Shahid Bahonar University of Kerman
Journal of Mahani Mathematical Research
Print ISSN: 2251-7952 Online ISSN: 2645-4505

UNSUPERVISED FEATURE SELECTION VIA LOW-RANK GRAPH EMBEDDING

F. Beiranvand ✉

Article type: Research Article
(Received: 02 October 2025, Received in revised form 19 February 2026)
(Accepted: 20 April 2026, Published Online: 20 April 2026)

Abstract. In unsupervised feature selection, the absence of class labels makes it necessary to rely on statistical measures like variance to assess feature importance. However, the presence of noise, outliers, and feature redundancy can significantly reduce the accuracy of this process, negatively affecting the performance of machine learning models. To address these issues, this paper proposes the UFSLRAGE algorithm, which begins by applying the LRAGE feature extraction method to minimize reconstruction error, adaptively updating the similarity between samples to remove noise and outliers while generating orthogonal and uncorrelated features. This paper then constructs a weighted bipartite graph that represents both original and extracted features, with cosine similarity determining the edge weights. The LAPJV algorithm is used to identify a maximum matching in this graph, where the vertices corresponding to the original features are selected as the most informative ones. This paper evaluates UFSLRAGE on five widely-used image datasets: Jaffe, Yale, ORL, COIL-20, and pixraw10P, using metrics such as accuracy, normalized mutual information (NMI), precision, recall, and F-measure. Experimental results demonstrate that UFSLRAGE consistently outperforms other state-of-the-art unsupervised feature selection methods, achieving an average NMI of 0.9358 and an average accuracy of 0.9333 on the pixraw10P face image dataset with 10,000 features.

Keywords: Bipartite graph matching, Adaptive graph embedding, LAPJV algorithm, Cosine similarity, Noise and outliers
2020 MSC: 68T20.

1. Introduction

In recent years, the rapid growth of high-dimensional data in applications such as pattern recognition, image processing, and text analysis has created significant challenges for data analysis and learning systems. High-dimensional datasets often contain a large number of redundant, irrelevant, or noisy features, as well as outliers, which can severely degrade learning performance if not properly handled. These issues increase computational complexity and storage requirements, reduce model interpretability, and negatively affect generalization performance [1]. As a result, effective dimensionality reduction has

✉ beiranvand.fi@fe.lu.ac.ir, ORCID: 0000-0002-1908-8386
https://doi.org/10.22103/jmmr.2026.26040.1881
Publisher: Shahid Bahonar University of Kerman
How to cite: F. Beiranvand, *Unsupervised feature selection via low-rank graph embedding*, J. Mahani Math. Res. 2027; 16(1): 75-107.

become a crucial preprocessing step in modern machine learning pipelines. Dimensionality reduction techniques are generally categorized into feature extraction and feature selection. Feature extraction methods aim to transform the original high-dimensional data into a lower-dimensional space by learning new representations, for example through subspace learning or neural-based models, which can effectively reconstruct underlying data structures and handle noise [2]. However, the transformed features often lose their physical meaning, which limits interpretability. In contrast, feature selection directly identifies a subset of informative features from the original feature space, preserving semantic interpretability while reducing redundancy and computational cost. For this reason, feature selection remains a fundamental topic in machine learning research, particularly in high-dimensional settings [3].

Feature selection methods are commonly divided into three categories: filter, wrapper, and embedded approaches. Filter methods evaluate features based on intrinsic statistical properties of the data, independently of any learning algorithm. Wrapper methods incorporate a learning model to assess feature subsets, typically achieving higher accuracy at the expense of increased computational cost. Embedded methods integrate feature selection directly into the training process of the learning model [4]. From another perspective, feature selection approaches can be classified as supervised, semi-supervised, or unsupervised, depending on the availability of class labels. Among these, unsupervised feature selection (UFS) is particularly challenging and important, as it must identify informative features solely based on the intrinsic structure of unlabeled data [5].

In unsupervised feature selection, the absence of labels forces the algorithm to rely on statistical or geometric criteria such as distances, variances, or similarity measures to assess feature importance. However, these criteria are often highly sensitive to noise and outliers, which can distort similarity relationships and lead to suboptimal feature subsets. Moreover, many existing UFS methods either fail to adequately model the underlying data structure or suffer from redundancy among the selected features. This highlights a key gap in current UFS research: how to robustly learn informative and non-redundant feature representations while maintaining a principled and interpretable selection mechanism in the absence of labels. However, many existing unsupervised feature selection methods either fail to capture the intrinsic structure of high-dimensional data or select redundant and less informative features, particularly in the presence of noise and outliers. This highlights the need for a method that can robustly learn informative and non-redundant feature representations while maintaining interpretability in the absence of labels.

To address this gap, the proposed UFSLRAGE integrates Low-Rank Adaptive Graph Embedding (LRAGE) with weighted bipartite graph matching. The core motivation for this integration is twofold. First, LRAGE performs adaptive

probabilistic neighborhood graph embedding and subspace learning by minimizing reconstruction error, dynamically updating sample similarities to enhance robustness against noise and outliers. Second, bipartite graph matching selects a compact, non-redundant subset of original features that best correspond to the informative orthogonal representations learned in the transformed space, thereby preserving interpretability. It is important to emphasize that the proposed UFSLRAGE framework is, in essence, a filter-based unsupervised feature selection method. Specifically, it consists of an initial data-driven representation learning stage based on LRAGE, which learns an orthogonal projection matrix in an unsupervised manner, followed by a purely filter-style feature selection step implemented through weighted bipartite graph matching. This design allows UFSLRAGE to benefit from powerful unsupervised representation learning while retaining the simplicity, efficiency, and interpretability of filter-based feature selection.

The key novelty of UFSLRAGE lies in its combination of three complementary components: (i) adaptive similarity update through LRAGE, which dynamically captures intrinsic data structures and mitigates noise and outliers; (ii) orthogonal feature transformation that generates uncorrelated and robust feature representations; and (iii) bipartite graph matching for selecting a compact, non-redundant, and interpretable subset of features. Importantly, UFSLRAGE operates independently of statistical labels or external supervision, making it robust and reliable in fully unsupervised settings. This integrated design distinguishes UFSLRAGE from existing unsupervised feature selection methods.

The main contributions of this study can be summarized as follows:
LRAGE algorithm is employed to learn an orthogonal and low-rank feature representation that adaptively captures intrinsic data structure while suppressing noise and outliers, without performing additional dimensionality reduction. A cosine-similarity-based relationship is defined between original features and the learned orthogonal features, enabling a principled comparison between the two feature spaces of equal dimensionality. Feature selection is formulated as a weighted bipartite graph matching problem and utilize the LAPJV algorithm to identify the optimal correspondence between original and transformed features, resulting in a compact and informative subset of selected features.

The remainder of this paper is organized as follows: Section 2 reviews related work on feature selection methods. Section 3 presents the proposed UFSLRAGE framework in detail. Experimental results and analysis are provided in Section 4, and conclusions along with future research directions are discussed in Section 5.

2. **Related work**

In the field of feature selection, extensive research has been conducted. This section reviews notable studies in unsupervised feature selection (UFS).

Graph-based and adaptive graph methods have gained attention for preserving data structure. A unified framework combining an explicit selection matrix with a structured adaptive graph has been proposed [1]. Low-rank graph embedding with sparsity regularization has been integrated with latent representations [5]. Adaptive graph manifold learning has also been explored [12]. While these approaches improve structural preservation, they often rely on pre-defined or iteratively learned graphs that remain sensitive to noise and outliers or require careful parameter tuning. The proposed UFSLRAGE surpasses these limitations by adaptively updating similarities through reconstruction error minimization in LRAGE, without dependence on fixed graphs or statistical measures vulnerable to noise.

Subspace and low-rank methods focus on robust representation. Robust Subspace Representation (RSR) using $\ell_{2,1}$-norm regularization has been introduced [10]. Adaptive graphs with label correlation have been employed in multi-label settings [6]. These methods effectively capture global structure but typically do not guarantee orthogonal features or direct mapping back to original dimensions, limiting interpretability and noise resilience. UFSLRAGE addresses this by explicitly enforcing orthogonality ($P^TP = I$) in the transformed space and using bipartite matching to select original features aligned with these orthogonal representations.

Autoencoder and projection-free approaches capture non-linear relationships. Autoencoder-inspired feature selection with group lasso has been proposed [8]. Projection-free methods based on $\ell_{2,0}$-norm constraints [9] and random subspace collaboration [11] have been developed. These methods handle non-linearity well but can be computationally intensive and lack guaranteed orthogonality or optimal correspondence to original features. In contrast, UFSLRAGE provides a lightweight filter-based solution with orthogonal transformation and maximum-weight matching for superior performance and interpretability.

Spectral and clustering-based methods group features for importance scoring. Spectral clustering with centroid distances has been utilized [7]. Standard deviation and cosine similarity-based clustering extensions have also been presented [13]. Although effective for feature grouping, these approaches are sensitive to clustering parameters and noise in similarity computation. UFSLRAGE overcomes this through data-driven orthogonal embedding and optimal assignment.

Other techniques include ant colony optimization [14], bipartite graph-based methods [15–17], and PCA combined with bipartite matching [18]. These methods offer diverse strategies but often lack adaptive handling of noise/outliers or orthogonal guarantees. The proposed UFSLRAGE effectively addresses these weaknesses by combining adaptive reconstruction error minimization in LRAGE with weighted bipartite graph matching, achieving robust orthogonal representations and optimal feature selection.

One of the main weaknesses in many feature selection methods is their inability

to adaptively update similarity relationships between samples, particularly in the presence of noise and outliers. Statistical methods are especially vulnerable in such cases, leading to reduced performance accuracy. The proposed UFSLRAGE effectively mitigates these issues.

3. The Proposed UFSLRAGE Method

In unsupervised feature selection, traditional methods often rely on statistical measures (e.g., variance) that are highly sensitive to noise, outliers, and feature correlations, leading to suboptimal performance in high-dimensional data without labels [19]. To address these challenges, the proposed UFSLRAGE algorithm combines unsupervised representation learning with filter-style feature selection. The method consists of three key stages. First, Low-Rank Adaptive Graph Embedding (LRAGE) [20] is applied as a feature extraction step to generate orthogonal and uncorrelated features in a transformed space. LRAGE minimizes reconstruction error to adaptively update sample similarities, effectively mitigating noise and outliers without requiring pre-defined graphs or statistical criteria. Importantly, no dimensionality reduction is performed at this stage. Second, cosine similarity is computed between the original features and the transformed orthogonal features. Third, a weighted bipartite graph is constructed with cosine similarity as edge weights, and the LAPJV algorithm [22] is used to find the maximum-weight matching. This identifies the subset of original features most aligned with the robust orthogonal representations. The selected feature indices are sorted in ascending order, and the KNN classifier is evaluated on the test set using these features.

