Pattern Recognition 174 (2026) 112997

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

RFAG: Random forest clustering based on anchor graph

Jinyu Li [1], Congyu Wang [1], Mingjing Du *

Jiangsu Key Laboratory of Educational Intelligent Technology, School of Artificial Intelligence and Computer Science, Jiangsu Normal University, Xuzhou, 221116, China

ARTICLE INFO

Keywords:
Random forest
Decision trees
Clustering ensemble selection
Clustering

ABSTRACT

Random forest (RF) clustering uses decision trees to extract similarity information for clustering, but existing methods often suffer from high computational costs and the equal treatment of decision trees regardless of their contributions. To address these limitations, we propose a random forest clustering method based on anchor graph (RFAG). The method comprises two phases: training and clustering. During training, representative anchors are generated to build an anchor graph, reducing computational cost. A binary random forest classifier is trained on a pseudo-labeled dataset formed by anchors and generated negatives. For clustering, anchor similarities are extracted via a virtual-path-based measure from the trained trees. A stability-aware ensemble selection strategy selects high-contribution trees to partition the anchors, and clustering results are mapped to data points through the anchor graph. Experiments on 8 real-world datasets show that RFAG significantly reduces computation time while maintaining or improving clustering performance compared to 7 baselines.

1. Introduction

A random forest (RF) [1] is an ensemble learning model composed of numerous decision trees, widely recognized for its robustness and effectiveness in pattern recognition and machine learning. This advantage arises from the randomness introduced during the training of individual trees, which are subsequently aggregated into a powerful final model. While RFs perform well in regression and classification tasks, their potential in clustering applications remains relatively underexplored.

As an unsupervised learning method, RF clustering does not directly utilize category labels information. A common approach to address this limitation is to generate negative samples through marginal random sampling, subsequently transforming the clustering task into a binary classification problem. RF clustering extracts similarity information between samples using its ensemble of decision trees, converts this information into a similarity matrix, and ultimately produces clustering results through a consensus clustering algorithm. However, existing RF clustering methods still face several challenges. First, similarity information in RF clustering is obtained by tracing the path of each data point across all decision trees, which requires traversing the entire tree structure for every data point. This process significantly increases computational time and memory consumption. Additionally, these methods use all decision trees, neglecting the contribution of individual decision trees to the final clustering result.

To address these issues, this paper proposes a novel random forest clustering based on anchor graph, called RFAG. The proposed method is divided into two main phases: the training phase and the clustering phase. Specifically, in the training phase, the hierarchical K-means approach based on balanced K-means is first used to generate representative anchors. These anchors, being significantly fewer in number than the original data points, help reduce the computational complexity of subsequent similarity calculations [2]. Subsequently, a parameter-free neighbor assignment strategy is implemented to construct an anchor graph based on the anchors. Negative samples are generated through marginal random sampling of the anchors and, along with the anchors, are used to train the RF. In the clustering phase, the trained decision trees serve as the base clusterers to extract the similarity information between anchors, which are then converted into similarity matrices and clustered. Next, two clustering ensemble selection strategies are designed based on the dataset's stability and instability. These strategies are used to select specific decision trees, which are then combined to construct a new RF. Finally, the newly formed RF clusters the anchors. The anchor graph, combined with these clustering results, establishes a hierarchical association that links data points to anchors and then to clusters.

The contributions of this paper are summarized as follows.

- We develop an anchor-driven random forest training approach that constructs a pseudo-labeled dataset from representative anchors and generated negative samples, enabling efficient binary classifier training with significantly reduced computational cost.

* Corresponding author.
E-mail addresses: jinyu@jsnu.edu.cn (J. Li), wangcongyu@jsnu.edu.cn (C. Wang), dumj@jsnu.edu.cn (M. Du).
[1] Equal contribution.

https://doi.org/10.1016/j.patcog.2025.112997
Received 14 March 2025; Received in revised form 31 October 2025; Accepted 22 December 2025
Available online 27 December 2025

- We propose a virtual-path based similarity measure for anchors, which captures more nuanced relationships between them than conventional leaf co-occurrence, improving the quality of the similarity matrix and enhancing clustering performance.
- We introduce a stability-aware ensemble selection approach, which designs two strategies based on the stability and instability of the dataset to overcome the negative impact of low-contribution decision trees on clustering quality.

The remaining parts of the paper are organized as follows: Section 2 discusses the related work, Section 3 includes the methodology, Section 4 presents the experimental evaluations, and Section 5 concludes the paper.

2. Related work

2.1. Random forest clustering

RF clustering extracts similarity information between samples and applies it to clustering algorithms. Compared to traditional distance-based clustering methods, this method uses the structural characteristics of decision trees instead of directly calculating distances. Specifically, this method uses the partitioning information from each decision tree in the RF to capture the similarity between samples.

This elegant concept is first proposed by [3]. In this method, when two samples ultimately fall into the same leaf node, it can be inferred that they are similar because they have undergone the same partitioning process along their respective paths. On this basis, [4] extends the definition of RF similarity proposed in [3], arguing that the condition of considering two samples as similar only when they are in the same leaf node is overly strict and does not fully utilize the structure of the decision tree. Specifically, they propose to use the length of the common path between two samples when traversing to the leaf node as the similarity between the two samples. This improvement overcomes the limitation of using leaf node co-occurrence as the only measure of similarity. Recently, [5] further refines this concept by acknowledging that samples deeper in the decision tree may still have potential similarities. This enhancement can capture more subtle relationships in the data, thereby improving the performance of RF methods in a wider range of applications. In addition, [6] proposed an innovative solution that effectively overcomes the inherent limitations of random forest clustering when dealing with non-vector data by using a measure of the difference between paired samples instead of the traditional vector representation. However, despite these advances, the high computational cost of RF clustering remains a significant challenge, particularly when applied to large datasets, limiting its practical performance.

2.2. Anchor-based model

In recent years, anchor-based models have gained great attention in large-scale spectral clustering and multi-view clustering [7,8], mainly because they can accelerate the clustering process. These models reduce the computational complexity by selecting a small number of representative anchors, eliminating the need to calculate pairwise affinities between all data points. Typically, anchor-based models select m anchors from a set of n data points and construct a matrix $\boldsymbol{B} \in \mathbb{R}^{n\times m}$ to capture the similarities between data points and anchors.

Two traditional approaches for generating anchors are the K-means strategy and the random selection strategy [9]. Among them, K-means is the most widely used, and the centers obtained from the algorithm serve as the anchors. This strategy efficiently captures the global structure of the data, using a limited number of anchors to represent key features. Several studies have expanded on the strategy. For example, Zhang and Kwok [10] propose a K-means-based sampling technique for large manifold learning and dimensionality reduction, which reduces computational costs by selecting representative anchors; Cai and Chen [11] introduce a spectral clustering approach that uses landmark-based sparse representation, where landmarks are selected through K-means, and data points are expressed as sparse linear combinations of these landmarks, improving clustering efficiency. Liu et al. [12] combine anchor graph clustering and sparse projection to enhance efficiency and robustness in large-scale, high-dimensional data. Wang et al. [13] use high-density anchor points to automatically select clustering strategies, effectively handling clusters of various shapes and densities. Nie et al. [14] develop an unsupervised graph embedding algorithm for large data, where anchors are created using K-means, followed by the construction of a doubly stochastic and positive semi-definite similarity matrix for clustering. However, the K-means strategy is computationally expensive because selecting anchors through this strategy has a time complexity of $O(ndmt)$, where n represents the number of data points, d the dimensionality, m the number of anchors, and t the number of iterations. As datasets grow in size, the computational cost of K-means escalates rapidly, which hinders its scalability for large datasets. On the other hand, the random selection strategy involves choosing data points at random as anchors. This approach is inexpensive and simple to implement, with no iterative steps required. However, because of its randomness, the quality of the selected anchors can be inconsistent, leading to variable clustering results and lower accuracy. This variability limits the application of the random selection method in tasks that require high-quality clustering.

To address these limitations, Zhu et al. [15] propose a novel approach for anchor generation called Balanced K-means-based Hierarchical K-means (BKHK). This approach employs a balanced binary tree structure to select anchors, reducing computational complexity while ensuring a good representation of the data structure. Compared to traditional K-means, BKHK optimizes both computational time and storage, making it well-suited for large datasets [16]. It strikes an effective balance between computational efficiency and anchor quality, reducing processing time while maintaining high clustering accuracy, making it ideal for handling large data.

