Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 10, No. 3, June, 2026

A Hierarchical Hybrid Closest Access Point–Medoids Algorithm for Improved Clustering-Based Fingerprint Localization

Abdulmalik Shehu Yaro [1, 2*], Filip Maly [1], Kateřina Frončková [1]

[1] *Department of Informatics and Quantitative Methods, Faculty of Informatics and Management, University of Hradec Kralove, Hradec Kralove 500 03, Czech Republic.*

[2] *Department of Electronics and Telecommunications Engineering, Ahmadu Bello University, Zaria 810106, Nigeria.*

Abstract

Traditional clustering algorithms in fingerprint-based localization often struggle with outliers, overlapping clusters, and irregular RSS variations in fingerprint databases, which reduces clustering accuracy. To address these issues, this study proposes a hierarchical hybrid approach, the closest access point–medoids (CAP-medoids) algorithm, which combines the closest access point (CAP) method with k-medoids clustering. The CAP algorithm generates initial clusters based on the strongest received signal strength (RSS) from nearby wireless access points (APs), while k-medoids refines clusters by selecting actual fingerprint vectors as cluster centers, improving robustness against noise and irregular RSS variations. The algorithm was evaluated on four publicly available fingerprint databases of varying size and density. Performance was assessed using Euclidean, Manhattan, and cosine similarity distances as similarity metrics, with silhouette scores and Davies Bouldin (DB) indices as clustering performance metrics. Results show that the CAP-medoids algorithm consistently produces more compact and well-separated clusters than standard k-medoids in small databases, with silhouette scores increasing up to 75% and DB indices decreasing up to 63%. For larger, high-density databases, performance declines, indicating sensitivity to database size. Comparisons with other hybrid algorithms, including CAP+k-means++ and k-density-based spatial clustering of applications with noise (k-DBSCAN) algorithms, confirm its overall robustness and adaptability.

Keywords:

Fingerprint-Based Localization;
Hierarchical Hybrid;
K-Means;
Pre-Clustering Approach;
Proximity-Based Clustering.

Article History:

Received:	21	October	2025
Revised:	07	May	2026
Accepted:	11	May	2026
Published:	01	June	2026

1- Introduction

Fingerprint-based localization systems with received signal strength (RSS) measurement as a position-dependent signal parameter (PSDSP) have emerged as a pivotal technology in numerous applications, ranging from indoor navigation to asset tracking. The localization performance of the system consists of two phases: the offline and online phases [1]. The offline phase involves generating an RSS-based fingerprint database, while the online phase focuses on predicting an unknown target location based on fingerprint measurements. A fingerprint is a set of RSS measurements collected from spatially deployed wireless access points (APs) at a fixed reference location [1, 2]. RSS measures the signal power detected at the receiver after transmission from the wireless AP.

The density of the fingerprint database is essential for determining localization accuracy. Generally, a higher density results in better performance; however, this improvement is often accompanied by increased localization times [3]. To address this challenge, clustering algorithms for fingerprint databases are employed to divide larger databases into smaller, more manageable groups [4, 5]. However, the accuracy and efficiency of the clustering process depend on the robustness of the algorithm employed. Traditional clustering algorithms often struggle with handling non-Euclidean

* **CONTACT**: abdulmalik.yaro@uhk.cz

DOI: https://doi.org/10.28991/ESJ-2026-010-03-010

distance metrics, managing overlapping clusters, outlier fingerprints, and fingerprint database irregularities [6]. These limitations lead to suboptimal cluster formations, increased computational demands, and reduced reliability, especially in real-world scenarios marked by complexity and variability [6, 7]. Overcoming these challenges requires innovative approaches that combine the strengths of multiple clustering algorithms.

Proximity-based clustering algorithms, such as the closest access points (CAP) algorithm, which operates using the principle of dominant wireless access points (AP), efficiently group fingerprint vectors based on their spatial proximity to wireless APs [3, 8, 9]. These algorithms offer speed and simplicity in generating clusters, but their effectiveness is often limited when confronted with challenges like overlapping clusters. On the other hand, centroid- or medoid-based partitioning algorithms, such as k-medoids, are better at handling noise and outliers due to their robust nature [6, 10]. However, they tend to perform poorly when applied to unstructured or highly variable fingerprint databases, where the underlying signal patterns do not match the assumptions of the algorithm [10]. By combining multiple clustering algorithms to perform the clustering process in a hierarchical manner, the combined strength of both algorithms could mitigate their individual limitations and improve overall clustering performance [8, 11].

By leveraging the combined strengths of two clustering algorithms, this paper improves the clustering performance of a clustering-based fingerprinting system. It proposes a hierarchical hybrid clustering method called the CAP-medoids algorithm. In the first stage of the clustering process, the CAP algorithm quickly forms initial clusters based on proximity to dominant wireless APs, while in the second stage, the k-medoids algorithm is used to refine these clusters, thereby minimizing intra-cluster dissimilarity. This hierarchical structure uses the speed of the CAP algorithm to generate structured initial clusters and the robustness of k-medoids to improve cluster quality. The approach reduces the effect of overlapping clusters and noise. It also controls computational cost and improves performance on unstructured fingerprint databases. The main contributions are as follows. First, the development of a hierarchical hybrid clustering algorithm that integrates CAP and k medoids algorithms. Second, the evaluation of the proposed CA-medoids algorithm and the identification of its optimal clustering configuration.

The remainder of the paper is organized as follows: Section 2 presents the review of related works, while Section 3 presents the theoretical foundation and clustering methodology for the proposed CAP-medoids algorithm. The simulation results and discussion are presented in Section 5, with the conclusion and recommendation for future works in Section 5.

2- Literature Reviews

In the realm of fingerprint-based localization, several clustering algorithms have been proposed through the modification or hybridization of existing clustering techniques, each with unique strengths and limitations. This section reviews notable works in clustering algorithm hybridization, highlighting their shortcomings and justifying the need for the proposed clustering algorithm.

Lin et al. (2023) proposed a two-stage clustering algorithm combining the group matching method (GMM) and a modified k-NN algorithm to improve indoor fingerprinting system localization accuracy [12]. The GMM segments the database using reference signal received power (RSRP) values, while the modified k-NN assesses device locations within these sub-databases. However, the GMM algorithm has limitations, including sensitivity to parameter selection and the creation of uneven group sizes due to its reliance on grouping based on the few strongest signals within each fingerprint.

Another hybrid clustering algorithm, k-DBSCAN, was presented in Gholizadeh et al. [11], which uses k-means++ to generate initial clusters and refines them with the DBSCAN algorithm. However, k-means++ assumes spherical clusters and may struggle with non-convex or irregular data distributions, leading to suboptimal initializations. This can negatively impact the performance of DBSCAN, particularly in databases with varying densities or noise.

Yaro et al. (2023) presented a hybrid clustering algorithm based on the sequential hybridization of the CAP and k-means++ algorithms [8]. The CAP algorithm generates the initial clusters, which are refined using the k-means++ algorithm. However, k-means++ may produce suboptimal results in the presence of noise or outliers, as it relies on centroids that may not represent actual fingerprints and assumes spherical clusters, which can be problematic.

Guo et al. (2024) proposed a hybrid density peak clustering (DPC) and mean-shift algorithm to enhance accuracy and robustness, addressing limitations in traditional DPC methods [13]. While this approach offers improvements, it still faces challenges in dealing with noisy databases that have significant variations in density, which can affect clustering performance and stability.

Kaur & Singh (2015) proposed a hybrid clustering algorithm combining the Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) algorithm with k-means to enhance clustering performance [14]. BIRCH generates numerous initial clusters, later refined by k-means. However, managing these clusters becomes increasingly challenging in large, incrementally growing databases. Additionally, k-means' requirement for predefined cluster numbers can result in suboptimal performance when the true number of clusters varies.

