Available online at http://scik.org
Eng. Math. Lett. 2024, 2024:4
https://doi.org/10.28919/eml/8402
ISSN: 2049-9337

ANALYSIS OF TIME COMPLEXITY OF K-MEANS AND FUZZY C-MEANS CLUSTERING ALGORITHM

PRATIK SINGH THAKUR[1,*], ROHIT KUMAR VERMA[2], RAKESH TIWARI[1]

[1]Department of Mathematics, Govt. V.Y.T. PG. Autonomous College, Durg (C.G.), 491001, India

[2]Department of Mathematics, Govt. Chandulal Chandrakar Arts and Science College, Patan, Dist.- Durg (C.G.), 491111, India

Abstract. This paper compares the time complexity of the K-means and Fuzzy C-means (FCM) clustering algorithms for different cluster counts. The algorithms' performance is evaluated using several datasets, and the results show that, while the FCM algorithm has a higher time complexity than the K-means algorithm in general, it may be better suited for certain types of data and when a larger number of clusters are used. The paper concludes that both algorithms have advantages and disadvantages, and that the choice should be based on the specific requirements of the problem at hand.

Keywords: clustering; K-means; fuzzy C-means; time complexity.

2020 AMS Subject Classification: 62H30, 90C70, 68W40, 91C20.

1. INTRODUCTION

Clustering is a data mining technique that involves grouping similar data points in a dataset together. It is an unsupervised learning method that is used to discover natural patterns or structures in data without the use of labeled data. Clustering algorithms such as K-means and Fuzzy C-means are used to find clusters in a dataset, with each cluster represented by its centroid

*Corresponding author

E-mail address: spratik343@gmail.com

Received December 16, 2023

and each data point assigned to a cluster based on its similarity to the centroid. Clustering can be used to identify customer segments, detect fraud, and segment images, among other things. It is a powerful tool for uncovering hidden information in large datasets, and it can be used in a variety of applications. However, when selecting a clustering algorithm for a specific task, the algorithm's time complexity should also be considered. Time complexity is a measure of how long it takes an algorithm to run and complete its task, which is usually expressed in terms of the size of the input. Big O notation is used to express the upper bound of the time complexity, which is usually the worst-case scenario. We can make informed decisions about which algorithm is best suited for a specific task and the resources available by understanding the time complexity of clustering algorithms [11][16].

The time complexity of an algorithm refers to how long it takes the algorithm to run and complete its task, which is typically measured in big O notation. A common example is $O(n)$, which means that the time required to run the algorithm grows linearly with the size of the input. Some algorithms, such as $O(logn)$, have a lower time complexity, which means that the time required to run the algorithm increases logarithmically with the size of the input. Some algorithms, on the other hand, have a high time complexity, such as $O(n^2)$, which means that the time it takes to run the algorithm grows exponentially with the size of the input. When selecting an algorithm for a specific task, the time complexity of the algorithm is an important factor to consider [18].

2. BACKGROUND

A well-liked clustering algorithm that combines related data points is K-means. It functions by defining spherical clusters, with each cluster represented by the mean of its individual points. Until convergence, the algorithm repeatedly modifies the cluster means and redistributes data points to the nearest cluster. K, the number of clusters, must be predetermined. Although the approach can be sensitive to initial conditions and may not always find the global optimum, it is computationally efficient. It is frequently employed in a number of industries, including anomaly detection, market segmentation, and picture compression.

K-Means is a popular clustering algorithm that was first proposed by Stuart Lloyd in 1957[13] as a technique for vector quantization. However, the modern version of the algorithm was developed by MacQueen in 1967[14]. In the 1980s, several authors made significant contributions to the understanding and implementation of the K-Means algorithm. For example, Forgy [6] published a version of the algorithm that is more efficient for large data sets, and Hartigan and Wong [9] proposed an algorithm for determining the optimal number of clusters. In the 1990s, several authors extended the K-Means algorithm in different ways. For example, Kaufman and Rousseeuw [15] proposed the "K-Medians" algorithm, which is similar to K-Means but uses the median instead of the mean as the center of a cluster. In the 2000s, authors such as Bradley and Fayyad [4] proposed variations of the K-Means algorithm that are more robust to outliers. Also, authors like Hamerly and Elkan [8] proposed a variant of k-means that is more efficient than the traditional algorithm, the k-means|| algorithm. In recent years, authors like Arthur and Vassilvitskii [1] proposed a variant of k-means called k-means++ which improves the initialization of the centroids to avoid poor local optima.

