



**HAL**  
open science

## A deep learning object detection method to improve cluster analysis of two-dimensional data

Raphael Couturier, Pablo Gregori, Hassan Noura, Ola Salman, Abderrahmane Sider

► **To cite this version:**

Raphael Couturier, Pablo Gregori, Hassan Noura, Ola Salman, Abderrahmane Sider. A deep learning object detection method to improve cluster analysis of two-dimensional data. *Multimedia Tools and Applications*, 2024. hal-04528334

**HAL Id: hal-04528334**

**<https://hal.science/hal-04528334>**

Submitted on 1 Apr 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# A deep learning object detection method to improve cluster analysis of two-dimensional data

Raphaël Couturier\*, Pablo Gregori,  
Hassan Noura, Ola Salman and  
Abderrahmane Sider

the date of receipt and acceptance should be inserted later

**Abstract** Clustering is an unsupervised machine learning method grouping data samples into clusters of similar objects, used as a system support tool in numerous applications such as banking customers profiling, document retrieval, image segmentation, and e-commerce recommendation engines. The effectiveness of several clustering techniques is sensible to the initialization parameters, and different solutions have been proposed in the literature to overcome this limitation. They require high computational memory consumption when dealing with big data. In this paper, we propose the application of a recent object detection Deep Learning model (YOLO-v5) for assisting the initialization of classical techniques and improving their effectiveness on two-variate datasets, leveraging the accuracy and reducing dramatically the memory and time consumption of classical clustering methods.

**Keywords** Clustering algorithms; Clustering initialization methods; Clustering initialization metrics; Deep Learning object detection model

## 1 Introduction

2 Clustering is an efficient solution to split the observations in a dataset into  
3 groups, that are characterized by a high similarity of observations within each

---

R. Couturier (\* corresponding author, email: raphael.couturier@univ-fcomte.fr), H. Noura  
University of Franche-Comte (UFC), FEMTO-ST Institute, CNRS, France

P. Gregori

Instituto de Matemáticas y Aplicaciones de Castellón, Departamento de Matemáticas, Uni-  
versitat Jaume I de Castellón, E-12071 Castellón, Spain

O. Salman

American University of Beirut, Electrical and Computer Engineering Department, Lebanon

A. Sider

LIMED Laboratory, Faculty of Exact Sciences, University of Bejaia, 06000 Bejaia, Algeria

group and a high distance between different groups [1,2]. Mainly, the clustering methods are unsupervised machine learning methods that can be parametric [3] (i.e. probability-based) or non-parametric [4], and they serve as support tools for systems (see e.g. [5] for MRI images). The non-parametric clustering methods are often based on an empirical function that measures the similarity or dissimilarity between the data points [6].

In addition, the clustering methods can be classified into partitional and hierarchical ones. We shall deal with the former type, where each observation is initially assigned to a unique cluster, and they are rearranged according to an optimality criterium [7].

Partitional clustering algorithms suffer from several challenges, concerning their reliability and efficiency. The main challenge is that the stability and convergence of the clustering algorithms depend on the initialization parameters. In addition, they are generally applied to unlabeled datasets, where the number of clusters is unknown. However, some clustering methods (as k-means) require the choice of a given number of clusters. In order to estimate the most likely number of clusters, several metrics can be employed (e.g., see Table 1), some of them being computationally expensive when dealing with large-scale datasets.

We propose an efficient solution to estimate the number of clusters and their initialization parameters. It consists of training a supervised object detection neural network in order to (1) detect the number of clusters, and (2) optimize the initialization parameters of the current clustering algorithms. Without it, the efficiency of the classical algorithms is compromised, since it depends on fortunate initial guesses for the parameters, that are unlikely in high dimensional problems.

Our approach is assessed through extensive experimental results with methods such as k-means (in two implementations) and Fuzzy c-Means (FCM). The results show an improvement in both performance and stability of the considered clustering algorithms.

The rest of the paper is organized as follows: in Section 2, we briefly describe the selected clustering algorithms, the internal validation metrics, and the object detection Deep Learning (DL) tool. Then, Section 4 presents the new methodology that we propose for the optimization of the clustering problem. In Section 5, experimental results with real datasets are presented to validate the efficiency of the proposed solution. Finally, conclusions are depicted in Section 6.

## 2 Related Works

In the sequel  $X = \{x_1, x_2, \dots, x_n\}$  denotes a set of  $n$  data points in  $\mathbb{R}^d$ , the Euclidean space of dimension  $d$ . These data points can be clustered in  $m$  different clusters with centroids denoted by  $C = \{c_1, c_2, \dots, c_m\}$ .

## 2.1 Clustering Algorithms

Clustering algorithms have been extensively explored in the literature and implemented in a set of substantive areas [1, 8, 9]. We shall consider algorithms based on the Expectation-Maximisation (EM) algorithm [10]. The EM algorithm is an iterative algorithm, where each iteration consists of two steps: computing the expectation (E-Step) and maximizing it (M-Step). This probabilistic technique is generally used to solve Maximum Likelihood problems.

### 2.1.1 K-means

The k-means algorithm [11] tries to find the optimal  $m$  (given by the user) clusters in a dataset through the following steps:

1. Initialize  $m$  centroids by randomly selecting  $m$  data-points without replacement.
2. Iterate the following steps, until a maximum number of steps is reached, or until the difference between two successive sets of centers is below some error threshold.
  - (a) Update of the cluster membership: for each data-point, compute the Euclidean distance to each one of the present centroids, and assign that data-point (only) to the closest centroid (E step).
  - (b) Update the centroids: compute the new centroids as the average of the data-points within the corresponding updated cluster (M step).

The k-means algorithm tries to minimize the inertia, which is defined as the sum of the squared distances between data-points and their centroids. We can express it by the Equation

$$J_m(X, c) = \sum_{i=1}^n \sum_{k=1}^m u_{ki} \|x_i - c_k\|^2, \quad (1)$$

where  $u_{ki}$  is a membership matrix (with value 1 only if the data-point  $x_i$  belongs to the cluster with center  $c_k$ , and null otherwise).

The initial centroids are usually chosen at random from the dataset (Lloyd's implementation [12]). k-means++ [13] is a smarter heuristic for setting the initial centroids to achieve faster convergence. In any case, the number of clusters must be given beforehand, and the attainment of the minimum inertia is not ensured, as the algorithm might stick to a local minimum.

