Comparative Study of Clustering Methods: K-means, DBSCAN, and Their Optimizations

1024041124 Qian Xukun
Nanjing University of Posts and Telecommunications
1025345743@qq.com

*Ji Yimu**
Nanjing University of Posts and Telecommunications
jiym@njupt.edu.cn

***Abstract*—Clustering is a fundamental task in unsupervised learning with applications spanning various fields such as data mining, pattern recognition, and machine learning. This paper provides an in-depth study of clustering algorithms, focusing on K-means, DBSCAN, and their optimizations: K-means++, DBSCAN, and a hybrid approach combining DBSCAN and K-means. By evaluating these methods on synthetic and real-world datasets, we highlight their strengths, limitations, and suitability for different clustering scenarios. Performance metrics, including Silhouette Coefficient and Calinski-Harabasz Index, are used for quantitative analysis. The study reveals that hybrid approaches can leverage the advantages of both methods, improving clustering quality and robustness. Furthermore, practical implications and applications in real-world scenarios, such as customer segmentation and anomaly detection, are discussed, emphasizing the versatility and adaptability of these clustering algorithms in addressing diverse challenges.**

***Keywords*—Clustering, K-means, DBSCAN**

I. Introduction

Clustering is a widely studied problem in machine learning and serves as the foundation for understanding the inherent structure of data. It involves partitioning data into groups or clusters such that objects within the same cluster are more similar to each other than to objects in other clusters. Popular clustering methods include partition-based algorithms like K-means and density-based methods like DBSCAN. However, these algorithms have inherent limitations that have motivated the development of optimized and hybrid methods.

The significance of clustering extends beyond academic research, finding applications in domains such as healthcare, where clustering aids in patient segmentation for personalized treatments, and marketing, where customer segmentation enables targeted advertising. For instance, in healthcare, clustering patient data based on symptoms, genetic profiles, and treatment responses can help doctors identify subgroups of patients who may benefit from specific therapies. In marketing, businesses can analyze customer purchase histories, demographics, and online behaviors to group customers with similar preferences, allowing for more effective product recommendations and advertising campaigns.

Anomaly detection in cybersecurity is another crucial application area. By clustering network traffic data, unusual patterns that may indicate cyberattacks can be detected. For example, if a group of network connections exhibits significantly different behavior from the majority of the traffic, it could potentially be a sign of a malicious intrusion. This helps security teams take proactive measures to protect the network.

Clustering is essential in applications such as customer segmentation, anomaly detection, and image processing. Choosing the right clustering algorithm is critical to achieving meaningful results. This study systematically compares these clustering techniques, focusing on their theoretical foundations, practical implementations, and experimental performance. The aim is to provide a comprehensive guide for researchers and practitioners to select the most suitable method for their specific needs.

II. Related Work

Clustering algorithms have been a central focus of research in data mining and machine learning for decades. Classical methods, such as K-means, were introduced by MacQueen in 1967, providing a simple and efficient way to partition data into clusters by minimizing intra-cluster variance[1]. However, the sensitivity of K-means to initial centroid placement prompted the development of K-means++, proposed by Arthur and Vassilvitskii in 2007[2]. This enhancement improves initialization, reduces convergence time, and avoids suboptimal clustering results.

DBSCAN, introduced by Ester et al. in 1996[3], addresses the limitations of K-means by grouping data points based on density. It can identify clusters of arbitrary shapes and handle noise effectively. Recent studies, such as those by Campello et al. (2015)[4], propose hierarchical extensions to DBSCAN to improve its adaptability to varying densities. These hierarchical extensions allow DBSCAN to better handle datasets with complex density distributions, where different regions may have different levels of clustering granularity.

Hybrid approaches combining K-means and DBSCAN have gained traction in recent years. For example, Hahsler and Piekenbrock (2018)[5] demonstrated how DBSCAN can preprocess noisy data, which is then refined by K-means for improved clustering results. This method is particularly effective for datasets with high noise or irregular cluster shapes. In such datasets, DBSCAN can first remove the noise points and identify the rough structure of the clusters, and then K-means can further optimize the cluster centers and boundaries.

Recent advancements also explore the integration of deep learning with clustering algorithms. Xie et al. (2016)[6] proposed deep embedded clustering (DEC), which combines

autoencoders with K-means to learn feature representations while clustering data. This approach leverages the powerful feature extraction capabilities of autoencoders to transform the data into a more suitable space for clustering, improving the clustering performance. Similarly, Aljalbout et al. (2018)[7] reviewed hybrid methods leveraging neural networks to improve traditional clustering algorithms' performance.