2.1. K-Means Clustering Algorithm. It divides a set of n items into k clusters using the input parameter k, the number of clusters so that the resulting intra-cluster similarity is high while the inter-cluster similarity is low. To define k centroids, one for each cluster, is the main notion. These centroids should be positioned ingeniously because different locations yield various effects. The preferable option is to situate them as far apart from one another as you can. Next, each point from a particular data collection is taken and connected to the closest centroid. The first step is finished and an early groupage is finished when there are no points still open. At this stage, k new centroids must be recalculated. The identical data set points must now be bound to the closest new centroid once we have these k new centroids. There has been created a loop. This loop causes the k centroids to gradually shift positions until no more changes are made, as we might observe. To put it another way, centroids are no longer in motion. Last but not least, the objective of this procedure is to minimize an objective function, in this case, a squared error function. The objective function:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \|x_i^{(j)} - c_j\|^2 \tag{1}$$

where $\|x_i^{(j)} - c_j\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, is an indicator of the n data points' distance from their respective cluster centres.

(1) As initial centroids, choose k points.
(2) Create k clusters by assigning all points to the centroid that is closest to them.
(3) Calculate the centroid of each cluster again.
(4) Repeat, until the centroids stop changing.

2.2. Fuzzy C-Means Clustering. Fuzzy C-means (FCM) is a data clustering technique in which each data point belongs to a cluster to some extent defined by a membership grade. Bezdek first proposed this technique in 1981 [2, 3] as an improvement on previous clustering methods [5]. It describes how to divide data points that populate a multidimensional space into a specific number of clusters. The main advantage of *fuzzy c - means* clustering is that it allows data points to gradually join clusters as degrees in $[0,1]$. This allows you to express that data points can belong to more than one cluster. The FCM optimises the following objective function:

$$J_{FCM} = \sum_{j=1}^{k} \sum_{i=1}^{p} u_{j,i}^{q} d(x_i, m_j) \tag{2}$$

where q denotes the fuzziness exponent, and $q \geq 1$. The algorithm becomes more fuzzy as the value of q increases; $u_{j,i}$ is the membership value for the i^{th} pattern in the j^{th} cluster satisfying the following constraints:

(1) $u_{j,i} \geq 0,\ i = 1,2,\ldots,p$ and $j = 1,2,\ldots,k$
(2) $\sum_{j=1}^{k} u_{j,i} = 1,\ i = 1,2,\ldots,p$

For FCM the membership function is denifend as

$$u(m_j|x_i) = \frac{\|x_i - m_j\|^{-2/(q-1)}}{\sum_{j=1}^{k} \|x_i - m_j\|^{-2/(q-1)}} \tag{3}$$

and weight function is defined as

$$w(x_i) = 1 \tag{4}$$

As a result, FCM features a constant weight function as well as a soft membership function. FCM generally outperforms K-Means [8] and is less impacted by the existence of data uncertainty [12]. The user must yet define the number of clusters in the data set, just like in K-Means. Additionally, it could reach local optimum [10].

3. TIME COMPLEXITY

When evaluating clustering algorithms for practical applications, understanding their computational complexities is crucial for determining their scalability and efficiency, particularly with respect to handling large datasets and high-dimensional data. In this context, the time complexities of algorithms play a significant role in assessing their computational demands.

3.1. Complexity of K-Means Algorithm. The K-Means algorithm has a time complexity of $O(ncdi)$, where:

- n: Number of data points
- c: Number of clusters
- d: Number of dimensions or features
- i: Number of iterations

In the best-case scenario, where the algorithm converges in a small number of iterations, the time complexity reduces to $O(ncd)$. This occurs because the number of iterations (i) becomes constant. However, in the worst-case scenario, where the algorithm either fails to converge or the data is poorly separated, the time complexity escalates to $O(ncdi)$, potentially leading to significantly higher computational overhead.