2.3. Clustering ensemble selection

Clustering ensemble refers to combining the results of multiple base clusterers to achieve a more effective clustering outcome than that produced by a single clustering method [17] . The fundamental concept is to leverage the diversity and complementarity of different base clusterers to enhance the stability and accuracy of clustering results. Clustering ensemble selection is a subproblem within clustering ensemble methods, specifically how to choose the optimal subset from a pool of candidate base clusterers to attain the best clustering performance [18]. The main challenge lies in selecting base clusterers that exhibit both high quality and sufficient diversity, ensuring clustering accuracy while avoiding overfitting or unstable outcomes.

In recent years, in the research on clustering ensemble selection, only a few studies have considered how to balance the quality and diversity of base clusterers. There are two common diversity and quality selection criteria: the first approach is to select base clusterers through a base clusterer selection strategy based on the Adjusted Rand Index (ARI) [19] as a measure. For example, some studies use median diversity or resampling techniques to select a subset of base clusterers [20,21]. These strategies calculate the similarity between base clusterers to ensure that the final selected base clusterers have sufficient diversity in clustering results while maintaining high clustering quality. The second approach is to use Normalized Mutual Information (NMI) [22] as a measure of quality and diversity. For example, the clustering quality is evaluated by calculating the sum of the NMI of the results of each base clusterer, and then the results of these base clusterers are clustered again to select the base clusterer with the largest sum of NMI in each cluster [23]. This strategy can further refine the criteria for selecting base clusterers, thereby improving the overall performance of clustering results. In addition, other strategies include direct combination, cluster combination [24], extended evidence accumulation clustering [25], and combining

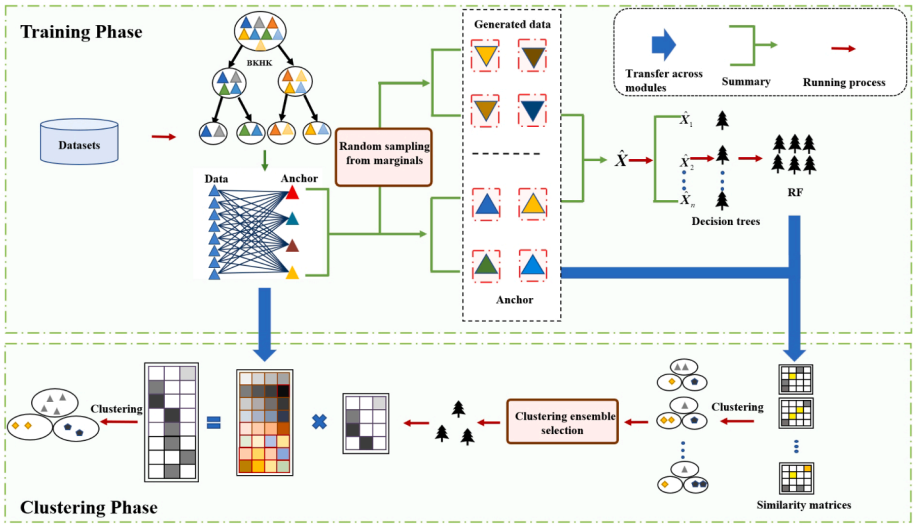


Fig. 1. Framework of RFAG.

transfer learning with clustering ensemble selection to optimize cluster member selection through a multi-objective self-evolution process [26]. These strategies further improve the accuracy and stability of clustering results through different combinations. In addition, [27] uses 5 internal evaluation indicators and NMI to evaluate the quality and diversity of base clusters respectively. Different from the aforementioned methods that rely on the K-means algorithm to generate base clusterers, our method uses decision trees as the base clusterers in a clustering ensemble framework.

3. Methodology

The proposed method is divided into two main phases: the training phase and the clustering phase, as illustrated in Fig. 1.

In short, the RFAG method follows a tightly integrated process, where each module builds naturally on the previous one to ensure efficient clustering on large-scale datasets.

The process begins with Anchor Selection (Section 3.1.1), where representative anchors are generated through BKHK. These anchors preserve the global structural characteristics of the original data while substantially reducing its size, thereby providing the basis for the subsequent modules. The selected anchors are then used in Anchor Graph Construction (Section 3.1.2) to model affiliations between original data points and anchors, serving as a critical input for the subsequent clustering phase. In Random Forest Training (Section 3.1.3), the anchors are used as positive samples, and corresponding negative samples are generated through marginal random sampling. These samples together form a pseudo-labeled dataset that is used to train a binary random forest classifier, whose primary role is to learn discriminative node-level split thresholds and provide the basis for extracting similarity information.

In Similarity Measurement (Section 3.2.1), the trained random forest is used to compute similarities among anchors through a virtual-path strategy, which extends beyond simple leaf co-occurrence to produce an anchor-level similarity matrix that captures both local and global data relationships. This matrix serves as the foundation for Clustering Ensemble Selection (Section 3.2.2), where clustering is performed and only informative decision trees are retained: on stable datasets, trees with high SNMI scores are chosen to maximize quality, while on unstable datasets, diversity is promoted through a balanced k-medoids algorithm. The resulting anchor-to-cluster assignments are then passed to Membership Mapping (Section 3.2.3), where they are propagated back to the full dataset through the anchor graph, generating the final membership matrix between data points and clusters.

3.1. RF training phase

3.1.1. Anchor selection

We employ the BKHK approach for anchor selection, which is highly efficient and particularly well-suited for large-scale clustering problems. BKHK iteratively partitions the dataset into two clusters, each containing an equal number of samples, ensuring that the selected anchors are representative and effectively encompass the data distribution. This approach reduces computational complexity and mitigates redundancy and bias in anchor selection, thereby enhancing both clustering accuracy and efficiency. Specifically, given a dataset $\boldsymbol{X} \in \mathbb{R}^{n\times d}$, where n represents the number of samples and d denotes the feature dimension, BKHK initially applies the biclustering K-means algorithm for preliminary partitioning. The objective is to segment the dataset into two clusters based on its inherent structure. The clustering problem can be formulated as:

$$\min_{\boldsymbol{C},\boldsymbol{G}\in\mathrm{Ind},\mathbf{1}^T\boldsymbol{G}=[K,L]} \left\| \boldsymbol{X} - \boldsymbol{G}\boldsymbol{C}^T \right\|_F^2 \tag{1}$$

where $\boldsymbol{G} \in \mathbb{R}^{n\times 2}$ is the cluster label matrix, each row corresponds to a sample, and each $\boldsymbol{g}_i \in \{0,1\}^2$ is a one-hot vector indicating the cluster assignment of sample $\boldsymbol{x}_i$. $\boldsymbol{C} \in \mathbb{R}^{d\times 2}$ is the cluster center matrix. K and L represent the number of samples in each cluster, where $K + L = n$. Usually, $K = \left\lfloor \frac{n}{2} \right\rfloor$ ensures the balance of the clusters.

3.1.2. Anchor graph construction

After selecting anchors, an undirected weighted graph can be constructed to represent the affiliations between data points and anchors. The nodes in this graph include both data points and anchors, while the weights of the edges represent the similarity between a data point and an anchor [28]. Specifically, the dataset $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ consists of n data points, each of which is connected to m anchors $\boldsymbol{A} \in \mathbb{R}^{m\times d}$, where m represents the number of samples and d denotes the feature dimension. The affiliation probability b_{ij}, an element of the matrix $\boldsymbol{B} \in \mathbb{R}^{n\times m}$, reflects the similarity between sample $\boldsymbol{x}_i$ and anchor $\boldsymbol{a}_j$. To optimize the affiliation probabilities b_{ij}, we solve the following optimization problem to minimize the distance and regularization terms:

$$\min_{\boldsymbol{b}_i{}^T\mathbf{1}=1,0\le b_{ij}\le 1} \sum_{j=1}^{m} \left((\|\boldsymbol{x}_i - \boldsymbol{a}_j\|_2^2) b_{ij} + \gamma b_{ij}^2 \right) \tag{2}$$

where b_{ij}^2 is the regularization term and γ is the regularization parameter. Follow [29], γ can be set as

$$\gamma = \frac{k}{2}\|\boldsymbol{x}_i - \boldsymbol{a}_{k+1}\|_2^2 - \frac{1}{2}\sum_{j=1}^{k}\|\boldsymbol{x}_i - \boldsymbol{a}_j\|_2^2. \tag{3}$$

The solution to problem (2) is

$$b_{ij} = \frac{\|\boldsymbol{x}_i - \boldsymbol{a}_{k+1}\|_2^2 - \|\boldsymbol{x}_i - \boldsymbol{a}_j\|_2^2}{\sum_{j'=1}^{k} \left(\|\boldsymbol{x}_i - \boldsymbol{a}_{k+1}\|_2^2 - \|\boldsymbol{x}_i - \boldsymbol{a}_{j'}\|_2^2\right)}. \tag{4}$$

Finally, the matrix $\boldsymbol{B} \in \mathbb{R}^{n\times m}$ is obtained by calculating the similarity of each data point with the anchor.