The x-means algorithm [14] is a modified version of k-means, which does not require a given number of clusters: it relies on the Bayesian Information Criterion (BIC) in order to decide the number of clusters. However, the stability of x-means still depends on its initialization. Let us mention that the computation of the BIC measure presents a high computation overhead with large datasets.

The Fuzzy c-Means (FCM) algorithm, first presented by Dunn [15] and later improved by Bezdek [16], leverages the fuzzy algebra to express the simultaneous membership of a data-point to different clusters. It computes a *soft*

84 partition of the dataset. In order to get the partition with FCM, it is sufficient  
 85 to consider that each data-point shall belong to the cluster with maximum  
 86 membership degree. The FCM algorithm tries to minimize the inertia, given  
 87 by the equation

$$J_z(X, c) = \sum_{i=1}^n \sum_{k=1}^m (u_{ki})^z \|x_i - c_k\|_A^2, \quad (2)$$

88 where  $z$  is called the fuzziness parameter, initialized to a value between 2  
 89 and 3, and  $\|\cdot\|_A^2$  stands for any mathematical distance. It entails the following  
 90 steps:

- 91 1. Initialize a random membership matrix  $U^0 = (u_{ki})$  such that  $\sum_{k=1}^m u_{ki} = 1$   
 92 for any data-point  $x_i$ ,  $i = 1, 2, \dots, n$ .
- 93 2. Iterate the following steps, until a maximum number of steps is reached,  
 94 or until the difference between two successive matrices is below some error  
 95 threshold ( $\|U^{t+1} - U^t\|^2 < \epsilon$ ):
  - 96 (a) Update the centroids using the membership matrix  $U^t$  (E-Step).
  - 97 (b) Update the membership matrix  $U^{t+1}$  using the computed centroids in  
 98 the previous step (M-Step).

99 Just like k-means, the FCM algorithm is very dependent on the initial mem-  
 100 bership matrix, which is randomly chosen. Our approach manages to handle  
 101 this limitation very efficiently since the initial  $U^0$  matrix will be tightly linked  
 102 to the detected centroids.

## 103 2.2 Validation Metrics

104 When the dataset contains labels specifying the group of each observation, the  
 105 true labels can be used to validate the performance of the clustering method.  
 106 Otherwise, there are several different approaches for the best estimation of the  
 107 number of clusters [25], and we list the most widely used ones in Table 1.

108 In general, the use of these metrics implies the execution of the clustering  
 109 methods for a sequence of values (of the number of clusters), and the selection  
 110 of the most likely value under an optimization criterium.

## 111 3 Proposed Model

112 Object detection is among the classical computer vision problems. It aims to  
 113 identify which objects are in the image and their corresponding locations. The  
 114 object detection issue is more complex than the classification problem, which  
 115 consists of recognizing objects but without indicating their locations in the  
 116 image.

117

Table 1: Several metrics to estimate the number of clusters for the k-means clustering algorithm

Reference	Metric	Description
[17]	Adjusted Rand Index (ARI)	It measures the similarity between the cluster assignments by making pair-wise comparisons. A higher score signifies greater similarity.
[18]	Bayesian Information Criterion (BIC)	BIC is a criterion for measuring and selecting models. It relies on the principles of Bayesian inference and probability. The complexity of the model is penalized by the BIC, so more complex models would have a lower score and therefore be less likely to be chosen.
[19]	Fowlkes-Mallows Index (FMI)	FMI performs the external evaluation using labels that are already known. Scores use pairwise precision and recall to assess how correctly cluster assignments were performed. The score is defined as the geometric mean of precision and recall. Higher similarity is indicated by a higher score.
[20]	Akaike Information Criterion (AIC)	It is suitable for models that fit into the maximum likelihood estimation system, like BIC. The lower are the AIC and BIC, the better is the clustering performance.
[15]	Dunn's index (DN)	DN defines sets of clusters that are compact, with a very small variation between cluster members, and large separation between clusters. The higher is the value of Dunn's index, the better is the clustering performance. The optimal amount of clusters is the number of clusters that maximises the Dunn's index.
[21]	Davies-Bouldin index (DB)	It calculates the average similarity between each cluster and its most similar one. The DB validity index aims to maximize the distances between clusters while minimizing distances between the cluster centroid and its data objects.
[22]	Silhouette Width (SIL)	It is a statistic that measures how similar an object is to its own cluster versus other clusters. The silhouette value ranges from $-1$ to $1$ . A high silhouette value is well suited to its own cluster but poorly related to neighboring clusters. Positive and negative high silhouette widths indicate the objects that are correctly clustered and those that are incorrectly clustered, respectively. It is well known that objects with a SW validity index of zero or less are difficult to be clustered.
[23]	Calinski and Harabasz index (CAL)	This metric is the ratio of the sum of between-cluster dispersion and inter-cluster dispersion for all clusters. It is known also as the Variance Ratio Criterion. The higher is this score, the better is the clustering performance.
[24]	Gap statistic (GAP)	It is a statistical hypothesis test-based cluster validity measure. At each value of the cluster number, the gap statistic compares the variation in within-cluster dispersion to that predicted under an appropriate reference null distribution. The smallest is the number of clusters, the best is the clustering performance.

118 In this paper, we use YOLO-v5 [26], a recent update of the YOLO family,  
119 the first object recognition model to merge bounding box estimation and ob-  
120 ject identification in one end-to-end differentiable network. In comparison to  
121 previous YOLO models, YOLO-v5 is the first one developed with the PyTorch  
122 framework [27,28], and it is more lightweight and simple to use compared to  
123 previous YOLO variants.

124 YOLO-v5 is based on a smart Convolutional Neural Network (CNN) for  
125 real-time object detection. This algorithm divides the image into regions and  
126 calculates the bounding boxes and probabilities for each region. The predicted

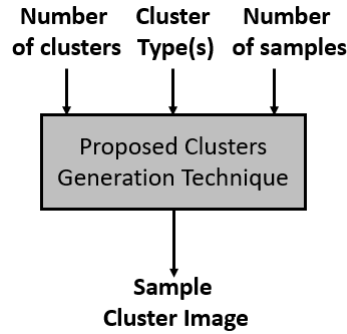


Fig. 1: Proposed Technique to generate label dataset that will be used to training the YOLO-v5 model

127 probabilities are used to weight these bounding boxes. The algorithm needs  
 128 only one forward propagation pass through the neural network to make pre-  
 129 dictions, so it *only looks once* at the image. It then outputs known objects  
 130 together with the bounding boxes after a non-max suppression (which ensures  
 131 that the object detection algorithm only recognizes each object once).