The proposed UFSLRAGE is fundamentally a filter-based approach that avoids classifier-dependent learning. It incorporates an initial unsupervised representation learning step via LRAGE to obtain robust orthogonal features, followed by a pure filter-style selection through bipartite matching. The overall procedure is illustrated in Figure 1.

3.1Formulation. The dataset is randomly split into 70% training and 30% test samples. All feature selection steps, including LRAGE fitting to obtain the projection matrix P, transformation of the training data to the new space$Y$, cosine similarity computation, and maximum-weight bipartite matching using the LAPJV algorithm, are performed exclusively on the training set. The selected feature indices are then directly applied to the test set for evaluation using the KNN classifier.

Each training sample is represented as a feature vector in the matrix $X \in \mathbb{R}^{m\times n}$, where m is the number of samples and n is the number of features. LRAGE is applied to extract uncorrelated features without relying on a prebuilt graph. Detailed steps of each stage are described in the following subsections.

3.1.1 Low-Rank Adaptive Graph Embedding (LRAGE). In the first stage, Low-Rank Adaptive Graph Embedding (LRAGE) [20] is applied as an

unsupervised feature extraction step to generate orthogonal and uncorrelated features that are robust to noise and outliers. If there are outliers among the data or the data is affected by noise, the pre-built graph is unable to correctly identify the similarity connections between the samples. To overcome these problems, the LRAGE algorithm simultaneously performs adaptive neighborhood graph embedding and subspace learning with continuous weights by minimizing the reconstruction error. The LRAGE algorithm is utilized to acquire the projection matrix P, and subsequently, new features are generated using $X^\top P$.
The objective function of LRAGE is shown by equation (3) [20]:

$$\begin{aligned} &\min_{P,R,W} \sum_{i,j} \left\| X_i^T - X_j^T PR \right\|_2^2 W_{ij} + \alpha \|W\|_F^2 + \lambda \|PR\|_{2,1} \\ &\text{s.t.} \quad \forall i, \quad W_i \geq 0, \quad \sum_j W_{ij} = 1, \quad P^T P = I \end{aligned} \tag{3}$$

Where α and β denote two non-negative balanced parameters, and the constraint is introduced to ensure the probability characteristics of W_i. It is incorporated to prevent scenarios where every element in each row of matrix W is equal to zero. P denotes a projection matrix and R denotes a regression matrix. The term is included to discourage trivial solutions and promote a prior assumption of uniform distribution.W is an adaptive similarity matrix that represents the likelihood of similarity among sample points X_i and X_j derived from the minimization of the reconstruction error.
Due to the presence of three variables in equation (3), simultaneous optimization poses a challenge. To address this issue, an iterative procedure is developed that updates one variable at a time while keeping the remaining variables constant. The detailed iterative update rules for W, P, R, and the diagonal matrix U, along with intermediate equations (former Equations 1–8), are provided in Appendix A for brevity.
A key property of the projection matrix is its orthogonality:

$$P^T P = I \tag{9}$$

Such a matrix P is called an orthogonal matrix. The initial features (X) are transformed into a new space where essential information and key data structures are preserved using the P matrix of the LRAGE algorithm. This transformation is defined as:

$$Y = X^\top P \tag{10}$$

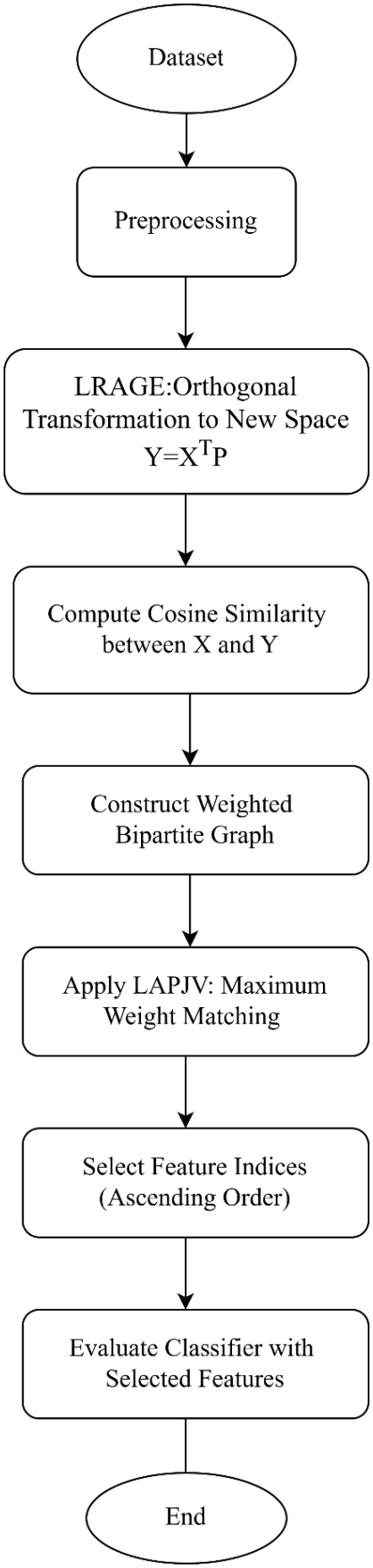


FIGURE 1. The Comprehensive flowchart of UFSLRAGE.

In this equation, Y represents the new set of features that, after the transformation, retain the key information and important structures of the data. It is essential to emphasize that all the extracted features are utilized without performing any dimensionality reduction. Algorithm 1 provides an overview of the LRAGE algorithm (detailed steps in Appendix A). The detailed iterative update rules and intermediate equations are provided in Appendix A for completeness.

Algorithm 1 : LRAGE Algorithm

Input: Training set $X \in \mathbb{R}^{m \times n}$, number of nearest neighbors k, regularization parameter λ, maximum iterations T.
Output: Projection matrix $P \in \mathbb{R}^{n \times n}$.
Initialize projection matrix $P = \mathbf{0}$
Compute initial k-NN similarity matrix W
for iter = 1 to T **do**
    Update similarity matrix W adaptively
    Update projection matrix P via eigen-decomposition (Equation 7)
    Save all eigenvectors in the columns of P
end for
return P (orthogonal projection matrix)

The original feature space is transformed as $Y = X^T P$ using the projection matrix P from Algorithm 1. Since the columns of P are orthogonal eigenvectors ($P^T P = I$), the features in Y are mutually orthogonal and uncorrelated. All eigenvectors are retained, ensuring no dimensionality reduction at this stage.

3.1.2 Cosine Similarity Computation. In the second stage, pairwise similarity between the original features (X) and the transformed orthogonal features ($Y = X^T P$) is quantified using cosine similarity. This measure is particularly suitable for high-dimensional data due to its robustness to magnitude variations and focus on directional alignment between vectors [23, 24]. Cosine similarity is defined as:

$$\text{Cosine similarity}(X, Y) = \frac{X^T Y}{\|X\|\|Y\|} \tag{11}$$

where $\cos(\theta)$ is the cosine of the angle θ between vectors X and Y, and $\|\cdot\|$ denotes the Euclidean norm. Values range from -1 (opposite directions) to 1 (perfect alignment), with 0 indicating orthogonality [24]. The resulting $n \times n$ similarity matrix is symmetric, $s_{ij} = s_{ji} \quad \forall i, j \in \{1, \ldots, n\}$, and serves as edge weights in the weighted bipartite graph for the subsequent matching stage [25]. Higher similarity values indicate stronger correspondence between original and transformed features, increasing their likelihood of selection in the maximum-weight matching process.

In practice, cosine similarity is directly used as the primary measure for quantifying the correspondence between original and transformed features.

Since cosine similarity may take negative values, a simple linear shifting is applied to ensure non-negative edge weights required by the LAPJV algorithm. Specifically, the edge weight is defined as Cosine similarity $= 1 -$ cosine distance. This transformation maps the similarity values from the interval $[-1, 1]$ to $[0, 2]$ while preserving the relative ordering of similarities. As a result, the maximum-weight matching objective and the ranking of selected features remain unchanged.

3.1.3 Construct Weighted Bipartite Graph Matching (WBGM). In the third stage, the original features (set X) and transformed orthogonal features (set$Y$) are modeled as partite sets in a weighted bipartite graph $G = (X \cup Y, E)$. Edge weights are defined by the cosine similarity matrix computed in the previous step.

The problem of graph matching is a captivating subject within the research community [26]. A bipartite graph consists of two disjoint vertex sets (X and Y) with edges only between the sets, each assigned a weight $w(e) \in \mathbb{R}^+$ [27]. A matching M is a set of edges without common vertices ($M \subseteq E$). In a weighted graph, the maximum-weight matching maximizes the sum of edge weights in M [28]. Figure 2 illustrates a maximum matching in a bipartite graph: edges (1; 9), (2; 6), (3; 8), and (5; 7) [18].

The objective is to find the maximum-weight perfect matching, which corresponds to the linear assignment problem. The LAPJV algorithm [22] is employed to efficiently solve this problem using the cosine similarity matrix as input. The resulting matching yields the indices of original features that best align with the orthogonal representations in Y, forming the selected subset.

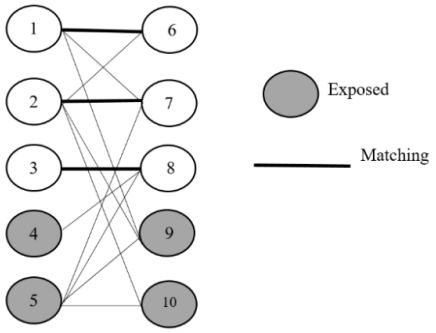


FIGURE 2. A maximum matching in a bipartite graph: (1; 9), (2; 6), (3; 8) and (5; 7) [18].

3.1.4 The LAPJV Algorithm to Find Maximum Weighted Bipartite Graph Matching. The LAPJV algorithm [22] is employed to solve the maximum-weight bipartite matching problem using the cosine similarity

matrix as the cost matrix. LAPJV is an efficient implementation of the Jonker -Volgenant algorithm, which solves the linear assignment problem in $O(n^3)$ time. It consists of two main phases: initialization (column reduction, reduction transfer, and augmenting row reduction) and augmentation (finding shortest augmenting paths using a modified Dijkstra's algorithm until a complete assignment is achieved).

The detailed steps of the LAPJV algorithm are presented in Algorithm 2.

Algorithm 2 :Maximum Weighted Bipartite Graph Matching using LAPJV Algorithm

Input: $n \times n$ cost matrix C (similarity weights on edges between sets X and Y), where n is the number of features and $|X| = |Y|$.
Output: Feature indices c that maximize the total weight.
for $i = 1$ **to** n **do**
    Perform column reduction
    Transfer reductions
    Apply augmenting row reduction
    Initialize arrays for shortest path lengths and predecessors
    Identify columns with updated shortest path values
    Scan rows to find augmenting paths
    Update column values and augment assignment
end for
Return: assignment indices c

3.2 Algorithm of the proposed UFSLRAGE method. The detailed steps of the proposed UFSLRAGE algorithm are presented in Algorithm 3. An illustrative example is provided in Figure 3, where an input matrix $X \in \mathbb{R}^{5\times3}$ is used and the selected features are highlighted in bold. Figure 3 demonstrates the contribution of each stage: LRAGE generates orthogonal features that capture the most informative directions in the data (due to zero correlation among them), while bipartite matching ensures optimal correspondence to the original features. This combination enables effective unsupervised feature selection, as confirmed by the superior classification performance in Section 4.