For density-based clustering, adaptive DBSCAN variants have emerged to handle parameter sensitivity. For instance, Sander et al. (2020)[8] proposed a method that dynamically adjusts eps based on local density estimations, enhancing DBSCAN's robustness across datasets with varying densities. By adaptively tuning the eps parameter, this method can better capture the density variations in different parts of the dataset, resulting in more accurate clustering. Another promising direction involves combining DBSCAN with hierarchical clustering methods to balance scalability and accuracy, as explored by Xu et al. (2021). Hierarchical clustering can provide a multi-level view of the data, while DBSCAN can handle the density-based clustering at each level, combining their advantages to handle large and complex datasets. Comprehensive reviews, such as those by Aggarwal and Reddy (2014)[9] and Xu and Tian (2015)[10], provide extensive surveys of clustering methods, highlighting emerging trends and challenges, including scalability, parameter sensitivity, and the integration of clustering with other machine learning paradigms. These reviews underscore the continued relevance and innovation within the field, offering a roadmap for future research.

III. Problem Statement

This research formalizes clustering as a data mining task aimed at grouping data points into clusters while maximizing intra-cluster similarity and minimizing inter-cluster similarity. However, several challenges arise in practical applications:

- Handling Noise and Outliers: Real-world datasets often contain noisy data points that can significantly impact clustering performance. For example, in sensor data collected from environmental monitoring, measurement errors or faulty sensors can introduce noise. In customer data, there may be outliers such as extremely high or low purchase amounts that can distort the clustering results.
- Scalability: Many clustering algorithms struggle to scale efficiently with large datasets. For example, the computational complexity of K-means can become a bottleneck for datasets with millions of data points. As the dataset size increases, the time and memory requirements of K-means can grow rapidly, making it impractical for very large datasets.
- Parameter Sensitivity: Algorithms like DBSCAN require careful tuning of parameters (eps, min_samples) to achieve optimal performance. Incorrect parameter choices can lead to over-clustering or under-clustering, particularly in heterogeneous datasets. In a dataset with varying densities, choosing the wrong eps value can result in either splitting a single cluster into multiple small ones or merging multiple distinct clusters into one.
- Initialization Issues: K-means is highly sensitive to initial centroid placement, which can lead to suboptimal solutions. Poor initialization often results in convergence to local minima, especially for datasets with overlapping clusters. When the initial centroids are poorly chosen, K-means may converge to a solution that is not the global optimum, resulting in inaccurate clustering.

The study evaluates K-means, DBSCAN, K-means++, and hybrid DBSCAN + K-means to address these challenges systematically. By using synthetic and real-world datasets, this research provides insights into the strengths and limitations of these methods.

IV. Algorithms

A. K-means

K-means partitions data points into clusters by minimizing the within-cluster sum of squares (WCSS). Despite its efficiency, the algorithm struggles with sensitivity to initial centroid placement and spherical cluster assumptions. The algorithm proceeds in the following steps:

- Randomly initialize centroids. This random initialization can lead to different clustering results in different runs, especially when the dataset has complex structures.
- Assign each data point to the nearest centroid. This step calculates the Euclidean distance between each data point and the centroids and assigns the point to the cluster with the closest centroid.
- Update centroids as the mean of assigned points. After all points are assigned, the centroids are recalculated as the mean of the points in each cluster.
- Repeat steps 2-3 until convergence (no significant changes in centroid positions). The convergence criterion is usually based on a small threshold for the change in centroid positions.

B. K-means++

K-means++ enhances K-means by ensuring a more strategic selection of initial centroids. This modification reduces convergence time and improves clustering quality. The initialization process ensures that centroids are well-spread, minimizing the likelihood of poor clustering results. It selects the initial centroids in such a way that they are far from each other, increasing the chance of finding a good clustering solution.

C. K-prototypes

K-prototypes extends K-means to handle mixed data types (numerical and categorical). Instead of solely relying on Euclidean distance, K-prototypes incorporates dissimilarity measures for categorical attributes and optimizes the following steps:

- Initialization: Randomly select initial cluster centroids for numerical and categorical features. This requires considering both the numerical and categorical aspects of the data when choosing the initial centroids.