3.2. Time Complexity of Fuzzy C-Means Algorithm. The time complexity of the Fuzzy C-Means (FCM) algorithm is generally considered to be $O(nc^2di)$ (see [7]), where:

- n: Number of data points
- c: Number of clusters
- d: Number of dimensions or features
- i: Number of iterations

The algorithm requires multiple iterations to converge, and each iteration involves updating the membership values for each data point and the cluster centroids. The calculation of the membership values and the cluster centroids is the most computationally intensive part of the algorithm, which is why the time complexity is affected by the number of iterations, data points, clusters, and features.

4. COMPARISON

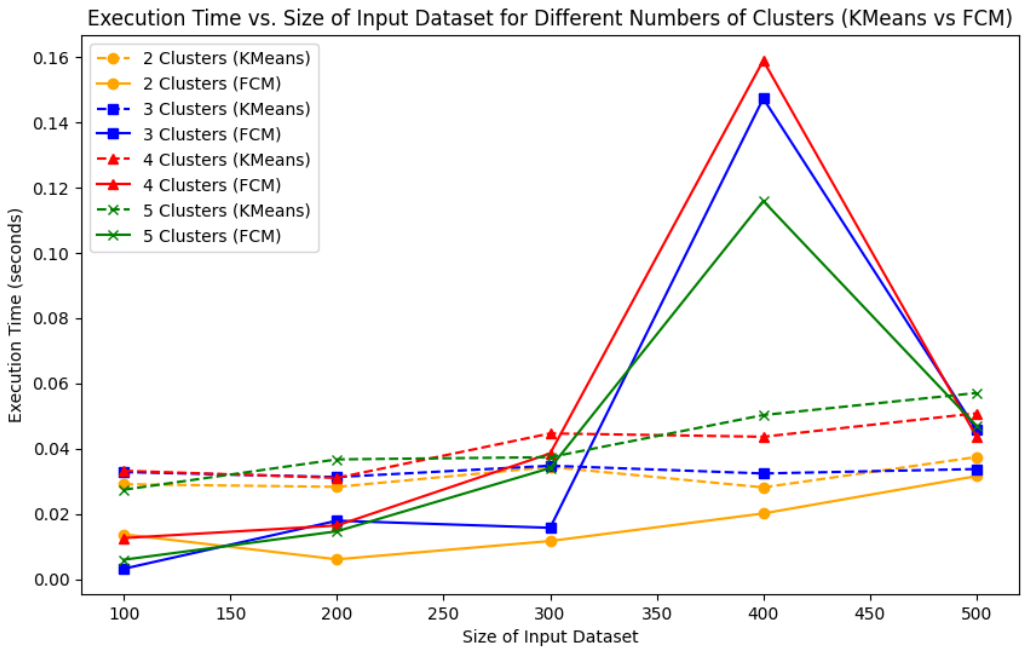

FIGURE 1. Execution Time

Fig. 1 illustrates the relationship between the execution time and the number of data points in the dataset for both the K-Means and Fuzzy C-Means (FCM) clustering algorithms. Each algorithm is evaluated across varying numbers of clusters, ranging from 2 to 5. In the legend, the K-Means results are depicted by dotted lines, while the FCM results are represented by solid lines. Each cluster count is differentiated by a distinct color and line style. As the number of data points increases, the execution time for both algorithms tends to rise due to the computational complexity associated with processing larger datasets. Notably, the plot highlights any performance differences between K-Means and FCM across different cluster counts. The comparison

allows for insights into how each algorithm scales with increasing dataset sizes and varying numbers of clusters. Additionally, the use of different line styles and colors aids in clearly distinguishing between the results for different algorithms and cluster configurations. Overall, the plot provides a comprehensive visualization of the execution time trends for K-Means and FCM under different experimental conditions, offering valuable insights for assessing their practical scalability and performance characteristics.