4. Experimental Results

The performance of the proposed UFSLRAGE method is evaluated through comparison with eight established unsupervised feature selection algorithms: UFSPCA-2022 [18], SCEFS-2021 [13], SCAFS-2021 [13], AEFS-2018 [8], DGUFS-2018 [9], RSR-2015 [10], SRCFS-2019 [11], and URAFS-2018 [12]. Experiments are conducted on five widely used image datasets (detailed in Table 1). All methods are implemented in MATLAB R2018b on an Intel Core i7 CPU with 8 GB RAM running Microsoft Windows 10.

Algorithm 3 :The proposed UFSLRAGE algorithm

1: **Input:** Training dataset $X \in \mathbb{R}^{m \times n}$
2: **Output:** Selected feature indices c
3: Compute projection matrix P using Algorithm 1.
4: Transform training data to new space: $Y_{n \times m} = X^T_{n \times m} P_{n \times n}$.
5: Compute cosine distance matrix $CD = \text{pdist2}(X, Y, \text{'cosine'})$.
6: Convert cosine distance to similarity: $CS = 1 - CD$ (non-negative edge weights).
7: Construct a weighted bipartite graph with edge weights CS.
8: Solve maximum-weight matching: $c = \text{LAPJV}(CS)$.
9: **Return:** selected feature indices c (sorted in ascending order if required).

4.1. **Datasets description.** The experiments are conducted on five publicly available image datasets: JAFFE (facial expressions), Yale and ORL (face images under varying conditions), COIL-20 (object images from multiple views), and pixraw10P (high-dimensional face images). Detailed characteristics and references are summarized in Table 1.

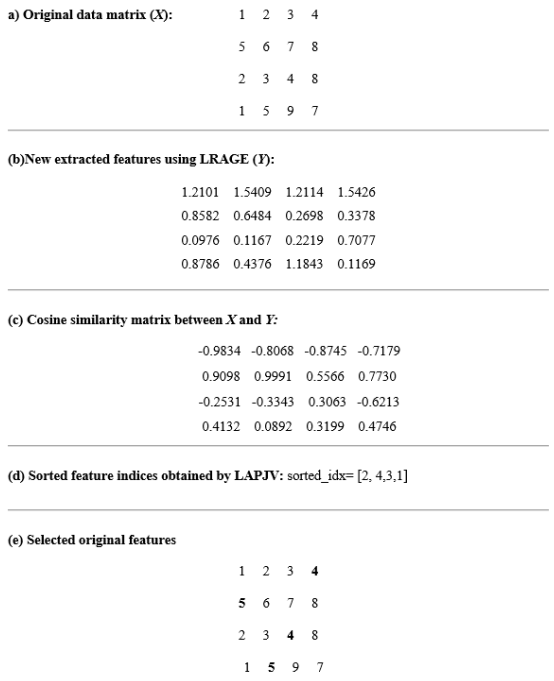


FIGURE 3. Step-by-step numerical illustration of the proposed UFSLRAGE framework

4.2. **Classifier.** The selected features are evaluated using the K-Nearest Neighbors (KNN) classifier [29]. KNN is a non-parametric, instance-based algorithm that performs classification by identifying the k closest training samples to a test instance based on a distance metric and assigning the majority class label among them.
Due to its simplicity and strong performance, KNN (with $k = 5$) is widely used as a benchmark classifier for comparing feature selection methods.

TABLE 1. Overview of datasets used in the experiments.

Dataset	**#Samples**	**#Features**	**#Classes**	**Reference**
JAFFE	213	676	10	Lyons et al., 1998 (https://zenodo.org/record/3451084)
Yale	165	1024	15	http://vision.ucsd.edu/content/yale-face-database
ORL	400	1024	40	https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
COIL-20	1440	1024	20	https://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php
pixraw10P	100	10000	10	http://featureselection.asu.edu/datasets.php

4.3. **Performance Evaluation Criteria.** The performance of the proposed method and baselines is evaluated using five standard metrics: Accuracy, Precision, Recall, F-measure, and Normalized Mutual Information (NMI). These metrics provide a comprehensive assessment of classification and clustering quality. Accuracy [30] is the ratio of correctly classified samples to the total number of samples:

$$\text{Accuracy} = \frac{\text{Number of Correctly Classified Samples}}{\text{Total Number of Samples}} \tag{12}$$

$$\text{Accuracy} = \frac{TN + TP}{TN + FN + TP + FP} \tag{13}$$

Where TP, TN, FP, and FN are true positives, true negatives, false positives, and false negatives, respectively.

Precision measures the accuracy of positive predictions [30]:

$$Precision = \frac{TP}{TP + FP} \tag{14}$$

Recall (sensitivity) quantifies the proportion of actual positives correctly identified [30]:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{15}$$

High Precision or Recall alone is insufficient, as one can be high while the other is low due to false positives or negatives [30].

F-measure is the harmonic mean of Precision and Recall [18]:

$$F\text{-measure} = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{16}$$

Additionally, this paper employs the Normalized Mutual Information (NMI) metric to assess the mutual information between the predicted and actual class labels. NMI is derived from the concept of entropy in information theory. The entropy of variable Y is calculated using equation (17) [18] :

$$H(Y) = -\mathbb{E}[\log p(y)] = -\sum\nolimits_{y} p(y)\, \log p(y) \tag{17}$$

The mutual information quantifies the degree of dependence between two random discrete variables, Y and X, and can be computed using equation (18) [18]:

$$I(Y; X) = H(Y) - H(Y \mid X) \tag{18}$$

NMI is computed using equation (19) [18]:

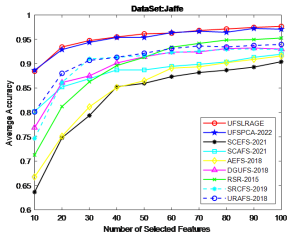
$$NMI(Y, X) = \frac{2I(Y; X)}{H(Y) + H(X)} \tag{19}$$

NMI = 0 indicates independence, and NMI = 1 indicates perfect agreement.

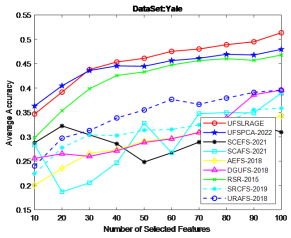
The experiments are repeated 30 times independently for each algorithm and dataset to ensure robust evaluation. In each run, the dataset is randomly split into 70% training and 30% test samples. For each method, the top m features are selected($m \in \{10, 20, \ldots, 100\}$), and classification is performed on the test set using these features. The average Accuracy, Precision, Recall, F-measure, and NMI across the 30 runs are reported. The proposed UFSLRAGE method is compared with eight established unsupervised feature selection algorithms: AEFS [8], DGUFS [9], RSR [10], SRCFS [11], URAFS [12] , SCAFS [13], SCEFS [13], and UFSPCA [18].

Figure 4 shows the average KNN accuracy as a function of the number of selected features (10 to 100 in increments of 10) across 30 runs. The proposed UFSLRAGE consistently achieves the highest accuracy on all datasets, with a steady increase as more features are selected. It particularly excels on the high-dimensional pixraw10P dataset (10,000 features). Figures 4–8 demonstrate that UFSLRAGE produces superior feature subsets across all metrics compared to the baselines.

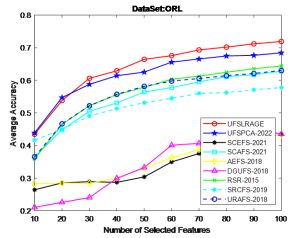
In Figure 5, the mean precision efficiency of the KNN classifier is illustrated across 30 separate iterations for the specified datasets. The range of features was systematically adjusted from 10 to 100, with increments of 10. In all figures, the y-axis denotes the classification precision, while the horizontal axis signifies the number of selected features.



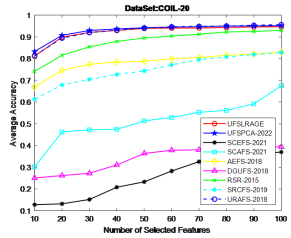
(A) Jaffe



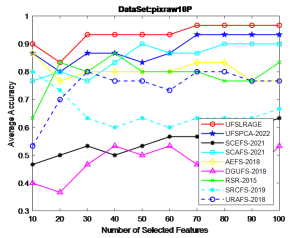
(B) Yale



(C) ORL



(D) COIL-20

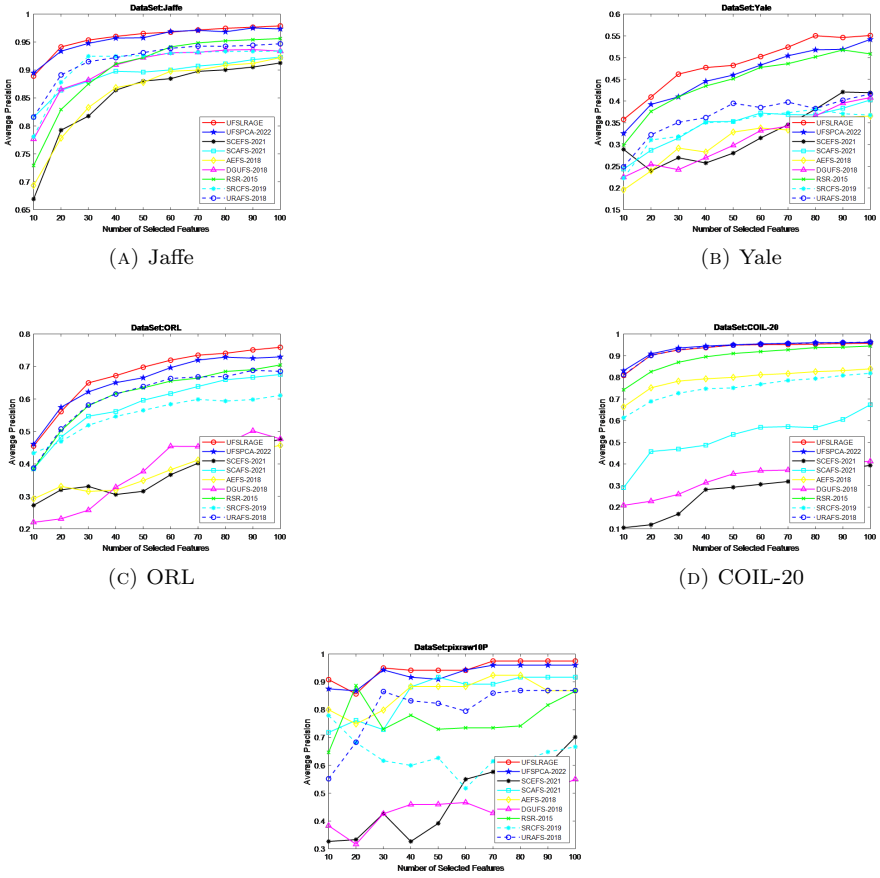


(E) pixraw10P

FIGURE 4. Average classification accuracy of the KNN classifier as a function of the number of selected features (10 to 100 in increments of 10) across 30 independent runs on five datasets.