Amiri et al. (2017) presented a hybrid clustering algorithm combining k-means with single linkage clustering [15]. This method uses k-means to generate initial clusters, followed by single linkage clustering for merging. However, k-means performs poorly on unstructured data, making it unsuitable for initial cluster generation, while single linkage clustering is prone to chaining effects, producing elongated or irregular clusters.

Liu et al. (2025) proposed a hybrid clustering algorithm combining k-means with particle swarm optimization (PSO) [16]. The PSO optimizes the cluster centroids instead of relying solely on random initialization, while k-means refines clusters using a fitness function based on minimum spanning tree (MST) and local centroids. This improves cluster compactness and separation. However, the method inherits k-means' sensitivity to the number of clusters to be generated and the spherical cluster assumption, and the added PSO and MST computations increase complexity for large databases.

Woo et al. (2025) proposed a two-phase hybrid clustering framework that integrates deep feature extraction with K-means (HDF-Kmeans) clustering [17]. Autoencoder and LSTM models extract latent features, which K-means then uses to form clusters. This improves clustering quality compared to using K-means alone. However, the method relies on supervised training for feature extraction, increases computational complexity, and depends on fixed data segmentation, which may limit flexibility.

Al-Nussairi et al. (2025) proposed a hybrid clustering framework that uses K‑means to initialize HDBSCAN, combining centroid-based and density-based clustering [18]. This approach improves cluster stability, accuracy, and noise handling compared to using HDBSCAN alone. However, it increases computational complexity and still depends on careful parameter tuning for optimal performance.

A summary of hybrid clustering approaches presented in earlier research works and their limitations is provided in Table 1.

Table 1. Summary of proposed hybrid clustering strategies and their limitations

Reference work	Hybrid clustering strategy	Limitations
Lin et al. [12]	GMM-Modified k-NN	GMM is sensitive to parameter selection and may create uneven group sizes due to grouping based on a few strongest RSRP signals
Guo et al. [13]	DPC-Mean-shift	Performance drops with noisy database and highly varying densities, affecting clustering stability
Yaro et al. [8]	CAP-k-means++	k-means++ assumes spherical clusters and is sensitive to noise/outliers; centroids may not represent real fingerprints
Kaur & Singh [14]	BIRCH -k-means	Difficult to manage large, growing database; k-means requires predefined cluster numbers, which can reduce performance if the true number of clusters varies
Amiri et al. [15]	k-means-Single Linkage	k-means performs poorly on unstructured data; single linkage is prone to chaining, leading to elongated/irregular clusters
Gholizadeh et al. [11]	k-DBSCAN	k-means++ struggles with non-convex or irregular distributions; suboptimal initialization can affect DBSCAN's performance, especially with noisy or varied-density databases
Liu et al. [16]	PSO-k-means	Inherits k‑means' sensitivity to the number of clusters and its assumption of spherical cluster shapes, while the additional PSO and MST computations increase complexity for large databases.
Woo et al. [17]	HDF-k-means	Relies on supervised training for feature extraction which increases computational complexity
Al-Nussairi et al. [18]	k-means- HDBSCAN	Increase in computational complexity and requires careful parameter tuning for optimal results.

Existing hybrid clustering approaches have improved clustering-based fingerprinting localization systems, but they face notable limitations, as shown in Table 1. The GMM algorithm used in the hybrid GMM-modified k-NN algorithm is sensitive to parameter selection and produces uneven cluster sizes. The DBSCAN algorithm used in the k-DBSCAN algorithm is computationally intensive. Additionally, the k-means++ algorithm assumes spherical clusters and struggles with non-convex or irregular data, which negatively affects DBSCAN algorithm performance in noisy or variable-density databases. A similar limitation affects the CAP-k-means++ algorithm due to the reliance on the k-means++ algorithm. The DPC algorithm used in the DPC-mean-shift algorithm struggles with noisy fingerprint databases with varying densities. The BIRCH algorithm in the hybrid BIRCH-k-means algorithm faces challenges in large, incrementally growing databases. The single linkage clustering used in the hybrid k-means-single linkage clustering suffers from chaining effects, producing elongated clusters, while k-means poorly handles unstructured fingerprint databases. The PSO-k-means algorithm inherits k-means' sensitivity to the number of clusters and spherical cluster assumption, and the additional PSO and MST computations increase complexity for large databases. The HDF-k-means algorithm relies on supervised training for feature extraction, which increases computational complexity. Finally, the k-means-HDBSCAN algorithm has increased computational complexity and requires careful parameter tuning for optimal results.

To address the limitations of previous hybrid clustering approaches, this paper proposes a hierarchical combination of the CAP algorithm and the k-medoids algorithm, referred to as the CAP-medoids algorithm. In this approach, the CAP algorithm performs coarse clustering by generating the initial clusters, while the k-medoids algorithm performs

refinement within each initial cluster to improve intra-cluster compactness. This hierarchical structure enables the CAP-medoids algorithm to achieve both computational efficiency and clustering robustness, offering a balance that other hybrid methods do not provide. For example, in the hybrid CAP-k-means++ algorithm, the k-means++ algorithm can be heavily affected by noise or outliers, and its assumption of spherical clusters often leads to suboptimal results. Similarly, the k-DBSCAN algorithm relies on the k-means++ algorithm for initialization, which can produce poor starting clusters that reduce the effectiveness of the DBSCAN algorithm. Additionally, the DBSCAN algorithm is computationally intensive compared to either CAP or k-medoids algorithm. In contrast, the CAP algorithm quickly forms initial clusters based on proximity to APs, providing a structured starting point, while the k-medoids algorithm refines these clusters by minimizing dissimilarities, handling noise, RSS outliers, and irregular cluster shapes more effectively.

In the next section of the paper, the CAP-medoids algorithm clustering methodology is presented.

3- Proposed CAP-Medoids Algorithm Theoretical Foundation and Clustering Methodology

This section presents the theoretical foundation and a detailed description of the clustering methodology employed by the proposed CAP-medoids algorithm. The proposed approach follows a hierarchical two stage structure. The first stage performs coarse partitioning using the CAP algorithm, which operates on the principle of wireless AP dominance. The second stage refines each initial partition using the k-medoids algorithm.

3-1- Theoretical Foundation of the Proposed CAP-Medoids Clustering Framework

RSS fingerprint vectors are strongly influenced by spatial proximity, multipath propagation, shadowing, and hardware variability. In indoor environments, signal strength generally decays with distance according to large-scale path loss models. As a result, the strongest RSS value in a fingerprint vector is often associated with the physically nearest wireless AP. However, in realistic indoor environments, factors such as multipath fading, interference, and dynamic obstacles can occasionally alter the dominant wireless AP measurement.

Despite these variations, the dominant wireless AP still provides a meaningful coarse indicator of spatial proximity. Fingerprint vectors that share the same dominant wireless AP tend to be closer to one another than to vectors dominated by other wireless APs. The CAP algorithm leverages this property for coarse clustering, grouping fingerprint vectors based on dominant wireless APs to partition the global database into subspaces that reflect the spatial distribution of signals. This approach reduces intra-cluster variance and limits the search space for subsequent refinement.

The second stage of the CAP-medoids framework applies the k-medoids algorithm to refine each coarse cluster. The theoretical foundation of k-medoids is based on robust partitioning and representative selection. Unlike centroid-based methods such as k-means, which compute cluster centers as the mean of all fingerprint vectors and assume convex cluster shapes, k-medoids selects actual fingerprint vectors known as medoids as cluster representatives. This makes the algorithm resilient to outliers, non-Gaussian distributions, and irregular cluster shapes, which are common in RSS fingerprint databases.

Within each coarse cluster, k-medoids iteratively assigns each fingerprint vector to the nearest medoid according to the chosen similarity metric, such as Euclidean or Manhattan distances, and updates medoids to minimize the total intra-cluster dissimilarity. This process theoretically ensures that clusters are compact and homogeneous, with medoids serving as physically meaningful representatives of the underlying data. By reducing intra-cluster distances and improving cluster cohesion, k-medoids enhances the separability of subclusters and preserves the intrinsic spatial relationships in the fingerprint space.