5. Conclusion

The key difference in time complexity between FCM and K-means lies in the additional factor of c^2, representing the computational overhead introduced by fuzzy membership calculations. This implies that as the number of clusters increases, the computational demand of FCM grows quadratically. The comparison of execution times for both algorithms (Fig. 1) reveals that the performance of K-Means and FCM is comparable. Despite the slight differences in execution time observed in our experiments, the choice between the two algorithms ultimately depends on the specific requirements of the clustering task. For exclusive clustering tasks where data points are expected to belong to distinct clusters, K-Means remains a suitable choice due to its computational efficiency and simplicity (see [17, 19]). On the other hand, for scenarios where data points may exhibit membership to multiple clusters simultaneously, as in overlapping clustering tasks, FCM offers a more flexible approach. By assigning membership values to data points, FCM accommodates the inherent fuzziness in the dataset, allowing for more nuanced clustering results. Therefore, the selection of the most appropriate algorithm hinges on factors such as the nature of the data, the desired level of granularity in cluster assignments, and computational constraints.

Conflict of Interests

The author(s) declare that there is no conflict of interests.

References

[1] D. Arthur, S. Vassilvitskii, k-means++: The advantages of careful seeding, Stanford, (2006).

[2] J.C. Bezdek, A convergence theorem for the fuzzy ISODATA clustering algorithms, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-2 (1980), 1–8. https://doi.org/10.1109/tpami.1980.4766964.

[3] J.C. Bezdek, Objective function clustering, in: Pattern Recognition with Fuzzy Objective Function Algorithms, Springer US, Boston, MA, 1981: pp. 43–93. https://doi.org/10.1007/978-1-4757-0450-1_3.

[4] P.S. Bradley, U.M. Fayyad, Refining initial points for k-means clustering, in: Proceedings of the 15th International Conference on Machine Learning (ICML98), J. Shavlik (ed.), pp. 91-99. Morgan Kaufmann, San Francisco, 1998.

[5] J.C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, J. Cybern. 3 (1973), 32–57. https://doi.org/10.1080/01969727308546046.

[6] C.L. Forgy, (1989). Rete: A fast algorithm for the many pattern/many object pattern match problem, in: Readings in artificial intelligence and databases, pp. 547–559. https://doi.org/10.1016/B978-0-934613-53-8.50041-8.

[7] S. Ghosh, S. Kumar, Comparative analysis of K-means and fuzzy C-means algorithms, Int. J. Adv. Computer Sci. Appl. 4 (2013), 35–39. https://doi.org/10.14569/IJACSA.2013.040406.

[8] G. Hamerly, C. Elkan, Learning the k in k-means, Adv. Neural Inform. Process. Syst. 16 (2003), 1–8.

[9] J.A. Hartigan, M.A. Wong, Algorithm AS 136: A k-means clustering algorithm. J. R. Stat. Soc. Ser. C. 28 (1979), 100–108.

[10] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: A review, ACM Comput. Surv. 31 (1999), 264–323. https://doi.org/10.1145/331499.331504.

[11] A. Levitin, Introduction to the design and analysis of algorithms, Segunda ediçao, Pearson International Edition, (2007).

[12] A. Liew, S. Leung, W. Lau, Fuzzy image clustering incorporating spatial continuity, IEE Proc.-Vis. Image Signal Process. 147 (2000), 185–192.

[13] S. Lloyd, Least squares quantization in PCM, IEEE Trans. Inform. Theory 28 (1982), 129–137. https://doi.org/10.1109/TIT.1982.1056489.

[14] J. MacQueen, Classification and analysis of multivariate observations, in: 5th Berkeley Symp. Math. Stat. Probab. 281–297, (1967).

[15] D.L. Massart, L. Kaufman, P.J. Rousseeuw, A. Leroy, Least median of squares: A robust method for outlier and model error detection in regression and calibration, Anal. Chim. Acta. 187 (1986), 171–179.

[16] P.N. Tan, M. Steinbach, V. Kumar, Data mining introduction, People's Posts and Telecommunications Publishing House, Beijing, (2006).

[17] P.S. Thakur, R.K. Verma, R. Tiwari, A comparison between the fuzzy C-means clustering algorithm and the K-mean clustering algorithm, Eng. Math. Lett., 2024 (2024), 1. https://doi.org/10.28919/eml/8403

[18] T.H. Cormen, C.E. Leiserson, R.L. Rivest, et al. Introduction to algorithms, The MIT Press, 2016.

[19] R.K. Verma, R. Tiwari, P.S. Thakur, Partition coefficient and partition entropy in fuzzy C means clustering, J. Sci. Res. Rep. 29 (2023), 1–6. https://doi.org/10.9734/jsrr/2023/v29i121812.