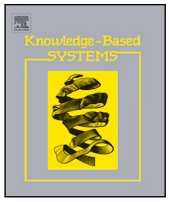
Knowledge-Based Systems 302 (2024) 112379

Contents lists available at ScienceDirect

Knowledge-Based Systems

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



Unsupervised feature selection based on bipartite graph and low-redundant regularization

Longyan Xiang [a,b,c,d], Hongmei Chen [a,b,c,d,*], Tengyu Yin [a,b,c,d], Shi-Jinn Horng [e,f], Tianrui Li [a,b,c,d]

[a] School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, 611756, China
[b] National Engineering Laboratory of Integrated Transportation Big Data Application Technology, Southwest Jiaotong University, Chengdu, 611756, China
[c] Engineering Research Center of Sustainable Urban Intelligent Transportation, Ministry of Education, Chengdu 611756, PR China
[d] Manufacturing Industry Chains Collaboration and Information Support Technology Key Laboratory of Sichuan Province, Southwest Jiaotong University, Chengdu 611756, PR China
[e] Department of Computer Science and Information Engineering, Asia University, Taichung 41354, Taiwan
[f] Department of Medical Research, China Medical University Hospital, China Medical University, Taichung 404327, Taiwan

ARTICLE INFO

Keywords:
Unsupervised feature selection
Bipartite graph learning
$l_{2,0}$ -norm constraint
Low-redundant regularization

ABSTRACT

Unsupervised feature selection (UFS) has attracted increasing attention because of the difficulty and high cost of obtaining data labels. Since the ignorance of redundancy between features and the use of a fixed similarity matrix, current UFS algorithms do not perform well. Moreover, because of the high computational complexity, existing UFS algorithms usually cannot handle large-scale data. A novel UFS method is proposed to solve these problems through bipartite graph learning with low-redundant regularization (BGLR). Firstly, BGLR uses a variance-based de-correlation anchor selection method to ensure that selected anchors can be evenly distributed across classes. Secondly, BGLR builds an adaptive bipartite graph by using selected anchors and original data in the projection space. $L_{2,0}$-norm constraint is applied to the projection matrix. Thus, the feature subset can be obtained directly. Finally, a regularization term is applied to control the redundancy of selected features. Moreover, BGLR is solved by an iterative method, which can converge quickly. Experiments are performed on public datasets in comparison with some relevant methods. The experimental results demonstrate the BGLR's effectiveness.

1. Introduction

With the rapid development of information technology, plenty of high-dimensional data exists in numerous research areas, including machine learning [1], medical images [2], genetic analysis [3], computer vision [4], etc. High-dimensional data can provide abundant information but increases the learning algorithms' computational burden and memory requirements [5]. Additionally, the noise in high-dimensional data will further reduce the algorithm's performance. Therefore, dimensionality reduction is crucial.

Dimensionality reduction algorithms are classified as feature extraction and feature selection according to how feature space is constructed [6]. The representative or optimal subset is chosen from original data [7] in feature selection, which can retain the semantics of the original feature. Whereas feature extraction transforms and compresses original features to map data into a low-dimensional space, which cannot retain the features' physical meaning [8]. Three categories: unsupervised feature selection(UFS), supervised feature selection, and semi-supervised feature selection [9] are classified in feature selection according the use of labels. Under the circumstances that a label is not available, UFS can effectively select discriminative features. Since obtaining labels is difficult and costly, UFS has attracted more and more attention.

UFS is split into three types: filter, embedded, and wrapper [10]. Filter methods evaluate each feature, filtering out unimportant features and retaining discriminative features. He et al. proposed to score the locality preservation ability of individual features by Laplacian score [11], which ignores whether the combination of features with a high score can represent the original data well. In comparison, wrapper approaches design learning algorithms to guarantee the discriminability of the selected feature subset [12]. However, it requires high computational costs since learning algorithms have to be trained repeatedly. Embedded approaches combine feature selection and model optimization into an objective function [13]. Thus, feature subsets can

* Corresponding author.
E-mail addresses: longyqnxiang@my.swjtu.edu.cn (L. Xiang), hmchen@swjtu.edu.cn (H. Chen), Tengyu@my.swjtu.edu.cn (T. Yin), horngsj@asia.edu.tw (S.-J. Horng), trli@swjtu.edu.cn (T. Li).

https://doi.org/10.1016/j.knosys.2024.112379
Received 18 February 2024; Received in revised form 24 June 2024; Accepted 11 August 2024
Available online 16 August 2024

be selected during the optimization process. This paper focuses on embedded UFS methods.

Existing UFS methods are generally considered to reduce feature redundancy or maintain the data's intrinsic characteristics. The first type usually assumes that features can be reconstructed by the linear combinations of their related features. That is, the features themselves can be linearly reconstructed. Miao et al. proposed an approach based on self-representation [14]. However, this approach performs poorly when the features are not linearly related. Huang et al. applied mutual information to evaluate the dependencies between features to eliminate redundant features [15]. Wang et al. realized that a sparse mapping matrix may result in data distribution distortion. Then, they proposed an exponential weighting mechanism to explore the data structure in feature space (LLSRFS) [16]. LLSRFS took more minor yet valuable features into account, which can better preserve data structure information. The above methods require ranking features and selecting the feature with the top rank. However, the combination of these features is not always the optimal subset.

The second type mainly considers preserving the data structure [17]. Since local manifold structures can more accurately characterize data than global structures [18], many embedded UFS methods focus on preserving the local manifold structure. Graph-based UFS methods can utilize the local structure well. Current graph-based UFS approaches mainly include two processes: (1) similarity matrix construction based on the distance between samples and (2) feature selection. Similar matrix construction and feature selection are carried out in two stages. Therefore, if the predefined similarity matrix remains fixed, the performance of the UFS method will highly rely on the similarity matrix's quality [19]. However, the similarity matrix constructed by the original data suffers from noise, which will degrade the model's performance. Some existing UFS methods can select features and construct a similarity matrix simultaneously. However, these methods generally use $l_{2,1}$-norm regularization or $l_{2,p}$-norm regularization for feature selection [20]. Both $l_{2,1}$-norm regularization and $l_{2,p}$-norm regularization need to introduce regularization parameters, but the sparsity effect is not so obvious. Moreover, these two regularization methods need to rank features and choose features with high scores, which ignore the dependency between features. Furthermore, the computational complexity of constructing the similarity matrix will rapidly increase when there are many samples [21]. Many machine learning methods introduce anchor techniques to reduce computational complexity and improve efficiency [22].

Current anchor-based methods typically obtain anchors through k-means clustering or random sampling [23]. Although these methods are time-saving, the quality of selected anchors cannot be guaranteed. Choosing samples randomly from all data points as anchors or using the clustering centers generated by k-means as anchors always result in randomness. The performance of corresponding algorithms will also be unstable. Zhang et al. took the data feature or pairwise relation into account and then designed an initialization-independent anchor selection strategy(ALG) [24]. However, ALG measures the pairwise relation between samples by their similarity. That is, ALG needs to learn a large similarity matrix first. This approach is time-consuming and cannot guarantee that the selected anchors contain enough information. Multiple repetitions of k-means clustering can lower the randomness caused by the original centroids. Wang et al. employed a binary tree manner to perform k-means (BKHK) which can accelerate the process of anchor selection and improve the quality of selected anchors [16]. BKHK can preserve the balanced structure of the data to some extent. The above two methods reduce the randomness associated with selected anchors. Zheng et al. proposed a dimensionality reduction method called Joint Structure Bipartite Graph Projection (JSBGP) for industrial process monitoring [25]. JSBGP improves the K-means centroid initialization method to select anchors with stability. Since the digital twin server cluster's load-balancing algorithm significantly impacts the computing nodes' computing performance, Tang et al. proposed an optimal complete matching of a weighted bipartite graph to allocate computing task blocks evenly [26]. Aigner-Horev et al. employed envy-free matchings in bipartite graphs to handle fair division problems with continuous or discrete resources. [27]. Nevertheless, they ignore that selected anchors must be able to provide sufficient information. A bipartite graph is constructed by all samples and selected anchors. That is, there some samples are missing. If the selected anchors cannot contain enough information to represent the original data, the quality of the bipartite graph will be significantly reduced. Moreover, most methods did not realize that selected anchors should reflect the intrinsic structure of the original data.

The differences among the algorithms above are compared in Table 1. As shown in Table 1, existing UFS methods pay little attention to the redundancy of the selected feature subset. Selecting the top-ranked features cannot guarantee that the selected feature subset is low-redundant. In addition, methods based on bipartite graphs always use k-means clustering or random sampling to select anchors for saving time. They ignored the anchor quality issue. If selected anchors cannot provide enough information, the quality of the bipartite graph will be affected. Motivated by the issues of anchor quality, redundancy of the selected feature subset, and feature ranking in the methods above, we propose a low-redundant UFS method based on bipartite graph learning (BGLR).

The general procedures of BGLR are shown in Fig. 1. BGLR selects anchors according to the variance and correlation of samples to ensure the quality of the selected anchors. Different from typical bipartite learning methods, BGLR builds an adaptive bipartite graph in the projection space to lower the effect of noise and redundant information. Additionally, $l_{2,0}$-norm constraint and low-redundant regularizer are incorporated to guarantee the sparsity of W. Thus, we can obtain an optimal feature subset. In a word, the main contributions of BGLR are as follows.

(i) Different from other methods that use k-means or random sampling to select anchors, BGLR selects anchors from the original data according to the variance of samples and correlation between samples. Thus, selected anchors can preserve the balanced structure of the original data and provide sufficient information. We demonstrate the anchor selection process in a real application scenario in Fig. 2.

(ii) BGLR demonstrates adaptability by constructing a bipartite graph in the projection space to mitigate noise and redundant features. Notably, BGLR performs feature selection and similarity matrix updates concurrently, allowing the bipartite graph to be updated adaptively.

(iii) The Projection matrix is updated by incorporating a similarity matrix under the $l_{2,0}$-norm constraint, which realizes the group feature selection. Additionally, a low-redundant regularization term is applied to eliminate redundant features.

(iv) Experimental results on eight benchmark datasets verify BGLR's effectiveness.

In real applications, using meaningful features to build an effective learning model is crucial, as it aims to capture the data's intrinsic characteristics, lower the noise effect, and improve learning efficiency. In the proposed method BGLR, the significance of data and features are considered simultaneously, which can be applied to different scenarios in the real world, e.g., river pollution analysis, text processing, and fault diagnosis. Fig. 2 illustrates the river pollution analysis procedure using BGLR.

River pollution impacts people's lives [32]. Currently, the main types of pollutants are analyzed as chemicals, heavy metals, and oils [33]. Identifying the types of pollutants is crucial for better managing contaminated rivers [34]. It is difficult to effectively identify the pollution source if the number of rivers is polluted and the contaminated areas are large. The number of samples is large due to the need

Table 1
The differences between BGLR and related methods.

Methods	Dynamic graph	Sparse regularization	Feature redundancy	Bipartite graph	Anchor selection method
RSOGFS [19]	✔	✔	✘	✘	/
VCSDFS [28]	✘	✔	✘	✘	/
BGCFS [29]	✔	✔	✘	✔	K-means
FOLPPFS-R [23]	✘	✔	✘	✔	Random sampling
FSDK [30]	✘	✔	✘	✘	/
DUFS [31]	✘	✔	✔	✘	/
SFESA	✔	✔	✘	✔	Balanced k-means
OBCLSP	✔	✔	✘	✘	/
BGLR	✔	✔	✔	✔	Based on variance and correlation of samples

✔ Indicates that the method uses the technique.
✘ Indicates that the technique is not used.

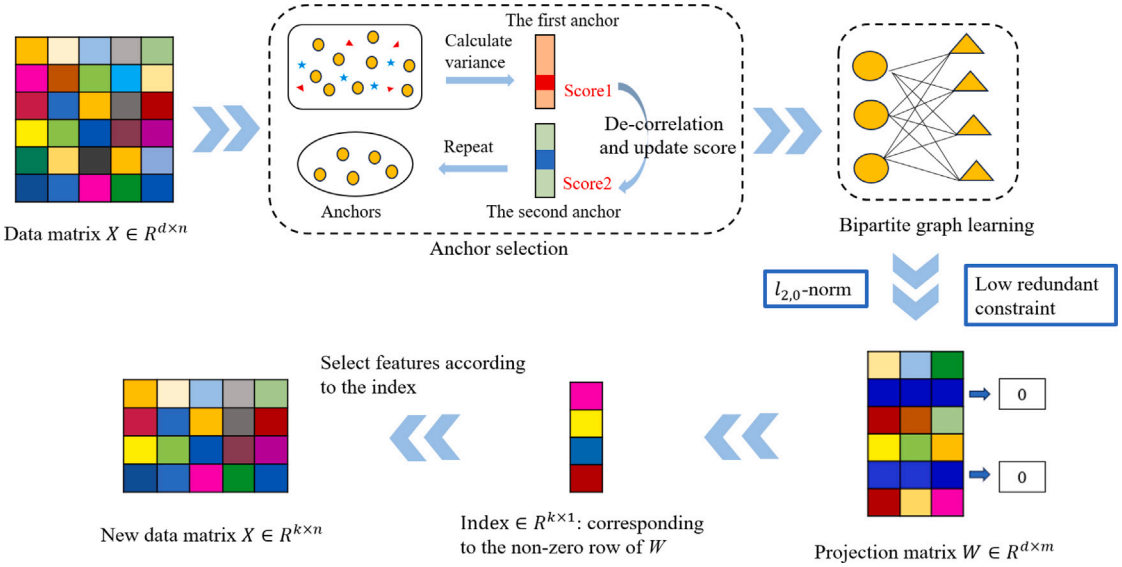


Fig. 1. The process of anchor selection in real application scenario. Selected anchors can provide sufficient information for feature selection.