- Assignment: Assign each data point to the nearest centroid based on combined numerical and categorical distances. The distance calculation takes into account the differences in both types of data.
- Update: Update centroids by calculating the mean for numerical attributes and the mode for categorical attributes. For numerical data, the mean is used to update the centroid, while for categorical data, the mode is used.

This algorithm is particularly effective for applications like customer segmentation, where datasets often include demographic and behavioral data. For example, in a customer dataset with attributes such as age (numerical) and gender (categorical), K-prototypes can handle both types of data to group customers with similar characteristics.

D. K-medoids

K-medoids, also known as Partitioning Around Medoids (PAM), is a robust alternative to K-means. Instead of centroids, K-medoids selects actual data points (medoids) as cluster centers, making it less sensitive to outliers. The algorithm operates as follows:

- Initialization: Select medoids randomly from the dataset. These initial medoids are used as the starting points for the clustering process.
- Assignment: Assign each data point to the nearest medoid. Similar to K-means, the distance between each point and the medoids is calculated to determine the assignment.
- Optimization: Iteratively swap medoids with non-medoids and reassign clusters if this reduces the total cost (sum of distances). This step tries to find the optimal set of medoids by exploring different combinations.

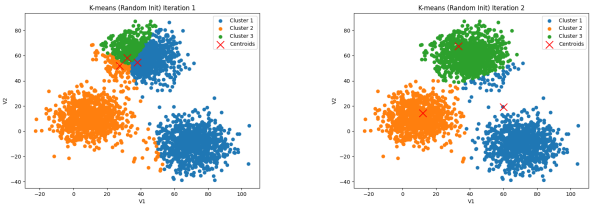
K-medoids is particularly useful for datasets with irregular cluster shapes and datasets containing significant noise or outliers. In a dataset with outliers, since the medoids are actual data points, they are less affected by the outliers compared to the centroids in K-means. surrounding equation.

E. DBSCAN

DBSCAN relies on density connectivity to form clusters. It classifies points as core, border, or noise based on: eps, Maximum distance between points to be considered neighbors. min_samples, Minimum number of points required to form a dense region.

The algorithm proceeds as follows:

- Identify all core points with at least min_samples neighbors within distance eps. These core points form the basis of the clusters.
- Connect core points to form clusters. If two core points are within the eps distance of each other, they are part of the same cluster.
- Classify points that are not part of any cluster as noise. Points that do not belong to any dense region are considered noise.



(a) (b)

Fig. 1. Initial cluster using K-means

F. DBSCAN + K-means Hybrid

The hybrid approach applies DBSCAN to identify dense regions and remove noise. K-means is then used to refine the clusters, improving performance in terms of accuracy and noise handling. This two-step process is particularly useful for datasets with irregularly shaped clusters and outliers. First, DBSCAN can handle the noise and identify the rough structure of the clusters, and then K-means can fine-tune the cluster centers and boundaries to improve the clustering quality.

V. Evaluation

A. Data Characteristics

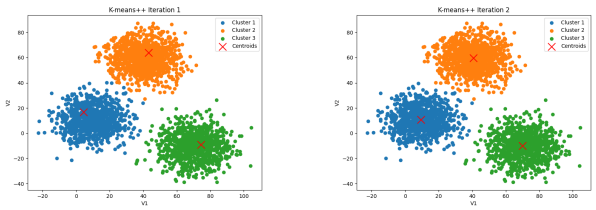
Data sets for clustering can be downloaded from GitHub, i.e., address to https://github.com/mubaris/friendly-fortnight/blob/master/xclara.csv. I have downloaded it and you can use it directly. The data scale is 3000*2. The dataset contains two-dimensional data points, which can be used to evaluate the performance of different clustering algorithms in a relatively simple setting. However, in real-world applications, datasets can have much higher dimensions and more complex structures.

B. Experimental Results

a) K-means Iterative Process: The K-means algorithm was applied to the synthetic dataset. Random initialization of centroids led to suboptimal clustering results in the first two iterations. Figure 1(a) and 1(b) illustrate how poor initialization can result in overlapping or poorly separated clusters. In the initial iterations, the centroids may be placed in such a way that some clusters are not clearly distinguishable, leading to incorrect assignments of data points.