Notably, Figure 5 underscores the superior Precision of the proposed method in comparison to the other mentioned methodologies across all subsets of features. An insightful observation from the figures is the direct correlation between the classification precision and the number of selected features, signifying the relevance of the chosen features. The consistently higher Precision of the proposed UFSLRAGE method suggests its superior performance relative to

other approaches.



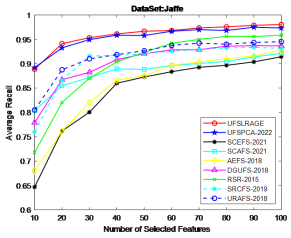
(A) Jaffe (B) Yale

(C) ORL (D) COIL-20

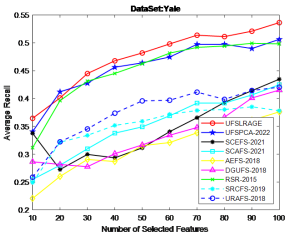
(E) pixraw10P

FIGURE 5. Average Precision of the KNN classifier as a function of the number of selected features (10 to 100 in increments of 10) across 30 independent runs on five datasets.

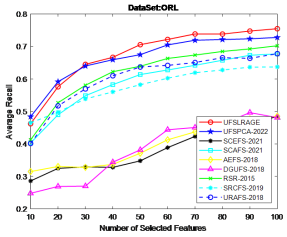
Figure 6 displays the KNN classifier's mean recall efficiency over 30 separation iterations across the mentioned datasets. In all figures, the x-axis signifies the quantity of selected features, whereas the y-axis denotes the average recall for classification. Notably, Figure 6 illustrates the effectiveness of the proposed UFSLRAGE algorithm for all datasets and datasets with high dimensionality such as pixraw10P, that have 10,000 features.



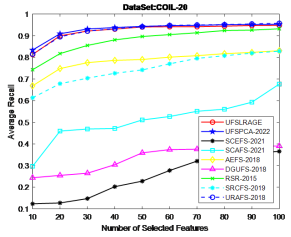
(A) Jaffe



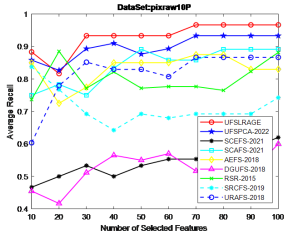
(B) Yale



(C) ORL



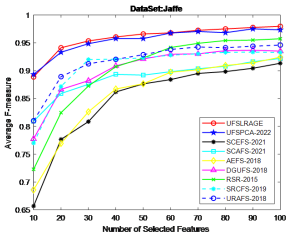
(D) COIL-20



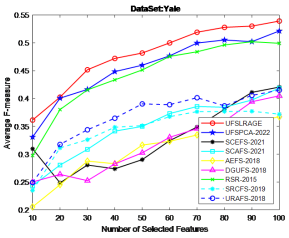
(E)
pixraw10P

FIGURE 6. Average Recall of the KNN classifier as a function of the number of selected features (10 to 100 in increments of 10) across 30 independent runs on five datasets.

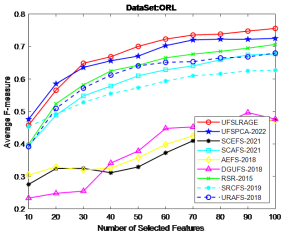
In Figure 7, the average F-measure efficiency of the KNN classifier is displayed in 30 independent iterations for the mentioned datasets. The x-axis corresponds to the quantity of selected features while all figures show the average F-measure on the y-axis. Figure 7 demonstrates the efficacy of the proposed UFSLRAGE algorithm for all datasets and datasets with high dimensionality such as pixraw10P that have 10,000 features.



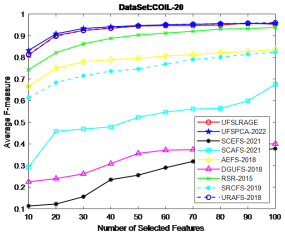
(A) Jaffe



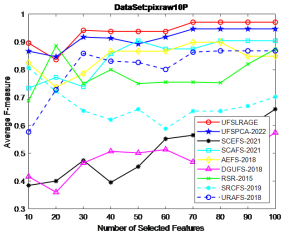
(B) Yale



(C) ORL



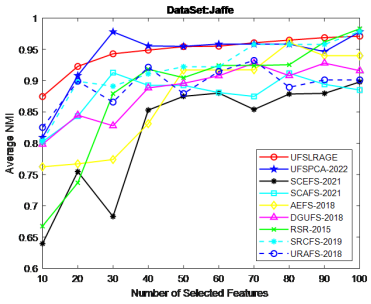
(D) COIL-20



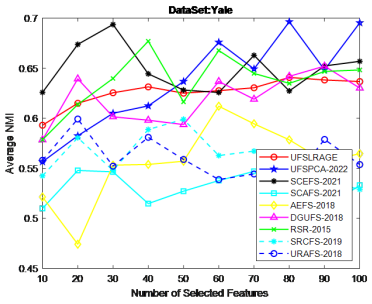
(E) pixraw10P

FIGURE 7. Average F-measure of the KNN classifier as a function of the number of selected features (10 to 100 in increments of 10) across 30 independent runs on five datasets.

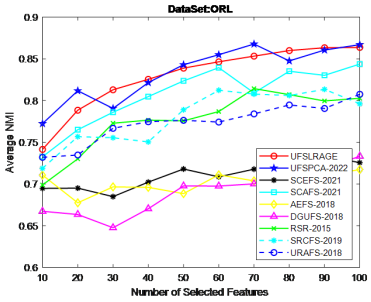
In Figure 8, the average NMI for the KNN classifier is depicted in 30 independent iterations for the mentioned datasets. Across all figures, the x-axis illustrates the quantity of selected features, and the y-axis represents the average NMI for classification. Figure 8 demonstrates the effectiveness of the proposed UFSLRAGE algorithm for all datasets and datasets with high dimensionality such as pixraw10P that have 10,000 features.



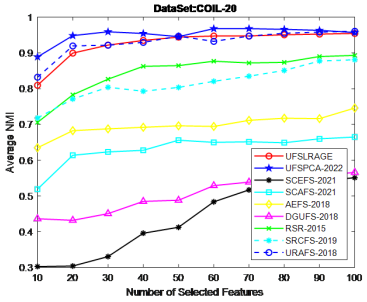
(A) Jaffe



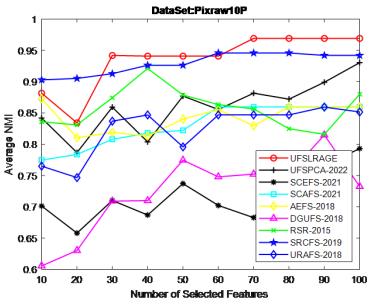
(B) Yale



(C) ORL



(D) COIL-20



(E) pixraw10P

FIGURE 8. Average Normalized Mutual Information (NMI) of the KNN classifier as a function of the number of selected features (10 to 100 in increments of 10) across 30 independent runs on five datasets.

Tables 2–6 present a comprehensive comparison of the KNN classifier performance metrics (Accuracy, Precision, Recall, F-measure, and NMI) across the five datasets. The proposed UFSLRAGE method is compared with eight established unsupervised feature selection algorithms. The best performance for each dataset is highlighted in bold. Statistical significance is assessed using the two-sided paired Wilcoxon signed-rank test($p < 0.05$), with significant improvements over baselines marked by asterisks (*). Overall, the results demonstrate the consistent superiority of UFSLRAGE across all metrics and datasets. Since multiple datasets and performance metrics are evaluated, no explicit multiple-comparison correction is applied; therefore, the reported significance results should be interpreted accordingly. Bold values indicate the best

Table 2. Average accuracy for selected 10 features subset.

Dataset	UFSLRAGE	UFSPCA	SCEFS	SCAFS	AEFS	DGUFS	RSR	SRCFS	URAFS
Jaffe	**0.9537***	0.9506	0.8331	0.8829	0.8459	0.8963	0.8924	0.8987	0.9103
Yale	**0.4546***	0.4308	0.3273	0.3253	0.2941	0.3170	0.4295	0.3271	0.3523
ORL	**0.6374***	0.6171	0.3401	0.5444	0.3501	0.3420	0.5602	0.5218	0.5562
COIL20	0.9215	**0.9290**	0.2537	0.5135	0.7831	0.3385	0.8778	0.7487	0.9254
pixraw10P	**0.9333***	0.8833	0.55	0.85	0.8033	0.4667	0.79	0.6567	0.7433
Wilcoxon		+	+	+	+	+	+	+	+

performance for each dataset, and the symbol (*) denotes that UFSLRAGE significantly outperforms the corresponding baseline according to the two-sided paired Wilcoxon signed-rank test at a significance level of ($p < 0.05$). Table 2 reports the average classification accuracy obtained using a subset of 10 selected features for all compared algorithms. As shown in Table 2, UFSLRAGE consistently demonstrates superior performance compared to the other eight feature selection methods across most datasets. Specifically, on the Jaffe dataset, UFSLRAGE achieves the highest average accuracy of 0.9537, outperforming all competing approaches, which highlights its effectiveness in selecting discriminative features. For the Yale dataset, although the overall accuracy values are relatively low due to the challenging nature of the dataset, UFSLRAGE attains the best performance with an accuracy of 0.4546, significantly outperforming methods such as SCEFS (0.3273) and SCAFS (0.3253). Similarly, on the ORL dataset, UFSLRAGE achieves the highest average accuracy of 0.6374, confirming its ability to identify informative features in high-dimensional settings. In the case of the COIL20 dataset, UFSPCA slightly outperforms UFSLRAGE; nevertheless, UFSLRAGE still exhibits competitive performance with an accuracy of 0.9215. Finally, for the pixraw10P dataset, which contains 10,000 features, UFSLRAGE attains the highest accuracy of 0.9333, outperforming all other algorithms and further demonstrating its robustness and scalability in high-dimensional feature selection scenarios. Overall, the experimental results indicate that UFSLRAGE provides strong and consistent feature selection performance across diverse datasets. The statistical significance confirmed by the Wilcoxon signed-rank test further supports the superiority of UFSLRAGE over

most baseline methods.