By combining CAP for coarse partitioning and k-medoids for fine refinement, the hierarchical CAP-medoids framework maintains robustness under realistic indoor conditions, producing clusters that are both physically meaningful and statistically compact. This theoretical foundation supports improved localization accuracy and reliability in subsequent processing.

3-2- CAP-Medoids Clustering Methodology

This section describes the step-by-step implementation of the proposed CAP-medoids clustering algorithm. As earlier stated, the clustering methodology follows a hierarchical two-stage approach. The first stage performs coarse partitioning of the fingerprint database using the CAP algorithm, exploiting the dominance of the strongest wireless AP to create physically meaningful clusters. The second stage refines these initial clusters using the k-medoids algorithm, improving cluster compactness and reducing intra-cluster variability.

3-2-1- First-Stage: Generation of Initial Clusters

The initial clusters are generated using the CAP algorithm, which partitions fingerprint vectors according to their dominant wireless AP. The dominant wireless AP is defined as the access point with the highest RSS measurement within each fingerprint vector, typically corresponding to the physically nearest wireless AP [9].

Let the fingerprint database (**F**) consist of N fingerprint vectors:

$$\mathbf{F} = \{\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \dots, \mathbf{f}_N\} \quad (1)$$

$$\mathbf{f}_i = \left[rss_{i,1}, rss_{i,2}, \dots, rss_{i,M}\right] \quad \text{for } 1 \le i \le N \quad (2)$$

where, $\mathbf{f}_i$ is the *i*-th fingerprint vector M represents the number of wireless APs.

The clustering procedure of the CAP algorithm used for coarse clustering is described in Steps 1 and 2:

Step 1: Dominant Wireless AP Selection:

For each fingerprint vector, $\mathbf{f}_i$, determine the index of the dominant wireless AP. The index of the dominant wireless AP is determined mathematically using Equation 3

$$k_i = \text{arg\_max}\{rss_{i,1}, rss_{i,2}, \dots, rss_{i,M}\} \quad (3)$$

where, k_i denotes the index of the dominant wireless AP for $\mathbf{f}_i$.

Step 2: Cluster Assignment:

Assign each fingerprint vector $\mathbf{f}_i$ to the cluster corresponding to its dominant wireless AP. This process generates M initial clusters. The *j*-th cluster (C_j), is defined as shown in Equation 4:

$$C_j = \{\mathbf{f}_i \in \mathbf{F} | k_i = j\} \quad for\ 1 \le j \le M \quad (4)$$

By grouping fingerprint vectors according to their dominant wireless AP, the global database F is partitioned into physically meaningful subspaces. The resulting clusters, C_j, are then refined using the k-medoids algorithm in the second stage of the CAP-medoids clustering process. A detailed description of the cluster refinement stage of the CAP-medoids algorithm is provided in the following section.

3-2-2- Second-stage: Cluster Refinement

The k-medoids algorithm is employed to further refine the clusters generated by the CAP algorithm (C_j) by partitioning each cluster into *K* subclusters. This is done as follows:

Step 1: Medoids Initialization:

Select *K* initial medoids, $\{\mathbf{m}_1, .., \mathbf{m}_K\}$, from C_j here each **m** represents a medoid. These medoids are representatives fingerprint vectors from the cluster.

Step 2: Cluster Assignment:

Assign each fingerprint vector $\mathbf{f}_{j,1} \in C_j$ where $\mathbf{f}_{j,1}$ represents the i-th fingerprint in cluster C_j, to the nearest medoid based on the chosen distance‐based fingerprint similarity metric.

$$S_k = \left\{ \mathbf{f}_{j,1} \in C_j \middle| \mathbf{m}_k = \underset{\mathbf{m}_k \in \{\mathbf{m}_1, .., \mathbf{m}_k\}}{\text{arg\_max}} d(\mathbf{f}_{j,i}, \mathbf{m}_k) \right\} \quad (5)$$

where: S_k is the set of fingerprint vectors assigned to medoids $\mathbf{m}_k$, and $d(\cdot,\cdot)$ denotes the chosen fingerprint similarity metric.

Step 3: Medoid Update:

Update each medoid by selecting a new medoid from S_k that minimizes the total intra-cluster distance determined using Equation 6.

$$\mathbf{m}_k = \underset{\mathbf{f} \in S_k}{\text{arg\_max}} \sum_{\mathbf{f}_{j,i}} d(\mathbf{f}, \mathbf{f}_{j,i}) \quad (6)$$

Step 4: Iteration:

Repeat Steps 2 and 3 until the medoids stabilize (no changes) or the improvement in the total intra-cluster distance falls below a predefined threshold.

Step 5: Clustering Result:

The refined clusters C_j is partitioned into K clusters:

$$C_j = \bigcup_{k=1}^{K} S_k \tag{7}$$

where, $S_k \cap S_{\acute{k}} = \emptyset$ for $k \neq \acute{k}$.

The cluster refinement from steps 1 to 5 guarantees that the fingerprint vectors within each subcluster are closely clustered around their respective medoids, resulting in a more refined clustering structure. This refinement improves the accuracy of subsequent localization processes. Figure 1 shows an overview of the clustering process used in the CAP-medoids algorithm.

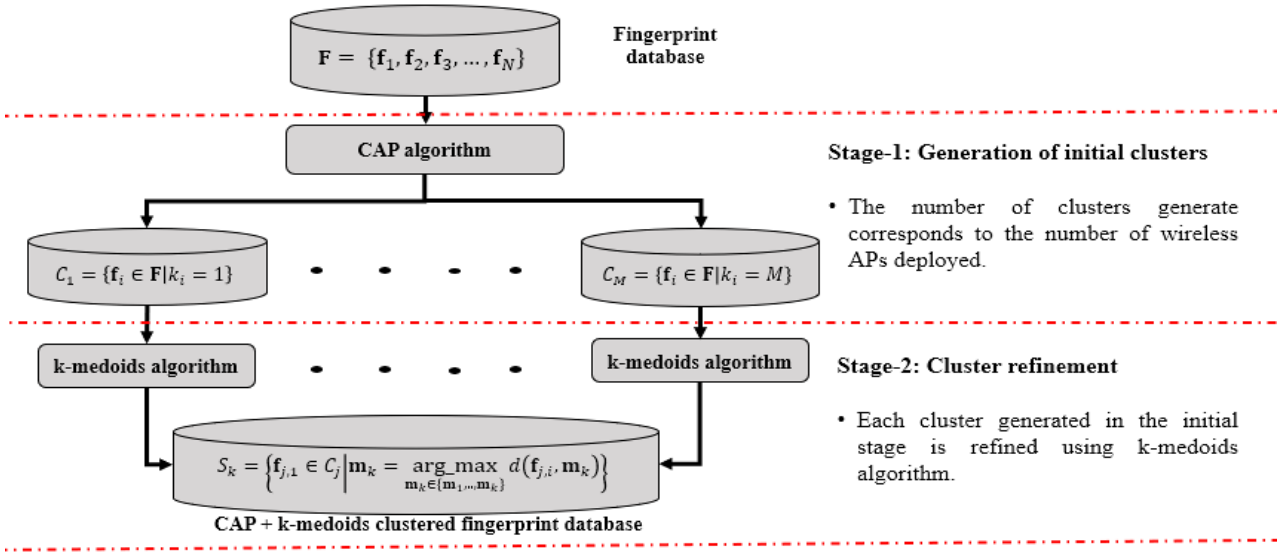


Figure 1. The hierarchical clustering methodology of CAP-medoids algorithm

The next section of the paper provides a clustering performance analysis of the CAP-medoids algorithm across different fingerprint databases and clustering configurations.