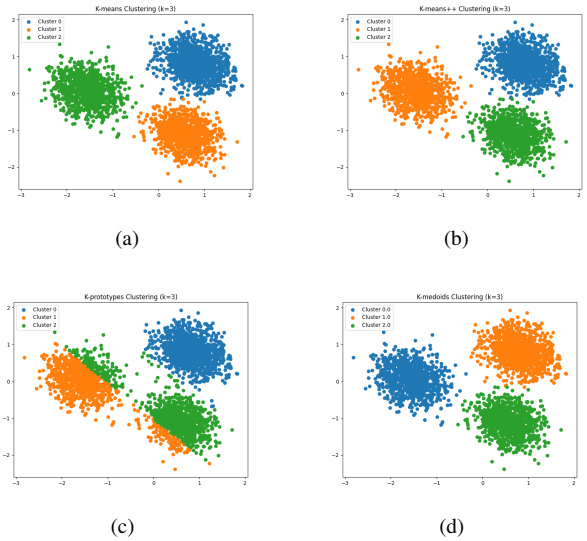
To address the issue, K-means++ was employed to enhance centroid initialization. As shown in Figure 2, the clusters were well-separated after the first iteration, demonstrating the advantage of selecting centroids with a probabilistic spread. K-means++ spreads the initial centroids more evenly across the dataset, reducing the chance of getting stuck in a local minimum.

b) Comparative Analysis of Optimized K-means Variants: The performance of K-means was compared with its optimized versions: K-means++, K-prototypes, and K-medoids, using two evaluation metrics: Silhouette Coefficient and Calinski-Harabasz Index. As depicted in Figures 3–5, K-means++ consistently performed on par with traditional K-means, while



(a) (b)

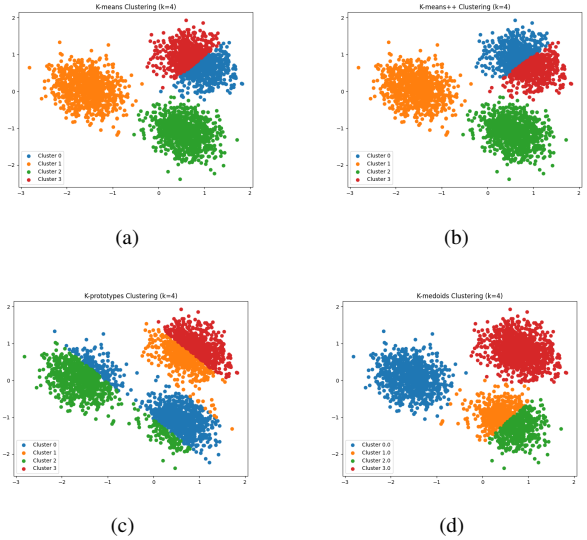
Fig. 2. Improved centroid initialization using K-means++



(a) (b)

(c) (d)

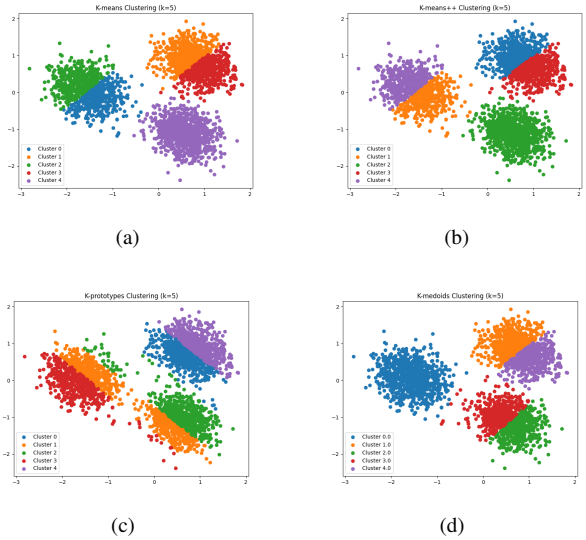
Fig. 3. Visual comparison of clustering results for different K-means variants at (k = 3)



(a) (b)

(c) (d)

Fig. 4. Visual comparison of clustering results for different K-means variants at (k = 4)



(a) (b)

(c) (d)