Bold values indicate the best performance for each dataset, while the symbol

Table 3. Average Precision for selected 10 features subset

Dataset	UFSLRAGE	UFSPCA	SCEFS	SCAFS	AEFS	DGUFS	RSR	SRCFS	URAFS
Jaffe	**0.9579***	0.9548	0.8524	0.8913	0.8591	0.9024	0.9019	0.9097	0.9191
Yale	**0.4862***	0.4599	0.3218	0.3447	0.3086	0.3134	0.4462	0.3415	0.3663
ORL	**0.6741***	0.6574	0.3670	0.5829	0.3723	0.3763	0.6117	0.5518	0.6104
COIL20	0.9293	**0.9363**	0.2686	0.5227	0.7917	0.3298	0.8911	0.7503	0.9323
pixraw10P	**0.9440***	0.9295	0.4827	0.8540	0.8581	0.4426	0.7668	0.6367	0.8017
Wilcoxon		+	+	+	+	+	+	+	+

(*) denotes that UFSLRAGE significantly outperforms the corresponding baseline according to the two-sided paired Wilcoxon signed-rank test ($p < 0.05$).

Table 3 presents the average precision obtained using a subset of 10 selected features for all competing methods. As reported in Table 3, UFSLRAGE consistently achieves superior precision compared to the other eight feature selection algorithms across most datasets. On the Jaffe dataset, UFSLRAGE attains the highest precision value of 0.9579, outperforming all competing methods, which highlights its effectiveness in selecting highly discriminative features. Although UFSPCA and URAFS also yield competitive results, UFSLRAGE maintains a clear advantage. For the Yale dataset, where precision values are generally lower due to increased data complexity, UFSLRAGE still achieves the best performance with a precision of 0.4862, surpassing UFSPCA (0.4599) and RSR (0.4462). This demonstrates its superior capability in identifying relevant features under challenging conditions. Similarly, on the ORL dataset, UFSLRAGE achieves the highest precision of 0.6741, outperforming strong competitors such as UFSPCA and RSR, further confirming its robustness in precision-oriented feature selection. In the case of the COIL20 dataset, UFSPCA slightly outperforms UFSLRAGE (0.9363 versus 0.9293). Nevertheless, UFSLRAGE remains among the top-performing methods and shows a substantial advantage over most other algorithms. Finally, for the high-dimensional pixraw10P dataset containing 10,000 features, UFSLRAGE achieves the highest precision value of 0.9440, significantly outperforming all competing approaches. This result highlights the scalability and effectiveness of UFSLRAGE in handling high-dimensional data. Overall, UFSLRAGE demonstrates strong and consistent precision performance across diverse datasets, achieving the highest precision in four out of five cases. The statistical significance confirmed by the Wilcoxon signed-rank test further validates the superiority of UFSLRAGE over existing feature selection methods, particularly in scenarios where precision is a critical evaluation metric. The best results for each dataset are highlighted in bold, and the symbol (*) indicates that UFSLRAGE significantly outperforms the corresponding baseline according to the two-sided paired Wilcoxon signed-rank test ($p < 0.05$).

Table 4. Average Recall for selected 10 features subset

Dataset	UFSLRAGE	UFSPCA	SCEFS	SCAFS	AEFS	DGUFS	RSR	SRCFS	URAFS
Jaffe	**0.9588***	0.9544	0.8434	0.8845	0.8558	0.9025	0.8999	0.9032	0.9158
Yale	**0.4741***	0.4566	0.3463	0.3516	0.3121	0.3328	0.4512	0.3509	0.3738
ORL	**0.6757***	0.6647	0.3819	0.5916	0.3951	0.3846	0.6199	0.5767	0.6035
COIL20	0.9227	**0.9303**	0.2487	0.5115	0.7848	0.3330	0.8792	0.7484	0.9265
pixraw10P	**0.9300***	0.8992	0.5433	0.84	0.8310	0.5195	0.8010	0.7132	0.8169
Wilcoxon		+	+	+	+	+	+	+	+

Table 4 reports the average recall obtained using a subset of 10 selected features for all competing methods. As shown in Table 4, UFSLRAGE consistently achieves superior recall performance across most datasets when compared with the other eight feature selection methods. On the Jaffe dataset, UFSLRAGE attains the highest recall value of 0.9588, outperforming all competing approaches, which indicates its strong ability to identify relevant features and reduce false negatives. For the Yale dataset, where recall values are generally lower due to increased data variability, UFSLRAGE still achieves the best performance with a recall of 0.4741, surpassing UFSPCA and RSR. This result highlights the robustness of UFSLRAGE in challenging unsupervised scenarios. Similarly, on the ORL dataset, UFSLRAGE obtains the highest recall value of 0.6757, significantly outperforming methods such as SCEFS and AEFS. This demonstrates its effectiveness in selecting informative features that enhance recall performance. In the case of the COIL20 dataset, UFSPCA slightly outperforms UFSLRAGE (0.9303 versus 0.9227). Nevertheless, UFSLRAGE remains among the top-performing methods and shows a clear advantage over most competing algorithms. Finally, for the high-dimensional pixraw10P dataset containing 10,000 features, UFSLRAGE achieves the highest recall value of 0.9300, significantly outperforming all other methods. This result confirms the scalability and effectiveness of UFSLRAGE in handling high-dimensional data while maintaining strong recall performance. Overall, UFSLRAGE achieves the highest recall in four out of five datasets, demonstrating its robustness and consistency across diverse data distributions. The statistical significance validated by the Wilcoxon signed-rank test further confirms the superiority of UFSLRAGE over existing unsupervised feature selection methods, particularly in applications where recall is a critical evaluation metric.

Table 5. Average F-measure for selected 10 features subset

Dataset	UFSLRAGE	UFSPCA	SCEFS	SCAFS	AEFS	DGUFS	RSR	SRCFS	URAFS
Jaffe	**0.9583***	0.9546	0.8476	0.8877	0.8572	0.9024	0.9008	0.9063	0.9173
Yale	**0.4786***	0.4562	0.3289	0.3483	0.3069	0.3186	0.4436	0.3442	0.3664
ORL	**0.6740***	0.6617	0.3706	0.5897	0.3837	0.3787	0.6200	0.5671	0.6043
COIL20	0.9260	**0.9333**	0.2568	0.5169	0.7881	0.3309	0.8851	0.7492	0.9294
pixraw10P	**0.9369***	0.9140	0.5063	0.8467	0.8441	0.4774	0.7833	0.6723	0.8088
Wilcoxon		+	+	+	+	+	+	+	+

Bold values indicate the best performance for each dataset, and the symbol (*) denotes statistically significant superiority of UFSLRAGE based on the two-sided paired Wilcoxon signed-rank test ($p < 0.05$).

As shown in Table 5, UFSLRAGE consistently achieves superior F-measure performance across most datasets. On the Jaffe dataset, UFSLRAGE obtains the highest F-measure value of 0.9583, outperforming all competing methods and slightly surpassing UFSPCA. This result confirms its strong ability to maintain an effective balance between precision and recall. For the Yale dataset, which is generally more challenging due to lower overall performance levels, UFSLRAGE again achieves the best result with an F-measure of 0.4786. This demonstrates its robustness and stability under difficult unsupervised learning conditions. Similarly, on the ORL dataset, UFSLRAGE achieves the highest F-measure (0.6740), significantly outperforming most competing methods, including SCEFS and AEFS, and confirming its effectiveness in selecting discriminative features. In the case of the COIL20 dataset, UFSPCA slightly outperforms UFSLRAGE (0.9333 versus 0.9260). Nevertheless, UFSLRAGE remains among the top-performing methods and clearly outperforms the majority of baseline algorithms. Finally, for the high-dimensional pixraw10P dataset containing 10,000 features, UFSLRAGE achieves the best F-measure value of 0.9369, substantially outperforming all competing methods. This result highlights the scalability and effectiveness of UFSLRAGE in preserving a balanced performance in high-dimensional feature spaces. Overall, UFSLRAGE achieves the highest F-measure in four out of five datasets, demonstrating its consistent superiority and robustness across diverse data distributions. The statistical significance confirmed by the Wilcoxon signed-rank test further validates that UFSLRAGE outperforms existing unsupervised feature selection methods in achieving an optimal balance between precision and recall.

Table 6 reports the average Normalized Mutual Information (NMI) obtained using a subset of 10 selected features for UFSLRAGE and eight competing unsupervised feature selection methods. NMI measures the agreement between clustering results and ground truth labels, and higher values indicate better preservation of data structure. As shown in Table 6, UFSLRAGE demonstrates strong and consistent performance in terms of NMI across most datasets. On the Jaffe dataset, UFSLRAGE achieves the highest NMI value of 0.9465, clearly outperforming all competing methods, which confirms its effectiveness in preserving the intrinsic data structure. For the Yale dataset, although UFSPCA and RSR achieve slightly higher NMI values, UFSLRAGE still delivers competitive performance (0.6264) and remains among the top-performing methods. This result indicates that UFSLRAGE maintains robust clustering quality even in challenging datasets with high intra-class variability. On the ORL dataset, UFSLRAGE achieves an NMI of 0.8294, which is comparable to the best-performing method and higher than most baseline algorithms. This demonstrates its capability to preserve meaningful cluster structures in medium-scale datasets. In the COIL20 dataset, UFSPCA attains the highest NMI, while

UFSLRAGE achieves a competitive value of 0.9263 and significantly outperforms several baseline methods such as SCEFS and SCAFS. This highlights the stability of UFSLRAGE in maintaining mutual information across diverse data distributions. Finally, for the high-dimensional pixraw10P dataset with 10,000 features, UFSLRAGE achieves the highest NMI value of 0.9358, substantially outperforming all competing methods. This result underscores the scalability and robustness of the proposed algorithm in high-dimensional settings. Overall, UFSLRAGE achieves the best or near-best NMI performance across the majority of datasets. The Wilcoxon signed-rank test further confirms that UFSLRAGE consistently outperforms or remains statistically competitive with existing unsupervised feature selection methods, demonstrating its effectiveness in preserving data structure and clustering quality.

Table 6. Average NMI for selected 10 features subset

Dataset	UFSLRAGE	UFSPCA	SCEFS	SCAFS	AEFS	DGUFS	RSR	SRCFS	URAFS
Jaffe	**0.9465***	0.9224	0.8197	0.8791	0.8728	0.8844	0.8827	0.9203	0.8932
Yale	0.6264*	0.6359	**0.6491**	0.5300	0.5567	0.6191	0.6368	0.5617	0.5599
ORL	0.8294*	**0.8337**	0.7102	0.8070	0.7031	0.6901	0.7765	0.7807	0.7736
COIL20	0.9263	**0.9518**	0.4362	0.6314	0.6979	0.5023	0.8448	0.8155	0.9303
pixraw10P	**0.9358***	0.8608	0.7195	0.8305	0.8421	0.7230	0.8582	0.9296	0.8244
Wilcoxon		+	+	+	+	+	+	+	+

Bold values indicate the best performance for each dataset, and * denotes that UFSLRAGE significantly outperforms the corresponding baseline according to the two-sided paired Wilcoxon signed-rank test ($p < 0.05$).

