoint clusters with small and parameterizable overlaps. In IEEE International Conference on Computer Applications Technology (ICCAT), pages 1–6. https://doi.org/10.1109/iccat.2013.6522010

[11] BenN'Cir, C., Essoussi, N., & Bertrand, P. (2010). Kernel overlapping k-means for clustering in feature space. In International Conference on Knowledge discovery and Information Retrieval (KDIR), pp 250–256. https://doi.org/10.5220/0003095102500256

[12] Berkhin P. (2006) A survey of clustering data mining techniques. In: Kogan J., Nicholas C., Teboulle M. (eds) Grouping Multidimensional Data. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-28349-8_2

[13] Boundaillier, E., & Hebrail, G. (1988). Interactive interpretation of hierarchical clustering. Intelligent Data Analysis.

[14] Chagas, G. O., Lorena, A., Dos Santos, R. (2019). A hybrid Heuristic for the overlapping Clustering problem. Applied Soft Computing. 81(105482), 1-48. https://doi.org/10.1016/j.asoc.2019.105482

[15] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1-72. https://doi.org/10.1145/1541880.1541882

[16] Celebi, M., Kingravi, H., & Vela, P. (2013). A comparative study of efficient initialization methods for the K-means clustering algorithm. Expert Systems with Applications. 40 (1). 200-210. https://doi.org/10.1016/j.eswa.2012.07.021

[17] Cleuziou, G. (2008). An extended version of the k-means method for overlapping clustering. In International Conference on Pattern Recognition ICPR, pp 1–4. https://doi.org/10.1109/icpr.2008.4761079

[18] Cleuziou, G. (2009). Two variants of the okm for overlapping clustering. Advances in Knowledge Discovery and Management. pp 149–166. https://doi.org/10.1007/978-3-642-00580-0_9

[19] Danganan, A., Sison, A., Medina, R. (2019). OCA: Overlapping Clustering application unsupervised approach for data analysis. Indonesian Journal of Electrical Engineering and Computer Science, 14 (3) pp. 1473-1478. https://doi.org/10.11591/ijeecs.v14.i3.pp1471-1478

[20] Elisseeff, A., & Weston, J. (2001). A kernel method for multi-labelled classification. In T.G. Dietterich, S. Becker, and Z. Ghahramani, (eds), Advances in Neural Information Processing Systems.

[21] Gan, G., & Ng, M. K. (2017). K-means clustering with outlier removal. Pattern Recognition Letters,

90,8-14. https://doi.org/10.1016/j.patrec.2017.03.008

[22] Höppner, F., Klawonn, F., Kruse, R., & Runkler, T. (1999). Fuzzy cluster analysis: Methods for classification, data analysis and image recognition. Wiley.

[23] Hrushka, E. R., Campello, R., Freitas, A., & Carvalho, A. (2009). A survey of evolutionary algorithms for clustering. IEEE Transactions on Systems, Man, and cybernetics, Part C. (Applications and Reviews), 39(2), 133-155. https://doi.org/10.1109/tsmcc.2008.2007252

[24] Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern Recognition Letters 31(8) 651–666. https://doi.org/10.1016/j.patrec.2009.09.011

[25] Jain, A. K., & Dubes, R. C. (1988). Algorithms for clustering data, Prentice Hall.

[26] Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. ACM Computing Surveys 31(3) 264–323. https://doi.org/10.1145/331499.331504

[27] Kadam, N. V., & Pund, M. A. (2013). Joint approach for outlier detection. International Journal of Computer Science Application, 6 (2), 445–448.

[28] Lam, D., & Wunsch, D. (2014). Clustering. Academic Press Library in Signal Processing, Signal Processing Theory and Machine Learning, (1). https://doi.org/10.1016/b978-0-12-396502-8.00020-6

[29] Liu, H., Li, J., Wu, Y., & Fu, Y. (2018). Clustering with outlier ermoval. Proceedings of ACM Sig on Knowledge Discovery and Data Mining (KDD). ACM, New York, NY, USA.

[30] McQueen, J. B. (1967). Some methods of classification and analysis of multivariate observations, In: Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297.

[31] Ramaswamy, S., Rastogi, R., & Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In SIGMOD. https://doi.org/10.1145/335191.335437

[32] Saxena, A., Prasad, M., … Gupta, A. (2017). A review of clustering techniques and developments. International Journal of Neurocomputing. 267. Pp 664-681. https://doi.org/10.1016/j.neucom.2017.06.053

[33] Trohidis, K., Tsoumakas, G., Kalliris, G., & Vlahavas, I. (2008). Multilabel classification of music into emotions. Proceeding of the 2008 International Conference on Music Information Retrieval (ISMIR 2008), pp. 325-330, Philadelphia, PA, USA. https://doi.org/10.1186/1687-4722-2011-426793

[34] Tsoumakas, G., Katakis, I., & Vlahavas, I. (2010). Mining multi-label data, data mining and knowledge discovery handbook, O. Maimon, L. Rokach (Ed.), Springer, 2nd ed., 2010. https://doi.org/10.1007/978-0-387-09823-4_34

[35] Yu, Q., Luo, Y., Chen, & C., Ding, X. (2016). Outlier-eliminated k-means clustering algorithm based on differential privacy preservation. Applied Intelligence, 45 (4). 1179–1191. https://doi.org/10.1007/s10489-016-0813-z

[36] Zhang, J. S., & Leung, Y. (2003). Robust clustering by pruning outliers. IEEE Trans. on Systems, Man, and Cybernetics – Part B 33 (6) 983–999. https://doi.org/10.1109/tsmcb.2003.816993