#### 132 4 Experimental Procedure

133 Let us assume that we want to apply a cluster analysis to 2D datasets  $X =$   
 134  $\{x_1, \dots, x_n\}$ , where  $x_i \in \mathbb{R}^2$ . In order to apply our model, we need to transform  
 135 such datasets into images. We have chosen squares of resolution  $640 \times 640$   
 136 pixels.

137 The YOLO-v5 learner has been trained with 1000 of such datasets, each  
 138 one presenting between 2 and 12 clusters, made of between 20000 and 50000  
 139 points, generated from different bivariate Gaussian densities (as sketched in  
 140 Fig. 1). We point out that the flexibility of this model would be enhanced if  
 141 the training phase would cover a larger family of cluster shapes.

142  
 143 Fig. 2 presents the workflow of the proposed solution that consists of:

- 144 1. The labeled dataset will be processed to convert samples to image repre-  
 145 sentations in 2D.
- 146 2. Split the dataset into training and testing sets
- 147 3. Training the YOLO-v5 DL-object detection model by using the training  
 148 set. After fitting this model, the trained model can be used to predict the  
 149 cluster parameters (number of clusters and their corresponding centroid).
- 150 4. The trained model will be evaluated using the test set, and if it reaches  
 151 good performance, the trained model can be used with new datasets to  
 152 quickly detect the cluster initialization parameters. Otherwise, the retrain-  
 153 ing process will be applied after modifying the DL parameters.

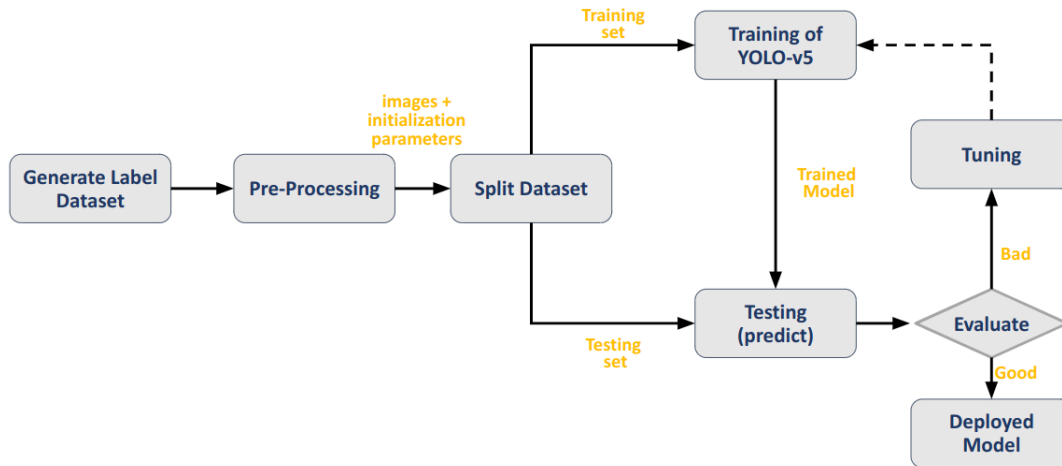


Fig. 2: The workflow process of the proposed solution

154 We show in Fig. 3, a visual show of the proposed scheme with a random  
 155 training image (correspondent to dataset samples in 2D image representation)  
 156 and another one for testing: after the training step, for each new test dataset,  
 157 the trained learner detects its clusters, and it feeds the clustering method  
 158 (k-means, FCM , and x-means) with the initial parameters, improving their  
 159 efficiency.

160 We present a few examples of the training images in Fig. 4.

161 We have used the YOLO-v5 model implementation of [26], trained with  
 162 the SGD optimizer by using an initial value of  $10^{-2}$ , a batch of the size of 16,  
 163 and the rest of the parameters with the default values.

164 In the training phase, the YOLO-v5 model learns to detect all the clusters  
 165 with 20 epochs. At the inference phase, once the model detects the bounding  
 166 boxes, the center for each cluster is computed, and this value will be the  
 167 initial value for the initial cluster. YOLO-v5 is very efficient and lightweight;  
 168 it quickly detects the objects (clusters) and can be implemented on GPU or  
 169 CPU (after being trained).

## 170 5 Results

171 In this section, we compare the performance of k-means [29], x-means [30], and  
 172 FCM [31] under their usual initialization, against our DL-based initialization,  
 173 over a set of 100 test datasets (i.e. images).



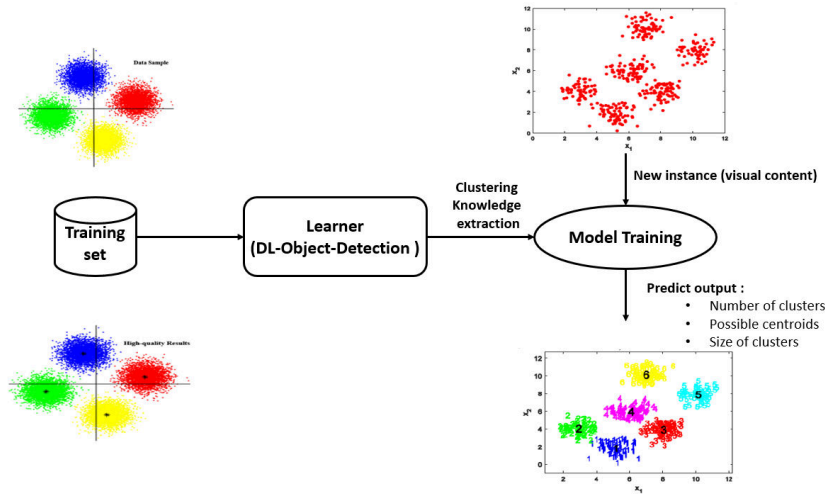


Fig. 3: The proposed solution consists of training the YOLO-v5 model to detect the number of clusters and centroids, and later the trained model will be used to extract initialization parameters (the number of clusters and centroids) for new unknown clustering datasets