3.1.3. Random forest training

Since no true labels are available for the clustering task, we adopt a pseudo-labeling strategy, which is commonly used in random forest clustering [5], to enable the supervised training required by random forests. Following this principle, representative anchors are treated as positive samples, while negative samples are generated through marginal uniform random sampling, ensuring they do not structurally align with any anchor point. Algorithm 1 demonstrates the process of generating negative samples. The combination of positive anchors and generated negatives constitutes the pseudo-labeled training set. The resulting dataset is then used to train a binary random forest classifier. In this way, the model learns underlying data structures and relationships that are essential for subsequent similarity extraction.

Algorithm 1: Negative sample generation.

Input: Data matrix $\boldsymbol{X} \in \mathbb{R}^{n\times d}$, where n is the number of samples and d is the number of features
Output: Negative sample matrix $\boldsymbol{X}' \in \mathbb{R}^{n\times d}$

```
Initialize X' ← 0^{n×d}
// Randomly shuffle each feature column independently
for j = 1 to d do
    Let x_j ← X_{:,j}
    Generate a random permutation I_j of {1, 2, ..., n}
    Set X'_{:,j} ← x_j(I_j)
return X'
```

To train the decision trees in an RF, we employ a resampling technique on the training set $\hat{\boldsymbol{X}}$ to generate multiple data subsets. Training individual decision trees on these subsets of the data introduces diversity and helps mitigate overfitting. The decision tree grows starting from the root node, which represents the entire dataset. This root node is then divided into two subsets based on the selection of specific features and their corresponding binary split thresholds (T). The resulting subsets are assigned to the left and right child nodes. The splitting process is guided by a greedy algorithm that aims to optimize the Gini index. Once the root node is split, the recursive process continues. At each node, the best feature and corresponding threshold are selected to further divide the data, creating two additional subsets. This iterative process refines the decision tree by determining the threshold for each node until the tree is fully developed. After the decision tree training is completed, multiple such trees are combined to form an RF.

3.2. Clustering phase

3.2.1. Similarity measurement

In decision tree-based clustering, after training a binary random forest classifier, extracting similarity information between anchors becomes a critical step. Traditional random forest clustering measures the similarity between two data points based on the length of their common path in the decision tree: the longer the shared path, the higher the similarity, and vice versa. However, this approach assumes that two points with identical paths are equally similar, regardless of potential differences in other features, which can lead to inaccurate similarity judgments, as illustrated in Fig. 2.

Fig. 2 shows three samples: $\boldsymbol{x}_1(0.4, 0.4)$, $\boldsymbol{x}_2(0.6, 0.4)$ and $\boldsymbol{x}_3(0.7, 0.8)$. The first feature is selected with a threshold $T_1 = 0.5$ to split the dataset, and the second feature is then selected with a threshold $T_2 = 0.5$ for further splitting. As shown in Fig. 2, $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, as well as $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$, share the same common path length (zero), resulting in identical similarity measures, which is clearly inadequate.

To address this limitation, we adopt the approach described in [5] for the anchor similarity metric. Specifically, virtual paths are generated by comparing a data point with the thresholds at each node along the path of another data point, enabling more accurate similarity computation between data points. As shown in the right subfigure of Fig. 2, when computing the similarity between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, $\boldsymbol{x}_1$ is compared with the threshold $T_2 = 0.5$ of the second feature, generating a left branch (since the second feature value is less than 0.5) and forming a virtual path. The virtual path of $\boldsymbol{x}_1$ overlaps more with $\boldsymbol{x}_2$ than with $\boldsymbol{x}_3$, indicating that $\boldsymbol{x}_1$ is more similar to $\boldsymbol{x}_2$ than to $\boldsymbol{x}_3$ under this similarity measure. Therefore, this similarity measure can extract the real similarity information between anchors more precisely.

The similarity measure is defined by the following formula:

$$s(\boldsymbol{y}, \boldsymbol{z}) = \frac{|\boldsymbol{Y} \cap \boldsymbol{Z}|}{|\boldsymbol{Y} \cap \boldsymbol{Z}| + |\boldsymbol{Y} - \boldsymbol{Z}| + |\boldsymbol{Z} - \boldsymbol{Y}|} \tag{5}$$

where $|\boldsymbol{Y} \cap \boldsymbol{Z}|$ denotes the number of times samples $\boldsymbol{y}$ and $\boldsymbol{z}$ share the same partition on all nodes along their respective paths to each other. $|\boldsymbol{Y} - \boldsymbol{Z}|$ represents the number of times sample $\boldsymbol{y}$ does not share the same partition as sample $\boldsymbol{z}$ on all nodes along the path to $\boldsymbol{z}$, while $|\boldsymbol{Z} - \boldsymbol{Y}|$ indicates the number of times sample $\boldsymbol{z}$ does not share the same partition as sample $\boldsymbol{y}$ on all nodes along the path to $\boldsymbol{y}$.

3.2.2. Clustering ensemble selection

To mitigate the negative impact of low-contribution decision trees on clustering quality, we design a clustering ensemble selection approach tailored for decision tree selection. Notably, unsupervised clustering tasks differ from supervised tasks, as they lack external information (e.g., labels) to evaluate clustering quality. In order to evaluate the performance of clustering algorithms, predefined class labels [30] or similar external information are usually used instead of the true basic structure. By comparing the clustering results with the predefined class labels, the accuracy and consistency of clustering can be quantified, thereby evaluating the quality of the clustering solution. However, in clustering ensemble selection, this external evaluation approach is not applicable. Because ensemble selection aims to use the information of the clustering results themselves to determine the best set of clusters, rather than relying on external criteria. Therefore, in this case, internal evaluation approaches are usually used in clustering literature to evaluate the performance of clustering ensemble selection. Internal evaluation approaches do not rely on external label information, but analyze the characteristics and clustering structure of the data itself to evaluate the quality of clustering. ARI and NMI are commonly used clustering quality evaluation criteria. This paper adopts NMI to evaluate clustering quality. Specifically, given a set of base clusterers $\boldsymbol{\Pi} = \{\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_v\}$, a good clustering ensemble should maximize the following criteria [31]:

$$SNMI(\boldsymbol{\pi}, \boldsymbol{\Pi}) = \sum_{i=1}^{v} NMI(\boldsymbol{\pi}, \boldsymbol{\pi}_i). \tag{6}$$

We cluster the similarity matrix extracted from a single decision tree and regard each decision tree as a base clusterer. From the perspective of a subset of decision trees is selected. Spectral clustering, specifically the regularized version of Ng-Jordan-Weiss [32], is employed for this purpose. After obtaining the clustering results from all decision trees, we derive the unique clustering result (the consensus solution) $\boldsymbol{\pi}^*$. Specifically, the core of the consensus function is to construct an $m \times m$ co-occurrence matrix, which is generated based on the co-occurrence concept in clustering ensemble. After obtaining V clustering results, we construct V $m \times m$ matrices. Each matrix element $\boldsymbol{E}_{ij}$ indicates whether anchors i and j are assigned to the same cluster in this clustering result (1 or 0). These V matrices $\boldsymbol{E}$ are summed and normalized to obtain the final co-occurrence matrix, where each element $\boldsymbol{E}_{ij}$ represents the similarity between anchors i and j across the V clustering results.

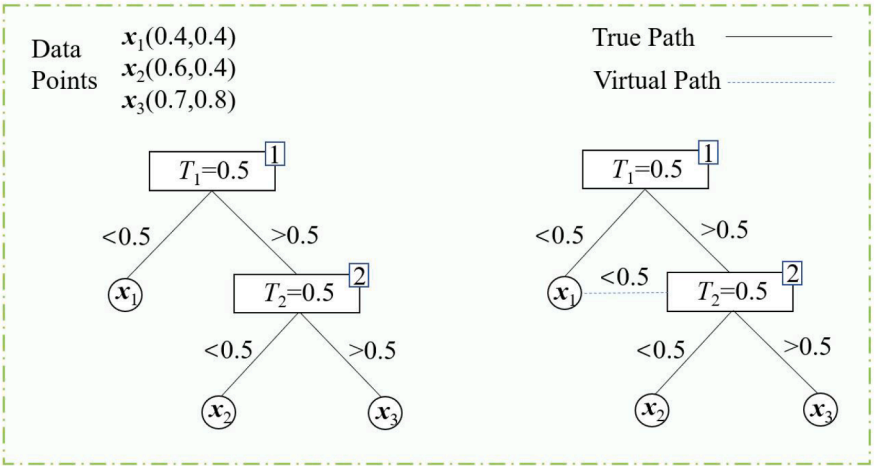


Fig. 2. Partial path.