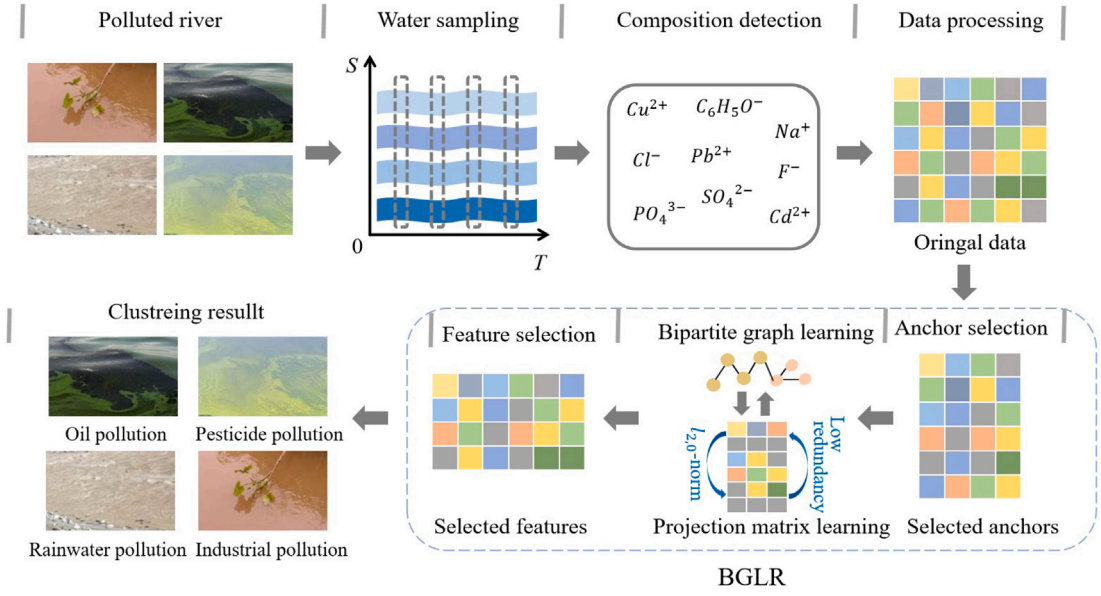


Fig. 2. Scenario of BGLR use in the real world. T represents different time and S represents the different section of the river.

to collect water samples from multiple river reaches at different times. Additionally, the composition of the polluted water source is complex, resulting in high feature dimensions in the samples. Therefore, using the proposed BGLR to reduce the dimension of features is essential for identifying the types of pollutants. As shown in Fig. 2, samples are taken at different polluted river sections, followed by compositional analysis and data processing. BGLR is used to analyze the composition of water samples in the presence of multiple water samples.

By eliminating highly repetitive components and retaining the main pollutant components, the types of water pollutants can be determined effectively. In this way, we can determine the cause of pollution and treat the polluted river better.

2. Related work

Feature learning is important since data with high dimension is ubiquitous. Du et al. proposed a few-shot class-incremental learning method called Branch Fusion Strategy based Network Intrusion Detection (BFS-NID) [35]. BFS-NID employs a feature extractor to learn better feature representations. Jia et al. proposed a novel attack detection framework called ACAM [36]. ACAM includes a knowledge extraction module which extracts the latest cyber security knowledge from original data to lower computational pressure.

In recent years, UFS has gained interest due to the increasing abundance of unlabeled data in real-world scenarios. These data often contain noise, which can significantly impact the effectiveness of data analysis. To address this, Nie et al. proposed a structured optimal graph feature selection (SOGFS) method [18], which adaptively updates the similarity matrix. Zhu et al. introduced an unsupervised spectral feature selection method via dynamic hyper-graph learning (DHLFS) [37]. The above two methods use $l_{2,1}$-norm regularization or $l_{2,p}$-norm regularization to sparsify the feature selection matrix. However, these methods rank the features, which may only sometimes result in the sub-optimal subset due to the ignorance of the correlation between features. Nie et al. designed a novel UFS approach with an optimization graph and $l_{2,0}$-norm constraints (RSOGFS) to obtain the optimal feature subset directly [19]. Zhu et al. also designed a UFS algorithm based on graph learning and $l_{2,0}$-norm sparse constraints (GLUFS) [38], which realizes group feature selection. However, the computational complexity of constructing the similarity matrix is $O(n^2)$, making these methods unsuitable for processing large-scale data.

Anchor technology has been proposed to address the issue of computational complexity effectively. It reduces the complexity of constructing the similarity matrix to O(nh), where h denotes the number of anchors. Chen et al. proposed a fast UFS method with a bipartite graph and row sparsity constraints (BGCFS) [29], which constructs a bipartite graph in the projection space. However, BGCFS uses the k-means method to select anchors, which can be influenced by the initial clustering centers and may require multiple repetitions to reduce randomness. In order to accelerate the process of anchor selection, Wang et al. designed a binary tree manner to perform k-means (BKHK) [39]. Anchors selected by BKHK can preserve the balanced structure of the data to a certain extent. To avoid the long running time caused by k-means clustering, Zhu et al. adopted random sampling to select anchors (FOLPPFS-R) [23]. However, since the quality of anchors is not stable, FOLPPFS-R does not always perform well. To eliminate randomness caused by selected anchors, Xia et al. designed an anchor selection strategy based on the variance of each sample and the correlation between samples [40]. It ensures that the selected anchors are evenly distributed across each category and represent a rough approximation of the information contained in all data points. Zhang et al. proposed an initialization-independent anchor selection strategy that considers the intrinsic properties of original instances. It ensures that categories with more samples have multiple corresponding anchor points, resulting in a more even distribution across the data space [24]. Yang et al. proposed a fast spectral clustering method via adaptive bipartite graph learning (FSBGL) [41], which updates the bipartite graph and obtains clustering results simultaneously, improving clustering accuracy. Recently, it has been found that maximal bicliques include meaningful information hidden in bipartite graphs [42]. Chen et al. designed two exact algorithms dedicated to small dense and large sparse bipartite graphs to mine the bipartite graph effectively [43].

Since redundant features will further reduce the algorithms' performance, several methods considering feature relevance have been proposed. Huang et al. proposed a novel UFS approach through adaptive graph learning and dependency score (AGDS) [15]. AGDS uses mutual information and entropy to evaluate the reliance between features and eliminate feature redundancy. Li et al. integrated self-paced learning and low-redundant regularization into a framework (SPLR). SPLR computes the pairwise similarity of features to lower the redundancy of selected feature subsets [44]. Since mapping data into a low space may result in data distribution distortion, Wang et al. proposed an exponential weighting mechanism to preserve the feature structure of original data (LLSRFS) [16]. LLSRFS considers the correlation between features, which can avoid ignoring smaller yet valuable features.

Table 2
Relevant notations.

Symbols	Description
$X \in R^{d\times n}$	The original data matrix
$x_i \in R^{d\times 1}$	The ith column of X
$x^j \in R^{1\times n}$	The jth row of X
$A \in R^{d\times h}$	The anchor matrix
$W \in R^{d\times m}$	The projection matrix
$E \in R^{n\times h}$	The similarity matrix
$S \in R^{(n+h)\times(n+h)}$	The affinity matrix of relevant bipartite graph
$L_S \in R^{(n+h)\times(n+h)}$	The Laplacian matrix of S
$D \in R^{(n+h)\times(n+h)}$	The degree matrix of S
$\check{X} \in R^{d\times n}$	The normalized data matrix
$\mathbf{1}_{d\times d} \in R^{d\times d}$	A matrix with all elements 1

3. Notations

For any matrix $A \in R^{m\times n}$, a^i means the ith row of A, a_j means the jth column of A, and the ith row jth element of A is represented by a_{ij}. I is the identity matrix. Let A^T denote the transpose of A. trace(A), A^{-1}, and A^+ denote its trace, inverse, and Penrose inverse, respectively. Additionally, $\mathbf{1}_n \in R^n$ denotes a column vector with all elements 1. More details are shown in Table 2.

4. The proposed method

This section shows the main strategies of the proposed BGLR.

4.1. Anchor selection

Assume data matrix $X = [x_1, x_2, \ldots, x_n] \in R^{d\times n}$ has d features and n samples. h anchors are selected from the original data to form an anchor matrix $A = [a_1, a_2, \ldots, a_h] \in R^{d\times h}$. The adjacency matrix is constructed based on these anchors and original instances. The adjacency matrix significantly impacts the algorithm's performance.

The sample with a large variance contains much information. Selecting anchors according to the variance of samples and correlation between samples can reduce the impact of missing samples [40]. In this way, the selected anchors can provide sufficient information and preserve the balanced structure of the original data. We select the sample with the largest variance as the initial anchor to guarantee that the selected anchors contain enough information. In order to properly maintain the data's internal structure while characterizing all the data, the sample's variance is updated according to the correlation between the selected anchors and the original sample. The initial variance of the sample is calculated as

$$\theta_i = \frac{var\left(x_i\right)}{max(var\left(x_i\right))}(i \in \{1, \ldots, n\}). \tag{1}$$

$\Theta = [\theta_1, \theta_2, \ldots, \theta_i, \ldots, \theta_n] \in R^n$ denotes the variance vector, in which θ_i are the normalized variance value. We choose the instance with the largest variance as the initial anchor. The index of the maximum value of θ_i is

$$index = \arg max_i \theta_i. \tag{2}$$

The initial anchor point obtained based on the maximum variance is $a_1 = X_{index}$. There may be a high correlation between samples. Anchor point selection based on the original variance may result in these remarkably similar anchor points, which cannot provide sufficient information and are concentrated in specific categories. To maximize the information the anchor points provide, consider updating the variance by calculating the correlation between instances. Let θ_{index} denote the variance of the anchor points selected in the latest round, and the correlation coefficient δ_i between θ_i and θ_{index} is calculated as

$$\delta_i = \frac{1}{1 + \|\theta_{index} - \theta_i\|^2}, \tag{3}$$

where $\delta = [\delta_1, \delta_2, \ldots, \delta_n] \in R^n$ is the correlation coefficient vector. δ is used to update the variance of the sample.

$$\theta_i = \theta_i \times (1 - \delta_i) \tag{4}$$

The variance is updated based on the correlation coefficient between samples. Thus, the probability that anchor points are selected from the same category as the previous anchors will be reduced in the next iteration. This way, the anchor points will be evenly distributed in different categories, preserving the internal balance data structure. Moreover, the selected anchors' variance will be updated to 0. That is, the data points would not be repeatedly selected as anchors. $h-1$ round iterations can obtain the required h anchors based on the initial anchors.

4.2. Bipartite graph construction

Let $X = [x_1, x_2, \ldots, x_n] \in R^{d\times n}$ represent the original data. h samples are selected to form the anchor matrix $A = [a_1, a_2, \ldots, a_n] \in R^{d\times h}$ by the above variance-based decorrelation anchor selection method. We can obtain the similarity matrix $E \in R^{n\times h}$ by solving the following equation.

$$\min_{e^i \mathbf{1}_h = 1, e_{ij} \geq 0} \sum_{i=1}^{n}\sum_{j=1}^{h} \|x_i - a_j\|_2^2 e_{ij} \tag{5}$$

In Eq. (5), e_{ij} denotes the probability that x_i is a neighbor of anchor a_j. The trivial solutions may be obtained if x_i has a similarity of 1 with the nearest anchor and 0 with the remaining anchors. Therefore, the information entropy $e_{ij}loge_{ij}$ balances the similarity matrix E to avoid trivial solutions. Then, the formulation is

$$\min_{e^i \mathbf{1}_h = 1, e_{ij} \geq 0} \sum_{i=1}^{n}\sum_{j=1}^{h} \|x_i - a_j\|_2^2 e_{ij} + \alpha \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij}loge_{ij}, \tag{6}$$

where α is the regularization parameter ($\alpha > 0$). To lower the impact of noise and highly similar features, we map original data into a low-dimensional space via linear transformation $W^T X$, where $W \in R^{d\times m}$ is the projection matrix, m is the dimension of subspace. By plugging $W^T X$ into Eq. (6), we can obtain

$$\begin{aligned} &\min_{E,W} \sum_{i=1}^{n}\sum_{j=1}^{h} \|W^T x_i - W^T a_j\|_2^2 e_{ij} + \alpha \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij}loge_{ij} \\ &s.t.\ e^i \mathbf{1}_h = 1, e_{ij} \geq 0, W^T W = I. \end{aligned} \tag{7}$$

The constraint $W^T W = I$ is applied to construct a low-dimensional space with uncorrelated dimensions. The full connection matrix of the bipartite graph is $S = \begin{bmatrix} 0 & E \\ E^T & 0 \end{bmatrix} \in R^{(n+h)\times(n+h)}$.