These results demonstrate that UFSLRAGE consistently achieves a superior trade-off between classification accuracy and clustering quality.

4.4. **Computational Complexity.** Time complexity characterizes the computational efficiency of an algorithm in terms of the time it requires to run on a computer. It is typically denoted with Big-O notation and stated as a function of the input size. In the worst case, time complexity signifies the maximum duration of the method for each input size [18]. The proposed UFSLRAGE method consists of three main components, where n denotes the number of features (dimensionality), and m denotes the number of training samples. LRAGE optimization: The dominant operations involve matrix multiplications and eigen-decomposition in each iteration. With a maximum of T iterations, the complexity is $O(T \cdot n^2 \cdot m + T \cdot n^3)$ [20]. Construction of the pairwise similarity matrix: Cosine similarity is computed between the n original features and the n transformed features, producing an $n \times n$ similarity matrix. Each pairwise similarity requires a dot product of dimension m, yielding a total complexity of $O(n^2 \cdot m)$. The time complexity of computing a single cosine similarity is $O(m)$, making it efficient for high-dimensional data [23, 24].

Matching solver: The LAPJV implementation of the Jonker–Volgenant algorithm solves the maximum-weight bipartite matching on an $n \times n$ cost matrix with worst-case complexity $O(n^3)$ [22]. Overall, the computational complexity of UFSLRAGE is dominated by $O(n^3)$, mainly due to the eigen-decomposition in the LRAGE stage and the LAPJV matching solver.

4.5. **Statistical Analysis.** The statistical significance of performance differences between the proposed UFSLRAGE method and the baselines is assessed using the two-sided paired Wilcoxon signed-rank test at significance level $\alpha = 0.05$. The test is paired across the identical 30 random train/test splits for each dataset and evaluation metric. No multiple-comparison correction (e.g., Holm–Bonferroni) is applied, given the exploratory nature of the study and the consistent superiority of UFSLRAGE across most datasets and metrics. Significant improvements over individual baselines ($p < 0.05$) are indicated by asterisks (*) in Tables 2–6.

4.6. **Implementation Details and Reproducibility.** The hyperparameters used for the proposed UFSLRAGE method are listed in Table 7. For LRAGE, the number of nearest neighbors k is set to 5, the $\ell_{2,1}$-norm regularization parameter λ to 100000 (selected empirically to provide strong robustness against noise and outliers), and the maximum number of iterations to 5 (sufficient for convergence).The subspace dimensionality was kept equal to the original number of features, with no dimensionality reduction performed. Cosine similarity was computed using MATLAB's pdist2 function with the 'cosine' metric, and similarity values were obtained by the transformation $1 - \text{distance}$. The maximum-weight bipartite matching was solved using the LAPJV implementation of the Jonker-Volgenant algorithm. For all baseline methods (AEFS, DGUFS, RSR, SRCFS, URAFS, SCEFS, SCAFS, and UFSPCA), the hyperparameters recommended by the original authors in their published papers or publicly available code are followed.

TABLE 7. Hyperparameter settings used in the proposed UFSLRAGE method.

Component	Parameter	Value	Description
LRAGE	k (nearest neighbors)	5	Number of nearest neighbors for adaptive graph construction
	λ ($\ell_{2,1}$-norm regularization)	100000	Strong regularization empirically chosen for robustness
	Maximum iterations	5	Sufficient for convergence
	Subspace dimensionality	Original	No dimensionality reduction
Similarity Computation	Distance metric	Cosine	pdist2(..., 'cosine'); similarity = 1 − distance
Bipartite Matching	Solver	LAPJV	Jonker-Volgenant algorithm

All experiments are repeated over 30 independent runs with random 70/30 train-test splits, using identical random seeds across methods for fair comparison.

A sensitivity analysis with respect to the neighborhood size k and the regularization parameter λ is provided in Appendix B to further examine the robustness of the proposed method.

4.7. **Reproducibility and Confidence Intervals.** To ensure reproducibility and evaluate performance stability, all experiments are repeated 30 times with independent random seeds (using rng(42 + iter) in MATLAB). For each number of selected features, the mean and 95% confidence interval (CI) of Accuracy, Precision, Recall, F-measure, and NMI are reported across the 30 runs. The narrow confidence intervals observed indicate high stability of the proposed UFSLRAGE method across independent runs. Tables 8–12 present the mean performance and 95% confidence intervals for UFSLRAGE on the five datasets. In all cases, performance metrics improve as the number of selected features increases, demonstrating the method's ability to effectively identify discriminative features. The consistently small CIs further confirm the robustness of UFSLRAGE against randomness in data partitioning.

TABLE 8. Mean and 95% confidence interval (CI) of Accuracy, Precision, Recall, F-measure, and NMI for UFSLRAGE on the Jaffe dataset over 30 independent runs

NumFeat	Accuracy ± 95% CI	Precision ± 95% CI	Recall ± 95% CI	F-measure ± 95% CI	NMI ± 95% CI
10	0.8577 ± 0.0216	0.8679 ± 0.0214	0.8624 ± 0.0208	0.8650 ± 0.0208	0.8514 ± 0.0201
20	0.9201 ± 0.0140	0.9247 ± 0.0130	0.9227 ± 0.0136	0.9236 ± 0.0129	0.9101 ± 0.0144
30	0.9471 ± 0.0111	0.9498 ± 0.0108	0.9481 ± 0.0111	0.9489 ± 0.0106	0.9405 ± 0.0117
40	0.9556 ± 0.0106	0.9569 ± 0.0107	0.9607 ± 0.0091	0.9587 ± 0.0096	0.9507 ± 0.0108
50	0.9587 ± 0.0116	0.9598 ± 0.0110	0.9641 ± 0.0089	0.9619 ± 0.0097	0.9545 ± 0.0111
60	0.9624 ± 0.0111	0.9640 ± 0.0101	0.9661 ± 0.0091	0.9650 ± 0.0093	0.9565 ± 0.0113
70	0.9672 ± 0.0097	0.9691 ± 0.0090	0.9706 ± 0.0076	0.9699 ± 0.0082	0.9609 ± 0.0100
80	0.9720 ± 0.0104	0.9735 ± 0.0099	0.9749 ± 0.0081	0.9742 ± 0.0089	0.9675 ± 0.0104
90	0.9757 ± 0.0071	0.9768 ± 0.0065	0.9783 ± 0.0052	0.9775 ± 0.0057	0.9730 ± 0.0069
100	0.9783 ± 0.0074	0.9789 ± 0.0068	0.9804 ± 0.0055	0.9796 ± 0.0060	0.9739 ± 0.0072

TABLE 9. Mean and 95% confidence interval (CI) of Accuracy, Precision, Recall, F-measure, and NMI for UFSLRAGE on the Yale dataset over 30 independent runs

NumFeat	Accuracy (Mean ± CI)	Precision (Mean ± CI)	Recall (Mean ± CI)	F-measure (Mean ± CI)	NMI (Mean ± CI)
10	0.3354 ± 0.0232	0.3578 ± 0.0282	0.3640 ± 0.0244	0.3580 ± 0.0240	0.6010 ± 0.0142
20	0.3721 ± 0.0265	0.4020 ± 0.0372	0.3994 ± 0.0293	0.3984 ± 0.0319	0.5999 ± 0.0167
30	0.3966 ± 0.0212	0.4351 ± 0.0314	0.4305 ± 0.0232	0.4308 ± 0.0260	0.6136 ± 0.0135
40	0.4320 ± 0.0248	0.5051 ± 0.0354	0.4647 ± 0.0275	0.4828 ± 0.0300	0.6185 ± 0.0133
50	0.4388 ± 0.0264	0.5072 ± 0.0387	0.4656 ± 0.0271	0.4839 ± 0.0315	0.6210 ± 0.0172
60	0.4456 ± 0.0205	0.5141 ± 0.0381	0.4852 ± 0.0229	0.4967 ± 0.0291	0.6242 ± 0.0173
70	0.4571 ± 0.0258	0.5180 ± 0.0345	0.4893 ± 0.0259	0.5019 ± 0.0289	0.6323 ± 0.0145
80	0.4714 ± 0.0234	0.5381 ± 0.0365	0.5033 ± 0.0232	0.5179 ± 0.0282	0.6333 ± 0.0155
90	0.4673 ± 0.0244	0.5215 ± 0.0347	0.5045 ± 0.0268	0.5116 ± 0.0297	0.6367 ± 0.0171
100	0.4707 ± 0.0251	0.5279 ± 0.0346	0.5099 ± 0.0272	0.5174 ± 0.0299	0.6377 ± 0.0179

On the JAFFE dataset (Table 8), high performance is achieved even with few features, with Accuracy exceeding 0.97 when using all 100 features. Similar trends are observed on Yale (Table 9) and ORL (Table 10), where metrics

TABLE 10. Mean and 95% confidence interval (CI) of Accuracy, Precision, and NMI for UFSLRAGE on the ORL dataset over 30 independent runs

Num. of Features	Accuracy (Mean ± CI)	Precision (Mean ± CI)	NMI (Mean ± CI)
10	0.4242 ± 0.0212	0.4457 ± 0.0266	0.7460 ± 0.0069
20	0.5478 ± 0.0188	0.5842 ± 0.0251	0.7904 ± 0.0076
30	0.6050 ± 0.0197	0.6520 ± 0.0194	0.8121 ± 0.0093
40	0.6406 ± 0.0195	0.6826 ± 0.0179	0.8243 ± 0.0090
50	0.6558 ± 0.0182	0.6965 ± 0.0193	0.8386 ± 0.0086
60	0.6703 ± 0.0178	0.7127 ± 0.0199	0.8419 ± 0.0078
70	0.6786 ± 0.0162	0.7167 ± 0.0197	0.8462 ± 0.0069
80	0.6872 ± 0.0184	0.7281 ± 0.0199	0.8507 ± 0.0076
90	0.6961 ± 0.0191	0.7435 ± 0.0174	0.8515 ± 0.0085
100	0.7036 ± 0.0175	0.7483 ± 0.0171	0.8554 ± 0.0076

TABLE 11. Mean and 95% confidence interval (CI) of Accuracy, Precision, Recall, F-measure, and NMI for UFSLRAGE on the COIL-20 dataset over 30 independent runs