The co-occurrence matrix no longer uses the original features to describe the anchor but instead uses a new "relational feature" to describe it. The higher the similarity between two anchors, the closer their co-occurrence value (connection strength) will be to 1. This matrix captures the intrinsic relationships between anchors from the clustering perspective. The co-occurrence matrix itself is then treated as a similarity matrix, and clustering algorithms are applied to it to obtain the final consensus partition $\boldsymbol{\pi}^*$. The algorithm for generating the co-occurrence matrix is shown in Algorithm 2. Since the consensus function needs to handle V base clusterers, for each base clusterer, all possible point pairs need to be checked. The number of point pairs is $\binom{m}{2} \approx \frac{m(m-1)}{2}$, which is asymptotically $O(m^2)$. Therefore, the total complexity of constructing the co-occurrence matrix is: $O(V\frac{m(m-1)}{2}) \approx O(Vm^2)$. The stability of the dataset is assessed by the average NMI between $\boldsymbol{\pi}^*$ and the clustering results of each decision tree, as defined by the following formula [33].

$$NMI(\boldsymbol{\pi}^*, \boldsymbol{\Pi}) = \frac{1}{v}\sum_{i=1}^{v} NMI(\boldsymbol{\pi}^*, \boldsymbol{\pi}_i). \tag{7}$$

Algorithm 2: Consensus clustering via co-association matrix.

Input: Base clusterers $\{\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \dots, \boldsymbol{\pi}_v\}$, number of clusters c
Output: Consensus clustering $\boldsymbol{\pi}^*$
// Initialize co-association matrix
Initialize $\boldsymbol{E}^* \leftarrow \boldsymbol{0}^{m\times m}$
// Accumulate co-occurrences from base clusterers
foreach $\boldsymbol{\pi}_t \in \{\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_v\}$ **do**
  foreach *pair of anchors* (a_i, a_j) **do**
    if $\boldsymbol{\pi}_t$ *assigns* a_i *and* a_j *to the same cluster* **then**
      $\boldsymbol{E}^*_{ij} \leftarrow \boldsymbol{E}^*_{ij} + 1$
// Normalize co-association matrix
$\boldsymbol{E}^* \leftarrow \boldsymbol{E}^*/v$
// Apply spectral clustering
$\boldsymbol{\pi}^* \leftarrow$ SpectralClustering$(\boldsymbol{E}^*, c)$
return $\boldsymbol{\pi}^*$

Based on the stability assessment, if the value is greater than 0.5, the dataset is categorized as stable; otherwise, it is marked as unstable. In the stable case, base clusterers with higher NMI relative to $\boldsymbol{\pi}^*$ are selected for the final clustering ensemble. Conversely, in the unstable case, a subset of the most dissimilar base clusterers is chosen. To implement these principles, we propose two strategies for selecting base clusterers. For stable datasets, SNMI is used to evaluate the quality of base clusterers, which are then sorted in descending order, and a threshold is applied to select the appropriate decision trees. Conversely, for unstable datasets, inspired by BKHK, we introduce a balanced k-medoids algorithm to select decision trees, ensuring the diversity of the base clusterers. The algorithm partitions the dataset into multiple clusters and selects the medoid (center) of each cluster as the representative decision tree. By representing the key characteristics of their respective clusters, the selected medoids collectively cover different regions of the dataset, which enhances representativeness, reduces redundancy, and promotes diversity in the selection [34].

3.2.3. Membership mapping

In this subsection, we detail how to establish the relationship between data points and clusters. Specifically, anchors are used as intermediaries to connect data points and clusters.

First, we construct a membership matrix $\boldsymbol{U} \in \mathbb{R}^{m\times c}$ for the anchors through RF clustering, where u_{ij} represents the degree to which the ith anchor belongs to the jth cluster. In the context of hard clustering, we set $u_{ij} = 1$ to indicate that anchor $\boldsymbol{a}_i$ belongs to cluster j, and $u_{ij} = 0$ to indicate that it does not. This means that each anchor is assigned to exactly one cluster, simplifying the computation while providing clear clustering results.

Next, we define a connection matrix $\boldsymbol{B} \in \mathbb{R}^{n\times m}$, which is computed in a previous section. Here, b_{ij} represents the connection weight between data point $\boldsymbol{x}_i$ and anchor $\boldsymbol{a}_j$. This connection weight reflects the similarity or association between data point $\boldsymbol{x}_i$ and anchor $\boldsymbol{a}_j$ in the feature space.

Finally, to represent the affiliation between the data points and clusters, we define an indicator matrix $\boldsymbol{F} = \{f_{ij}\} \in \mathbb{R}^{n\times c}$, where f_{ij} indicates the degree to which data point $\boldsymbol{x}_i$ belongs to cluster j. The membership of data point $\boldsymbol{x}_i$ to cluster j is determined by the weighted sum of the memberships of all anchors associated with the cluster. Specifically, we calculate f_{ij} as:

$$f_{ij} = \sum_{k=1}^{m} b_{ik} \cdot u_{kj} \tag{8}$$

where b_{ik} is the connection weight between data point $\boldsymbol{x}_i$ and anchor $\boldsymbol{a}_k$, and u_{kj} is the membership of anchor $\boldsymbol{a}_k$ to cluster j. In matrix form, the membership matrix $\boldsymbol{F}$ is computed as $\boldsymbol{F} = \boldsymbol{BU}$.

Once the membership matrix $\boldsymbol{F}$ is obtained, any classical clustering algorithm can be applied to further process the clustering problem. In this method, we use the K-means clustering algorithm for the final cluster assignment.

3.3. Time complexity analysis

The computational complexity of the proposed RFAG clustering method can be analyzed by examining the major procedures involved in anchor selection, anchor graph construction, random forest training, similarity measurement, clustering ensemble selection, and final membership mapping.

In the anchor selection step, BKHK obtains m anchors by recursively partitioning the dataset $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ into balanced clusters with a time complexity of $O(nd\log m)$. The anchor graph construction computes the similarity between each of the n data points and the m anchors, and thus the cost of this stage is $O(ndm)$. In the random forest training stage, a pseudo-labeled dataset is constructed consisting of m positive anchors and an equal number of negative samples, resulting in $O(m)$ training instances. Training a forest with T decision trees on $O(m)$ samples with d features requires $O(Tmd\log m)$ time. In the clustering ensemble selection stage, each decision tree is treated as a base clusterer, and spectral clustering is performed on the similarity matrices derived from the selected trees. The eigen-decomposition of an $m\times m$ similarity matrix has a cost of $O(m^3)$ in the worst case. Finally, in the membership mapping stage, the anchor-to-cluster membership matrix $\boldsymbol{U} \in \mathbb{R}^{m\times c}$ is propagated to the entire dataset through the connection matrix $\boldsymbol{B} \in \mathbb{R}^{n\times m}$. This requires computing $\boldsymbol{F} = \boldsymbol{BU}$, which involves $O(nmc)$ operations. Since $c \ll m$, the cost is dominated by $O(nm)$.

By combining the above analysis, the overall complexity of the RFAG method is dominated by the most expensive steps, resulting in $O(ndm + Tm^3)$.

3.4. Space complexity analysis

The space complexity of the RFAG clustering method can also be analyzed by examining the key steps such as anchor selection, anchor graph construction, random forest training, similarity measurement, and final membership mapping.

First, in the anchor selection phase, anchors are chosen using the BKHK algorithm. Although the number of anchors is much smaller than the number of data points, the features of each anchor still need to be stored. Therefore, the space complexity of this phase is $O(nd)$, where n is the total number of samples and d is the feature dimension of the data.

In the anchor graph construction phase, the algorithm needs to store the similarity matrix between each data point and each anchor. Let the number of anchors be m, then the space complexity of the similarity matrix is $O(nm)$. This matrix stores the similarity information between each data point and anchor, and the space required is mainly determined by the number of data points n and anchors m.

In the random forest training phase, the algorithm needs to store the structure of the decision trees. The depth of each tree is h, so the space complexity for storing each tree is $O(h)$. For T decision trees, the total space complexity is $O(Th)$. Additionally, the training process uses a pseudo-labeled dataset containing m anchors, and the space consumption for storing this dataset is minimal and can be ignored.

In the clustering phase, the similarity measurement requires storing an $m\times m$ similarity matrix to compute the similarities between anchors. Therefore, the space complexity for this step is $O(m^2)$.