4.3. Sparse feature selection with bipartite graph

Most UFS methods use $l_{2,1}$-norm as the sparsity term. These methods score a single feature and select the feature with a high score. However, combining single features with good performance is not necessarily the best feature subset in many cases. Compared with $l_{2,1}$-norm, $l_{2,0}$-norm can achieve better sparsity. We can directly obtain the best-performing feature combination if we apply $l_{2,0}$-norm to the projection matrix. Therefore, we use $l_{2,0}$-norm constraint to ensure the sparsity of W in Eq. (7). The formulation is

$$\begin{aligned} &\min_{E,W} \sum_{i=1}^{n}\sum_{j=1}^{h} \|W^T x_i - W^T a_j\|_2^2 e_{ij} + \alpha \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij}loge_{ij} \\ &s.t.\ e^i \mathbf{1}_h = 1, e_{ij} \geq 0, W^T W = I, \|W\|_{2,0} = k, \end{aligned} \tag{8}$$

where k is the number of selected features ($m \leq k \leq d$). $||W||_{2,0} = k$ ensures the projection matrix W has k non-zero rows corresponding to the selected features. Therefore, the optimal feature subset can be obtained directly.

4.4. The low-redundant regularization

UFS aims to obtain a representative feature subset with uncorrelated features. Eq. (8) considers the sparsity of the projection matrix, but it ignores the redundancy between features. Therefore, a low-redundancy regularization term is applied to reduce the redundancy of the selected feature subset [45].

Assuming that the selected feature subset is F, f_i, $f_j \in F (i, j \in \{1, \ldots, n\})$, the redundancy of F can be calculated by

$$R(F) = \frac{1}{k^2} \sum_{f_i, f_j \in F} \rho(f_i, f_j) = \frac{1}{k^2} \sum_{f_i, f_j \in F} \frac{cov(f_i, f_j)}{\sqrt{D(f_i)D(f_j)}}, \tag{9}$$

where $\rho(f_i, f_j) = \frac{cov(f_i, f_j)}{\sqrt{D(f_i)D(f_j)}}$ is the Pearson correlation coefficient between f_i and f_j. $D(f_*)$ is the standard deviation of f_*, and $cov(f_i, f_j)$ is the covariance between f_i and f_j.

Let $\check{X}$ denote the normalized X, in which the standard deviation of features f_i is 1. Then, the Pearson correlation coefficient between features is $\rho(f_i, f_j) = cov(f_i, f_j) = f_i^T f_j$, and $\rho(f_i, f_j) = f_i^t f_j$. $R(F)$ in Eq. (9) is converted to

$$R(F) = \frac{1}{k^2} \sum_{f_i, f_j \in F} f_i^T f_j = Tr\left(W^T \check{X} \check{X}^T W \mathbf{1}_{d\times d}\right) = Tr\left(W^T R W \mathbf{1}_{d\times d}\right), \tag{10}$$

where $\mathbf{1}_{d\times d} \in R^{d\times d}$ is a matrix and its all elements is 1, $R = \check{X}\check{X}^T$.

4.5. The objective function of BGLR

The objective function is obtained by integrating Eq. (10) into Eq. (8) as

$$\begin{aligned} &\min_{E,W} \sum_{i=1}^{n}\sum_{j=1}^{h} \|W^T x_i - W^T a_j\|_2^2 e_{ij} + \alpha \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij} \log e_{ij} + \beta Tr(W^T R W \mathbf{1}_{d\times d}) \\ &s.t.\ e^i \mathbf{1}_h = 1, e_{ij} \geq 0, W^T W = I, \|W\|_{2,0} = k, \end{aligned} \tag{11}$$

where $\alpha > 0, \beta > 0$ are balance parameters, and k is the number of selected features.

The objective function of BGLR has three parts: adaptive bipartite graph, low-redundant regularization, and sparse constraint. The first part is applied to obtain an adaptive bipartite graph, where information entropy is used to avoid trivial solutions. The second part aims to control the redundancy of selected feature subsets. The third part is $l_{2,0}$-norm constraint, which is employed to guarantee the sparsity of the projection matrix. Currently, many methods use random sampling or the k-means method to determine anchor points. BGLR selects anchors according to the variance of samples and correlation between samples, which can ensure selected anchor points contain enough information and preserve the balanced structure of data. Moreover, existing methods always choose $l_{2,1}$-norm constraint to realize sparsity. The proposed BGLR uses $l_{2,0}$-norm constraint to ensure the projection matrix has accurate k nonzero rows. That is, we can obtain feature subsets directly.

5. Optimization and convergence analysis

The section presents the optimization for the objective function of BGLR in Eq. (11), which is non-convex. Two variables, W and E, need to be learned. Consequently, the alternating iteration method is adopted to solve it. Moreover, the convergence and computational complexity of BGLR are analyzed theoretically.

5.1. Optimization

The variables W and E are alternately updated as follows.

5.1.1. Fix E and update W

With E fixed, Eq. (11) is simplified as

$$\begin{aligned} &\min_{W} \sum_{i=1}^{n}\sum_{j=1}^{h} \|W^T x_i - W^T a_j\|_2^2 e_{ij} + \beta Tr\left(W^T R \mathbf{1}_{d\times d} W\right) \\ &s.t.\ W^T W = I, \|W\|_{2,0} = k. \end{aligned} \tag{12}$$

Let $\widetilde{X} = [X\ A] \in R^{d\times(n+h)}$, $S = \begin{bmatrix} 0 & E \\ E^T & 0 \end{bmatrix} \in R^{(n+h)\times(n+h)}$. According to the properties of matrix rank, we have $Tr\left(W^T R W \mathbf{1}_{d\times d}\right) = Tr(W^T R \mathbf{1}_{d\times d} W)$. Eq. (12) is equivalent to

$$\begin{aligned} &\min_{W} Tr\left(W^T \widetilde{X} L_S \widetilde{X}^T W\right) + \beta Tr\left(W^T R \mathbf{1}_{d\times d} W\right) \\ &s.t.\ W^T W = I, \|W\|_{2,0} = k, \end{aligned} \tag{13}$$

where L_S is the normalized Laplacian matrix of S, $L_S = I - D^{-\frac{1}{2}} S D^{\frac{1}{2}}$. $D \in R^{(n+h)\times(n+h)}$ is the degree matrix of S. D is a diagonal matrix with diagonal elements $D_{ii} = \sum_j s_{ij}$. Let $M = \widetilde{X} L_S \widetilde{X}^T + \beta R \mathbf{1}_{d\times d}$, then Eq. (13) can be rewritten as

$$\begin{aligned} &\min_{W} Tr\left(W^T M W\right)\ \ s.t.\ W^T W = I, \|W\|_{2,0} = k \\ &\Leftrightarrow \max_{W^T W = I, \|W\|_{2,0}=k} Tr(W^T(\lambda I - M) W) \\ &\Leftrightarrow \max_{W^T W = I, \|W\|_{2,0}=k} Tr\left(W^T Z W\right), \end{aligned} \tag{14}$$

where $Z = \lambda I - M = \lambda I - (\widetilde{X} L_S \widetilde{X}^T + \beta R \mathbf{1}_{d\times d})$. Setting λ to the largest eigenvalue of M ensures that Z is a positive semi-definite matrix. So Eq. (13) is equivalent to

$$\max_{W} Tr\left(W^T Z W\right)\ \ s.t.\ W^T W = I, \|W\|_{2,0} = k. \tag{15}$$

Considering the cases $rank(Z) \le m$ or $rank(Z) > m$, then the solving processes are discussed as follows.

(1) $rank(Z) \le m$

when $k = m$, Eq. (15) converts to

$$\max_{W^T W = I, \|W\|_{2,0}=m} Tr\left(W^T Z W\right). \tag{16}$$

Since $W^T W = I$, W can be decomposed as $W = QU$, where $U \in R^{k\times m}$ $(U^T U = I)$ consists of nonzero rows of W. $Q \in R^{d\times k}$ is an indicator matrix, in which each element satisfies $q_{ij} = 1$ if $i = index(j)$ and $q_{ij} = 0$ otherwise. That is, $Q^T 1_d = 1_m$ and $Q^T Z Q$ is a principal submatrix of order m of Z. Eq. (16) can be rewritten as

$$\begin{aligned} &\max_{W^T W = I, \|W\|_{2,0}=k} Tr(U^T Q^T Z Q U) \\ &\Leftrightarrow \max_{U^T U = I, U\in R^{k\times m}} Tr(U^T Q^T Z Q U) \\ &\Leftrightarrow \max_{U^T U = I, U\in R^{k\times m}, \widetilde{Z}\in M_m(Z)} Tr\left(U^T \widetilde{Z} U\right). \end{aligned} \tag{17}$$

U is a square matrix when $k = m$. That is, $U^T U = U U^T = I_{m\times m}$, $Tr\left(U^T \widetilde{Z} U\right) = Tr\left(\widetilde{Z} U^T U\right) = Tr(\widetilde{Z})$. Eq. (17) is equivalent to

$$\max_{\widetilde{Z}\in M_m(Z)} Tr\left(\widetilde{Z}\right). \tag{18}$$

The first k largest diagonal elements of Z are the solution to the problem. The key to the above argument is to assume $k = m$. Obviously, $\sum_{i=1}^{m} \lambda_i(Q^T Z Q) = Tr(Q^T Z Q)$ and $rank(\widetilde{Z}) = rank(Q^T Z Q) \le m$. It can be found that, for any $k(m \le k \le d)$, $rank(Z) \le m$ and $rank(\widetilde{Z}) \le m$ while $\widetilde{Z} \in M_m(Z)$. The following can be obtained when $rank(Z) \le m$, even $k \ne m$.

$$\begin{aligned} &\max_{W^T W = I, \|W\|_{2,0}=k} Tr\left(W^T Z W\right) \\ &\Leftrightarrow \max_{\widetilde{Z}\in M_m(Z)} \sum_{i=1}^{k} \lambda_i(\widetilde{Z}) \\ &\Leftrightarrow \max_{\widetilde{Z}\in M_m(Z)} Tr(\widetilde{Z}) \end{aligned} \tag{19}$$

By sorting the diagonal elements of Z, choose the rows and columns corresponding to the top k most prominent diagonal elements to form the k-order principal sub-matrix $\widetilde{Z}$. The global optimal solution of W can be obtained by eigenvalue decomposition of $\widetilde{Z}$. Algorithm 1 summarizes the solution steps of Eq. (16).

Algorithm 1 Algorithm for solving Eq. (15) ($rank(Z) \le m$)

Input: Z in Eq. (15), number of selected features k.
Output: The projection matrix $W \in R^{d\times m}$.

1: Sort $diag(Z)$ in descending order to obtain the vector $index$.
2: Calculate $\widetilde{Z}$ via $Z_{index(1:k),index(1:k)}$.
3: Calculate Q by $Q_{ij} = \begin{cases} 1 & for\ i = index(j) \\ 0 & for\ otherwise \end{cases}$.
4: Compute the eigenvector composition U corresponding to the first m largest eigenvalues of $\widetilde{Z}$.
5: Calculate W by $W = QU$.

(2) $rank(Z) > m$

The above analysis shows that in the case of low-rank Z,Eq. (15) can be solved by Algorithm 1. So when the rank of Z is large, we can define a low-rank surrogate matrix F. Let $rank(F) \le m$. Then, Algorithm 1 can be used to solve Eq. (15). The low-rank proxy matrix F is defined as

$$F = ZW(W^T Z W)^+ W^T Z. \tag{20}$$

The following proposition holds.

Proposition 1. *When $W^T W = I$, F is semi-positive definite and $rank(F) \le m$.*

Proof. Z is a semi-positive definite matrix. Let $Z = RR^T$. Based on the property of generalized inverse, $G^+ = G^+ G G^+$, Eq. (20) is converted to

$$\begin{aligned} &F = ZW(W^T Z W)^+ W^T Z \\ &\Leftrightarrow F = ZW(W^T Z W)^+ W^T ZW(W^T Z W)^+ W^T Z \\ &\Leftrightarrow F = ZW(W^T Z W)^+ W^T RR^T W(W^T Z W)^+ W^T Z \\ &\Leftrightarrow F = MM^T. \end{aligned} \tag{21}$$

In Eq. (21), it can be observed $M = ZW(W^T Z W)^+ W^T R$. Then, it is proved that F is semi-positive definite. Based on the property of matrix rank, $rank(F) \le rank(W)$. Because $rank(W) = m$, then $rank(F) \le m$. □

When $rank(Z) > m$, the problem in Eq. (17) becomes

$$\max_{W^T W = I, \|W\|_{2,0}=m} Tr\left(W^T F W\right). \tag{22}$$

F is used as input of Algorithm 1 to obtain the solution of Eq. (22), i.e. the eigenvalue decomposition is performed on $\widetilde{F} \in M_m(F)$ to update the non-zero rows of W. However, the first m eigenvectors obtained are uncorrelated when $rank(\widetilde{F}) = m$. It

cannot guarantee that the eigenvectors are uncorrelated when $rank\left(\widetilde{F}\right) < m$. The feature vectors are uncorrelated with each other. Therefore, another method is used to solve the problem in Eq. (17). F is rewritten as $F = PGP^T$, where $P = ZW$ and $G = (W^TZW)^+$. It follows that $\widetilde{F} = \widetilde{W}G\widetilde{W}^T$, and that any orthogonal basis of $\widetilde{W}$ can be used to obtain W. Considering W and $\hat{W}$ denote the solution obtained by the above two methods. There exists a matrix K satisfying $KK^T = I_{m\times m}$ and $\hat{W} = WK$. It is obvious that

$$\begin{aligned} Tr\left(\hat{W}^TF\hat{W}\right) &= Tr\left(K^TW^TFWK\right) = Tr\left(W^TFWKK^T\right) \\ &= Tr\left(W^TFW\right) \end{aligned} \tag{23}$$

The solutions obtained by both methods correspond to the same objective function. Since $W = QU$, W can be further refined by updating U when fixing Q, i.e., it is solved by

$$\max_{U^TU=I} Tr\left(U^TQ^TZQU\right). \tag{24}$$

Eq. (24) does not need to be solved by a low-rank proxy matrix, and its solution can be obtained by directly performing an eigenvalue decomposition of Z. It also converges more quickly. Algorithm 2 summarizes the steps for solving Eq. (15) in the case $rank(Z) > m$.