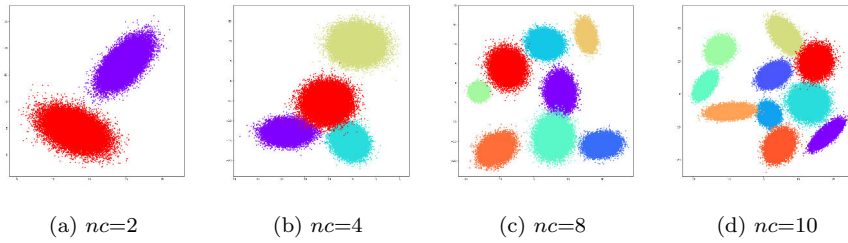


Fig. 4: Examples of simulated datasets used to train the YOLO-v5 object detection model

#### 174 5.1 Detection of the number of clusters

175 Our test set has been generated with varying numbers of clusters, that we have  
 176 labeled in order to test the accuracy of the clustering algorithms. We show in  
 177 Fig. 5 the percentage of correct cluster number identification, in the test set of  
 178 100 images, for: (1) the k-means algorithm with initialization, assisted with the  
 179 metrics of Table 1; (2) the x-means algorithm; and (3) the k-means algorithm  
 180 assisted with our proposal (i.e., with the number of clusters estimated in the  
 181 object detection phase).

182 We can see that the k-means algorithm assisted with our proposal, as well  
 183 as with the random initialization plus the metrics BIC, AIC, FMI, ARI, and  
 184 CAL, have yielded the exact number of clusters in all the instances. The met-  
 185 rics DB, SIL, and GAP have produced a very high rate of correct guesses too,  
 186 and only the x-means has shown a poor rate of correct guesses. Let us men-  
 187 tion that the advantage of our proposed solution, compared to the other ones,  
 188 using the metrics, is that we require less computation and memory overhead  
 189 (see Fig. 12).

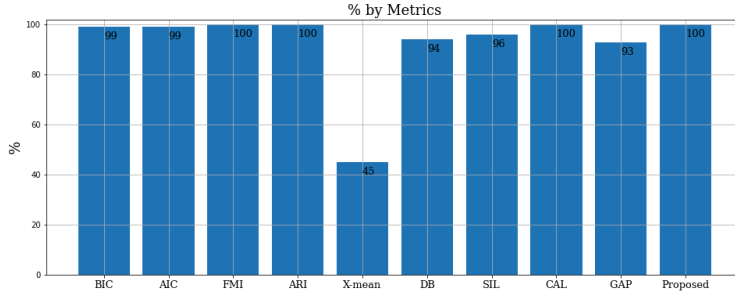


Fig. 5: Percentage of correctly detected number of clusters for 100 tests, where clusters for each test iteration have been generated randomly with different cluster numbers, sizes, and locations.

## 190 5.2 Centroids Detection Correctness

191 In this part, we have analyzed the correctness of the detected centroids. Fig. 6  
 192 presents a few of the images in our test set. We can remark visually that  
 193 the identified centroids are very close to the generated ones for the different  
 194 clusters (see Fig. 9). Moreover, we measure the Euclidean distance between  
 195 the identified and true centroids, and we have obtained the results presented  
 196 in Fig. 7, showing that the generated and detected centroids are very close in  
 197 almost all the instances.

## 198 5.3 Accuracy Rate

The clustering accuracy rate ( $AR$ ) is the proportion of correctly classified observations:

$$AR = \sum_{i=1}^k \frac{n(c_k)}{n} \quad (3)$$

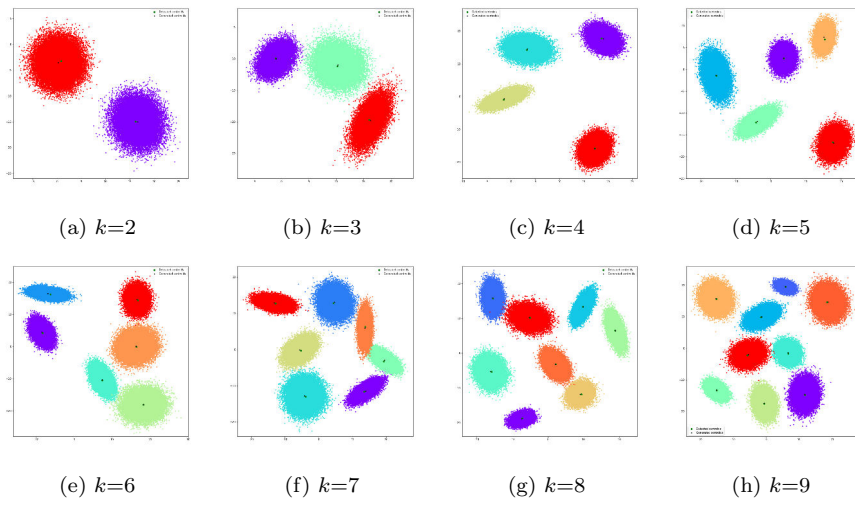


Fig. 6: An example of validation datasets with generated and identified centroids (consequently of each cluster)

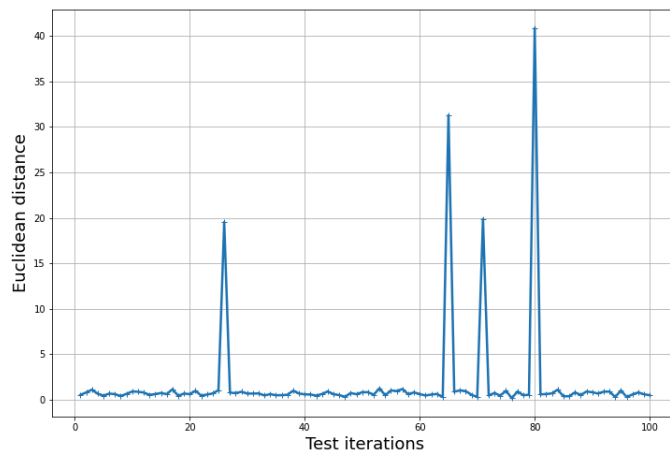


Fig. 7: Average Euclidean distance between generated and detected centroids for the 100 test images with k-means of `scikit-learn` and our initialization

199 where  $n(c_k)$  is the number of data points that were correctly included in cluster  
 200  $k$ , and  $n$  is the total number of data points. The higher is the  $AR$ , the better  
 201 is the clustering detection.