4- Simulation Results, Comparison and Discussion

This section evaluates the clustering performance of the CAP-medoids algorithm across a range of experimentally generated fingerprint databases and clustering configurations. First, the simulation parameters used for the evaluation are presented, followed by the clustering simulation results and comparative analysis.

4-1- Simulation Parameter

The CAP-medoids algorithm is evaluated across four experimentally generated, publicly accessible, RSS-based fingerprint databases: SEUG_IndoorLoc [19], IIRC_IndoorLoc [20], PIEP_UM_IndoorLoc [21], and MSI_IndoorLoc [22]. A summary of the characteristics of each fingerprint database, including the number of wireless APs, the number of RLs, wireless technology, and the coverage area in which the fingerprint measurements were taken, is presented in Table 2.

Table 2. Database characteristics considered for performance evaluation

Fingerprint Database	Database characteristic			
	Wireless technology	Number of APs (M)	Number of RL (N)	Coverage area (m^2)
SEUG_IndoorLoc	Wi-Fi	3	49	33
IIRC_IndoorLoc	ZigBee	3	68	161
PIEP_UM_IndoorLoc	Wi-Fi	8	1000	1000
MSI_IndoorLoc	Wi-Fi	11	631	1000

Table 2 presents the characteristics of the fingerprint databases used to evaluate the CAP-medoids algorithm. The databases differ in the number of wireless APs, ranging from 3 in SEUG_IndoorLoc and IIRC_IndoorLoc to 11 in MSI_IndoorLoc. Coverage areas vary from 33 m² in SEUG_IndoorLoc to 1000 m² in both PIEP_UM_IndoorLoc

and MSI_IndoorLoc. The number of reference locations (RLs) also varies considerably, with 49 in SEUG_IndoorLoc, 68 in IIRC_IndoorLoc, 1000 in PIEP_UM_IndoorLoc, and 631 in MSI_IndoorLoc. These databases provide diverse scenarios for assessing the performance of the CAP-medoids algorithm across different wireless configurations.

Three commonly used similarity assessment metrics, namely Euclidean distance, Manhattan distance, and cosine similarity distance, are considered as the fingerprint similarity metrics. To ensure optimal performance evaluation, the number of clusters generated by the k-medoids algorithm is determined using the elbow method [23]. Additionally, the silhouette score and Davies–Bouldin (DB) index are used as clustering performance metrics to evaluate the quality of the clusters formed by the algorithm [24, 25]. The silhouette score for a fingerprint vector is defined as [24, 25]:

$$s(\mathbf{f}_i) = \frac{b(\mathbf{f}_i) - a(\mathbf{f}_i)}{\max\{b(\mathbf{f}_i), a(\mathbf{f}_i)\}} \tag{8}$$

where, $a(\mathbf{f}_i)$ is the average intra-cluster distance and $b(\mathbf{f}_i)$ is the average inter-cluster distance. The silhouette score ranges from −1 to 1, with higher values indicating better cluster separation and cohesion.

The DB index is defined as [26]:

$$DB = \frac{1}{K}\sum_{i=1}^{K} \max_{j \neq 1} \left(\frac{d_i + d_j}{M_{ij}}\right) \tag{9}$$

where K is the number of clusters, d_i is the average distance between fingerprint vectors in cluster C_i and its medoid, and M_{ij} is the distance between the medoids of cluster C_i and C_j. Lower DB values indicate more compact and well-separated clusters.

4-2- Clustering Performance Comparison

In this section of the paper, the clustering performance of the CAP-medoids algorithm is evaluated using silhouette score and DB index as clustering performance metrics. Its performance is compared with the standard k-medoids (sk-medoids) algorithm, as well as other hybrid clustering algorithms reported in the literature, across all fingerprint databases with characteristics provided in Table 2. First, the performance comparison between the CAP-medoids and sk-medoids algorithms is presented, followed by the performance comparison between CAP-medoids and other hybrid clustering algorithms.

4-2-1- Performance Comparison: CAP-medoids vs. sk-medoids Algorithms

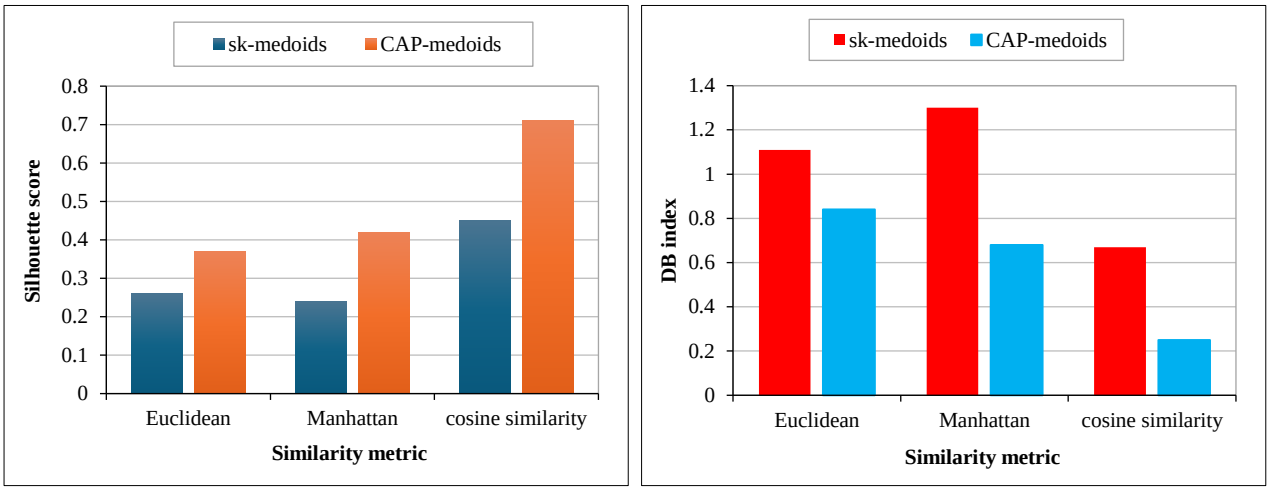
Table 3 compares the silhouette scores of the CAP-medoids and sk-medoids algorithms on the SEUG_IndoorLoc, IIRC_IndoorLoc, PIEP_UM_IndoorLoc, and MSI_IndoorLoc databases. Entries highlighted in green indicate the clustering algorithm with the best performance based on the silhouette scores and DB index.

Table 3. Comparison of silhouette scores and DB indices between CAP-k-medoids and sk-medoids across the SEUG_IndoorLoc, IIRC_IndoorLoc, PIEP_UM_IndoorLoc, and MSI_IndoorLoc database

Database	Similarity metric	Silhouette scores		DB index	
		sk-medoids	CAP-medoids	sk-medoids	CAP-medoids
SEUG_IndoorLoc	Euclidean distance	0.26	0.37	1.11	0.84
	Manhattan distance	0.24	0.42	1.30	0.68
	Cosine similarity	0.45	0.71	0.67	0.25
IIRC_IndoorLoc	Euclidean	0.30	0.34	1.04	0.84
	Manhattan	0.24	0.36	1.21	0.86
	Cosine similarity	0.56	0.63	0.47	0.28
PIEP_UM_IndoorLoc	Euclidean distance	0.14	0.10	1.81	2.30
	Manhattan distance	0.18	0.09	1.60	2.67
	Cosine similarity	0.27	0.21	1.55	2.54
MSI_IndoorLoc	Euclidean distance	0.20	0.17	1.67	1.76
	Manhattan distance	0.18	0.16	1.56	1.91
	Cosine similarity	0.33	0.23	1.60	2.10

Performance Comparison Across SEUG_IndoorLoc Database

Examining the performance across the SEUG_IndoorLoc database, which can be considered a small-size database with a fingerprint vector dimension of 3 and density of 49, it can be seen that the CAP-medoids algorithm generated clusters with higher silhouette scores and lower DB indices compared to the sk-medoids algorithm. This is irrespective of the similarity metric used, as shown graphically in Figure 2. Specifically, with the Euclidean distance as the similarity metric, the CAP-medoids algorithm generated clusters with a mean silhouette score of 0.37, while the sk-medoids algorithm generated clusters with a mean score of 0.26. This corresponds to an increase of about 42% in cluster cohesion and separation. The mean DB index for the clusters generated decreased from 1.11 for the sk-medoids algorithm to 0.84 for the CAP-medoids algorithm, which represents a reduction of about 24%. These results indicate that the CAP-medoids algorithm forms more compact clusters with improved inter-cluster separation with the Euclidean distance as the similarity metric in the SEUG_IndoorLoc database.