Algorithm 2 Algorithm for solving Eq. (15) ($rank(Z) > m$)

Input: Z in Eq. (15), number of selected features k.
Output: The projection matrix $W \in R^{d\times m}$.
1: **repeat**
2: Calculate $F = ZW(W^TZW)^+W^TZ$.
3: Sort $diag(F)$ in descending order to obtain the vector $index$.
4: Calculated $\widetilde{Z}$ by $Z_{index(1:k),index(1:k)}$.
5: Calculate Q by $Q_{ij} = \begin{cases} 1 & for\ i = index(j) \\ 0 & for\ otherwise \end{cases}$.
6: Compute the eigenvector composition U corresponding to the first m largest eigenvalues of $\widetilde{Z}$.
7: Calculation W by $W = QU$.
8: **until** converge

5.1.2. Update E and fix W

When W is fixed, Eq. (11) becomes

$$\min_{E} \sum_{i=1}^{n}\sum_{j=1}^{h} \|W^Tx_i - W^Ta_j\|_2^2 e_{ij} + \alpha \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij}loge_{ij} \ \ s.t.\ e^i\mathbf{1}_h = 1, e_{ij} \geq 0. \tag{25}$$

The Lagrangian function of Eq. (25) is

$$\begin{aligned} L(E) = &\sum_{i=1}^{n}\sum_{j=1}^{h}(\|W^Tx_i - W^Ta_j\|_2^2 e_{ij} + \alpha e_{ij}loge_{ij}) - \sum_{i=1}^{n} v_i\Big(\sum_{j=1}^{h} e_{ij} - 1\Big) \\ &- \sum_{i=1}^{n}\sum_{j=1}^{h} \mu_{ij}e_{ij}, \end{aligned} \tag{26}$$

where v_i and μ_{ij} are the Lagrange multipliers. The above equation satisfies the following KKT condition.

$$\begin{cases} \|W^Tx_i - W^Ta_j\|_2^2 e_{ij} + \alpha(loge_{ij} + 1) - v_i - \mu_{ij} = 0 \\ e_{ij} \geq 0 \\ \mu_{ij} \geq 0 \\ \mu_{ij}e_{ij} = 0 \end{cases} \tag{27}$$

From Eq. (27), it can obtain

$$e_{ij} = exp(-1)\,exp\left(\frac{v_i}{\alpha}\right) exp\left(-\frac{\|W^Tx_i - W^Ta_j\|_2^2}{\alpha}\right). \tag{28}$$

Because $\sum_{j=1}^{h} e_{ij} = 1$, the optimal solution of E is

$$e_{ij} = \frac{exp\left(-\frac{\|W^Tx_i - W^Ta_j\|_2^2}{\alpha}\right)}{\sum_{j=1}^{m} exp\left(-\frac{\|W^Tx_i - W^Ta_j\|_2^2}{\alpha}\right)}. \tag{29}$$

The Eq. (11) is solved by alternately optimizing W and E. Algorithm 3 summarizes the specific solution steps.

Algorithm 3 Optimizing Eq. (11)

Input: Data matrix $X \in R^{d\times n}$, the selected feature number k, the number of anchors h, the regularization parameters σ and β, the projection dimension m.
Output: The projection matrix $W \in R^{d\times m}$.
1: **Initialize:** Generate the matrix $W \in R^{d\times m}$ randomly, satisfying $W^TW = I$.
2: Obtain the anchor matrix $A \in R^{d\times h}$ by using the variance-based decorrelation anchor selection method.
3: **repeat**
4: Update E by Eq. (29).
5: Update W by Algorithm 2.
6: **until** converge

5.2. Convergence analysis

The solution of Eq. (11) is divided into two subproblems, i.e., the problems in Eqs. (12) and (25). Eq. (25) is convex. Then, solving Eq. (12) is analyzed to examine the convergence of Algorithm 2.

Lemma 1. *Assuming $C \in R^{n_1\times n_2}$, $D \in R^{n_2\times n_1}$, and $n_1 \leq n_2$, there following hold.*

$$\lambda_i(DC) = \begin{cases} \lambda_i(CD) & 1 \leq i \leq n_1 \\ 0 & n_1 + 1 \leq i \leq n_2 \end{cases} \tag{30}$$

Corollary 1. *Let $H = Z^TW(W^TZZ^TW)^+W^TZ$, the eigenvalues of H satisfy*

$$\lambda_i(H) = \begin{cases} 1 & 1 \leq i \leq l \\ 0 & l + 1 \leq i \leq d \end{cases} \tag{31}$$

where $l = rank(Z^TW) \leq m$.

Proof. Let $M = (W^TZZ^TW)^+W^TZ$, $N = Z^TW$. For any $1 \leq i \leq d$, $\lambda_i(\beth) = \lambda_i(NM)$, and $MN = (W^TZZ^TW)^+W^TZZ^TW$, the following is hold by combining with Lemma 1.

$$\lambda_i(\beth) = \begin{cases} \lambda_i(MN) & 1 \leq i \leq m \\ 0 & m + 1 \leq i \leq d \end{cases} \tag{32}$$

Since $rank(MN) = r \leq m$ and $MN = \begin{bmatrix} I_{r\times r} & 0 \\ 0 & 0 \end{bmatrix}$, the above corollary holds. The proof is finished. □

Lemma 2. *For any symmetric matrices $A \in R^{n\times n}$, $B \in R^{n\times n}$, the equation $Tr(AB) \leq \sum_{i=1}^{n} \lambda_i(A)\lambda_i(B)$ holds.*

Proposition 2. *Assuming Z is a semi-positive definite matrix, $m \leq k \leq d$, the loss function given in Eq. (12) increases monotonically at each iteration of Algorithm 2.*

Proof. Let $\widetilde{W}$ be the result of the maximization Eq. (22). According to the property of the generalized inverse, $G = GG^{+}G$, and the Cholesky decomposition $Z = RR^{T}$, the following holds.

$$\begin{aligned} &Tr\left(W_t^T Z W_t\right) = Tr(W_t^T Z W_t (W_t^T Z W_t)^{+} W_t^T Z W_t) \\ &\le Tr(\widetilde{W}_{t+1}^T Z W_t (W_t^T Z W_t)^{+} W_t^T Z \widetilde{W}_{t+1}) \\ &= Tr(R^T W_t (W_t^T Z W_t)^{+} W_t^T R R^T \widetilde{W}_{t+1} \widetilde{W}_{t+1}^T R) \\ &= Tr(\beth\aleph) \le \sum_{i=1}^{d} \lambda_i(\beth)\lambda_i(\aleph) \le \sum_{i=1}^{m} \lambda_i(\aleph) = Tr(\aleph) = Tr(\widetilde{W}_{t+1} Z \widetilde{W}_{t+1}^T) \end{aligned} \tag{33}$$

In above formulas, $\beth = R^T P_t (P_t^T Z P_t)^{+} P_t^T R$ and $\aleph = R^T \widetilde{P}_{t+1} \widetilde{P}_{t+1}^T R$. $Tr\left(W_t^T Z W_t\right) \le Tr(\widetilde{W}_{t+1} Z \widetilde{W}_{t+1}^T)$, the proof is finished. □

Since Z is a semi-positive definite matrix, it can be inferred that

$$\begin{aligned} &\max_{W^T W = I, \|W\|_{2,0} = k} Tr\left(W^T Z W\right) \\ &\le \max_{W^T W = I} Tr(W^T Z W) \\ &\le \sum_{i=1}^{m} \lambda_i(Z) \le \sum_{i=1}^{d} \lambda_i(Z) = Tr(Z). \end{aligned} \tag{34}$$

The objective function monotonically increases using Algorithm 2 to solve Eq. (22). Moreover, the objective function has an upper bound, which ensures the convergence of Algorithm 2.

5.3. Computational complexity analysis

In Algorithm 1, it takes $O(d \log d)$ to sort diag(Z) to get the index, $O(k^3)$ to decompose the eigenvalue of $\widetilde{Z}$, and $O(km)$ to construct W. So, the computational complexity required by Algorithm 1 is $O(maxd \log d, k^3)$. Algorithm 2 needs to calculate F, sort its diagonal elements to get index I, and calculate the selection matrix Q from the index. The computational complexity of these steps is $O(maxd \log d, dkm)$, and the subsequent feature refinement part of updating W takes $O(k^3)$. Algorithm 2's total computational complexity is $O(maxdkm, d \log d, k^3 t)$, where t is iteration times.

For Algorithm 3, anchors are selected according to the variance of the sample and the correlation between samples. h iterations generate h anchors. The computational complexity of anchor selection is $O(nh)$. Because of the use of anchors, the computational complexity of updating E is $O(nh)$. Finally, the computational complexity to update W is $O(maxdkm, d \log d, k^3 t)$. Therefore, the total computational complexity of Algorithm 3 is $O\left(nh + (nh + maxdkm, d \log d, k^3 t) iter\right)$, where iter is iteration times.

6. Experiments and analyses

We design experiments on eight benchmark datasets to validate the effectiveness of BGLR. There are six different kinds of experiments: (1) The feature selection approaches are applied to data sets. Then, the clustering method k-means is applied to the selected features. Performance is evaluated using ACC and NMI metrics. (2) The proposed BGLR's convergence is verified experimentally. (3) The performance of BGLR with different anchors and regularization parameters is discussed. (4) The stability of the BGLR is analyzed. (5) The ablation experiment is implemented to verify the strategies' effectiveness. (6) BGLR's running time is analyzed. We implement all experimental codes on a Windows 11 Server with MATLAB R2022a, 3.0 GHz Intel i5-812500 CPU, and 32 GB RAM.

6.1. Datasets

Eight public datasets are used in the experiments, including a handwriting dataset, (USPS[1]) a speech signal dataset (Isolet[2]), four image datasets, (Umist[2] ORL[2], COIL20[2] and AR[2]), a manual dataset (Madelon[2]), and a binary dataset (PCMAC[2]). Table 3 provides details.

6.2. Comparison algorithms

BGLR[3] is compared with eight state-of-the-art methods. A brief description of these methods is as follows.

Baseline: Clustering is performed using raw data.

LS [11]: It evaluates the importance of features based on their ability to maintain local structure and selects highly rated features.

UDFS [46]: It combines discriminant analysis and $l_{2,1}$-norm into a joint framework for feature selection.

RSOGFS [19]: It adaptively updates the similarity matrix and directly solves $l_{2,0}$-norm constraints to realize group feature selection.

OBCLFS [47]: It combines orthogonal basis clustering with adaptive graph learning, which preserves the local structure of data while achieving good cluster separation.

VCSDFS [28]: It takes the correlation between features into account and proposes a variance–covariance distance-based feature selection framework that excludes features with slight differences from the original feature set and selects a discriminative subset of features.

BGCFS [29]: It uses the k-means method to select anchors from original data. Besides, it utilizes $l_{2,0}$-norm constraint to ensure sparsity.

FSDK [30]: It performs feature selection without similarity matrix construction and spectral analysis. Therefore, it can handle large-scale data quickly.

FOLPPFS-R [23]: It adopts a random sampling method to choose anchors and uses LPP to preserve the manifold structure of data.

SFESA [39]: To shorten the time for anchor selection, SFESA employs a binary tree manner to perform k-means. Moreover, it performs spectral analysis to maintain the low-dimensional embedding of data manifold structure.

6.3. Evaluation metrics

We use clustering accuracy (ACC) and normalized mutual information (NMI) as evaluation indicators for UFS algorithms. The algorithm with higher ACC and NMI values performs better.

Let p_i be the predicted clustering label and r_i be the true label. If $a = b$, then $\delta(a, b) = 1$, otherwise $\delta(a, b) = 0$. $map(\bullet)$ is the optimal mapping function that aligns clustering labels with real labels. Then, $ACC = \frac{\sum_{i=1}^{n} \delta(r_i, map(p_i))}{n}$. For any two random variables R and Q, $MI(R, Q)$ is the mutual information between R and Q, and $H(R)$ and $H(Q)$ are the entropy of R and Q. The NMI of R and Q is defined as $NMI = \frac{MI(R;Q)}{\sqrt{H(R)H(Q)}}$ ($NMI \in [0, 1]$). When R is equal to Q, $NMI = 1$; When R and Q are independent, $NMI = 0$.