202 Fig. 8 represents the  $AR$  results of k-means with and without the identified  
 203 centroids in the test set. These results validate that the clustering accuracy (at  
 204 the level of the samples) is enhanced (to be close to 1) by using the identified  
 205 centroids for a naive implementation of k-means and also for the optimized  
 206 scikit-learn k-means implementation.

207 Therefore, k-means (or any other clustering algorithm) can provide better  
 208 clustering accuracy and low computational overhead (low number of iterations  
 209 to converge and consequently low delay) by using the proposed initialization  
 210 parameters detection method. For example, as shown in Fig. 9-(e) (detected  
 211  $nc = 6$ ), the accuracy is enhanced from 0.854 to 0.999 by using the identified  
 212 clusters number and centroids.

#### 213 5.4 Iterations for convergence and time consumption

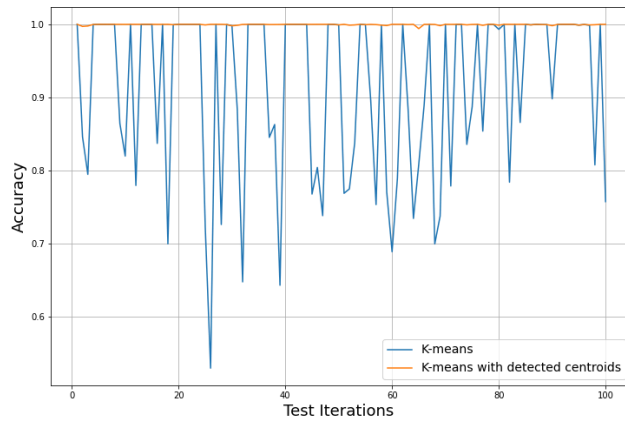
214 Fig. 11 confirms the reduction in the number of iterations until convergence, for  
 215 k-means and FCM, when used with the proposed approach, compared to the  
 216 traditional clustering approaches, for the 100 test set images. As an example,  
 217 in the case of  $nc = 7$  (Fig. 9-f), the required number of iterations to converge  
 218 is decreased from 4 to 2 by using the identified centroids.

219 Regarding the time delay of the clustering algorithms, we remark that it  
 220 depends on three factors:

- 221 1. The number of iterations in the algorithm.
- 222 2. The amount of data points.
- 223 3. The time needed to find out the clustering centers and data points parti-  
 224 tions.

225 We have examined the computer time of the proposed solution to detect  
 226 cluster initialization parameters. Fig. 12 shows the execution times (in seconds)  
 227 for the k-means assisted by two of the well-known metrics (AIC and BIC), as  
 228 well as our proposed method, versus the dataset size. We observe a lower  
 229 execution time for our proposal, compared to the other methods, making it  
 230 the best choice when working on large datasets. The proposed solution has  
 231 the advantage of requiring very low computational demand, and consequently,  
 232 low delay, since it is practically independent of the number of observations in  
 233 the dataset.

234 Moreover, we quantify the effectiveness of using these initialization pa-  
 235 rameters to help the clustering algorithm to converge fast. Fig. 13 shows the  
 236 execution time ratio between using clustering algorithms with the identified  
 237 centroids and without it. These results show clearly that using the identified  
 238 centroids reduces the testing time to half on average. This indicates that the  
 239 proposed solution reduces significantly the clustering testing time and makes  
 240 it suitable for large tabular datasets.



(a) k-means naive

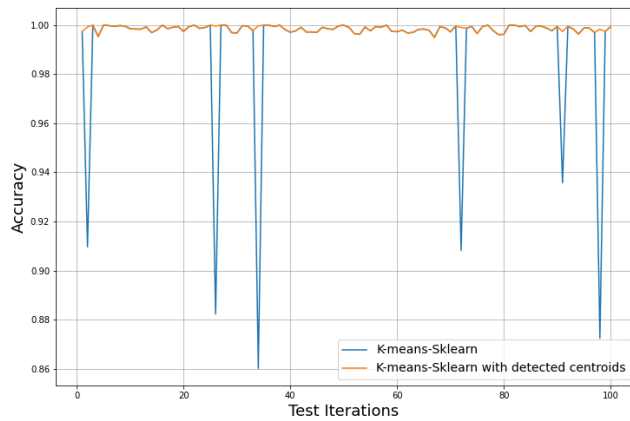
(b) k-means of `scikit-learn`

Fig. 8: Accuracy of k-means clustering algorithms with and without using the proposed approach for two k-means implementations

## 241 6 Discussion

242 The proposed solution involves the implementation of a Deep Learning (DL)  
 243 approach, utilizing Yolo-v5 with its default configuration to identify cluster  
 244 initialization parameters. This method demonstrates superior results in com-  
 245 parison to statistical approaches, showcasing enhanced efficiency. Yolo-v5, em-  
 246 ployed as an object detection model in this context, serves as a proof of con-