(a) Silhouette scores comparison (b) DB indices comparison

Figure 2. Graphical comparison of silhouette scores and DB indices for CAP-k-medoids and sk-medoids on the SEUG_IndoorLoc database

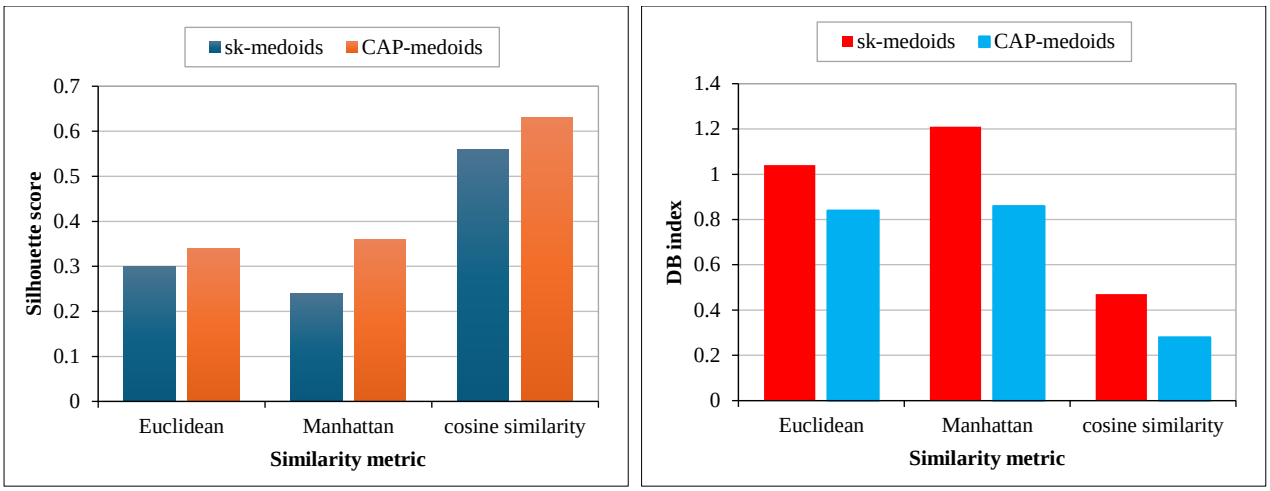
With the Manhattan distance as the similarity metric, the CAP-medoids algorithm generated clusters with a mean silhouette score of 0.42, while the sk-medoids algorithm generated clusters with a mean score of 0.24. This reflects an increase of about 75% in cluster cohesion and separation. The mean DB index for the clusters generated decreased from 1.30 for the sk-medoids algorithm to 0.68 for the CAP-medoids algorithm, which corresponds to a reduction of about 48%. The magnitude of improvement with the Manhattan distance as a similarity metric suggests that the CAP-medoids algorithm better captures the structure of the fingerprint space when distance accumulation along dimensions is considered.

The most pronounced performance improvement across the SEUG_IndoorLoc database was observed when the cosine similarity distance was used as the similarity metric. The mean silhouette score of the clusters generated increased from 0.45 for the sk-medoids algorithm to 0.71 for the CAP-medoids algorithm, which represents an increase of about 59%. The mean DB index decreased from 0.67 to 0.25, which corresponds to a reduction of about 63%. The very low DB index and high silhouette score indicate strong intra-cluster similarity and clear inter-cluster boundaries.

These results confirm that the CAP-medoids algorithm consistently forms more compact and well-separated clusters for the SEUG_IndoorLoc database. Across Euclidean, Manhattan, and cosine similarity distances as similarity metrics, the CAP-medoids algorithm generated clusters with higher silhouette scores and lower DB index values, with improvements ranging from about 42% to 75% in silhouette score and reductions of about 24% to 63% in DB index. The strongest performance was observed with the cosine similarity distance as a similarity metric, where the algorithm produced clusters with a mean silhouette score of 0.71 and a mean DB index of 0.25. This indicates strong intra-cluster consistency and clear inter-cluster separation. Overall, the CAP-medoids algorithm demonstrates superior clustering quality for this database, regardless of the similarity metric applied.

Performance Comparison Across IIRC_IndoorLoc Database

Extending the analysis to the IIRC_IndoorLoc database, which has slightly higher density but the same fingerprint vector dimension and is also categorized as small sized, the CAP-medoids algorithm again achieved better clustering quality across all evaluated similarity metrics, as illustrated in Figure 3.



(a) silhouette scores comparison

(b) DB indices comparison

Figure 3. Graphical comparison of silhouette scores and DB indices for CAP-k-medoids and sk-medoids on the IIRC_IndoorLoc database

With the Euclidean distance as the similarity metric, the CAP-medoids algorithm generated clusters with a mean silhouette score of 0.34, while the sk-medoids algorithm generated clusters with a mean score of 0.30. This represents an increase of about 13%. The mean DB index decreased from 1.04 for the sk-medoids algorithm to 0.84 for the CAP-medoids algorithm, which corresponds to a reduction of about 19%. These results indicate moderate improvement in cluster compactness and separation with the Euclidean distance as a similarity metric.

Using Manhattan distance as the similarity metric, the CAP-medoids algorithm generated clusters with a mean silhouette score of 0.36, while the sk-medoids algorithm generated clusters with a mean score of 0.24. This corresponds to an increase of about 50%. The mean DB index decreased from 1.21 for the sk-medoids algorithm to 0.86 for the CAP-medoids, which represents a reduction of about 29%. The improvement with the Manhattan distance as a similarity metric is more pronounced than under the Euclidean distance.

Just like in the SEUG_IndoorLoc database, the strongest performance in the IIRC_IndoorLoc database was observed with the cosine similarity distance as a similarity metric. The mean silhouette score of the clusters generated increased from 0.56 for the sk-medoids algorithm to 0.63 for the CAP-medoids algorithm, which corresponds to an increase of about 13%. The mean DB index decreased from 0.47 to 0.28, which represents a reduction of about 40%. The lower DB index indicates improved inter-cluster separation with tighter intra-cluster grouping.