Finally, in the membership mapping phase, the algorithm needs to store an $n\times m$ connection matrix $\boldsymbol{B}$ and an $m\times c$ membership matrix $\boldsymbol{U}$, where c is the number of clusters. The space complexity is $O(nm + mc)$.

In summary, the space complexity of the RFAG algorithm is primarily dominated by the need to store the anchor graph and the matrices in the final membership mapping phase, resulting in an overall space complexity of $O(nm)$.

4. Experimental evaluation

4.1. Preparations

We conduct experiments to evaluate the performance of RFAG on 8 different datasets. All experiments are carried out on a Lenovo Y9000P (equipped with an Intel i9-12900H processor, 14 cores, 20 threads, operating at 2.50GHz, with 32GB of RAM), running Windows 11. The RFAG method and the 7 comparison methods are implemented using MATLAB R2022b and Python.

4.1.1. Datasets

The performance of the proposed method is evaluated on 8 real-world datasets. The real-world datasets include Satimage, FMNIST, KMNIST, MNIST, Optdigits, Penbased, USPS, and YTF. These benchmark datasets can be found in some published benchmarks, such as the UCI Machine Learning Data Repository and clustering benchmark datasets. Table 1 summarizes the detailed information of these datasets. It is worth noting that FMNIST, KMNIST, MNIST, and USPS are reduced to 5 dimensions using UMAP as in [35], while YTF is processed directly following [36].

4.1.2. Comparison methods

To evaluate the performance of our method, we compare it with 7 representative baselines, including 2 anchor-based clustering methods and 5 RF-based clustering methods. The methods are listed below:

- APC [13]: This method selects clustering strategies based on high-density anchor points, combining the strengths of DPC and DBSCAN to handle clusters of varying shapes and densities.
- RSFCAG [12]: This method combines anchor graph clustering with sparse projection, where an anchor graph is constructed and a possibilistic neighbor-based similarity matrix is used to guide the clustering process.
- *m_Shi* [3]: This method is the first to use RF to extract similarity information for clustering. The similarity information is measured by the frequency with which two samples fall into the same leaf node.
- *m_Zhu2* [4]: This method uses the length of the common path between two samples as a similarity measure.
- *m_Zhu3* [4]: This method, also referred to as ClustRF-Strct-Adp, extends *m_Zhu2* by weighting each node in the common path, with the node's weight being computed as the inverse of the number of points that reach the node.
- *m_Ting* [37]: This method defines the distance between two samples as the ratio of the points in the training set that reach the lowest common ancestor (LCA).
- RatioRF [5]: This method applies the Tversky Axiomatic Model and defines the similarity between two samples as the number of times they respond similarly at threshold nodes during partitioning. Even if two samples do not share a common path, there may still be a possibility of similarity.

4.1.3. Parameter setup

In our proposed method, there are four key parameters: the number of nearest points (k), the number of anchors (m), the total number of decision trees (t), and the number of selected decision trees (t'). In our experiments, we set $k = 20$ to ensure local structure representation. To accurately capture the internal structure of the dataset, the number of anchors (m) is determined based on the dataset size and set to 2^i, ensuring it approximates 5% of the total number of data points. The total number of decision trees (t) is set to 100, consistent with the recommendation in [5], while the number of selected decision trees (t') is set to 2^5. All methods are executed 5 times, and the average results are recorded.

4.1.4. Implementation details

In our approach, the RF is trained 5 times, with each training representing an independent starting point for calculating the similarities between data points. After each training, we construct a relationship network between the data points by computing their similarities and dissimilarities. Once the similarity matrix is obtained, clustering is performed using four different algorithms: First, spectral clustering, which

Table 1
Datasets.

Name	#Samples	#Features	#Clusters
Satimage	6435	36	6
FMNIST	10,000	5	10
KMNIST	10,000	5	10
MNIST	10,000	5	10
Optdigits	5620	64	10
Penbased	10,992	16	10
USPS	4649	5	10
YTF	10,000	10	41

is a typical algorithm widely used in recent RF clustering research. This algorithm employs the Ng-Jordan-Weiss normalized version and repeats the inner K-means optimization 20 times during each clustering process to improve clustering accuracy. Next, the classic K-means clustering algorithm, which is a well-known and efficient distance-based clustering algorithm, is applied. Third, two hierarchical clustering algorithms, Complete-Link and Ward-Link, are used. These algorithms capture relationships between data points through different hierarchical structures, providing more refined clustering results. The implementations of these algorithms are based on the statistics and machine learning toolbox in Matlab.

4.1.5. Evaluation metrics

In our experiments, we use three metrics to evaluate the performance of RFAG and comparison methods. These metrics include NMI [38], ARI and clustering accuracy (ACC) [39]. ARI and NMI are commonly used to assess the performance of clustering methods, both evaluating the degree of agreement between true class labels and clustering results. Generally, higher values indicate better clustering performance. NMI ranges from 0 to 1, with 1 indicating perfect agreement, while ARI ranges from -1 to 1, where 1 indicates perfect agreement and negative values suggest poor clustering. ACC measures the proportion of correctly assigned samples, with higher values indicating better accuracy.

4.2. Comparison experiments

We test the clustering performance of the baseline methods and RFAG on the dataset in Table 1, analyzing the clustering performance of each method by averaging the results of the 7 clustering algorithms. The baseline methods are grouped into two categories: anchor-based methods (APC and RSFCAG) and random forest-based methods (m_Shi, m_Zhu2, m_Zhu3, m_Ting, and RatioRF). Table 2 shows the ACC, ARI, and NMI values of the baseline methods and RFAG on 8 real-world datasets. The highest scores for each dataset under the corresponding metrics are highlighted in bold.

The proposed RFAG method outperforms or is comparable to the baseline methods considered in terms of ACC, ARI and NMI in most cases. However, we observe that RFAG does not achieve the best clustering results on FMNIST dataset. This is because RFAG is based on the sampling idea. Although this approach is highly effective in accelerating the method, it inevitably loses some structural information of the data during the sampling process. However, compared to other methods, our proposed method strikes a good balance between efficiency and clustering accuracy.

For anchor-based clustering methods, both APC and RJSFCAG achieve competitive results. APC performs well on certain datasets, such as USPS, by selecting anchors away from dense regions to capture diverse structures. RJSFCAG enhances clustering performance by leveraging anchor graphs and a fast ternary-tree clustering approach (3KHK). Although both methods achieve strong performance in our experiments, RFAG still outperforms them on most datasets due to its anchor-based similarity extraction and stability-aware clustering ensemble selection strategy.

Random forest-based clustering methods exhibit notable differences in performance, primarily resulting from the use of different similarity measures. From Table 2, it is clear that the m_Shi method performs relatively poorly in terms of clustering effectiveness. This method primarily focuses on the frequency with which two samples fall into the same leaf node when extracting similarity information using decision trees. The limitation of this method lies in its failure to fully leverage the structural information of the decision tree, resulting in similarity information that does not accurately reflect the true similarity between samples. Relying solely on leaf node information often fails to capture subtle differences between samples, leading to unsatisfactory clustering results.

The m_Zhu2 and m_Zhu3 methods, as improvements on m_Shi, enhance the similarity calculation by introducing the common path of two samples as a similarity measure. This approach not only considers the similarity of leaf nodes but also extends to the entire decision tree's path information, fully utilizing the structural characteristics of the decision tree. This multi-dimensional similarity evaluation approach makes the clustering results more accurate, leading to significant improvements over mshi on several datasets.

Similarly, the m_Ting method shares a similar idea with m_Zhu2, using the common ancestor of two samples as a similarity measure. This method fully utilizes the structural information of the decision tree, thereby improving the accuracy of clustering. In the experimental results, it can be observed that mting shows considerable improvement over mshi on certain datasets. On some datasets, the ACC, NMI and ARI values of m_Ting are close to those of m_Zhu2 and m_Zhu3.

The RatioRF is a further extension of the similarity measure proposed by m_Zhu2 and m_Zhu3. It considers that samples that are more deeply separated might also be similar. Compared to m_Zhu2 and m_Zhu3, it refines the similarity relationship between two samples. From the clustering results, it can be seen that RatioRF shows some improvement over m_Zhu2 and m_Zhu3 on most datasets.

The RFAG is consistent with RatioRF in similarity measurement, but there are obvious differences in the training and clustering stages of random forest. Specifically, RFAG uses anchors that are much smaller than the original dataset to train the random forest, which helps to reduce the impact of similar feature values. RatioRF uses the entire dataset to train the random forest. At the same time, RFAG introduces a clustering ensemble selection strategy in the clustering stage, selecting only decision trees with higher gains to participate in the ensemble, thereby improving the performance of the model.