6.4. Parameter settings

The parameters of comparison algorithms are set according to the referenced papers. We fix the neighborhood number as 5. We set c according to the actual number of classes. The Gaussian kernel bandwidth parameter σ in LS is fixed as 10. Relevant balance parameters are searched in $[10^{-3}, \ldots, 10^{3}]$. The number of selected features (k) is set in $[20, 40, \ldots, 180, 200]$. Select different amounts of anchors according to specific datasets. Table 3 shows the number of anchors for each dataset. The value of m is adjusted in $[k-1, k-2, \ldots, k-10]$ with step 1. k-means is repeated 20 times on selected features, and the average results are recorded. Besides, to further reduce the randomness caused by the random initialization of the relevant matrix, all methods run ten times with the optimal combination of parameters. Relevant results are recorded for comparison.

[1] http://featureselection.asu.edu/datasets.php .

[2] https://jundongl.github.io/scikit-feature/datasets.html.

[3] https://github.com/amist08/BGLR.

Table 3
Datasets description.

Dataset	Instances	Features	Classes	Type	Anchors
ORL	400	1024	40	Face images	200
Umist	575	644	40	Face images	200
COIL20	1440	1024	20	Object images	500
Isolet	1560	617	26	Speech signal	500
AR	1680	1024	120	Face images	500
PCMAC	1943	3289	2	Digit images	500
madelon	2600	500	2	Artificial	800
USPS	9258	256	10	Digit images	1500

Table 4
Comparison of ACC on eight datasets (ACC ± STD).

Algorithms	ORL	Umist	COIL20	Isolet	PCMAC	madelon	USPS	AR	Mean
baseline	51.18 ± 0.85	41.20 ± 0.55	58.09 ± 2.44	58.23 ± 0.60	50.65 ± 0.11	51.75 ± 0.98	64.87 ± 0.63	35.40 ± 0.28	51.42
LS [11]	47.97 ± 0.75	39.25 ± 0.54	55.75 ± 1.22	52.48 ± 1.03	50.83 ± 0.01	55.41 ± 0.01	64.77 ± 0.75	36.34 ± 0.32	50.35
UDFS [46]	48.90 ± 0.58	43.44 ± 0.50	53.18 ± 0.84	52.28 ± 1.48	50.60 ± 0.13	59.76 ± 0.01	61.08 ± 0.80	34.52 ± 0.11	50.47
RSOGFS [19]	51.22 ± 1.14	**43.80 ± 1.48**	55.83 ± 1.38	55.17 ± 2.11	50.49 ± 0.11	53.73 ± 1.69	58.22 ± 1.89	38.90 ± 1.55	50.92
OBCLFS [47]	45.23 ± 0.38	43.72 ± 0.75	54.97 ± 0.99	59.35 ± 0.90	50.50 ± 0.01	50.67 ± 0.07	64.58 ± 0.35	30.23 ± 0.01	49.91
VCSDFS [28]	50.75 ± 1.07	41.65 ± 2.15	57.99 ± 1.41	53.67 ± 2.19	51.01 ± 0.18	50.89 ± 0.85	65.65 ± 0.98	38.61 ± 1.48	51.28
FSDK [30]	49.43 ± 1.57	40.56 ± 2.63	57.89 ± 1.56	55.03 ± 2.07	51.54 ± 0.94	53.69 ± 2.83	64.36 ± 1.07	**43.45 ± 2.88**	51.99
BGCFS [29]	50.25 ± 1.45	40.67 ± 2.02	57.13 ± 1.80	55.41 ± 2.76	50.60 ± 0.25	54.63 ± 3.53	63.51 ± 1.14	33.51 ± 2.33	50.71
FOLPPFS-R [23]	46.38 ± 1.87	38.17 ± 3.26	55.48 ± 0.61	37.46 ± 0.33	50.56 ± 0.05	51.76 ± 0.64	64.67 ± 0.73	42.90 ± 1.59	48.42
SFESA [39]	50.34 ± 1.73	41.48 ± 1.45	55.70 ± 1.91	53.82 ± 2.87	50.60 ± 0.01	56.22 ± 3.05	63.89 ± 1.00	33.81 ± 0.83	50.73
BGLR	**51.64 ± 0.21**	41.62 ± 0.14	**58.65 ± 0.75**	**60.96 ± 0.58**	**52.02 ± 0.30**	**60.73 ± 0.68**	**65.73 ± 1.15**	39.28 ± 1.12	**53.83**

Table 5
Comparison of NMI on eight datasets (NMI ± STD).

Algorithms	ORL	Umist	COIL20	Isolet	PCMAC	madelon	USPS	AR	Mean
baseline	71.11 ± 0.51	62.88 ± 0.31	72.48 ± 1.07	74.25 ± 0.37	0.16 ± 0.01	0.36 ± 0.26	60.73 ± 0.24	68.07 ± 0.17	51.26
LS [11]	69.07 ± 0.34	56.95 ± 0.62	69.46 ± 0.56	69.90 ± 0.45	**0.49 ± 0.01**	0.85 ± 0.01	60.98 ± 0.37	69.51 ± 0.20	49.65
UDFS [46]	70.13 ± 0.35	**63.71 ± 0.44**	66.38 ± 0.61	68.08 ± 0.63	0.13 ± 0.21	2.77 ± 0.03	56.52 ± 0.44	67.15 ± 0.01	49.36
RSOGFS [19]	71.05 ± 0.72	63.21 ± 1.24	71.08 ± 1.94	71.58 ± 1.76	0.06 ± 0.05	0.52 ± 0.34	55.43 ± 1.91	69.67 ± 1.14	50.36
OBCLFS [47]	66.62 ± 0.41	61.08 ± 1.01	70.85 ± 0.60	73.90 ± 0.68	0.20 ± 0.13	1.28 ± 0.13	61.96 ± 0.29	62.83 ± 0.22	49.84
VCSDFS [28]	71.41 ± 0.80	61.28 ± 2.26	71.81 ± 0.97	66.48 ± 1.30	0.14 ± 0.12	0.04 ± 0.07	61.30 ± 0.26	68.99 ± 1.01	50.18
FSDK [30]	70.15 ± 1.06	57.70 ± 3.38	70.16 ± 0.32	67.36 ± 2.80	0.38 ± 0.29	0.61 ± 0.64	60.03 ± 0.91	**72.94 ± 2.12**	49.92
BGCFS [29]	71.01 ± 0.82	60.63 ± 1.97	72.40 ± 0.84	70.91 ± 2.07	0.18 ± 0.16	1.01 ± 1.30	60.25 ± 0.63	66.12 ± 1.70	50.31
FOLPPFS-R [23]	67.49 ± 1.95	57.98 ± 2.94	69.32 ± 0.36	53.48 ± 0.21	0.15 ± 0.03	0.11 ± 0.01	61.41 ± 0.19	71.26 ± 1.31	47.65
SFESA [39]	71.11 ± 1.01	61.37 ± 1.87	70.98 ± 1.24	70.21 ± 2.10	0.15 ± 0.01	1.44 ± 1.16	59.81 ± 0.83	66.71 ± 0.68	50.22
BGLR	**71.52 ± 0.23**	61.90 ± 1.63	**72.89 ± 0.76**	**74.91 ± 0.63**	0.21 ± 0.03	**3.84 ± 0.32**	**62.01 ± 0.85**	69.17 ± 0.73	**52.02**

6.5. Analysis of experiment results

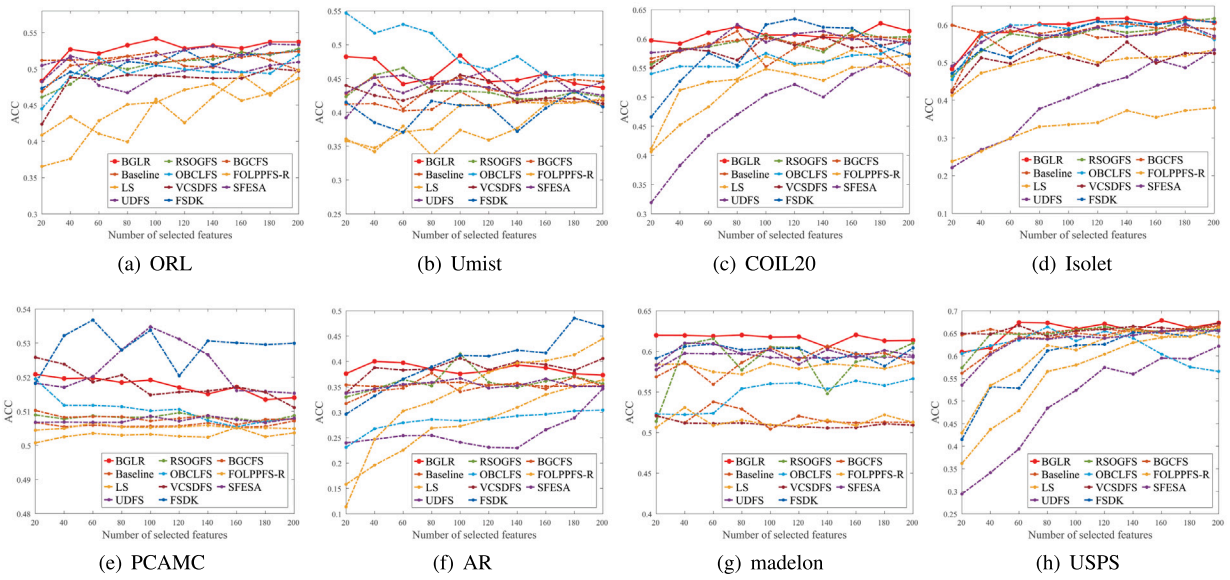
The BGLR and comparison algorithms are feature selection algorithms. k-means is applied to data after selection, and the performance of ACC and NMI is compared.

6.5.1. Clustering performance analysis

Tables 4 and 5 give the relevant results, where the values indicated in bold indicate the best results. Figs. 3 and 4 display the change of evaluation metrics values while the number of selected features varies.

Table 4 demonstrates that BGLR obtains higher ACC values on ORL, COIL20, Isolet, PCMAC, madelon, and USPS datasets. Table 5 indicates BGLR gets better NMI values on ORL, COIL20, Isolet, Madelon, and USPS datasets. FSDK obtains the best ACC and NMI values on AR dataset. OBCLFS obtains the highest ACC values on Umist data. UDFS has the highest NMI values on Umist data. On all datasets, BGLR performs better than the baseline, illustrating the necessity of feature selection. BGLR performs better than the comparison algorithms on most datasets for the following reasons: (1) BGLR selects anchors according to the variance of samples and correlation between samples. In this way, the selected anchors will contain enough information and can reduce the effect of missing samples on a similar matrix. Moreover, anchors are evenly distributed in the data space, which can preserve the internal structure of data well. (2) Since BGLR simultaneously updates the bipartite graph and projection matrix, the local structure of data can be preserved better. (3) BGLR takes redundant information between selected features into account. Combining the low redundant regularization term and $l_{2,0}$-norm constraint can guarantee that we can directly obtain the optimal feature subset while selected features are relatively independent. RSOGFS, which uses $l_{2,0}$-norm constraint and adaptive graph, obtains the best results on the Umist dataset. It proves that $l_{2,0}$-norm constraint is better for solving sparse problems, which can directly obtain feature subsets. Moreover, FSDK obtains the best performance on the AR datasets for its learned pseudolabel matrix to conduct feature selection.

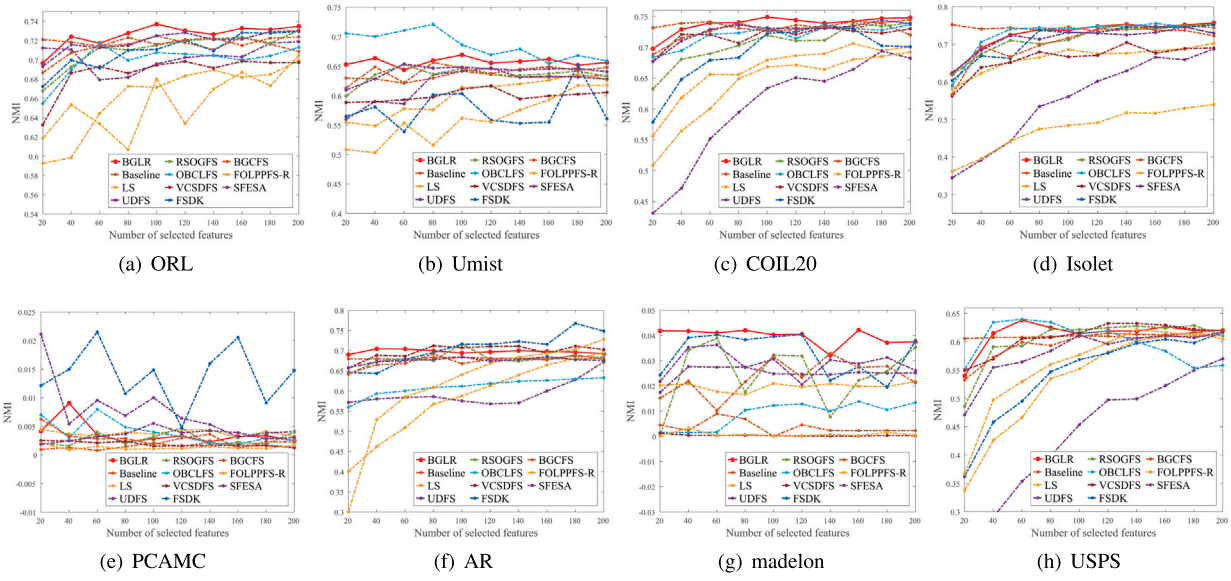
In Fig. 3, BGLR performs well on COIL20 and Madelon datasets. When the number of selected features varies, BGLR achieves better ACC than other algorithms. On ORL and Isolet datasets, BGLR achieves the maximum value of ACC. In Fig. 4, on the madelon dataset, the NMI values obtained by BGLR are higher than those obtained by other algorithms. On ORL, COIL20, and Isolet datasets, while we select more than 80 features, the NMI obtained by BGLR is superior to other algorithms. The NMI obtained by the BGLR algorithm on the three data sets of Umist, PCMAC, and AR is always higher than that of the baseline method in some cases, although not as good as that of other comparison algorithms, which indicates the effectiveness of feature selection. Moreover, because of noise and redundant information, ACC and NMI values do not always rise with the increase of selected feature numbers. Compared with the similarity matrix constructed by RSOGFS and OBCLFS using complete data, BGLR using a bipartite graph may miss some information. However, since we use the variance-based decorrelation anchor selection method, the selected anchor contains enough information to represent the original data, so BGLR's performance in specific applications is still excellent.