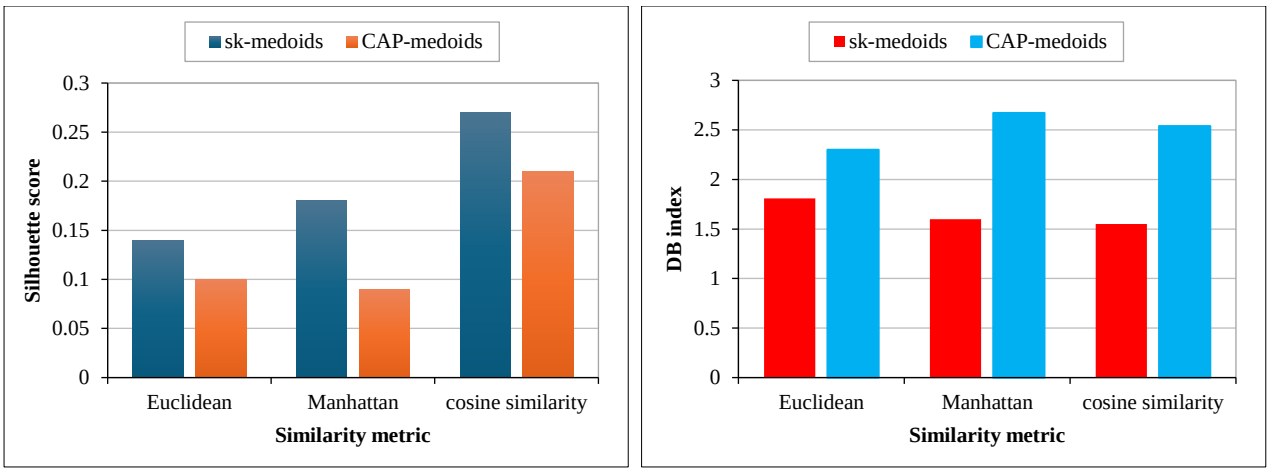
Overall, the CAP-medoids algorithm maintains superior clustering performance for the IIRC_IndoorLoc database across all similarity metrics considered. The consistent gains in silhouette score and reductions in DB index confirm its robustness even with slightly increased database density.

Performance Comparison Across PIEP_UM_IndoorLoc Database

The performance of the CAP-medoids algorithm on the PIEP_UM_IndoorLoc database shows a contrasting trend compared to the smaller SEUG_IndoorLoc and IIRC_IndoorLoc databases. This database is considerably larger and denser, with a fingerprint vector dimension of 8 and a density of 1000, which increases the complexity of clustering and the variability in fingerprint vectors.

Across all similarity metrics considered, the CAP-medoids algorithm produced lower silhouette scores and higher DB indices than the sk-medoids algorithm, as shown in Figure 4.

Using the Euclidean distance as a similarity metric, the mean silhouette score of the clusters generated decreased from 0.14 for the sk-medoids algorithm to 0.10 for the CAP-medoids algorithm, while the DB index increased from 1.81 to 2.30. This respectively represents a decrease of about 29% and an increase of about 27% in the mean silhouette scores and mean DB index. Using Manhattan distance, the mean silhouette score of the clusters generated dropped from 0.18 for the sk-medoids algorithm to 0.09 for the CAP-medoids algorithm, a reduction of 50%. The DB index rose from 1.60 to 2.67, which is an increase of about 67%. With cosine similarity distance, the silhouette score of the clusters generated dropped from 0.27 to 0.21, a decrease of about 22%, while the DB index increased from 1.55 to 2.54, representing a rise of about 64%.



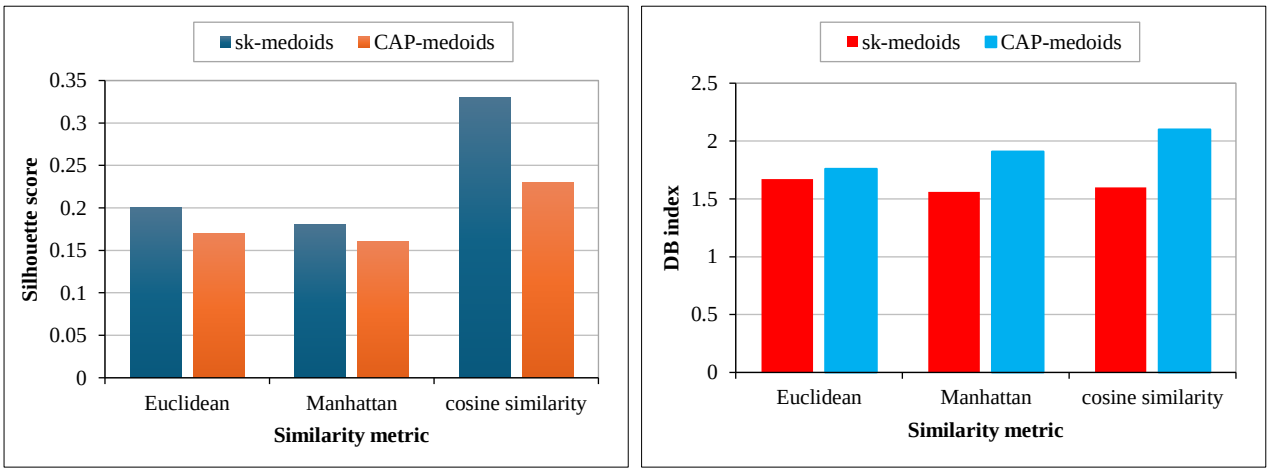
(a) Silhouette scores comparison (b) DB indices comparison

Figure 4. Graphical comparison of silhouette scores and DB indices for CAP-k-medoids and sk-medoids on the PIEP_UM_IndoorLoc database

These results indicate that the CAP-medoids algorithm struggles to maintain cluster compactness and separation in larger, high-density databases. The consistent decreases in silhouette scores and increases in DB indices of the clusters generated suggest weaker intra-cluster cohesion and greater overlap between clusters. This contrasts with its performance on smaller databases, highlighting that the CAP-medoids algorithm is more effective for small-sized databases but less suitable as the dimension and density increase, likely due to the increased diversity and sparsity of RSS patterns in the fingerprint vectors.

Performance Comparison Across PIEP_UM_IndoorLoc Database

Figure 5 presents a graphical comparison of clustering performance for the PIEP_UM_IndoorLoc database across all evaluated similarity metrics, highlighting the differences between the CAP-medoids and sk-medoids algorithms.



(a) Silhouette scores comparison (b) DB indices comparison

Figure 5. Graphical comparison of silhouette scores and DB indices for CAP-k-medoids and sk-medoids on the MSI_IndoorLoc database

The results for the MSI_IndoorLoc database show a similar trend to the PIEP_UM_IndoorLoc database, where the CAP-medoids algorithm does not outperform the sk-medoids algorithm. This database is also large and dense, with a fingerprint vector dimension of 11 and a density of 631. Compared to the smaller SEUG_IndoorLoc and IIRC_IndoorLoc databases, the increase in database size and dimension appears to reduce the effectiveness of the CAP-medoids algorithm.

Using Euclidean distance as a similarity metric, the CAP-medoids algorithm generated clusters with a mean silhouette score of 0.17, compared to 0.20 for the sk-medoids algorithm, representing a decrease of about 15%. The DB index increased from 1.67 to 1.76, which is an increase of about 5%. With the Manhattan distance, the silhouette score decreased from 0.18 to 0.16, a reduction of 11%, while the DB index increased from 1.56 to 1.91, corresponding to an increase of about 22%. For cosine similarity distance, the silhouette score dropped from 0.33 to 0.23, a decrease of 30%, and the DB index increased from 1.60 to 2.10, representing a rise of about 31%.

These results indicate that the CAP-medoids algorithm struggles to maintain cluster compactness and separation in larger, high-density databases. The consistent declines in silhouette scores and increases in DB indices of the clusters generated suggest weaker intra-cluster cohesion and greater cluster overlap.

Overall Performance Analysis and Comparison: CAP-Medoids vs sk-Medoids Algorithms

The comparative analysis across all four evaluated fingerprint databases highlights a clear relationship between database size, density, and the relative performance of the CAP-medoids and sk-medoids algorithms.

For small, dense databases such as SEUG_IndoorLoc and IIRC_IndoorLoc, the CAP-medoids algorithm consistently outperforms the sk-medoids algorithm across all similarity metrics considered. The algorithm generated clusters with higher silhouette scores, with increases ranging from about 13% to 75%, and lower DB indices, with reductions of approximately 19% to 63%. The largest gains were observed using cosine similarity distance as a similarity metric, where the CAP-medoids algorithm achieved strong intra-cluster cohesion and clear inter-cluster separation. These results confirm that the CAP-medoids algorithm effectively captures the natural partitioning in small, dense fingerprint databases, producing compact and well-separated clusters.