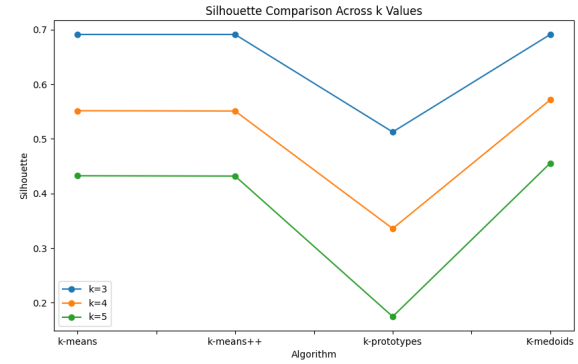
Fig. 5. Visual comparison of clustering results for different K-means variants at (k = 5)

demonstrating improved initialization stability. K-medoids exhibited robustness, particularly at, achieving higher Silhouette Coefficient values. Conversely, K-prototypes struggled due to its sensitivity to mixed-type data handling, resulting in lower clustering quality. K-prototypes' difficulty in handling mixed data types is mainly because combining the distance measures for numerical and categorical data is a complex task, and small errors in the distance calculation can lead to incorrect clustering assignments.

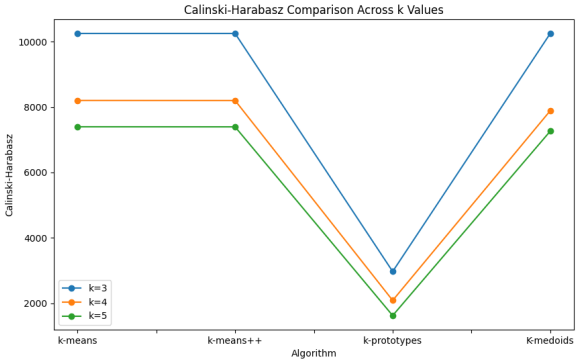
c) DBSCAN Performance and Parameter Tuning: The DBSCAN algorithm was applied to the synthetic dataset, focusing on parameter optimization. A k-distance graph was utilized to determine the optimal eps value, while min_samples was adjusted to balance cluster density and noise filtering. After tuning, DBSCAN effectively identified clusters of arbitrary shapes and outliers, as shown in Figure 7. By analyzing the k-distance graph, we can get an estimate of the appropriate eps value based on the density distribution of the data. Adjusting min_samples helps to control the minimum size of the clusters and filter out noise points.

DBSCAN achieved a Silhouette Coefficient of 0.7237 and a Calinski-Harabasz Index of 12456.24, outperforming K-means for non-spherical clusters and noisy datasets. These results underscore the algorithm's strength in handling data heterogeneity, making it a preferred choice for applications such as anomaly detection and spatial data analysis. In anomaly detection, DBSCAN can identify the abnormal patterns that do not fit into the normal density regions, while in spatial data analysis, it can handle the complex shapes of geographical clusters.

d) Combining DBSCAN and K-means: A hybrid approach was implemented to leverage the strengths of DBSCAN and K-means. DBSCAN was first applied to filter noise and initialize clusters, followed by K-means for refinement.

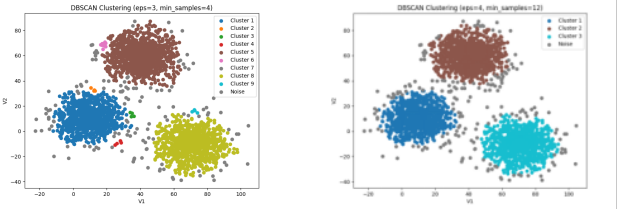


(a)



(b)

Fig. 6. Visual comparison of clustering results for different K-means variants



(a) (b)

Fig. 7. Clusters formed by DBSCAN after parameter tuning

This two-step process yielded superior clustering results, as illustrated in Figure 8. The hybrid method effectively balanced noise handling with cluster compactness, achieving the highest Silhouette Coefficient (0.7145) and Calinski-Harabasz Index (11906.29) among all methods tested. The combination of DBSCAN's noise removal ability and K-means' refinement of cluster centers makes this hybrid approach very effective in handling datasets with noise and complex cluster shapes.

The hybrid approach demonstrated its robustness in noisy and high-dimensional datasets, as shown in Figure 9. By combining DBSCAN's noise-filtering capabilities with K-means' ability to optimize cluster centers, the hybrid approach can

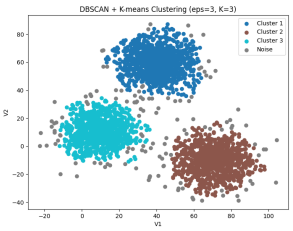


Fig. 8. Final refined clusters using K-means after DBSCAN preprocessing.

