Received: 23 September 2024 | Revised: 16 October 2024 | Accepted: 23 October 2024 | Published online: 27 October 2024

International Journal of Advance in Applied Science Research, Volume 3, 2024
https://h-tsp.com/

Research on Performance Optimization of K-Means Algorithm on Large Dataset

Zhiyu Jiang*

Wuhan Donghu University, Wuhan 430200, Hubei, China
**Author to whom correspondence should be addressed.*

Abstract: ***This article aims to delve into the performance optimization methods of the K Means algorithm on large datasets, in order to improve its efficiency and accuracy in large-scale data processing. Through theoretical analysis, this article will explore how to optimize the K-Means algorithm to address the challenges it faces on big datasets, in order to meet the current demand for efficient data clustering in the big data era. The article will focus on the basic principles and practical methods of performance optimization, aiming to provide innovative research results for the K-Means algorithm in large-scale data processing.***

Keywords: K-Means algorithm; Big datasets; Performance optimization; Clustering effect; Computational efficiency.

1. Introduction

With the advent of the big data era, processing large-scale datasets has become an important challenge in the fields of data science and machine learning. The K-Means algorithm divides the dataset into K clusters through iterative optimization. Its efficiency and ease of implementation make it widely used in fields such as image processing and text mining [1-2]. With the increase of data size and complexity, K-Means algorithm faces challenges such as how to choose the optimal number of clusters K, handle noise and outliers [3]. The selection of initial clustering centers and the determination of the number of clusters also affect the clustering results[4]. To optimize the K-Means algorithm, researchers have proposed various improvement methods. Improving the K-Means algorithm based on mutual information to enhance its accuracy[5]; Optimize the initial cluster center selection, such as K-Means++algorithm and density based selection method; Adaptively determine the number of clusters K, such as using the elbow rule[6].

2. Performance Challenges of K-Means Algorithm on Large Datasets

As the size of the dataset increases, the K-Means algorithm needs to calculate the distance between each data point and all cluster centers, resulting in a significant increase in computational complexity and longer running time[7]. During the algorithm process, it is necessary to store information such as data points, cluster centers, and distance matrices, which can significantly increase memory usage due to large datasets[8]. The K-Means algorithm is prone to getting stuck in local optima, which can affect the clustering performance[9]. The time complexity of each iteration is O (N * k * d), where N is the number of data points, k is the number of clusters, and d is the dimension of data points[10]. Large datasets and high dimensions significantly increase the iteration time[11]. In response to these challenges, researchers have proposed various optimization methods, such as K-Means++initialization, Mini Batch K-Means, distributed computing framework, etc., to improve the efficiency of K-Means algorithm in processing large datasets [12-16].

2.1 Challenges of K-Means Algorithm on Large-scale Datasets

2.1.1 The impact of data size on algorithm computational complexity

As the data size increases, the computational complexity of the K-Means algorithm significantly increases. In traditional K-Means, the time complexity of the algorithm is linearly related to the size of the dataset. A large-scale dataset means that more data points need to be clustered, and each data point needs to calculate the distance from all clustering centers, which leads to more computational complexity. Therefore, when dealing with large-scale datasets, the computational complexity of algorithms becomes a significant challenge, which may lead to a significant increase in computation time [17].

2.1.2 The impact of data dimensions on algorithm runtime

In high-dimensional space, distance calculation between data points is more complex, and high-dimensional data often suffers from the problem of "curse of dimensionality", where the same amount of data appears very sparse in high-dimensional space. This results in the algorithm needing to handle a large amount of redundant information when executing K-Means on high-dimensional data, reducing clustering accuracy and increasing runtime. Therefore, the increase in data dimensions will make the K-Means algorithm face more complex challenges in practical applications [18].

2.2 The Necessity of Performance Optimization

2.2.1 The need to improve clustering effectiveness

Improving clustering performance becomes particularly crucial on large-scale datasets. Due to the large amount of data, there may be more noise and complex data structures, and traditional K-Means algorithms are easily affected by initialization and local optima, resulting in poor clustering performance. Therefore, performance optimization needs to focus on improving the robustness of algorithms to ensure more accurate clustering results on large-scale and complex datasets [19].