In contrast, for larger and high-density databases such as PIEP_UM_IndoorLoc and MSI_IndoorLoc, the performance advantage of the CAP-medoids algorithm diminishes. Across Euclidean, Manhattan, and cosine similarity distances, the CAP-medoids algorithm consistently produced clusters with lower silhouette scores and higher DB indices compared to the sk-medoids algorithm. The declines in silhouette scores ranged from approximately 11% to 50%, while DB indices increased by 5% to 67%. The largest discrepancies occurred with cosine similarity distance as a similarity metric, suggesting that the CAP-medoids algorithm has difficulty maintaining angular separation in large and complex fingerprint databases.

These observations indicate that the effectiveness of the CAP-medoids algorithm is strongly dependent on database characteristics. It excels in small, dense databases with limited fingerprint vector dimensions and densities, but its advantages decrease as database size and density increase. The sk-medoids algorithm, while slightly less effective in small databases, provides more stable clustering outcomes in larger and dense fingerprint databases.

Overall, the comparison demonstrates that the CAP-medoids algorithm is the preferred choice for small, dense fingerprint databases, whereas the sk-medoids algorithm may be more reliable for large-scale and high-density databases. This trend is visually supported by Figures 2 to 5, which show the performance differences across all similarity metrics and databases.

4-2-2- Performance comparison: CAP-medoids vs CAP-k-mean++ vs k-mean++-DBSCAN

Following the analysis of CAP-medoids versus sk medoids, it is important to examine how the CAP-medoids algorithm performs relative to other hybrid clustering algorithms reported in the literature. This comparison highlights its competitiveness not only against the sk-medoids algorithm but also against other hybrid approaches. Specifically, the CAP-medoids algorithm is compared with CAP-k-means++ [8] and k-DBSCAN [11] to assess its relative effectiveness in forming compact and well-separated clusters across the SEUG_IndoorLoc and IIRC_IndoorLoc databases. Table 4 provide the silhouette score comparison between the CAP-k-medoids, CAP-k-means++, and k-means++-DBSCAN algorithms across the SEUG_IndoorLoc and IIRC_IndoorLoc databases. Entries highlighted in green indicate the clustering algorithm with the best performance, that is, the algorithm with the highest silhouette scores for each similarity metric.

Table 4. Comparison of silhouette scores between CAP-k-medoids, CAP-k-means++, and k-DBSCAN on the SEUG_IndoorLoc and IIRC_IndoorLoc databases

Database	Similarity metric	Silhouette scores		
		CAP-k-mean++	k-DBSCAN	CAP-medoids
IIRC_IndoorLoc	Euclidean	0.37	0.13	0.37
	Manhattan	0.37	0.24	0.42
	Cosine similarity	0.64	0.25	0.71
SEUG_IndoorLoc	Euclidean	0.37	0.29	0.37
	Manhattan	0.34	0.30	0.42
	Cosine similarity	0.69	0.29	0.71

The results show that the CAP-medoids algorithm generally generates clusters with higher mean silhouette scores across both the IIRC_IndoorLoc and SEUG_IndoorLoc databases. For the IIRC_IndoorLoc database, using Euclidean distance as a similarity metric, the CAP-medoids and CAP-k-means++ algorithms both generated clusters with a mean

silhouette score of 0.37, while k-DBSCAN generated clusters with a mean score of 0.13. Using Manhattan distance, the CAP-medoids algorithm generated clusters with a mean score of 0.42, slightly higher than those of the CAP-k-means++ algorithm at 0.37 and significantly higher than the k-DBSCAN algorithm at 0.24. With cosine similarity distance, the CAP-medoids algorithm again performed best, generating clusters with a mean score of 0.71, compared to 0.64 for CAP+k-means++ and 0.25 for k-DBSCAN.

A similar trend is observed for the SEUG_IndoorLoc database. Using Euclidean distance, the CAP-medoids and CAP-k-means++ algorithms both produced clusters with a mean score of 0.37, while k-DBSCAN produced clusters with a mean score of 0.29. Using Manhattan distance, CAP-medoids generated clusters with a mean score of 0.42, outperforming 0.34 for CAP-k-means++ and 0.30 for k-DBSCAN. With cosine similarity distance, CAP medoids again generated clusters with the highest mean score of 0.71, slightly higher than 0.69 for CAP-k-means++ and significantly higher than 0.29 for k-DBSCAN.

These results indicate that the CAP-medoids algorithm consistently matches or outperforms other hybrid clustering approaches, particularly for similarity metrics that capture angular relationships in fingerprint vectors. The CAP-k-means++ algorithm shows competitive performance in some cases, while k-DBSCAN consistently underperforms across all evaluated metrics. For the Euclidean distance metric, where the CAP-medoids and CAP-k-means++ algorithms show comparable performance in both databases, the advantage of using the CAP-medoids algorithm arises from the second stage of its processing. In this stage, the k-medoids algorithm selects actual fingerprint vectors as cluster centers, reducing sensitivity to outliers and irregular RSS variations that frequently occur in fingerprinting-based indoor localization environments. Overall, the CAP-medoids demonstrates robust clustering quality and strong adaptability across different fingerprint databases, confirming its advantage over both traditional and hybrid alternatives.

5- Conclusion

The paper presents a hierarchical hybrid clustering approach that integrates CAP and k-medoids algorithms to enhance clustering operations in fingerprint-based indoor localization systems. In the first stage, the CAP algorithm quickly generates initial clusters based on the strongest RSS values from wireless APs, leveraging the natural dominance property of fingerprint vectors. The second stage applies k-medoids to refine these clusters by selecting actual fingerprint vectors as cluster centers, reducing sensitivity to noise, outliers, and irregular RSS variations. This approach ensures both computational efficiency and improved cluster quality, addressing challenges commonly encountered in fingerprinting-based indoor localization systems.

Simulation result and analysis across four fingerprint databases namely SEUG_IndoorLoc, IIRC_IndoorLoc, PIEP_UM_IndoorLoc, and MSI_IndoorLoc databases, demonstrates that the CAP-medoids algorithm consistently produces more compact and well-separated clusters than the sk-medoids algorithm for small, dense fingerprint databases. In SEUG_IndoorLoc and IIRC_IndoorLoc databases, silhouette scores of the clusters generated increased by 13% to 75%, and DB indices decreased by 19% to 63% across Euclidean, Manhattan, and cosine similarity distance metrics. These results highlight the algorithm's ability to maintain strong intra-cluster cohesion and clear inter-cluster separation. For larger and high-density databases such as PIEP_UM_IndoorLoc and MSI_IndoorLoc, the CAP-medoids algorithm showed reduced effectiveness, with silhouette scores of generated clusters decreasing by 11% to 50% and DB indices increasing by up to 67% compared to the sk-medoids algorithm, suggesting that database size and density significantly influence clustering performance.

Comparison with other hybrid algorithms, including CAP-k-means++ and k-DBSCAN algorithms, further confirms the robustness of the CAP-medoids algorithm. It consistently matched or exceeded the CAP-k-means++ algorithm performance and outperformed the k-DBSCAN algorithm across all similarity metrics considered, particularly for angular metrics such as the cosine similarity distance. This demonstrates its adaptability across different fingerprint databases and clustering contexts.

Overall, the CAP-medoids algorithm hybrid approach provides a reliable and flexible solution for fingerprinting-based indoor localization clustering operations, particularly in small to moderately sized databases. Future research could extend its applicability by exploring pattern-based similarity metrics, such as the context similarity coefficient (CSC), or integrating CAP with alternative clustering techniques like affinity propagation or DBSCAN to further improve clustering quality and robustness.