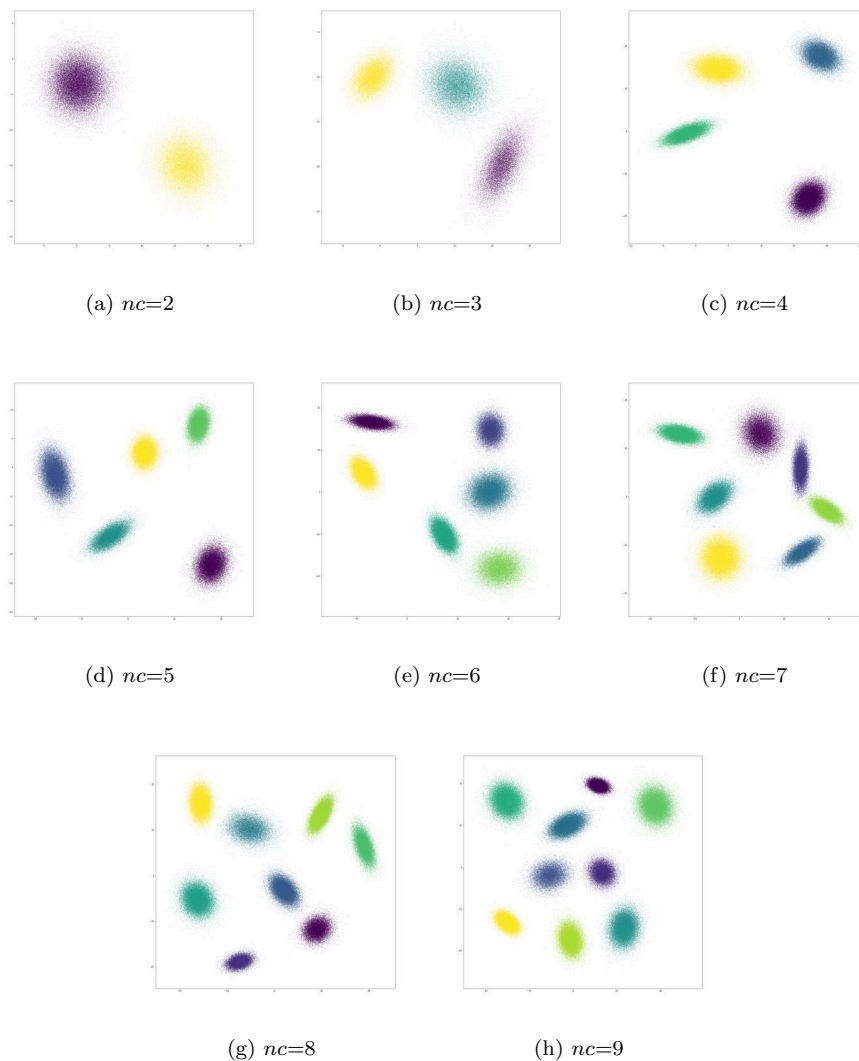


Fig. 9: The k-means clustering results for the test datasets, listed in Fig. 6, by using the detected cluster number and identified centroids (proposed method)

247 cept. Consequently, based on the achieved outcomes, any proficient DL object  
248 detection model could substitute YOLO for detecting initial clustering param-  
249 eters. The solution leverages the capabilities of DL models for image analysis,  
250 emphasizing that the contribution lies in the overall framework rather than  
251 the specific DL model used. Additionally, the work includes a comprehen-  
252 sive comparison between classical initialization methods and the proposed DL  
253 YOLO-v5 initialization, validating its efficiency and robustness.

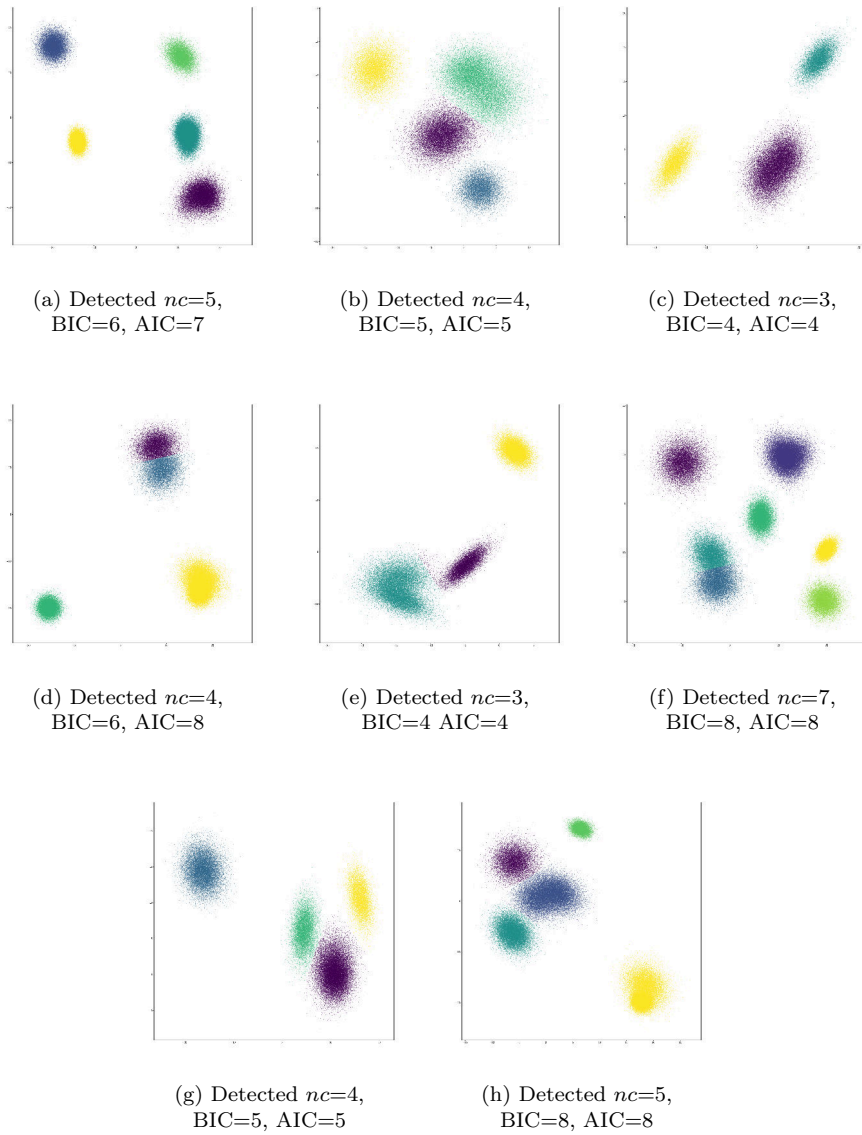


Fig. 10: Results of the overlapping model for which the number of centroids detected by the proposed approach is different compared to other classical approaches (AIC, and BIC)

254 The proposed work is a new clustering initialization method that can de-  
255 termine the number of clusters of a 2D dataset, in addition to their possible  
256 centroids and sizes, improving the speed and results of the most popular clus-  
257 tering algorithms by using the DL object detection model, which is the main  
258 contribution of this work. The advantages of the proposed solution are light-  
259 ness, speed, and robustness with different cluster volumes, shapes, and noise.  
260 The proposed solution has been tested with several configurations, proving its  
261 efficiency compared to other existing approaches, especially in terms of accu-  
262 racy, time consumption, and resource overhead, as can be seen in Figs. 8, 11,  
263 12 and 13.