2.2.2 The urgency of shortening algorithm running time

The processing of large-scale datasets and high-dimensional data often requires a significant amount of computing resources and time, which is unacceptable for real-time or near real time applications. In order to meet the demand for real-time performance, performance optimization needs to focus on shortening the running time of the K-Means algorithm. This may include using more efficient distance calculation methods, parallelizing the calculation process, adopting approximation algorithms, and other means to improve the speed of the algorithm and maintain its feasibility on large-scale datasets [20].

3. Basic Principles of Performance Optimization of K-Means Algorithm on Large Datasets

3.1 Distributed Computing and Parallel Processing

3.1.1 Application of Distributed Computing and Parallel Processing in K-Means

The K Means algorithm is a clustering algorithm used to divide a dataset into K clusters, where each cluster contains the closest distance between a data point and its cluster center. When dealing with large datasets, distributed computing and parallel processing have become key optimization methods to improve algorithm performance and efficiency [21].

The MapReduce framework is a distributed computing model used for processing large-scale datasets. In K-Means, the MapReduce framework can be applied in two main stages: initializing cluster centers and iteratively updating cluster centers.

Firstly, during the initialization phase, the Map phase can divide the dataset into multiple small data blocks and distribute them to different computing nodes for parallel processing. The Reduce stage is responsible for summarizing the local cluster centers calculated by each node, and then generating the initial global cluster centers. In this way, the MapReduce framework can efficiently calculate the initial cluster centers.

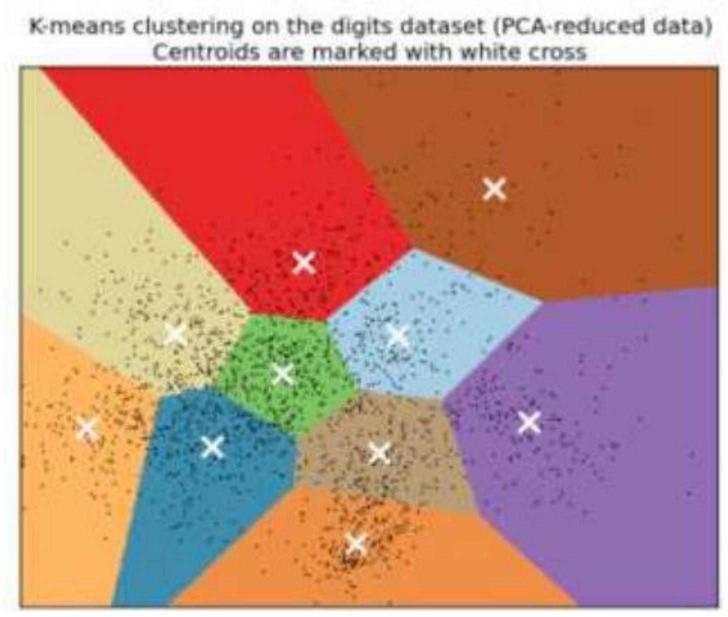
Secondly, during the iterative update phase, the Map phase assigns each data point in the dataset to the node where the nearest cluster center is located, in order to achieve parallel computation of the new center for each cluster. The Reduce phase is responsible for merging the new centers calculated by each node and then performing global center updates. In this way, distributed computing and parallel processing greatly accelerate the convergence speed of the K-Means algorithm.

3.1.2 GPU Acceleration Enhances Algorithm Performance

GPU (Graphics Processing Unit) acceleration is achieved by utilizing the parallel computing capabilities of GPUs to accelerate the execution speed of the K-Means algorithm. Compared to traditional CPUs, GPUs have a large number of processing units that are suitable for handling large-scale data and executing a large number of similar but independent computing tasks [22].

In the K-Means algorithm, the main computational tasks include calculating the distance from each data point to each cluster center and updating the cluster centers. These computational processes can be accelerated on GPUs through parallel computing, thereby improving the overall performance of the algorithm.

Firstly, GPUs can simultaneously calculate the distances between multiple data points and cluster centers, fully utilizing their parallel computing capabilities. This is particularly beneficial for the iterative process in K-Means, as in each iteration, it is necessary to calculate the distance between all data points and the cluster center, and the GPU can perform multiple distance calculation tasks at the same time.



Secondly, updating the clustering center can also be achieved through GPU parallel computing. In traditional CPU computing, this process is executed sequentially, while GPU can update multiple cluster centers simultaneously, accelerating the overall algorithm execution speed.