Num. of Features	Accuracy (Mean ± CI)	Precision (Mean ± CI)	Recall (Mean ± CI)	F-measure (Mean ± CI)	NMI (Mean ± CI)
10	0.8133 ± 0.0128	0.8151 ± 0.0148	0.8139 ± 0.0121	0.8145 ± 0.0133	0.8172 ± 0.0125
20	0.8948 ± 0.0093	0.9017 ± 0.0088	0.8957 ± 0.0088	0.8987 ± 0.0087	0.8994 ± 0.0090
30	0.9175 ± 0.0091	0.9246 ± 0.0087	0.9182 ± 0.0083	0.9214 ± 0.0085	0.9244 ± 0.0078
40	0.9318 ± 0.0053	0.9402 ± 0.0043	0.9322 ± 0.0048	0.9362 ± 0.0045	0.9370 ± 0.0050
50	0.9364 ± 0.0042	0.9454 ± 0.0037	0.9369 ± 0.0038	0.9411 ± 0.0036	0.9433 ± 0.0036
60	0.9424 ± 0.0047	0.9507 ± 0.0043	0.9425 ± 0.0045	0.9466 ± 0.0043	0.9484 ± 0.0037
70	0.9465 ± 0.0044	0.9553 ± 0.0038	0.9472 ± 0.0044	0.9512 ± 0.0040	0.9529 ± 0.0038
80	0.9474 ± 0.0041	0.9562 ± 0.0033	0.9480 ± 0.0041	0.9521 ± 0.0036	0.9533 ± 0.0034
90	0.9478 ± 0.0046	0.9573 ± 0.0034	0.9486 ± 0.0042	0.9529 ± 0.0038	0.9540 ± 0.0040
100	0.9483 ± 0.0036	0.9583 ± 0.0027	0.9486 ± 0.0034	0.9534 ± 0.0030	0.9549 ± 0.0029

TABLE 12. Mean and 95% confidence interval (CI) of Accuracy, Precision, Recall, F-measure, and NMI for UFSLRAGE on the pixraw10P dataset over 30 independent runs (with λ=1000)

NumFeat	Accuracy ± 95% CI	Precision ± 95% CI	Recall ± 95% CI	F-measure ± 95% CI	NMI ± 95% CI
10	0.9000 ± 0.0036	0.9083 ± 0.0036	0.8833 ± 0.0036	0.8957 ± 0.0036	0.8817 ± 0.0036
20	0.8333 ± 0.0036	0.8567 ± 0.0036	0.8167 ± 0.0036	0.8362 ± 0.0036	0.8343 ± 0.0036
30	0.9333 ± 0.0036	0.9500 ± 0.0036	0.9333 ± 0.0036	0.9416 ± 0.0036	0.9420 ± 0.0036
40	0.9333 ± 0.0036	0.9417 ± 0.0036	0.9333 ± 0.0036	0.9375 ± 0.0036	0.9410 ± 0.0036
50	0.9333 ± 0.0036	0.9417 ± 0.0036	0.9333 ± 0.0036	0.9375 ± 0.0036	0.9410 ± 0.0036
60	0.9333 ± 0.0036	0.9417 ± 0.0036	0.9333 ± 0.0036	0.9375 ± 0.0036	0.9410 ± 0.0036
70	0.9667 ± 0.0036	0.9750 ± 0.0036	0.9667 ± 0.0036	0.9708 ± 0.0036	0.9693 ± 0.0036
80	0.9667 ± 0.0036	0.9750 ± 0.0036	0.9667 ± 0.0036	0.9708 ± 0.0036	0.9693 ± 0.0036
90	0.9667 ± 0.0036	0.9750 ± 0.0036	0.9667 ± 0.0036	0.9708 ± 0.0036	0.9693 ± 0.0036
100	0.9667 ± 0.0036	0.9750 ± 0.0036	0.9667 ± 0.0036	0.9708 ± 0.0036	0.9693 ± 0.0036

steadily improve with more features. For ORL, Recall and F-measure are not reported in some cases due to undefined values when certain classes are not predicted in individual runs; only reliable metrics are included following common practice. On COIL-20 (Table 11), Accuracy exceeds 94% with 60 or more

features, with narrow CIs indicating stable performance. Table 12 reports the mean values and 95% confidence intervals (CI) of Accuracy, Precision, Recall, F-measure, and NMI obtained by the proposed UFSLRAGE algorithm on the pixraw10P dataset (10,000 features) over 30 independent runs. All metrics show a consistent improvement as the number of selected features increases, with Accuracy reaching approximately 0.93–0.97 for 70 or more features. The narrow confidence intervals across the evaluated range further highlight the stability and reliability of UFSLRAGE in high-dimensional settings, where many traditional methods struggle due to noise and redundancy.

5. Conclusion

This paper proposes UFSLRAGE, a novel filter-based unsupervised feature selection method that does not rely on statistical measures such as variance, which are often sensitive to noise and outliers. Instead, Low-Rank Adaptive Graph Embedding (LRAGE) is employed as an initial feature extraction step to adaptively update sample similarities and generate orthogonal, uncorrelated representations through reconstruction error minimization. This effectively mitigates noise, outliers, and feature correlations. Subsequent weighted bipartite graph matching using the LAPJV algorithm selects the original features most aligned with these robust representations, ensuring a compact and interpretable subset. Experimental results on five benchmark image datasets demonstrate that UFSLRAGE consistently outperforms eight established unsupervised feature selection methods across all evaluated metrics (Accuracy, Precision, Recall, F-measure, and NMI). The framework is flexible and can be extended to supervised settings by replacing LRAGE with methods such as LDA, or to 2D inputs using techniques like 2DLDA. The use of the LAPJV algorithm also contributes to efficient matching. Future work will focus on further optimizing computational efficiency.

6. Aknowledgement

I would like to thank the reviewers for their thoughtful comments and efforts towards improving my manuscript.

7. Data Availability Statement

The data generated during the study are subject to a data-sharing mandate and are available in public repositories. All datasets used in this study have been appropriately cited in the manuscript.

8. Ethical considerations

The author avoided data fabrication and falsification.

9. **Funding**

This research did not receive any specific grant from the public, commercial, or not-for-profit funding agencies.

10. **Conflict of interest**

The author declare no conflict of interest

Appendix A: Detailed LRAGE Optimization Steps

The LRAGE algorithm consists of three steps. The initial step involves assessing the fundamental correlation structure of the data through the application of a low-rank constraint. In the second step of the subspace learning process, the structure of preservation of the local manifold is essential. In the third step, the similarity connection among samples is adaptively updated using iterative update learning of two projection and similarity matrices simultaneously. The adaptive graph undergoes continuous refinement during the subspace learning procedure, and it is more suitable and accurate for describing the similarity association among dataset samples, particularly in the presence of noisy data.

In the graph $G = (V, E)$, the embedding is a mapping

$$f : v_i \mapsto y_i \in \mathbb{R}^d, \quad \forall i \in [n],$$

such that

$$d \leq |V|.$$

The function f maintains the values of the measures specified in the graph G [21]. Therefore, the embedding associates each node with a low-dimensional feature vector and attempts to maintain strong connections between vertices. Embedding is a representation that describes graph data [21].

The general framework of graph embedding is as follows: Initially, a graph representing the local neighborhood is formed, denoted as $G = \{X, W\}$, where the set of vertices X lies in the space $\mathbb{R}^{d\times n}$, and the similarity matrix W is in $\mathbb{R}^{n\times n}$ [20]. Subsequently, the procedure utilizes Equation (1) to obtain a low-dimensional representation through learning [21].

$$\min_{Z^T\Theta Z=\sigma} \sum_{i\neq j} \|Z_i - Z_j\|_2^2 W_{ij} \;=\; \min_{Z^T\Theta Z=\sigma} Z^T L Z \tag{1}$$

Where σ represents a fixed value, Θ represents a matrix representing constraints crafted to prevent an insignificant solution, $L = D - W$ denotes the Laplacian matrix, with D being a matrix with entries only along the main diagonal meeting the condition $D_{ii} = \sum_{ji} W_{ij}$. Z has low dimensionality can be derived through $Z = P^T X$ that the P is the projection matrix [10]. In a

classification work, conventional linear regression endeavors to acquire a matrix for regression $Q \in \mathbb{R}^{d\times\varphi}$ by minimization of equation [20]:

$$\min_{Q} \|L - X^T Q\|_F^2 \tag{2}$$

where$L = [L_1, L_2, \ldots, L_n] \in \mathbb{R}^{n\times\varphi}$ is a matrix for labeling and φ represents the number of classes, the objective function is minimized [20] . The LRAGE has an objective function, and two parameters, namely the size of the neighborhood k and the tuning parameter λ , need to be set. The similarity matrix W, which determines the local structure of the data, is dependent on the k nearest neighbors of each sample. The objective function of LRAGE is shown by equation (3) [20]:

$$\min_{P,R,W} \sum_{i,j} \left\|X_i^T - X_j^T PR\right\|_2^2 W_{ij} + \alpha\|W\|_F^2 + \lambda\|PR\|_{2,1} \tag{3}$$

s.t. $\forall i : W_i \geq 0, \quad \sum_j W_{ij} = 1, \quad P^T P = I$ Where α and λ denote two non-negative balanced parameters, and the constraint $W_i \geq 0$ is introduced to ensure the probability characteristics of W_i and $\sum_j W_{ij} = 1$ is incorporated to prevent scenarios where every element in each row of matrix W is equal to zero. P denotes a projection matrix and R denotes a regression matrix. The term $\alpha\|W\|_F^2$ is included to discourage trivial solutions and promote a prior assumption of uniform distribution. W is an adaptive similarity matrix that represents the likelihood of similarity among sample points $X_i$and X_j derived from the minimization of the reconstruction error. Due to the presence of three variables in equation (3), simultaneous optimization poses a challenge. To address this issue, an iterative procedure is developed that updates one variable at a time while keeping the remaining variables constant. In line with the definition of the $\ell_{2,1}$-norm, they begin by establishing a diagonal matrix labeled U. Diagonal matrix U is obtained by equation (4) [20]:

$$U_{ii} = \frac{1}{2\|(PR)_i\|_2 + \varsigma} \tag{4}$$

In equation (4), $(PR)_i$ denotes the i-th row, and U_{ii} represents the i-th diagonal element of the matrix U. The symbol ς is utilized as an infinitesimally small constant to mitigate the occurrence of situations where the denominator reaches zero. As the matrix U is intricately linked to both the regression matrix R and projection matrix P, Formula (4) can be employed to iteratively update $U^{(k+1)}$ once $R^{(k+1)}$ and $P^{(k+1)}$ are determined. The objective function (3) can then be rewritten in the following matrix format:

$$\min_{P,R,W} \operatorname{tr}\left(X\psi X^T - 2XWX^T PR + R^T\left(P^T X\psi X^T P + \lambda P^T UP\right)R\right) + \alpha\|W\|_F^2 \tag{5}$$

Where$\psi_{ii} = \sum_{j=1}^{n} W_{ij}$. The output of solving equation (5) includes three matrices: matrix W, projection matrix P, and regression matrix R. The goal is to find the projection matrix P so that, using Eigen decomposition and

extract eigenvectors and eigenvalues for projecting into a new space. The best solution for $P^{(k+1)}$ can be derived through the resolution of the problem of maximizing the trace ratio [20]:

$$\max_{P^TP=I} \text{tr} \left[\left(P^{(k+1)T} \left(X\psi X^T + \lambda U^{(k)} \right) P^{(k+1)} \right)^{-1} \left(P^{(k+1)T} X W^{(k)T} X^T X W^{(k)} X^T P^{(k+1)} \right) \right] \tag{6}$$