6- Declarations

6-1- Author Contributions

Conceptualization, A.S.Y.; methodology, A.S.Y.; software, F.M.; validation, A.S.Y.; formal analysis, A.S.Y. and K.F.; investigation, A.S.Y.; resources, F.M.; data curation, A.S.Y. and K.F.; writing—original draft preparation, A.S.Y.; writing—review and editing, A.S.Y.; visualization, A.S.Y.; supervision, F.M.; project administration, F.M.; funding acquisition, F.M. All authors have read and agreed to the published version of the manuscript.

6-2- Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6-3- Funding and Acknowledgments

The author acknowledges funding from the UHK ReGa internal grant (Project No. UHK_ReGa_00006) of the University of Hradec Králové, Czech Republic.

6-4- Institutional Review Board Statement

Not applicable.

6-5- Informed Consent Statement

Not applicable.

6-6- Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

7- References

[1] Shang, S., & Wang, L. (2022). Overview of WiFi fingerprinting-based indoor positioning. IET Communications, 16(7), 725–733. doi:10.1049/cmu2.12386.

[2] Kolakowski, M. (2020). Automatic radio map creation in a fingerprinting-based BLE/UWB localisation system. IET Microwaves, Antennas & Propagation, 14(14), 1758–1765. doi:10.1049/iet-map.2019.0953.

[3] Yaro, A. S., Malý, F., & Malý, K. (2023). A Two-Nearest Wireless Access Point-Based Fingerprint Clustering Algorithm for Improved Indoor Wireless Localization. Emerging Science Journal, 7(5), 1762–1770. doi:10.28991/ESJ-2023-07-05-019.

[4] Xu, D., & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms. Annals of Data Science, 2(2), 165–193. doi:10.1007/s40745-015-0040-1.

[5] Shtar, G., Shapira, B., & Rokach, L. (2019). Clustering Wi-Fi fingerprints for indoor–outdoor detection. Wireless Networks, 25(3), 1341–1359. doi:10.1007/s11276-018-1753-9.

[6] Ezugwu, A. E., Ikotun, A. M., Oyelade, O. O., Abualigah, L., Agushaka, J. O., Eke, C. I., & Akinyelu, A. A. (2022). A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. Engineering Applications of Artificial Intelligence, 110, 104743. doi:10.1016/j.engappai.2022.104743.

[7] Djouzi, K., & Beghdad-Bey, K. (2019). A Review of Clustering Algorithms for Big Data. 2019 International Conference on Networking and Advanced Systems (ICNAS), 1–6. doi:10.1109/ICNAS.2019.8807822.

[8] Yaro, A. S., Maly, F., Prazak, P., & Malý, K. (2024). Improved Fingerprint-Based Localization Based on Sequential Hybridization of Clustering Algorithms. Emerging Science Journal, 8(2), 394–406. doi:10.28991/ESJ-2024-08-02-02.

[9] Quezada-Gaibor, D., Torres-Sospedra, J., Nurmi, J., Koucheryavy, Y., & Huerta, J. (2021). Lightweight Wi-Fi Fingerprinting with a Novel RSS Clustering Algorithm. 2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 1–8. doi:10.1109/IPIN51156.2021.9662612.

[10] Kenger, O. N., Kenger, Z. D., Ozceylan, E., & Mrugalska, B. (2023). Clustering of Cities Based on Their Smart Performances: A Comparative Approach of Fuzzy C-Means, K-Means, and K-Medoids. IEEE Access, 11, 134446–134459. doi:10.1109/ACCESS.2023.3333753.

[11] Gholizadeh, N., Saadatfar, H., & Hanafi, N. (2021). K-DBSCAN: An improved DBSCAN algorithm for big data. Journal of Supercomputing, 77(6), 6214–6235. doi:10.1007/s11227-020-03524-3.

[12] Lin, H., Purmehdi, H., Fei, X., Zhao, Y., Isac, A., Louafi, H., & Peng, W. (2023). Two-stage clustering for improve indoor positioning accuracy. Automation in Construction, 154, 104981. doi:10.1016/j.autcon.2023.104981.

[13] Guo, L., Qin, W., Cai, Z., & Su, X. (2024). Hybrid Clustering Algorithm Based on Improved Density Peak Clustering. Applied Sciences (Switzerland), 14(2), 715. doi:10.3390/app14020715.

[14] Kaur, J., & Singh, H. (2015). Performance evaluation of a novel hybrid clustering algorithm using birch and K-means. 2015 Annual IEEE India Conference (INDICON), 1–6. doi:10.1109/INDICON.2015.7443414.

[15] Amiri, S., Clarke, B. S., Clarke, J. L., & Koepke, H. (2019). A General Hybrid Clustering Technique. Journal of Computational and Graphical Statistics, 28(3), 540–551. doi:10.1080/10618600.2018.1546593.

[16] Liu, M., Zhu, Q., Yun, H., Wu, Z., & Chun, H. (2025). A hybrid particle swarm clustering algorithm with novel fitness function based on minimum spanning tree and local Centroids. Journal of King Saud University - Computer and Information Sciences, 37(10), 356. doi:10.1007/s44443-025-00378-8.

[17] Woo, M., Harby, A. A., Zulkernine, F., & Abdulsalam, H. M. (2025). A two-phase hybrid clustering framework exploring transitional activities in HAR. Discover Artificial Intelligence, 5(1), 233. doi:10.1007/s44163-025-00503-6.

[18] Al-Nussairi, A. K. J., Abdulazez, A. A., Hadi, A. A., Malik, S., Patro, S. G. K., Mahanty, C., Alamiery, A. A., Naveed, Q. N., Khan, S., & Zewude, A. (2025). LS-BMO-HDBSCAN as a hybrid memetic bacterial intelligence framework for efficient data clustering. Scientific Reports, 15(1), 40686. doi:10.1038/s41598-025-24380-2.

[19] Sadowski, S., Spachos, P., & Plataniotis, K. N. (2020). Memoryless Techniques and Wireless Technologies for Indoor Localization with the Internet of Things. IEEE Internet of Things Journal, 7(11), 10996–11005. doi:10.1109/JIOT.2020.2992651.

[20] Alhmiedat, T. (2023). Fingerprint-Based Localization Approach for WSN Using Machine Learning Models. Applied Sciences (Switzerland), 13(5), 3037. doi:10.3390/app13053037.

[21] Torres-Sospedra, J., Moreira, A., Mendoza-Silva, G. M., Joao Nicolau, M., Matey-Sanz, M., Silva, I., Huerta, J., & Pendao, C. (2019). Exploiting Different Combinations of Complementary Sensor's data for Fingerprint-based Indoor Positioning in Industrial Environments. 2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 8911758. doi:10.1109/IPIN.2019.8911758.

[22] Moreira, A., Silva, I., Meneses, F., Nicolau, M. J., Pendao, C., & Torres-Sospedra, J. (2017). Multiple simultaneous Wi-Fi measurements in fingerprinting indoor positioning. 2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 8115914. doi:10.1109/IPIN.2017.8115914.

[23] Wang, P. (2025). Archive Information Management System Based on K-Means Clustering with Elbow Method. 2025 3rd International Conference on Integrated Circuits and Communication Systems (ICICACS), 1–6. doi:10.1109/ICICACS65178.2025.10968391.

[24] Shahapure, K. R., & Nicholas, C. (2020). Cluster Quality Analysis Using Silhouette Score. 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), 747–748. doi:10.1109/DSAA49011.2020.00096.

[25] Lai, H., Huang, T., Lu, B. L., Zhang, S., & Xiaog, R. (2025). Silhouette coefficient-based weighting k-means algorithm. Neural Computing and Applications, 37(5), 3061–3075. doi:10.1007/s00521-024-10706-0.

[26] Ashari, I. F., Banjarnahor, R., Farida, D. R., Aisyah, S. P., Dewi, A. P., & Humaya, N. (2022). Application of Data Mining with the K-Means Clustering Method and Davies Bouldin Index for Grouping IMDB Movies. Journal of Applied Informatics and Computing, 6(1), 7–15. doi:10.30871/jaic.v6i1.3485.