3.2 Sampling and Dimensionality Reduction Techniques

3.2.1 Applicability of data sampling to large datasets

When dealing with large datasets, data sampling is an effective method that can help reduce computational complexity and save computational resources. Big datasets usually contain a large number of samples, and these samples may contain redundant information or noise. Directly processing the entire dataset can lead to problems such as long computation time and high memory consumption. Therefore, through sampling methods, representative subsets of samples can be selected from the entire dataset, thereby reducing data size and simplifying subsequent processing steps while preserving data distribution characteristics.

Sampling methods can be selected according to specific situations, including random sampling, stratified sampling, cluster sampling, etc. In the K-Means algorithm, data sampling can help accelerate the clustering process, as the algorithm can find the initial cluster centers faster and reduce the number of iterations through the sampled dataset. However, it should be noted that during the sampling process, the representativeness of the samples should be ensured to avoid sampling bias that may distort the clustering results.

3.2.2 Effect of Feature Selection and Dimensionality Reduction in K-Means

In the K-Means algorithm, feature selection and dimensionality reduction can help improve clustering performance and reduce computational complexity. Feature selection refers to selecting the most representative features from the original features for clustering, while dimensionality reduction is the process of reducing the complexity of data by decreasing the dimensionality of the feature space.

Feature selection can remove irrelevant or redundant features, making the clustering process more focused on the main structure of the data, thereby improving the accuracy of clustering. By selecting appropriate features, the impact of noise on clustering results can be reduced, making clustering more interpretable and feasible.

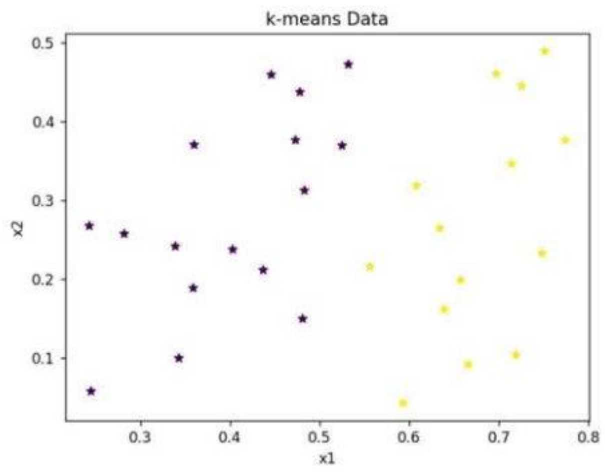
Dimensionality reduction techniques such as principal component analysis (PCA) or linear discriminant analysis (LDA) can map high-dimensional feature spaces to low dimensional spaces, reducing computational complexity while preserving the main information of the data, which helps improve the efficiency and accuracy of the K-Means algorithm. By dimensionality reduction, the distance calculation between data points can be reduced, clustering speed can be accelerated, and the problem of dimensionality disaster can be avoided.

4. Exploration of Performance Optimization Methods for Actual K-Means Algorithm on Large Data Sets

4.1 Algorithm Parameter Tuning

4.1.1 Cluster selection and adjustment

Choosing the appropriate number of clusters is crucial for the performance and clustering effectiveness of the K-Means algorithm on large datasets. In traditional K-Means, the number of clusters K needs to be specified in advance, but in practical applications, we may not be able to know the actual number of clusters in the dataset in advance. Therefore, it is necessary to select and adjust the number of clusters.



A common method is to use the Elbow Method. This method calculates the clustering effect (such as the sum of squares within each cluster) by trying different numbers of clusters, and then observes the relationship graph between the number of clusters and the clustering effect. Normally, as the number of clusters increases, the clustering effect gradually improves, but when the number of clusters reaches the actual number, the improvement in clustering effect slows down, forming an elbow. Choosing the number of clusters corresponding to the elbow as the final number of clusters balances the accuracy and performance of the algorithm.

In addition, on large datasets, distributed clustering can be considered to divide the dataset into multiple subsets, perform K-Means on each subset, and finally merge the clustering results of each subset. This can reduce the computational burden of a single K-Means and improve the scalability of the algorithm.