(a) ORL (b) Umist (c) COIL20 (d) Isolet

(e) PCAMC (f) AR (g) madelon (h) USPS

Fig. 3. ACC curve on eight datasets.



(a) ORL (b) Umist (c) COIL20 (d) Isolet

(e) PCAMC (f) AR (g) madelon (h) USPS

Fig. 4. NMI curve on eight datasets.

6.5.2. Parameter sensitivity analysis

There are three parameters α, β, and h (number of anchors) in BGLR. Here, we conduct experiments to demonstrate the impact of these three parameters. When validating the impact on the model of the regularization parameters α and β, we fix the selected feature number as 100, and the number of anchors is set as shown in Table 3. The adjusting range of regularization parameter α and β is $[10^{-3}, \ldots, 10^{3}]$. When verifying the impact of h, we set α and β as 1. The number of anchors h is changed according to the specific dataset.

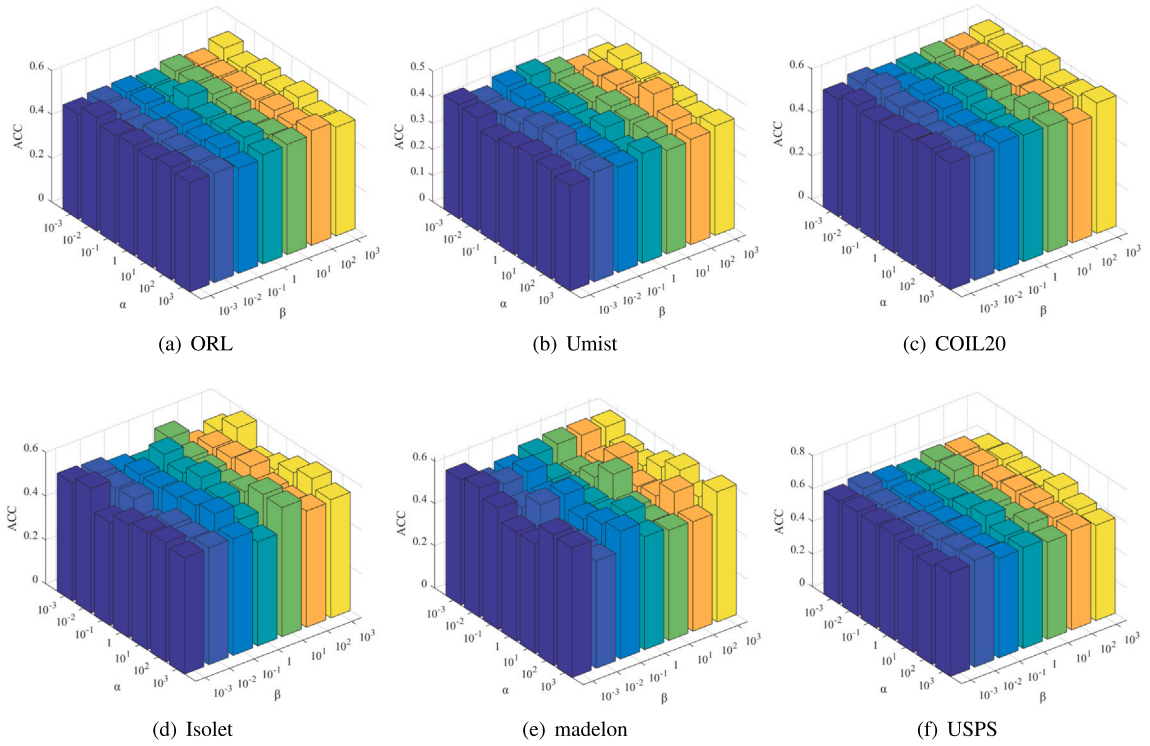
In Fig. 5, on ORL, COIL20, and USPS datasets, the ACC values of BGLR fluctuate slightly with the α and β values change. The ACC values change significantly with different parameter combinations on the Umist, Isolet, and Madelon datasets. The ACC value of BGLR on these six data sets increases first, then decreases, and then increases again, with different α. The ACC values of BGLR on ORL, Umist, COIL20, Isolet, and USPS datasets increase and decrease with different β. More minor β indicates more redundant features are selected, while larger β means that selected features are relatively independent. The above trend proves that redundant features will reduce the algorithm's performance. However, it may lead to overfitting if values of α and β are too big, reducing the algorithm's performance (see Fig. 6).

In Fig. 6, the NMI values of BGLR on ORL, Umist, COIL20, and USPS data sets are relatively stable with the change of α and β, which indicates that BGLR is robust. However, the NMI of BGLR fluctuates significantly with some parameter combinations, such as on Isolet and Madelon data sets, but the fluctuation value does not exceed 0.04.

Fig. 7 illustrates that the ACC value does not always rise when h (the number of anchors) increases. In practical application, we should set h according to the data size. The algorithm's performance will be compromised if the anchors are excessively large or small.

6.5.3. Ablation experiment

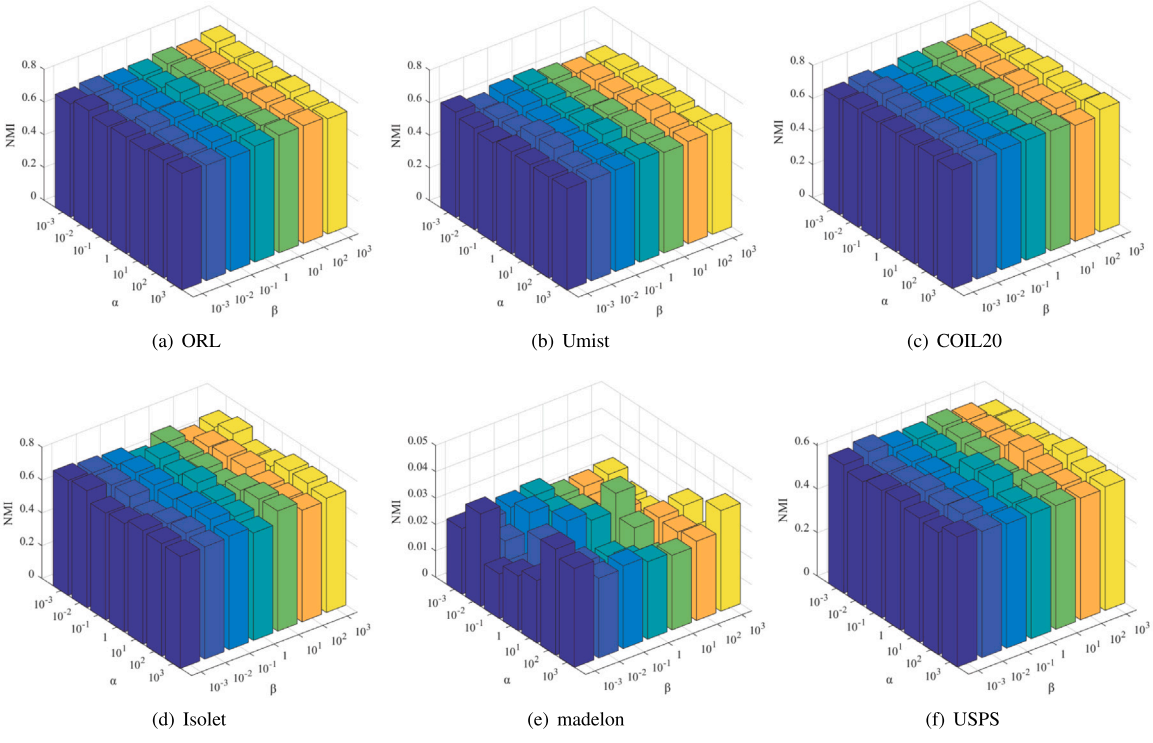
Different objective functions are designed by deleting or replacing a regularization item and keeping other items unchanged. The effectiveness of $l_{2,0}$-norm constraint of the projection matrix, the low-redundant regularization, and the variance-based decorrelation anchor selection method are testified. The following variants of BGLR are designed.



(a) ORL (b) Umist (c) COIL20

(d) Isolet (e) madelon (f) USPS

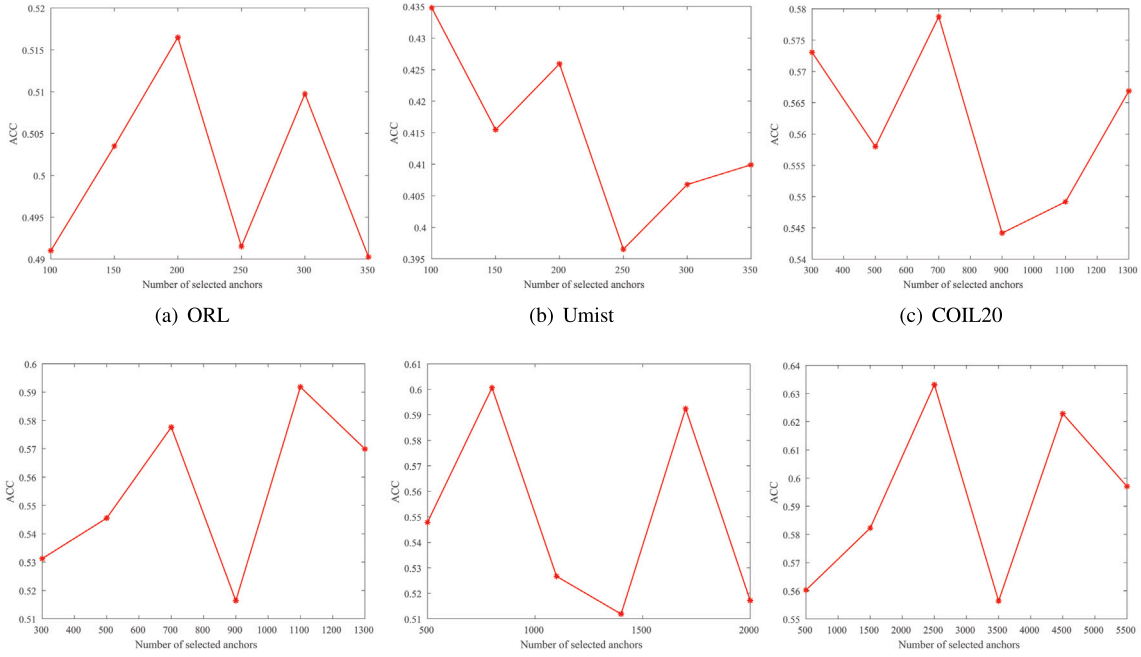
Fig. 5. The ACC of BGLR with different α and β.



(a) ORL (b) Umist (c) COIL20

(d) Isolet (e) madelon (f) USPS

Fig. 6. NMI of BGLR with different α and β.



(a) ORL (b) Umist (c) COIL20

(d) Isolet (e) madelon (f) USPS

Fig. 7. ACC of BGLR with the different number of anchors on six datasets.

(a) The $l_{2,0}$-norm constraint of projection matrix in Eq. (11) is replaced by $l_{2,1}$-norm to construct BGLR-21 to verify the validity of the $l_{2,0}$-norm. The formulation is

$$\min_{E,W} \sum_{i=1}^{n}\sum_{j=1}^{h}\|W^T x_i - W^T a_j\|_2^2 e_{ij} + \sigma \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij}\log e_{ij} + \alpha\|W\|_{2,1} + \beta Tr\left(W^T RW\mathbf{1}_{\mathbf{d}\times\mathbf{d}}\right) \; s.t.\; e^i\mathbf{1}_h = 1, e_{ij} \geq 0, W^T W = I, \|W\|_{2,0} = k \tag{35}$$

(b) The model $BGLR{-}\beta$ is constructed by deleting the $\beta Tr\left(P^T RP\mathbf{1}\right)$ term from Eq. (11) to verify the effectiveness of the low-redundant regularization in BGLR. The formulation is

$$\min_{E,W} \sum_{i=1}^{n}\sum_{j=1}^{h}\|W^T x_i - W^T a_j\|_2^2 e_{ij} + \sigma \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij}\log e_{ij} \quad s.t.\; e^i\mathbf{1}_h = 1, e_{ij} \geq 0, W^T W = I, \|W\|_{2,0} = k \tag{36}$$

(c) All items in Eq. (11) are unchanged, and the anchor selection method is changed to verify the effectiveness of the variance-based decorrelation anchor selection method. Two anchor selection methods are compared. (1) The anchor is selected by random sampling method, and the model variant BGLR-r is obtained; (2) The k-means method is used to select the anchor, and the model variant BGLR-k is obtained. That is, the target formula of BGLR-r and BGLR-k is consistent with that of BGLR.

$$\min_{E,W} \sum_{i=1}^{n}\sum_{j=1}^{h}\|W^T x_i - W^T a_j\|_2^2 e_{ij} + \sigma \sum_{i=1}^{n}\sum_{j=1}^{h} e_{ij}\log e_{ij} + \beta Tr(W^T RW\mathbf{1}_{\mathbf{d}\times\mathbf{d}}) \; s.t.\; e^i\mathbf{1}_h = 1, e_{ij} \geq 0, W^T W = I, \|W\|_{2,0} = k \tag{37}$$

Experiments on a data set with a small dimension are conducted for models BGLR and BGLR-21. The selected eigenvalue k is 5. The subspace dimension m is set to 3. The projection matrix is visualized in Fig. 8.