4.3. Stability comparison experiment

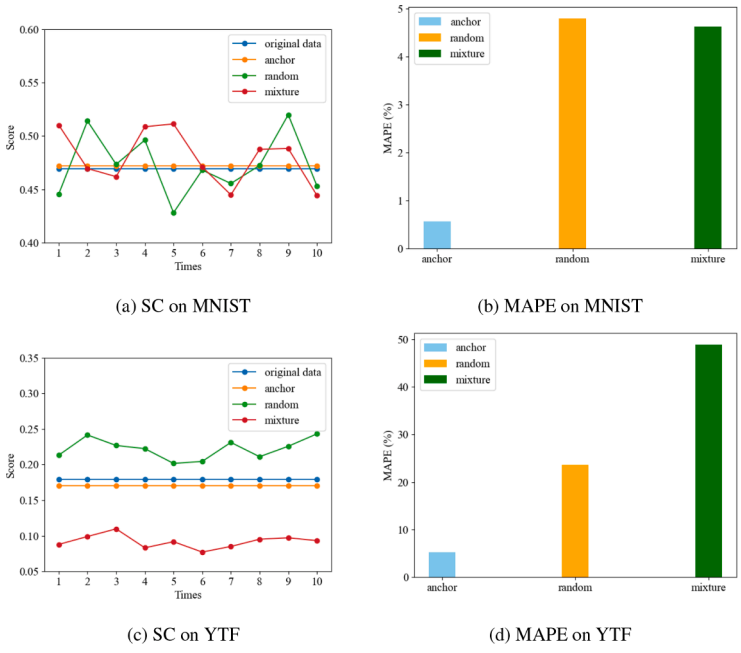
To validate the advantages of the anchor-based approach over random and mixed sampling, we conduct experiments comparing their stability and effectiveness. Specifically, we use the silhouette coefficient(SC) [40] with true class labels as the evaluation metric to assess whether the selected anchors maintain the cluster structure of the original data. The silhouette coefficient, ranging from -1 to 1, measures cluster cohesion and separation, with higher values indicating better cluster structure. To quantify the closeness of the silhouette coefficients between sampled points and the original data, we employ the mean absolute percentage error (MAPE), where lower MAPE values indicate higher structural consistency.

We apply the three sampling approaches (anchor, random, and mixed) on the YTF and MNIST datasets, running each approach 10 times. The silhouette coefficients and their corresponding MAPE values for each method are calculated and compared with the original data. Fig. 3(a) and (b) show the silhouette coefficients and MAPE for the YTF dataset, while Fig. 3(c) and (d) present the same for the MNIST dataset.

As illustrated in Fig. 3(a) and (c), the anchor-based approach demonstrates superior sampling stability, yielding a structure that more closely approximates the one derived from the original data. Fig. 3(b) and (d) further confirm that the anchor-based approach produces a structure most similar to the original data, as indicated by the MAPE values of

Table 2
Performance on real-world datasets.

	Dataset	APC	RJSFCAG	*m_Shi*	*m_Zhu2*	*m_Zhu3*	*m_Ting*	RatioRF	RFAG
ACC	FMNIST	0.551	0.51	0.263	**0.565**	0.463	0.564	0.525	0.564
	KMNIST	0.403	0.392	0.249	0.481	0.439	0.445	0.488	**0.498**
	MNIST	0.603	0.64	0.356	0.770	0.635	0.737	0.716	**0.802**
	Penbased	0.585	0.633	0.328	0.566	0.566	0.614	0.661	**0.683**
	Satimage	0.391	0.502	0.567	0.577	0.627	0.552	0.616	**0.684**
	Optdigits	0.45	0.685	0.269	0.486	0.399	0.489	0.599	**0.766**
	USPS	0.505	0.708	0.329	0.753	0.687	0.718	0.766	**0.838**
	YTF	0.472	0.539	0.325	0.590	0.490	0.512	0.634	**0.670**
ARI	FMNIST	0.265	**0.442**	0.049	0.407	0.234	0.407	0.336	0.375
	KMNIST	0.232	**0.284**	0.029	0.269	0.158	0.251	0.260	**0.284**
	MNIST	0.612	0.69	0.089	0.646	0.415	0.618	0.586	**0.703**
	Penbased	0.48	0.516	0.057	0.420	0.208	0.381	0.475	**0.525**
	Satimage	0.312	0.418	0.292	0.321	0.386	0.287	0.395	**0.443**
	Optdigits	0.527	0.621	0.041	0.280	0.140	0.269	0.411	**0.636**
	USPS	0.657	0.635	0.075	0.604	0.459	0.556	0.614	**0.722**
	YTF	0.39	0.505	0.050	0.390	0.189	0.321	0.485	**0.526**
NMI	FMNIST	0.538	**0.62**	0.236	0.581	0.465	0.570	0.514	0.559
	KMNIST	0.444	0.45	0.186	0.437	0.390	0.402	0.426	**0.454**
	MNIST	0.778	0.77	0.323	0.740	0.616	0.720	0.698	**0.779**
	Penbased	0.607	0.679	0.311	0.632	0.567	0.599	0.639	**0.680**
	Satimage	0.434	0.503	0.384	0.436	0.460	0.409	0.486	**0.542**
	Optdigits	0.685	0.738	0.209	0.417	0.325	0.392	0.540	**0.741**
	USPS	0.794	0.757	0.281	0.733	0.650	0.712	0.730	**0.818**
	YTF	0.736	0.736	0.364	0.676	0.554	0.615	0.720	**0.748**



(a) SC on MNIST (b) MAPE on MNIST

(c) SC on YTF (d) MAPE on YTF

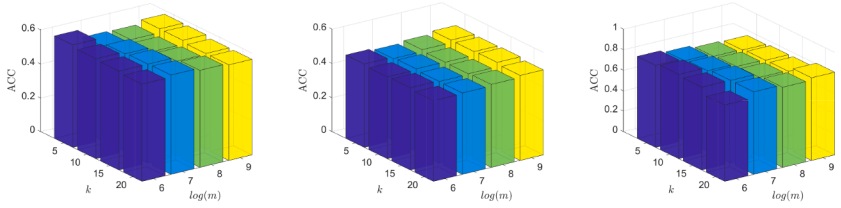
Fig. 3. SCs and MAPEs for different datasets.

the silhouette coefficients. This validates that the anchors capture the key information of the data. Consequently, when constructing decision trees, the model can quickly learn the primary patterns.

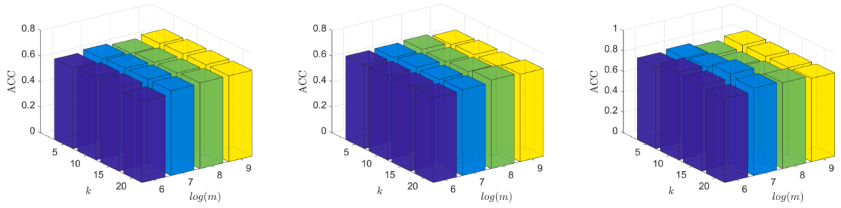
4.4. Ablation experiment

This subsection investigates the specific impact of introducing the anchor-based approach and the clustering ensemble selection mechanism on the performance of the proposed RF clustering method through ablation experiments. In the experiment, we first remove the anchor-based approach and the clustering ensemble selection mechanism from the RFAG method, resulting in two modified versions: RFAG-A (without the anchor-based approach) and RFAG-CES (without the clustering ensemble selection mechanism). Then, both the anchor-based approach and the clustering ensemble selection mechanism are removed, leading to the RFAG-ALL method. To ensure fairness, the parameter configurations for these modified methods are kept consistent with those of RFAG across all datasets. The experimental results, as shown in Table 3, present the average scores for each method on all datasets.

First, after removing the anchors, the average ACC, ARI, and NMI decreased to 0.641 (7.5% decrease), 0.458 (15% decrease), and 0.604 (10.1% decrease), respectively. This may be because the number of anchors is relatively small compared to the number of data points in the dataset, which helps to mitigate the impact of similar feature values. Second, after removing the clustering ensemble selection strategy, the average ACC, ARI, and NMI decreased to 0.677 (1.8% decrease), 0.515 (2.3% decrease), and 0.657 (1.2% decrease), respectively. This may be because some decision trees that have limited contribution to clustering or even have a negative impact are removed during the clustering ensemble selection process. Finally, the clustering performance



(a) ACC on FMNIST (b) ACC on KMNIST (c) ACC on MNIST



(d) ACC on YTF (e) ACC on Penbased (f) ACC on USPS

Fig. 4. ACC with different values of parameters k and m.

Table 3
Results of ablation experiments.