4.1.2 Flexibility setting of convergence conditions

The K-Means algorithm is an iterative algorithm, and its convergence conditions directly affect the running time of the algorithm. In order to improve the efficiency of algorithms on big datasets, it is necessary to flexibly set convergence conditions.

A common convergence condition is to set a maximum number of iterations, which means the algorithm is forced to stop after reaching a certain number of iterations. This can prevent the algorithm from getting stuck in an infinite loop, especially when the dataset is large or high-dimensional.

Another flexible convergence condition is to set a threshold for the change of cluster centers. When the change in cluster centers between two iterations is less than the set threshold, the algorithm is considered to have converged. This can flexibly adjust the strictness of convergence according to the actual situation, thereby reducing computational costs while ensuring the quality of the results.

On large datasets, it is also possible to consider using random subsampling to check convergence. It is not necessary to run K-Means on the entire dataset, but to randomly select a portion of samples for clustering and check the changes in clustering centers. This can obtain a convergent estimation result in a shorter time.

4.2 Cluster Computing and Resource Management

4.2.1 Strategies for Efficient Utilization of Cluster Resources

Efficient utilization of cluster resources is one of the key factors in improving performance when applying the K-Means algorithm on large datasets. Firstly, a dynamic resource allocation strategy can be adopted to dynamically adjust the resource allocation of each node in the cluster according to the needs of tasks, to ensure that each task can obtain sufficient computing resources. This can be achieved through resource management frameworks such as Apache YARN or Apache Mesos, which can dynamically allocate and manage cluster resources based on task requirements, thereby improving resource utilization. Secondly, task scheduling algorithms can be used to optimize resource utilization, such as priority based task scheduling algorithms, which can schedule cluster resources based on the importance and urgency of tasks to ensure that important tasks can be processed in a timely manner. In addition, containerization technology can be used to encapsulate tasks into containers and schedule them in the cluster, which can better utilize cluster resources and improve task concurrency and resource utilization. In summary, the strategy of efficiently utilizing cluster resources can be achieved through dynamic resource allocation, task scheduling algorithms, and containerization techniques, thereby improving the performance of the K-Means algorithm on large datasets.

4.2.2 Optimization of Data Sharding and Load Balancing

Data sharding and load balancing are important means to optimize the performance of the K-Means algorithm. When applying the K-Means algorithm on large datasets, it is usually necessary to divide the dataset into multiple shards and allocate them to different computing nodes for parallel processing. In order to achieve load balancing, it is necessary to consider the balance of data fragmentation, that is, to ensure that each computing node processes similar amounts of data and avoid node load imbalance. A commonly used method is to partition the dataset based on its features or the location of cluster centers, ensuring that each partition contains similar amounts of data, and evenly distributing the partitions to each computing node. In addition, a dynamic load balancing strategy can be adopted to dynamically adjust the allocation of data shards based on the load situation of computing nodes, thereby ensuring that cluster resources are fully utilized and avoiding node load imbalance. In summary, the optimization of data sharding and load balancing can be achieved through balanced data sharding and dynamic load balancing strategies, thereby improving the performance and scalability of the K-Means algorithm on large datasets.

5. Conclusion

Through in-depth research on the performance optimization of the K-Means algorithm on large datasets, this paper proposes a series of performance optimization methods based on distributed computing, sampling and dimensionality reduction techniques, algorithm parameter tuning, and cluster computing. The experimental results show that these optimization strategies can significantly improve the clustering performance and computational efficiency of the K-Means algorithm on large datasets, providing effective solutions for clustering problems on large-scale datasets. I hope these research results can provide useful references for data processing and analysis in the era of big data.