The colors correspond to different values in Fig. 8. The dark blue corresponds to 0. As shown in Fig. 8, BGLR-21 selects the top features by calculating feature scores and ranking them $l_{2,0}$-norm, the projection matrix obtained has accurate k non-zero rows, and the feature subset can be obtained by selecting the features corresponding to the number of non-zero rows. BGLR does not need to introduce additional regularization parameters and can directly get the best feature combination, improving the corresponding algorithm's performance.

We implement clustering experiments on selected features after applying BGLR and three sub-models on the ORL, Isolet, and Madelon datasets. The results of clustering experiments are displayed in Figs. 9 and 10.

In Figs. 9 and 10, BGLR achieved maximum ACC and NMI values on the ORL, Isolet, and Madelon datasets. BGLR-β performs less effectively than the other three models. It is concluded that the low redundancy regularization in BGLR is effective. Furthermore, the performance of the two sub-models that use random sampling and k-means methods to select anchors is not as good as the BGLR model because the randomness brought by these two methods in selecting anchors is relatively high. In most cases, the BGLR-r model performs worse than the BGLR-k on the three datasets. This may be due to the greater randomness in selecting anchors using random sampling methods.

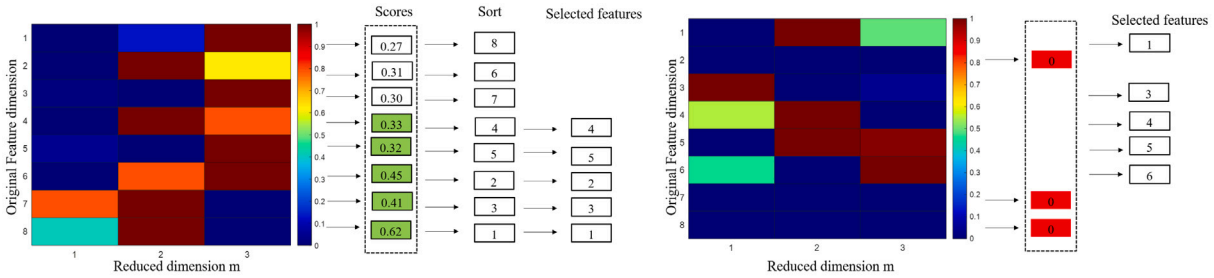
6.5.4. Convergence analysis

The convergence rate of BGLR is mainly determined by Eqs. (11) and (12). Here, we utilize experiments to validate the convergence rate of Eq. (12). On the ORL, COIL20, and Madelon datasets, we display the variations of the objective function values corresponding to Eq. (12) with different k and m. Besides, we depict the variations of the objective function values corresponding to Eq. (11).

From Fig. 11, the objective function values of Eq. (12) are monotonically increasing. Eq. (12) converges in 15 iterations regardless of the value of k and m. From Fig. 12, the objective function values of Eq. (11) decrease monotonically on different data sets. Eq. (11) converges within 50 iterations. In Fig. 12, the objective function values of Eq. (11) tend to be unchanged after thirty iterations.

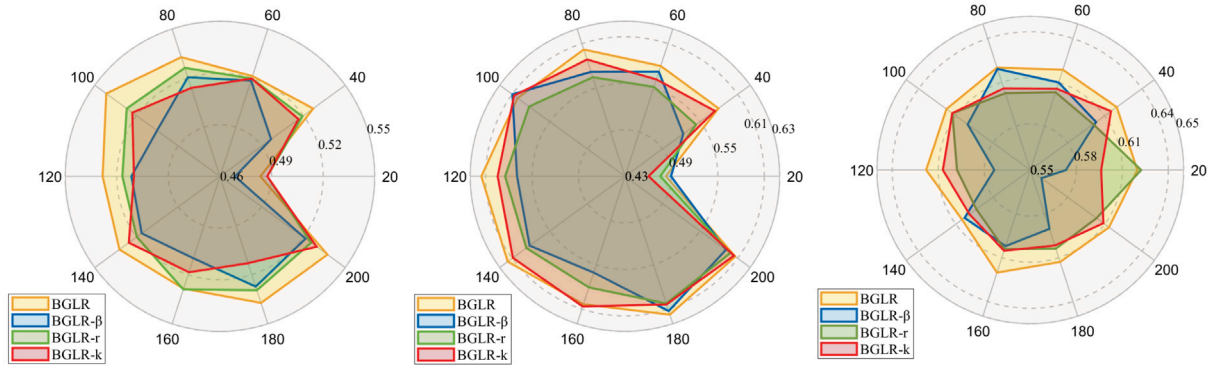
6.5.5. Running time analysis

The running time of BGLR is compared with other algorithms to verify its computation complexity. The running times of all algorithms



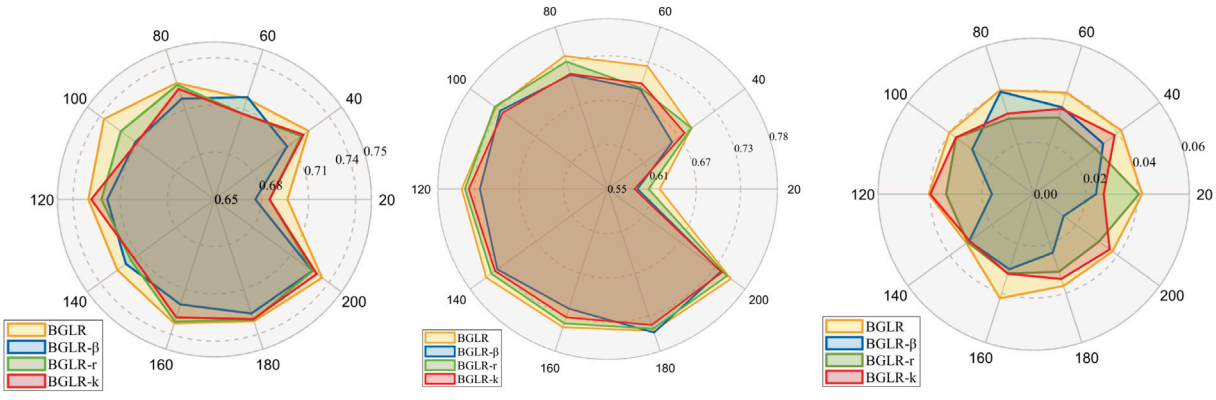
(a) $l_{2,1}$-norm (b) $l_{2,0}$-norm

Fig. 8. W gained via different methods.



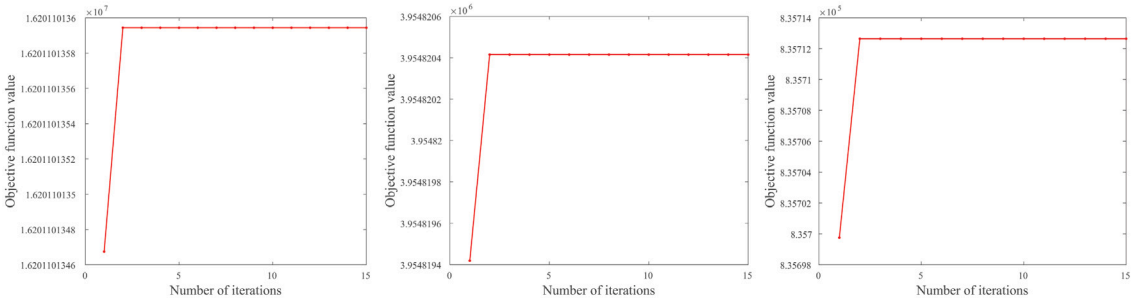
(a) ORL (b) Isolet (c) madelon

Fig. 9. ACC values for four methods on three datasets.



(a) ORL (b) Isolet (c) madelon

Fig. 10. NMI values for four methods on three datasets.



(a) COIL20,k=100,m=60 (b) ORL,k=100,m=70 (c) madelon,k=100,m=80

Fig. 11. The convergence curve of Eq. (12).

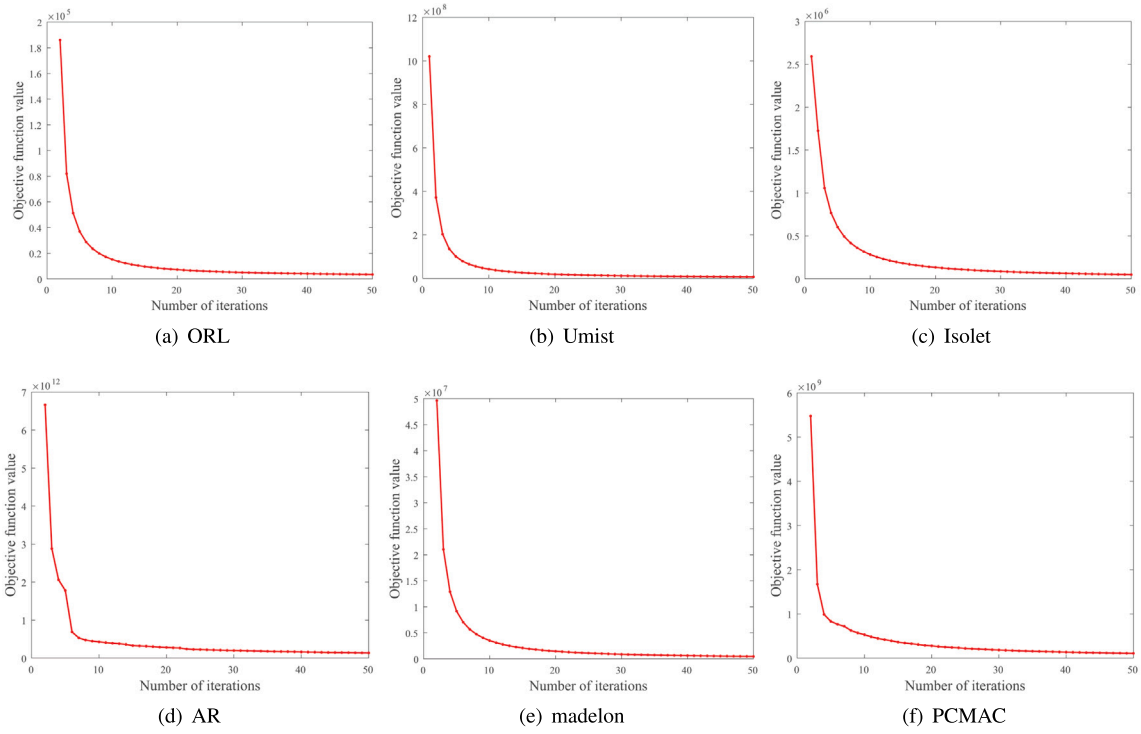


Fig. 12. The convergence curve of Eq. (11).

on eight datasets in Table 6. Since LS, VCSDFS, and FSDK perform feature selection without similarity construction, their running times are relatively low. Due to the adaptive graph learning, the running times of BGLR are longer than those of these methods. However, BGLR obtained higher means of ACC and NMI. Moreover, BGLR has an obvious advantage in ACC and NMI over these three methods on the madelon and Isolet datasets. For instance, on the madelon datasets, BGLR obtained ten percent higher ACC values than VCSDFS. On the Isolet dataset, BGLR obtained eight percent higher ACC values than LS. Table 6 shows that FOLPPFS-R runs faster than BGCFS, SFESA, and BGLR on the datasets with many samples. That is because FOLPPFS-R adopts a random sampling method to select anchors. Conversely, since SFESA adopts a binary tree manner to perform K-means, which will spend more time on anchor selection, SFESA runs relatively slowly. Moreover, it can be known from Table 6 that the methods employing graph learning run slowly. That is because the computational complexity of constructing the similarity matrix is relatively high. Since BGLR selects anchors according to the variance and the correlation of samples, selected anchors can preserve the balanced structure of the original data and provide sufficient information. In this way, the adaptive bipartite graph can preserve the structure of the data well and reduce the computational complexity of constructing the similarity matrix. Therefore, BGLR performs better on most datasets than these methods with worse runtime. Additionally, as shown in Tables 4 and 5, the overall means of BGLR are higher than those of other methods. In summary, it is obvious that BGLR performs better.

Table 6
Comparison of running time (s).