264 Therefore, this work presents a new simple, and fast way to set up the  
265 initial clustering parameters by using the DL object detection models such as  
266 YOLO-v5.

## 267 **7 Conclusion**

268 The proposed solution has proved to outperform the classical setups of cluster  
269 analysis, in accuracy as well as in time delay. It should also be noted that even  
270 if YOLO-v5 has been used in this paper as proof of work, any future efficient  
271 object detection model can be adapted and used instead of YOLO-v5 in the  
272 proposed solution.

273 As future work, we aim to design a new DL-based data transformation  
274 model for assisting clustering algorithms in higher-dimension data analysis,  
275 as well as envisaging a wider family of cluster shapes (further than Gaussian  
276 blobs).

## 277 **Conflict of Interest**

278 The authors declare that they have no conflict of interest.

## 279 **Data availability**

280 The code to generate data is available here: [https://github.com/rcouturier/  
281 data4clustering](https://github.com/rcouturier/data4clustering)

## 282 **Acknowledgement**

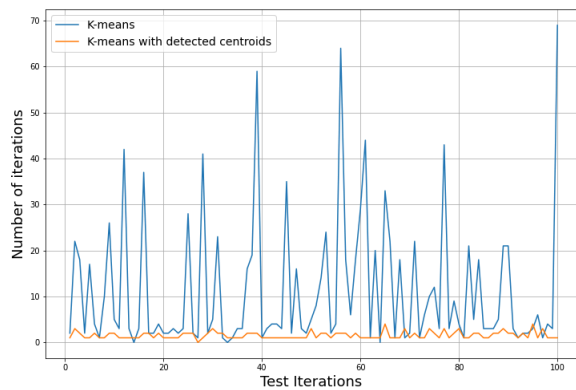
283 This work was partially funded by project ANER 2022 AGRO-IA-LIMENTAIRE  
284 and the EIPHI Graduate School (contract ANR-17-EURE-0002). The Mesocen-  
285 tre of Franche-Comté provided the computing facilities. This work was also  
286 partially sponsored by the General Directorate for Scientific Research and  
287 Technological Development, Ministry of Higher Education and Scientific Re-  
288 search (DGRSDT), Algeria.



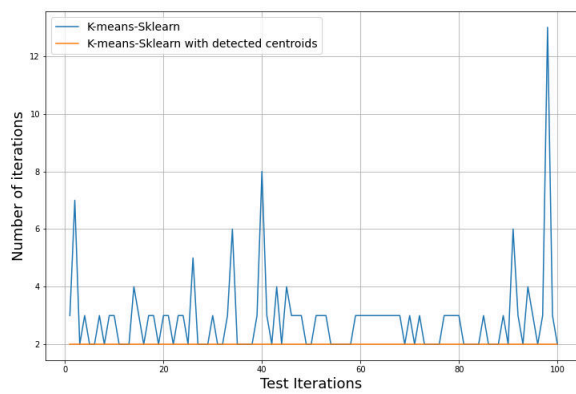
## References

- 290 1. Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms.  
291 *Annals of Data Science*, 2(2):165–193, 2015.
- 292 2. Christophe Guyeux, Stéphane Chrétien, Gaby Bou Tayeh, Jacques Demerjian, and  
293 Jacques Bahi. Introducing and comparing recent clustering methods for massive data  
294 management in the internet of things. *Journal of Sensor and Actuator Networks*, 8(4):56  
295 (25), dec 2019.
- 296 3. Miin-Shen Yang, Chien-Yo Lai, and Chih-Ying Lin. A robust em clustering algorithm  
297 for gaussian mixture models. *Pattern Recognition*, 45(11):3950–3961, 2012.
- 298 4. Kristina P Sinaga and Miin-Shen Yang. Unsupervised k-means clustering algorithm.  
299 *IEEE Access*, 8:80716–80727, 2020.
- 300 5. Myat Thet Nyo, F. Mebarek-Oudina, Su Su Hlaing, and Nadeem A. Khan. Otsu’s  
301 thresholding technique for mri image brain tumor segmentation. *Multimedia Tools and  
302 Applications*, 81(30):43837–43849, 2022.
- 303 6. Adil Fahad, Najlaa Alshatri, Zahir Tari, Abdullah Alamri, Ibrahim Khalil, Albert Y  
304 Zomaya, Sebti Foufou, and Abdelaziz Bouras. A survey of clustering algorithms for  
305 big data: Taxonomy and empirical analysis. *IEEE transactions on emerging topics in  
306 computing*, 2(3):267–279, 2014.
- 307 7. Maria Camila N Barioni, Humberto Razente, Alessandra MR Marcelino, Aagma JM  
308 Traina, and Caetano Traina Jr. Open issues for partitioning clustering methods: an  
309 overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*,  
310 4(3):161–177, 2014.
- 311 8. Mohammad Alhwarat and M Hegazi. Revisiting k-means and topic modeling, a com-  
312 parison study to cluster arabic documents. *IEEE Access*, 6:42740–42749, 2018.
- 313 9. Yinfeng Meng, Jiye Liang, Fuyuan Cao, and Yijun He. A new distance with derivative  
314 information for functional k-means clustering algorithm. *Information Sciences*, 463:166–  
315 185, 2018.
- 316 10. A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data  
317 via the em algorithm. *Journal of the Royal Statistical Society : Series B*, 39(1):1–38,  
318 1977.
- 319 11. J. B. MacQueen. Some methods for classification and analysis of multivariate obser-  
320 vations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and  
321 Probability*, pages 281–297. Berkeley, University of California Press, 1967.
- 322 12. Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information  
323 theory*, 28(2):129–137, 1982.
- 324 13. David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding.  
325 Technical Report 2006-13, Stanford InfoLab, June 2006.
- 326 14. Dan Pelleg and Andrew Moore. X-means: Extending k-means with efficient estimation  
327 of the number of clusters. In *Proceedings of the 17th International Conference on  
328 Machine Learning*, page 727. Citeseer, 2000.
- 329 15. Joseph C Dunn. A fuzzy relative of the isodata process and its use in detecting compact  
330 well-separated clusters. 1973.
- 331 16. James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*.  
332 Springer US, 1981.
- 333 17. Ka Yee Yeung and Walter L Ruzzo. Details of the adjusted rand index and clustering  
334 algorithms, supplement to the paper an empirical study on principal component analysis  
335 for clustering gene expression data. *Bioinformatics*, 17(9):763–774, 2001.
- 336 18. Robert E Kass and Adrian E Raftery. Bayes factors. *Journal of the american statistical  
337 association*, 90(430):773–795, 1995.
- 338 19. Tomasz Dziopa. Clustering validity indices evaluation with regard to semantic homo-  
339 geneity. In *FedCSIS (Position Papers)*, pages 3–9, 2016.
- 340 20. Hamparsum Bozdogan. Model selection and akaike’s information criterion (aic): The  
341 general theory and its analytical extensions. *Psychometrika*, 52(3):345–370, 1987.
- 342 21. David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transac-  
343 tions on pattern analysis and machine intelligence*, (2):224–227, 1979.
- 344 22. Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of  
345 cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