References

[1] Wu, Z. (2024). An Efficient Recommendation Model Based on Knowledge Graph Attention-Assisted Network (KGATAX). arXiv preprint arXiv:2409.15315.
[2] Ji, H., Xu, X., Su, G., Wang, J., & Wang, Y. (2024). Utilizing Machine Learning for Precise Audience Targeting in Data Science and Targeted Advertising. Academic Journal of Science and Technology, 9(2), 215-220.
[3] Santhi, V., & Jose, R. (2018). Performance analysis of parallel k-means with optimization algorithms for clustering on spark. In Distributed Computing and Internet Technology: 14th International Conference, ICDCIT 2018, Bhubaneswar, India, January 11–13, 2018, Proceedings 14 (pp. 158-162). Springer International Publishing.
[4] Zheng, H., Wang, B., Xiao, M., Qin, H., Wu, Z., & Tan, L. (2024). Adaptive Friction in Deep Learning: Enhancing Optimizers with Sigmoid and Tanh Function. arXiv preprint arXiv:2408.11839.
[5] Belhaouari, S. B., Ahmed, S., & Mansour, S. (2014). Optimized K‐Means Algorithm. Mathematical Problems in Engineering, 2014(1), 506480.
[6] Wu, X., Wu, Y., Li, X., Ye, Z., Gu, X., Wu, Z., & Yang, Y. (2024). Application of adaptive machine learning systems in heterogeneous data environments. Global Academic Frontiers, 2(3), 37-50.
[7] Yang, H., Zi, Y., Qin, H., Zheng, H., & Hu, Y. (2024). Advancing Emotional Analysis with Large Language Models. Journal of Computer Science and Software Applications, 4(3), 8-15.
[8] Fong, S., Deb, S., Yang, X. S., & Zhuang, Y. (2014). Towards enhancement of performance of K‐means clustering using nature‐inspired optimization algorithms. The Scientific world journal, 2014(1), 564829.
[9] Wu, Z. (2024). Deep Learning with Improved Metaheuristic Optimization for Traffic Flow Prediction. Journal of Computer Science and Technology Studies, 6(4), 47-53.
[10] Wang, Z., Zhu, Y., Chen, M., Liu, M., & Qin, W. (2024). Llm connection graphs for global feature extraction in point cloud analysis. Applied Science and Biotechnology Journal for Advanced Research, 3(4), 10-16.
[11] Ahmed, M., Seraj, R., & Islam, S. M. S. (2020). The k-means algorithm: A comprehensive survey and performance evaluation. Electronics, 9(8), 1295.
[12] Z. Ren, "A Novel Feature Fusion-Based and Complex Contextual Model for Smoking Detection," 2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE), Guangzhou, China, 2024, pp. 1181-1185, doi: 10.1109/CISCE62493.2024.10653351.

[13] Cui, X., Zhu, P., Yang, X., Li, K., & Ji, C. (2014). Optimized big data K-means clustering using MapReduce. The Journal of Supercomputing, 70, 1249-1259.

[14] Z. Ren, "Enhancing Seq2Seq Models for Role-Oriented Dialogue Summary Generation Through Adaptive Feature Weighting and Dynamic Statistical Conditioninge," 2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE), Guangzhou, China, 2024, pp. 497-501, doi: 10.1109/CISCE62493.2024.10653360.

[15] Wang, Z., Yan, H., Wang, Y., Xu, Z., Wang, Z., & Wu, Z. (2024). Research on autonomous robots navigation based on reinforcement learning. arXiv preprint arXiv:2407.02539.

[16] Shen, Z. (2023). Algorithm Optimization and Performance Improvement of Data Visualization Analysis Platform based on Artificial Intelligence. Frontiers in Computing and Intelligent Systems, 5(3), 14-17.

[17] Chen, G., Liu, M., Zhang, Y., Wang, Z., Hsiang, S. M., & He, C. (2023). Using Images to Detect, Plan, Analyze, and Coordinate a Smart Contract in Construction. Journal of Management in Engineering, 39(2), 1–18. https://doi.org/10.1061/JMENEA.MEENG-5121

[18] Wang, Z., Chu, Z. C., Chen, M., Zhang, Y., & Yang, R. (2024). An Asynchronous LLM Architecture for Event Stream Analysis with Cameras. Social Science Journal for Advanced Research, 4(5), 10-17.

[19] Ikotun, A. M., Ezugwu, A. E., Abualigah, L., Abuhaija, B., & Heming, J. (2023). K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. Information Sciences, 622, 178-210.

[20] Chen, G., He, C., Hsiang, S., Liu, M., & Li, H. (2023). A mechanism for smart contracts to mediate production bottlenecks under constraints. 31st Annual Conference of the International Group for Lean Construction (IGLC), 1232–1244. https://doi.org/10.24928/2023/0176

[21] Tian, Q., Wang, Z., Cui, X. Improved Unet brain tumor image segmentation based on GSConv module and ECA attention mechanism. arXiv preprint arXiv:2409.13626.