Method	ACC	ARI	NMI
RFAG	**0.689**	**0.527**	**0.665**
RFAG-A	0.641	0.458	0.604
RFAG-CES	0.677	0.515	0.657
RFAG-ALL	0.618	0.444	0.591

Table 4
Comparison in terms of running time (s).

Dataset	*m_Shi*	*m_Zhu2*	*m_Zhu3*	*m_Ting*	RatioRF	RFAG
FMNIST	807.69	13846.25	88678.49	71168.44	10253.44	**71.67**
KMNIST	810.59	14685.33	93984.64	78864.97	10863.20	**77.47**
MNIST	799.49	13095.39	83808.35	69548.23	9380.60	**68.64**
Penbased	401.31	31053.85	212460.20	166669.30	19510.21	**63.14**
Satimage	132.85	6025.78	41036.75	31247.32	3748.56	**26.53**
Optdigits	97.76	5417.51	37425.84	29751.61	3419.15	**24.01**
USPS	268.11	1854.03	11334.53	9407.97	1368.44	**24.62**
YTF	2440.19	22276.10	162615.5	131428.90	22021.75	**96.18**

of removing the cluster anchor mechanism and the clustering ensemble selection strategy is significantly reduced compared with RFAG, which indicates that the anchor mechanism and the clustering ensemble selection mechanism play a key role in improving the clustering performance of our proposed RFAG method, thereby improving the accuracy of the clustering results.

4.5. Running time comparison

This subsection provides a comparative analysis of runtime to evaluate the efficiency of the RFAG method. Since the primary focus of this work is to improve the efficiency of random forest clustering, the runtime comparison is conducted against existing RF-based clustering methods. The detailed results are reported in Table 4. The recorded runtime includes both the training time for constructing the random forest and the total time spent on extracting similarity information and executing the clustering process. The runtime values highlighted in bold represent the fastest processing speed for each dataset. The results in Table 4 show that the proposed RFAG method significantly optimizes runtime efficiency.

The performance improvement is primarily attributed to the significant reduction in the number of anchors compared to the original data points. This reduction not only effectively decreases memory usage but also substantially lowers the computational complexity during the similarity calculation and clustering stages, ultimately enhancing the overall speed of the algorithm. The analysis clearly demonstrates that the RFAG method exhibits outstanding efficiency in processing large datasets compared to existing RF clustering methods.

4.6. Parameter sensitivity

In this section, we delve into the impact of parameters on the clustering performance of the proposed RFAG method. The clustering performance of the RFAG method is influenced by five main parameters: the number of neighbors (k), the number of anchors (m), the total number of decision trees (t), the number of selected decision trees (t'), and stability threshold (θ). The choice of each parameter affects the algorithm's performance to some extent, so we conduct an experimental analysis of the sensitivity to these parameters.

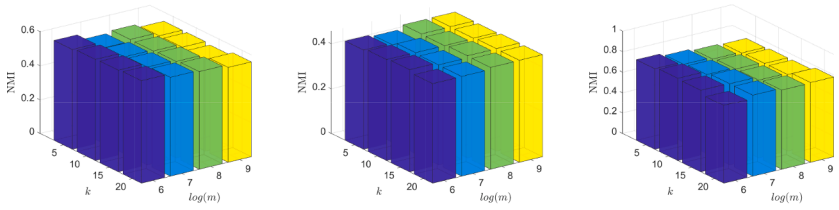
The parameter k determines the number of nearest neighbors considered for each data point when constructing the anchor graph. We test k values in the range $[5, 10, 15, 20]$. The parameter m controls the number of anchors used to represent the data's internal structure, with values set in the range $[2^6, 2^7, 2^8, 2^9]$. Since the BKHK algorithm generates anchors using a binary tree structure, the number of anchors is a power of 2. The parameter t governs the size of the random forest, with values tested in the range $[50, 75, 100, 125, 150]$. Finally, the parameter t' determines the number of decision trees selected for clustering ensemble selection, with values set in the range $[2^3, 2^4, 2^5, 2^6]$.

To comprehensively analyze the impact of these parameters on clustering performance, we evaluate the clustering results using ACC, ARI and NMI as primary metrics under different parameter configurations.

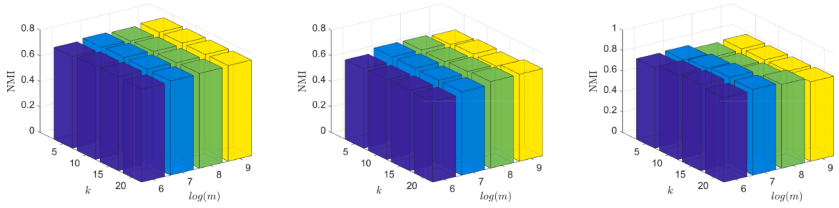
The parameters k and m, which are associated with the anchor-based approach, are analyzed together. Figs. 4 and 5 show the ACC and NMI values for different configurations of k and m across six datasets: FMNIST, KMNIST, MNIST, YTF, Penbased and USPS. The results indicate that clustering performance remains stable with changes in these parameters on most datasets, demonstrating that the method exhibits strong robustness with respect to k and m.

Next, we evaluate the effect of the parameter t. Fig. 6 presents the ARI and NMI values for different values of t on the same six datasets. The results show that RFAG's clustering performance is also robust to changes in t, with minimal fluctuations across datasets.

Moreover, we examine the impact of the parameter t'. Fig. 7 illustrates the ACC and NMI values for different values of t'. While a slight upward trend in performance is observed as t' increases, the changes

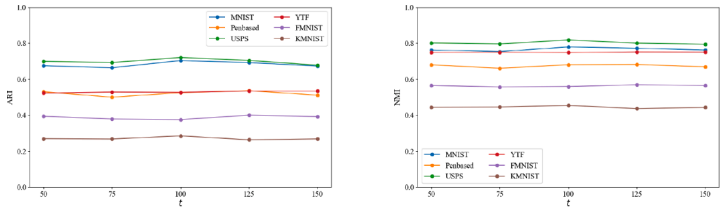


(a) NMI on FMNIST (b) NMI on KMNIST (c) NMI on MNIST



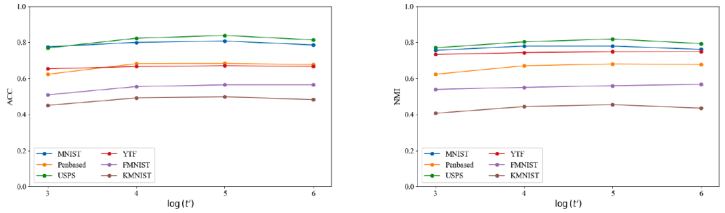
(d) NMI on YTF (e) NMI on Penbased (f) NMI on USPS

Fig. 5. NMI with different values of parameters k and m.



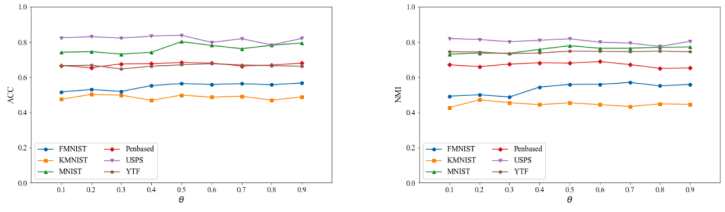
(a) ARI on different datasets (b) NMI on different datasets

Fig. 6. ARI and NMI with different values of parameter t.



(a) ACC on different datasets (b) NMI on different datasets

Fig. 7. ACC and NMI with different values of parameter t'.



(a) ACC on different datasets (b) NMI on different datasets

Fig. 8. ACC and NMI with different values of parameter θ.

are minimal, indicating that RFAG is relatively insensitive to variations in t'.

Finally, we examine the impact of the parameter θ. Fig. 8 shows the ACC and NMI values for different values of θ. It can be seen that our method exhibits strong robustness with a threshold $\theta > 0.5$ on most datasets, with minimal performance fluctuations. This analysis not only verifies the stability of the method but also supports the credibility of the 0.5 threshold proposed in [33].

In summary, although the parameters k, m, t, t', and θ do have some influence on the clustering results, their impact on performance is limited. This indicates that RFAG exhibits high stability and demonstrates strong generalization ability across different parameter configurations.

5. Conclusion

In this paper, we propose a novel random forest clustering, RFAG, which integrates anchor-based sampling and clustering ensemble selection to address the efficiency and accuracy challenges in traditional clustering frameworks. Experimental results across different datasets show that RFAG achieves statistically significant improvements in clustering accuracy while reducing running time compared to existing random

forest clustering methods. The advantages of RFAG stem from three main innovations: an anchor-driven training mechanism that reduces computational cost while retaining essential structural information; a virtual-path based similarity measure for anchors that captures more fine-grained relationships between anchors to enhance clustering accuracy; and a stability-aware anchor graph ensemble selection strategy that improves robustness by mitigating the influence of low-contribution decision trees.