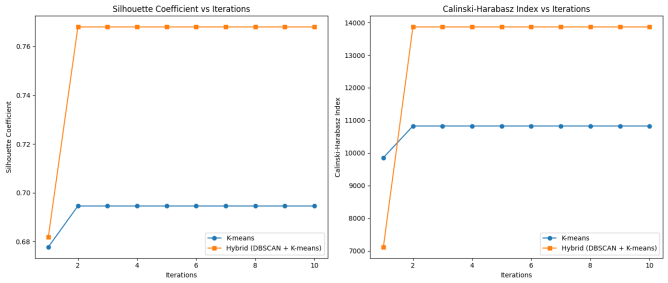


Fig. 9. Visual comparison of clustering results

handle datasets with a large amount of noise and complex structures. In high-dimensional datasets, DBSCAN can still identify the density-connected regions, and K-means can then work on these regions to improve the clustering quality. This makes the hybrid approach a promising solution for many real-world applications where data is often noisy and complex.

C. Complexity Analysis

K-means was computationally faster, but the hybrid approach achieved better trade-offs between complexity and clustering quality, making it more suitable for datasets with noise and irregularly shaped clusters. Although K-means has a relatively low time complexity for each iteration, it may require multiple iterations to converge, and its performance can be highly dependent on the initial centroid selection. In contrast, DBSCAN has a relatively higher time complexity due to the need to calculate the density connectivity of each data point. However, it does not require the specification of the number of clusters in advance and can handle datasets with arbitrary shapes. The hybrid approach combines the advantages of both methods. It first uses DBSCAN to quickly identify the rough structure of the clusters and remove noise, which reduces the computational burden of K-means in subsequent refinement steps. This makes it more efficient and effective in handling complex datasets.

VI. Conclusion

This study provides a comprehensive analysis of clustering methods, including K-means, DBSCAN, K-means++, and hybrid approaches combining DBSCAN with K-means. The experiments conducted on synthetic and real-world datasets have demonstrated several key insights:

- K-means is computationally efficient and effective for well-separated spherical clusters but struggles with noise and irregular shapes.
- K-means++ improves clustering stability by enhancing centroid initialization, making it less sensitive to initial placement.
- K-prototypes and K-medoids extend the applicability of clustering methods to mixed-type data and outlier-prone datasets, respectively, but with higher computational costs.
- DBSCAN excels in identifying arbitrarily shaped clusters and handling noise, but parameter tuning is critical for optimal performance.
- The hybrid DBSCAN + K-means approach combines the strengths of both methods, achieving superior results in scenarios with noise and complex cluster shapes.

These findings highlight the importance of selecting the right clustering algorithm based on data characteristics and specific application requirements. Future work could explore the integration of clustering algorithms with deep learning techniques to further enhance clustering performance and scalability. For example, combining DBSCAN with representation learning methods could improve clustering quality in high-dimensional spaces. Additionally, exploring adaptive algorithms that dynamically adjust parameters based on data distribution could address existing challenges in scalability and sensitivity.

The continued evolution of clustering techniques promises to address increasingly complex data challenges, paving the way for innovative applications across industries. This study serves as a foundational reference for researchers and practitioners seeking to leverage clustering for data-driven insights.

REFERENCES

[1] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[2] D. Arthur and S. Vassilvitskii, "K-means++: The Advantages of Careful Seeding," in Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, 2007.

[3] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, 1996.

[4] R. Campello, D. Moulavi, and J. Sander, "Density-Based Clustering Based on Hierarchical Density Estimates," in ACM Transactions on Knowledge Discovery from Data, 2015.

[5] M. Hahsler and K. Piekenbrock, "dbscan: Fast Density-Based Clustering with R," Journal of Statistical Software, 2018.

[6] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised Deep Embedding for Clustering Analysis," in Proceedings of the 33rd International Conference on Machine Learning, 2016.

[7] E. Aljalbout, V. Golkov, Y. Siddiqui, and D. Cremers, "Clustering with Deep Learning: Taxonomy and New Methods," in arXiv preprint arXiv:1801.07648, 2018.

[8] J. Sander, X. Xu, and M. Ester, "Density-Based Clustering Revisited: Fast DBSCAN with Local Density Adjustment," in Advances in Knowledge Discovery and Data Mining, 2020.

[9] C. Aggarwal and C. Reddy, "Data Clustering: Algorithms and Applications," CRC Press, 2014.

[10] R. Xu and D. Tian, "A Comprehensive Survey of Clustering Algorithms," in Annals of Data Science, 2015.