Algorithms	ORL	Umist	COIL20	Isolet	PCMAC	madelon	USPS	AR
LS [11]	0.02	0.03	0.28	0.06	0.52	0.11	0.79	0.17
UDFS [46]	1.76	1.81	13.35	7.64	229.97	35.73	1534.41	18.38
RSOGFS [19]	1.60	0.87	4.59	7.58	59.04	33.06	1254.51	6.93
OBCLFS [47]	8.38	5.04	15.22	9.81	224.33	45.31	997.17	20.21
VCSDFS [28]	0.22	0.36	1.21	1.17	2.67	3.84	8.19	1.39
FSDK [30]	0.11	0.17	0.61	0.48	3.11	3.31	105.75	0.80
BGCFS [29]	0.91	0.46	2.22	1.88	22.98	9.01	275.55	3.01
FOLPPFS-R [23]	1.57	0.53	2.42	1.55	36.46	7.95	177.07	3.15
SFESA [39]	1.10	0.80	4.51	3.18	21.71	14.58	312.37	4.58
BGLR	0.79	0.41	2.01	1.83	20.6	8.86	241.93	2.78
Average	1.65	1.05	4.64	3.52	62.14	16.18	490.77	6.14

7. Conclusions

Sparse graph-based learning captures the discriminative features to construct a learning model. However, constructing the similarity matrix comes with large computational costs. The bipartite graph is an effective way to lower this consumption. The crux of bipartite graph-based sparse graph learning is how to select anchors from samples. This study selects anchors according to the sample variance and pairwise correlations between samples. In this way, selected anchor points can better represent all samples and preserve the balanced structure of the data. Furthermore, the proposed BGLR can update the bipartite graph and projection matrix simultaneously. This study minimizes feature redundancy by evaluating the dependencies between features. The proposed BGLR solves the $l_{2,0}$-norm constraint problem to directly obtain a discriminative feature subset. Relevant experimental results illustrate the effectiveness of BGLR.

Nevertheless, BGLR has some limitations. Firstly, BGLR evaluates the similarity between features using the Pearson correlation coefficient, resulting in difficulty in determining the priority between two highly dependent features. Secondly, BGLR only focuses on reducing the redundancy of features, thus overlooking the preservation of feature structure. Therefore, it is worth noting how to make selected features relatively independent while preserving the feature structure. In future research, we will consider utilizing adaptive graphs to learn the local structure of features.

CRediT authorship contribution statement

Longyan Xiang: Writing – original draft, Visualization, Methodology, Formal analysis, Data curation, Conceptualization. **Hongmei Chen:** Writing – review & editing, Validation, Supervision, Investigation, Formal analysis. **Tengyu Yin:** Writing – review & editing.

Shi-Jinn Horng: Writing – review & editing. **Tianrui Li:** Writing – review & editing, Supervision, Funding acquisition.

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.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 62376230, 61976182, 62076171 and 62306196).

References

[1] M. Abi Kanaan, J.-F. Couchot, C. Guyeux, D. Laiymani, T. Atechian, R. Darazi, Combining a multi-feature neural network with multi-task learning for emergency calls severity prediction, Array 21 (2024) 100333.
[2] X. Hao, R. Wang, Y. Guo, Y. Xiao, M. Yu, M. Wang, W. Chen, D. Zhang, Alzheimer's Disease Neuroimaging Initiative, et al., Multi-modal self-paced locality preserving learning for diagnosis of Alzheimer's disease, IEEE Trans. Cogn. Dev. Syst. 15 (2) (2023) 832–843.
[3] Z. Xu, F. Yang, C. Tang, H. Wang, S. Wang, J. Sun, Y. Zhang, FG-HFS: A feature filter and group evolution hybrid feature selection algorithm for high-dimensional gene expression data, Expert Syst. Appl. 245 (2024) 123069.
[4] Z. Tu, H. Li, D. Zhang, J. Dauwels, B. Li, J. Yuan, Action-stage emphasized spatiotemporal VLAD for video action recognition, IEEE Trans. Image Process. 28 (6) (2019) 2799–2812.
[5] P. Zhu, G. Wang, J. He, Y. Dong, Y. Chang, An encrypted traffic identification method based on multi-scale feature fusion, Array (2024) 100338.
[6] J. Li, F. Qi, X. Sun, B. Zhang, X. Xu, H. Cai, Unsupervised feature selection via collaborative embedding learning, IEEE Trans. Emerg. Top. Comput. Intell. 8 (3) (2024) 2529–2540.
[7] C. Tang, X. Zheng, X. Liu, W. Zhang, J. Zhang, J. Xiong, L. Wang, Cross-view locality preserved diversity and consensus learning for multi-view unsupervised feature selection, IEEE Trans. Knowl. Data Eng. 34 (10) (2022) 4705–4716.
[8] X. Lu, J. Long, J. Wen, L. Fei, B. Zhang, Y. Xu, Locality preserving projection with symmetric graph embedding for unsupervised dimensionality reduction, Pattern Recognit. 131 (2022) 108844.
[9] B. Venkatesh, J. Anuradha, A review of feature selection and its methods, Cybern. Inf. Technol. 19 (1) (2019) 3–26.
[10] S. Solorio-Fernández, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, A review of unsupervised feature selection methods, Artif. Intell. Rev. 53 (2) (2020) 907–948.
[11] X. He, D. Cai, P. Niyogi, Laplacian score for feature selection, Adv. Neural Inf. Process. Syst. 18 (2005).
[12] J. Maldonado, M.C. Riff, B. Neveu, A review of recent approaches on wrapper feature selection for intrusion detection, Expert Syst. Appl. 198 (2022) 116822.
[13] R. Shang, J. Kong, W. Zhang, J. Feng, L. Jiao, R. Stolkin, Uncorrelated feature selection via sparse latent representation and extended OLSDA, Pattern Recognit. 132 (2022) 108966.
[14] J. Miao, Y. Ping, Z. Chen, X.-B. Jin, P. Li, L. Niu, Unsupervised feature selection by non-convex regularized self-representation, Expert Syst. Appl. 173 (2021) 114643.
[15] P. Huang, X. Yang, Unsupervised feature selection via adaptive graph and dependency score, Pattern Recognit. 127 (2022) 108622.
[16] C. Wang, J. Wang, Z. Gu, J.-M. Wei, J. Liu, Unsupervised feature selection by learning exponential weights, Pattern Recognit. 148 (2024) 110183.
[17] R. Shang, W. Zhang, M. Lu, L. Jiao, Y. Li, Feature selection based on non-negative spectral feature learning and adaptive rank constraint, Knowl.-Based Syst. 236 (2022) 107749.
[18] F. Nie, W. Zhu, X. Li, Structured graph optimization for unsupervised feature selection, IEEE Trans. Knowl. Data Eng. 33 (3) (2019) 1210–1222.
[19] F. Nie, X. Dong, L. Tian, R. Wang, X. Li, Unsupervised feature selection with constrained $l_{2,0}$-norm and optimized graph, IEEE Trans. Neural Netw. Learn. Syst. 33 (4) (2020) 1702–1713.
[20] Z. Li, F. Nie, J. Bian, D. Wu, X. Li, Sparse pca via L2, p-norm regularization for unsupervised feature selection, IEEE Trans. Pattern Anal. Mach. Intell. 45 (4) (2023) 5322–5328.
[21] S.-G. Fang, D. Huang, X.-S. Cai, C.-D. Wang, C. He, Y. Tang, Efficient multi-view clustering via unified and discrete bipartite graph learning, IEEE Trans. Neural Netw. Learn. Syst. (2023) 1–12.
[22] Y. Jiang, H. Lin, Y. Li, Y. Rong, H. Cheng, X. Huang, Exploiting node-feature bipartite graph in graph convolutional networks, Inform. Sci. 628 (2023) 409–423.
[23] J. Zhu, J. Chen, B. Xu, H. Yang, F. Nie, Fast orthogonal locality-preserving projections for unsupervised feature selection, Neurocomputing 531 (2023) 100–113.
[24] H. Zhang, F. Nie, X. Li, Large-scale clustering with structured optimal bipartite graph, IEEE Trans. Pattern Anal. Mach. Intell. 45 (8) (2023) 9950–9963.
[25] J. Zheng, Z. Wang, E. Chen, Joint structure bipartite graph projection and its application for industrial process monitoring, Process Saf. Environ. Prot. 184 (2024) 1502–1511.
[26] X. Tang, Y. Ding, J. Lei, H. Yang, Y. Song, Dynamic load balancing method based on optimal complete matching of weighted bipartite graph for simulation tasks in multi-energy system digital twin applications, Energy Rep. 8 (2022) 1423–1431, 2021 The 8th International Conference on Power and Energy Systems Engineering.
[27] E. Aigner-Horev, E. Segal-Halevi, Envy-free matchings in bipartite graphs and their applications to fair division, Inform. Sci. 587 (2022) 164–187.
[28] S. Karami, F. Saberi-Movahed, P. Tiwari, P. Marttinen, S. Vahdati, Unsupervised feature selection based on variance-covariance subspace distance, Neural Netw. 166 (2023) 188–203.
[29] H. Chen, F. Nie, R. Wang, X. Li, Fast unsupervised feature selection with bipartite graph and $l_{2,0}$-norm constraint, IEEE Trans. Knowl. Data Eng. 35 (5) (2022) 4781–4793.
[30] F. Nie, Z. Ma, J. Wang, X. Li, Fast sparse discriminative K-means for unsupervised feature selection, IEEE Trans. Neural Netw. Learn. Syst. (2023) 1–15.
[31] H. Lim, D.-W. Kim, Pairwise dependence-based unsupervised feature selection, Pattern Recognit. 111 (2021) 107663.
[32] Y. Guan, N. Zhang, C. Chu, Y. Xiao, R. Niu, C. Shao, Health impact assessment of the surface water pollution in China, Sci. Total Environ. 933 (2024) 173040.
[33] H. Feng, J.F. Schyns, M.S. Krol, M. Yang, H. Su, Y. Liu, Y. Lv, X. Zhang, K. Yang, Y. Che, Water pollution scenarios and response options for China, Sci. Total Environ. 914 (2024) 169807.
[34] Y. Lu, T. Ma, Q. Lan, B. Liu, X. Liang, Single entity collision for inorganic water pollutants measurements: Insights and prospects, Water Res. 248 (2024) 120874.
[35] L. Du, Z. Gu, Y. Wang, L. Wang, Y. Jia, A few-shot class-incremental learning method for network intrusion detection, IEEE Trans. Netw. Serv. Manag. 21 (2) (2024) 2389–2401.
[36] Y. Jia, Z. Gu, L. Du, Y. Long, Y. Wang, J. Li, Y. Zhang, Artificial intelligence enabled cyber security defense for smart cities: A novel attack detection framework based on the MDATA model, Knowl.-Based Syst. 276 (2023) 110781.
[37] X. Zhu, S. Zhang, Y. Zhu, P. Zhu, Y. Gao, Unsupervised spectral feature selection with dynamic hyper-graph learning, IEEE Trans. Knowl. Data Eng. 34 (6) (2020) 3016–3028.
[38] P. Zhu, X. Hou, K. Tang, Y. Liu, Y.-P. Zhao, Z. Wang, Unsupervised feature selection through combining graph learning and $l_{2,0}$-norm constraint, Inform. Sci. 622 (2023) 68–82.
[39] J. Wang, H. Wang, F. Nie, X. Li, Sparse feature selection via fast embedding spectral analysis, Pattern Recognit. 139 (2023) 109472.
[40] W. Xia, Q. Gao, Q. Wang, X. Gao, C. Ding, D. Tao, Tensorized bipartite graph learning for multi-view clustering, IEEE Trans. Pattern Anal. Mach. Intell. 45 (4) (2022) 5187–5202.
[41] X. Yang, M. Zhu, Y. Cai, Z. Wang, F. Nie, Fast spectral clustering with self-adapted bipartite graph learning, Inform. Sci. 644 (2023) 118810.
[42] L. Chen, C. Liu, R. Zhou, J. Xu, J. Li, Efficient maximal biclique enumeration for large sparse bipartite graphs, Proc. VLDB Endow. 15 (8) (2022) 1559–1571.
[43] L. Chen, C. Liu, R. Zhou, J. Xu, J. Li, Efficient exact algorithms for maximum balanced biclique search in bipartite graphs, in: SIGMOD/PODS '21: International Conference on Management of Data, 2021, pp. 248–260.
[44] W. Li, H. Chen, T. Li, J. Wan, B. Sang, Unsupervised feature selection via self-paced learning and low-redundant regularization, Knowl.-Based Syst. 240 (2022) 108150.
[45] S. Wang, W. Pedrycz, Q. Zhu, W. Zhu, Unsupervised feature selection via maximum projection and minimum redundancy, Knowl.-Based Syst. 75 (2015) 19–29.
[46] Y. Yang, H. Shen, Z. Ma, Z. Huang, X. Zhou, L2,1-norm regularized discriminative feature selection for unsupervised, in: International Joint Conference on Artificial Intelligence,International Joint Conference on Artificial Intelligence, 2011.
[47] X. Lin, J. Guan, B. Chen, Y. Zeng, Unsupervised feature selection via orthogonal basis clustering and local structure preserving, IEEE Trans. Neural Netw. Learn. Syst. 33 (11) (2022) 6881–6892.