The problem of maximizing the trace ratio (6) can be accomplished through Eigen-decomposition [20]:

$$\left(X\psi X^T + \lambda U^{(k)} \right)^{-1} \left(X W^{(k)T} X^T X W^{(k)} X^T \right) P^{(k+1)} = P^{(k+1)} \tilde{\Lambda} \tag{7}$$

Where $\tilde{\Lambda}$ signifies the eigenvalue matrix. This paper used the best solution for $P^{(k+1)}$ that involves all eigenvectors associated with all eigenvalues in the $(k+1)$-th iteration [20].

$$\psi_{ii} = \sum_{j=1}^{n} W_{ij} \tag{8}$$

The noteworthy point is the orthogonality of the projection matrix resulting from the objective function equation, which has the following properties:

$$P^T P = I \tag{9}$$

where P is a square matrix with dimensions $n \times n$. P^T is the transpose of the matrix P, which is obtained by swapping the rows and columns of P. I is the identity matrix, which is a square $n \times n$ matrix with 1s on the main diagonal and 0s elsewhere. Such a matrix P is called an orthogonal matrix. We transform the initial features(X) into a new space where essential information and key data structures are preserved using the P matrix of the LRAGE algorithm. This transformation is defined as:

$$Y = X^T P \tag{10}$$

In this equation, Y represents the new set of features that, after the transformation, retain the key information and important structures of the data. It is essential to emphasize that we utilize all the extracted features without performing any dimensionality reduction. Algorithm 1 provides an overview of the LRAGE algorithm.

Appendix B: Sensitivity Analysis

This appendix presents a concise sensitivity analysis of the proposed UFSLRAGE method with respect to two key hyperparameters: the neighborhood size k used in graph construction and the regularization parameter λ in the LRAGE model. The analysis was conducted on one representative dataset

(ORL). The number of selected features was fixed at 50, and all results were averaged over 30 independent runs.

Table 13 summarizes the classification accuracy trends under different values of k and λ, providing insight into the robustness of the proposed method with respect to hyperparameter variations. As shown in Table 13, the proposed method exhibits stable performance for k values in the range of 5 to 7, while a slight performance degradation is observed for very small or very large neighborhood sizes. Similarly, the accuracy remains relatively stable across different values of λ, with the best performance achieved at $\lambda = 10^4$. Overall, these results demonstrate that the proposed method is not overly sensitive to the choice of hyperparameters and maintains consistent performance within a reasonable parameter range.

Table 13. Sensitivity Analysis on ORL Dataset

Parameter	**Value**	**Mean Accuracy**
k ($\lambda = 10^5$)	3	0.5583
k ($\lambda = 10^5$)	5	0.6167
k ($\lambda = 10^5$)	7	0.6167
k ($\lambda = 10^5$)	9	0.5667
λ ($k = 5$)	10^4	**0.6667**
λ ($k = 5$)	10^5	0.6167
λ ($k = 5$)	10^6	0.5917

References

[1] Miao, J., Zhao, J., Yang, T., Fan, C., Tian, Y., Shi, Y., & Xu, M. (2024). Explicit unsupervised feature selection based on structured graph and locally linear embedding. *Expert Systems with Applications*, 255(June). https://doi.org/10.1016/j.eswa.2024.124568.

[2] Rossi, R., Murari, A., & Gelfusa, M. (2025). A Deep Learning Framework for Feature Selection and Dimensional Analysis: Variational Explainable Neural Networks. *Knowledge-Based Systems*. Published online 6 June 2025. https://doi.org/10.1016/j.knosys.2025.113940.

[3] Liang, S., Zhang, Y., Zheng, K., & Bai, Y. (2025). FeatureX: An explainable feature selection for deep learning. *Expert Systems with Applications*. Published online 21 April 2025. https://doi.org/10.1016/j.eswa.2025.127675.

[4] Moslemi, A., Bidar, M., & Ahmadian, A. (2023). Subspace learning using structure learning and non-convex regularization: Hybrid technique with mushroom reproduction optimization in gene selection. *Computers in Biology and Medicine*, 164, 107309. https://doi.org/10.1016/j.compbiomed.2023.107309.

[5] Yan, J., Hu, G., & Wei, G. (2025). Unsupervised feature selection via latent feature representation and modified graph embedding. *Engineering Applications of Artificial Intelligence*, 161, 112121. Published online 1 December 2025. https://doi.org/10.1016/j.engappai.2025.112121.

[6] Qin, Z., Chen, H., Mi, Y., Luo, C., Horng, S., & Li, T. (2024). Multi-label Feature selection with adaptive graph learning and label information enhancement. *Knowledge-Based Systems*, 285, 111363. https://doi.org/10.1016/j.knosys.2023.111363.

[7] Zuo, X., Zhang, W., Wang, X., Dang, L., Qiao, B., & Wang, Y. (2025). Unsupervised feature selection via maximum relevance and minimum global redundancy. *Pattern Recognition*, 164, 111483. Published online 1 August 2025. https://doi.org/10.1016/j.patrec.2025.111483.

[8] Han, K., Wang, Y., Zhang, C., Li, C., & Xu, C. (2018). Autoencoder Inspired Unsupervised Feature Selection. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* (pp. 2941–2945). https://doi.org/10.1109/ICASSP.2018.8462261.

[9] Guo, J., & Zhu, W. (2018). Dependence guided unsupervised feature selection. In *32nd AAAI Conference on Artificial Intelligence, AAAI 2018* (pp. 2232–2239).

[10] Zhu, P., Zuo, W., Zhang, L., Hu, Q., & Shiu, S. C. K. (2015). Unsupervised feature selection by regularized self-representation. *Pattern Recognition, 48*(2), 438–446. https://doi.org/10.1016/j.patcog.2014.08.006.

[11] Huang, D., Cai, X., & Wang, C. D. (2019). Unsupervised feature selection with multi-subspace randomization and collaboration. *Knowledge-Based Systems, 182*, 1–?. https://doi.org/10.1016/j.knosys.2019.07.027.

[12] Li, X., Zhang, H., Zhang, R., Liu, Y., & Nie, F. (2019). Generalized Uncorrelated Regression with Adaptive Graph for Unsupervised Feature Selection. *IEEE Transactions on Neural Networks and Learning Systems, 30*(5), 1587–1595. https://doi.org/10.1109/TNNLS.2018.2868847.

[13] Xie, J., Wang, M., Xu, S., Huang, Z., & Grant, P. W. (2021). The Unsupervised Feature Selection Algorithms Based on Standard Deviation and Cosine Similarity for Genomic Data Analysis. *Frontiers in Genetics, 12*, 1–17. https://doi.org/10.3389/fgene.2021.684100.

[14] Paniri, M., Dowlatshahi, M. B., & Nezamabadi-pour, H. (2020). MLACO: A multi-label feature selection algorithm based on ant colony optimization. *Knowledge-Based Systems, 192*, 105285. https://doi.org/10.1016/j.knosys.2019.105285.

[15] Beiranvand, F., & Mehrdad, V. (2026). Unsupervised feature selection based on the two-dimensional principal component analysis and bipartite graph for face image classification. *Journal of Mahani Mathematical Research, 15*(1), 1–37.

[16] Beiranvand, F. (2026). Unsupervised feature selection using orthogonal locality preserving projections and bipartite graph matching for face image classification. *Journal of Mahani Mathematical Research, 15*(2), 65–104.

[17] Hashemi, A., Dowlatshahi, M. B., & Nezamabadi-Pour, H. (2021). A bipartite matching-based feature selection for multi-label learning. *International Journal of Machine Learning and Cybernetics, 12*(2), 459–475. https://doi.org/10.1007/s13042-020-01180-w.

[18] Beiranvand, F., Mehrdad, V., & Dowlatshahi, M. B. (2022). Unsupervised feature selection for image classification: A bipartite matching-based principal component analysis approach. *Knowledge-Based Systems, 250*, 109085. https://doi.org/10.1016/j.knosys.2022.109085.

[19] Wahid, A., Khan, D. M., Hussain, I., Khan, S. A., & Khan, Z. (2022). Unsupervised feature selection with robust data reconstruction (UFS-RDR) and outlier detection. *Expert Systems with Applications, 201*, 117008. https://doi.org/10.1016/j.eswa.2022.117008.

[20] Hu, Q. (2020). Low-Rank Adaptive Graph Embedding for Unsupervised Feature Extraction. https://doi.org/10.1016/j.patcog.2020.107758.

[21] Goyal, P., & Ferrara, E. (2017). Graph Embedding Techniques, Applications, and Performance: A Survey.

[22] Jones, W., Chawdhary, A., & King, A. (2016). Optimising the Volgenant–Jonker algorithm for approximating graph edit distance. R, 0, 1–8. https://doi.org/10.1016/j.patrec.2016.07.024.

[23] Xia, P., Zhang, L., & Li, F. (2015). Learning similarity with cosine similarity ensemble. *Information Sciences, 307*, 39–52. https://doi.org/10.1016/j.ins.2015.02.024.

[24] Zhang, N., Xu, Y., Zhu, Q., & He, Y. (2022). Improved Locality Preserving Projections Based on Heat-Kernel and Cosine Weights for Fault Classification in Complex Industrial Processes. *IEEE Transactions on Reliability, PP*, 1–10. https://doi.org/10.1109/TR.2021.3139539.

[25] Goyal, P., & Ferrara, E. (2017). Graph Embedding Techniques, Applications, and Performance: A Survey.

[26] Armiti, A., & Gertz, M. (n.d.). Geometric Graph Matching and Similarity: A Probabilistic Approach. Categories and Subject Descriptors.

[27] Toroslu, I. H., & Arslanoglu, Y. (2007). Genetic algorithm for the personnel assignment problem with multiple objectives. *Information Sciences, 177*, 787–803. https://doi.org/10.1016/j.ins.2006.07.032.

[28] Erciyes, K., & Erciyes, K. (2018). Matching. In *Guide to Graph Algorithms: Sequential, Parallel and Distributed* (pp. 263–303).

[29] Wang, A., An, N., Chen, G., Li, L., & Alterovitz, G. (2015). Accelerating wrapper-based feature selection with K-nearest-neighbor. *Knowledge-Based Systems, 83*(1), 81–91. https://doi.org/10.1016/j.knosys.2015.03.009.

[30] Terven, J., Cordova-Esparza, D. M., Ramirez-Pedraza, A., & Chavez-Urbiola, E. A. (2023). Loss functions and metrics in deep learning: A review. arXiv preprint arXiv:2307.02694.

Firoozeh Beiranvand
Orcid number: 0000-0002-1908-8386
Department of Electrical Engineering
Lorestan University
Khoramabad, Iran
Email address: beiranvand.fi@fe.lu.ac.ir