Despite these advances, RFAG inevitably loses part of the structural information during anchor-based sampling, which may limit performance on datasets with complex topological structures. Future research will focus on improving the adaptability of RFAG to diverse data distributions and developing more effective strategies for anchor generation and ensemble optimization.

CRediT authorship contribution statement

Jinyu Li: Writing – original draft, Visualization, Software, Conceptualization; **Congyu Wang:** Writing – original draft, Resources, Conceptualization; **Mingjing Du:** Writing – review & editing, Supervision, Funding acquisition.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work is supported by the Qinglan Project of Jiangsu Province of China, National Natural Science Foundation of China (No. 62006104), Postgraduate Research & Practice Innovation Program of Jiangsu Normal University (No. 2024XKT2583).

References

[1] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.
[2] H. Jia, Q. Ren, L. Huang, Q. Mao, L. Wang, H. Song, Large-scale non-negative subspace clustering based on nyström approximation, Inf. Sci. 638 (2023) 118981.
[3] T. Shi, S. Horvath, Unsupervised learning with random forest predictors, J. Comput. Graph. Stat. 15 (1) (2006) 118–138.
[4] X. Zhu, C. Change Loy, S. Gong, Constructing robust affinity graphs for spectral clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1450–1457.
[5] M. Bicego, F. Cicalese, A. Mensi, RatioRF: a novel measure for random forest clustering based on the Tversky's ratio model, IEEE Trans. Knowl. Data Eng. 35 (1) (2021) 830–841.
[6] M. Bicego, DisRFC: a dissimilarity-based random forest clustering approach, Pattern Recognit. 133 (2023) 109036.
[7] C. Cui, Y. Ren, J. Pu, X. Pu, L. He, Deep multi-view subspace clustering with anchor graph, arXiv preprint arXiv:2305.06939 (2023).
[8] P. Zhang, S. Wang, L. Li, C. Zhang, X. Liu, E. Zhu, Z. Liu, L. Zhou, L. Luo, Let the data choose: flexible and diverse anchor graph fusion for scalable multi-view clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2023, pp. 11262–11269.
[9] Y. Zhang, S. Yan, L. Zhang, B. Du, Fast projected fuzzy clustering with anchor guidance for multimodal remote sensing imagery, IEEE Trans. Image Process. 33 (2024) 4640–4653.
[10] K. Zhang, J.T. Kwok, Clustered Nyström method for large scale manifold learning and dimension reduction, IEEE Trans. Neural Netw. 21 (10) (2010) 1576–1587.
[11] D. Cai, X. Chen, Large scale spectral clustering via landmark-based sparse representation, IEEE Trans. Cybern. 45 (8) (2014) 1669–1680.
[12] J. Liu, H. Zhang, K. Dong, F. Nie, Robust jointly sparse fast fuzzy clustering via ternary-tree-based anchor graph, IEEE Trans. Fuzzy Syst. 33 (2025) 2284–2294.
[13] Y. Wang, D. Wang, W. Pang, C. Miao, A.-H. Tan, Y. Zhou, A systematic density-based clustering method using anchor points, Neurocomputing 400 (2020) 352–370.
[14] F. Nie, W. Zhu, X. Li, Unsupervised large graph embedding, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2017.
[15] W. Zhu, F. Nie, X. Li, Fast spectral clustering with efficient large graph construction, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2017, pp. 2492–2496.
[16] D. Cheng, S. Liu, S. Xia, G. Wang, Granular-ball computing-based manifold clustering algorithms for ultra-scalable data, Expert Syst. Appl. 247 (2024) 123313.
[17] M.-S. Chen, J.-Q. Lin, C.-D. Wang, D. Huang, J.-H. Lai, Contrastive ensemble clustering, IEEE Trans. Neural Netw. Learn. Syst. 36 (8) (2025) 14678–14690.
[18] W. Wei, J. Wu, X. Guo, J. Yan, J. Liang, Self-Constrained clustering ensemble, IEEE Trans. Pattern Anal. Mach. Intell. 47 (2025) 11946-11960.
[19] M.J. Warrens, H. van der Hoef, Understanding the adjusted rand index and other partition comparison indices based on counting object pairs, J. Classif. 39 (3) (2022) 487–509.
[20] S.T. Hadjitodorov, L.I. Kuncheva, L.P. Todorova, Moderate diversity for better cluster ensembles, Inform. Fusion 7 (3) (2006) 264–275.
[21] Y. Hong, S. Kwong, H. Wang, Q. Ren, Resampling-based selective clustering ensembles, Pattern Recognit. Lett. 30 (3) (2009) 298–305.
[22] M.M. Gösgens, A. Tikhonov, L. Prokhorenkova, Systematic analysis of cluster similarity indices: how to validate validation measures, in: International Conference on Machine Learning, PMLR, 2021, pp. 3799–3808.
[23] X.Z. Fern, W. Lin, Cluster ensemble selection, Stat. Anal. Data Min. ASA Data Sci. J. 1 (3) (2008) 128–141.
[24] T. Ma, T. Yu, X. Wu, J. Cao, A. Al-Abdulkarim, A. Al-Dhelaan, M. Al-Dhelaan, Multiple clustering and selecting algorithms with combining strategy for selective clustering ensemble, Soft Comput. 24 (2020) 15129–15141.
[25] S.-o. Abbasi, S. Nejatian, H. Parvin, V. Rezaie, K. Bagherifard, Clustering ensemble selection considering quality and diversity, Artif. Intell. Rev. 52 (2) (2019) 1311–1340.
[26] Y. Shi, Z. Yu, C.L.P. Chen, J. You, H.-S. Wong, Y. Wang, J. Zhang, Transfer clustering ensemble selection, IEEE Trans. Cybern. 50 (6) (2018) 2872–2885.
[27] X. Zhao, J. Liang, C. Dang, Clustering ensemble selection for categorical data based on internal validity indices, Pattern Recognit. 69 (2017) 150–168.
[28] N. Zhang, X. Zhang, S. Sun, Efficient multiview representation learning with correntropy and anchor graph, IEEE Trans. Knowl. Data Eng. 36 (2023) 4632–4645.
[29] F. Nie, X. Wang, H. Huang, Clustering and projected clustering with adaptive neighbors, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 977–986.
[30] T.-E. Lin, H. Xu, H. Zhang, Discovering new intents via constrained deep adaptive clustering with cluster refinement, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2020, pp. 8360–8367.
[31] U. Von Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (2007) 395–416.
[32] R. Mondal, E. Ignatova, D. Walke, D. Broneske, G. Saake, R. Heyer, Clustering graph data: the roadmap to spectral techniques, Discover Artif. Intell.e 4 (1) (2024) 7.
[33] J. Azimi, X. Fern, Adaptive cluster ensemble selection, in: Twenty-First International Joint Conference on Artificial Intelligence, 2009.
[34] A.M. Ikotun, A.E. Ezugwu, L. Abualigah, B. Abuhaija, J. Heming, K-means clustering algorithms: a comprehensive review, variants analysis, and advances in the era of big data, Inf. Sci. 622 (2023) 178–210.
[35] D. Peng, Z. Gui, D. Wang, Y. Ma, Z. Huang, Y. Zhou, H. Wu, Clustering by measuring local direction centrality for data with heterogeneous density and weak connectivity, Nat. Commun. 13 (1) (2022) 5455.
[36] H. Averbuch-Elor, N. Bar, D. Cohen-Or, Border-peeling clustering, IEEE Trans. Pattern Anal. Mach. Intell. 42 (7) (2020) 1791–1797.
[37] K.M. Ting, Y. Zhu, M. Carman, Y. Zhu, Z.-H. Zhou, Overcoming key weaknesses of distance-based neighbourhood methods using a data dependent dissimilarity measure, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1205–1214.
[38] G.-Y. Zhang, D. Huang, C.-D. Wang, Large-scale tensorized multi-view kernel subspace clustering, ACM Trans. Intell. Syst. Technol. 16 (2025) 2157–6904.
[39] Z. Chen, M. Ma, T. Li, H. Wang, C. Li, Long sequence time-series forecasting with deep learning: a survey, Inform. Fusion 97 (2023) 101819.
[40] A.M. Bagirov, R.M. Aliguliyev, N. Sultanova, Finding compact and well-separated clusters: clustering using silhouette coefficients, Pattern Recognit. 135 (2023) 109144.