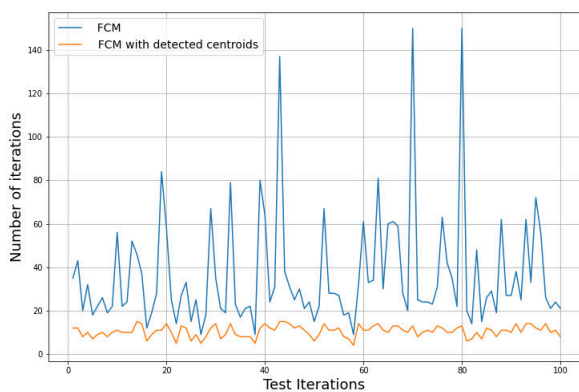
- 346 23. Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Com-*  
347 *munications in Statistics-theory and Methods*, 3(1):1–27, 1974.
- 348 24. Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of  
349 clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society:*  
350 *Series B (Statistical Methodology)*, 63(2):411–423, 2001.
- 351 25. Eréndira Rendón, Itzel Abundez, Alejandra Arizmendi, and Elvia M Quiroz. Internal  
352 versus external cluster validation indexes. *International Journal of computers and*  
353 *communications*, 5(1):27–34, 2011.
- 354 26. Yolov5 in pytorch. <https://github.com/ultralytics/yolov5>, 06 2020.
- 355 27. Sagar Imambi, Kolla Bhanu Prakash, and GR Kanagachidambaresan. Pytorch. In  
356 *Programming with TensorFlow*, pages 87–104. Springer, 2021.
- 357 28. Nikhil Ketkar and Jojo Moolayil. Introduction to pytorch. In *Deep learning with python*,  
358 pages 27–91. Springer, 2021.
- 359 29. Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering  
360 algorithm. *Pattern recognition*, 36(2):451–461, 2003.
- 361 30. Dan Pelleg, Andrew W Moore, et al. X-means: Extending k-means with efficient esti-  
362 mation of the number of clusters. In *Icml*, volume 1, pages 727–734, 2000.
- 363 31. Pradipta Maji and Sankar K Pal. Rfcml: a hybrid clustering algorithm using rough and  
364 fuzzy sets. *Fundamenta Informaticae*, 80(4):475–496, 2007.



(a) Naive K-means implementation



(b) Scikit-learn K-means implementation



(c) FCM

Fig. 11: Variation of the number of iterations required to converge for k-means with random start (a), with `kmeans++` start (b), and for FCM with random start (c), along the images in the test set. It can be seen that our solution to detect centroids is very efficient for all the tested algorithms.

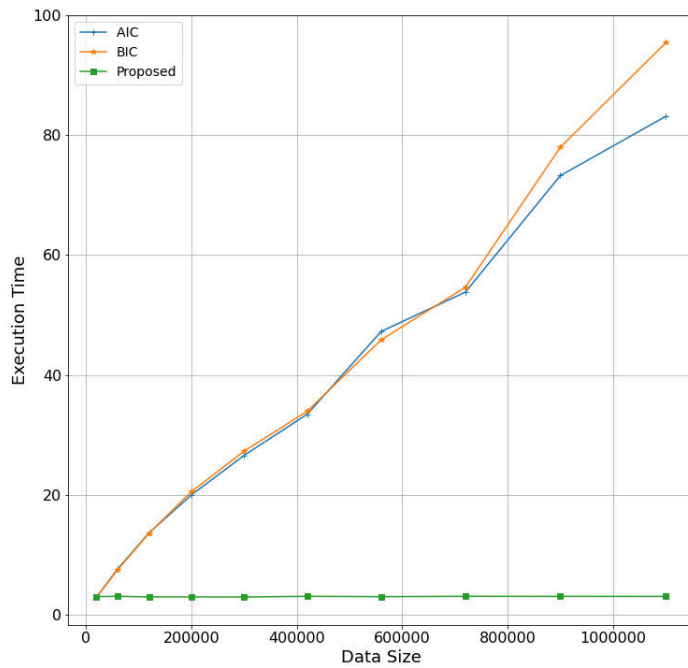


Fig. 12: Comparison of the execution time for the detection of the correct number of clusters, under AIC, BIC, and the proposed initialization method used with the k-means version of `scikit-learn`. It can be observed that the proposed approach is very efficient and scalable compared to other approaches.

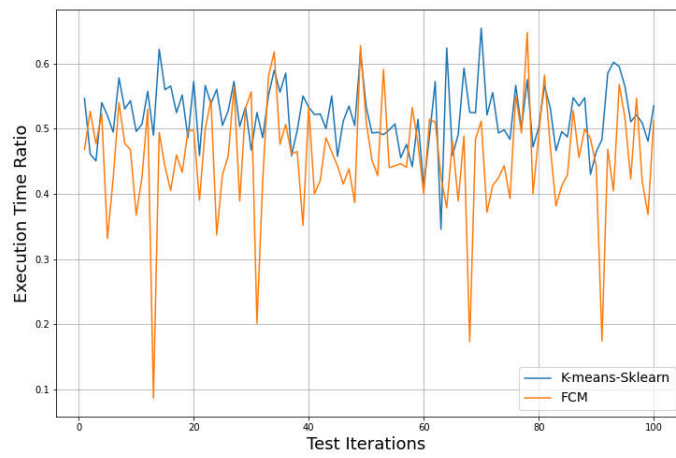


Fig. 13: Ratio of the execution time for k-means and FCM (between “with” over “without” the detected clusters initialization)