Information Fusion 100 (2023) 101910

Contents lists available at ScienceDirect

Information Fusion

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

Full length article

Bi-level ensemble method for unsupervised feature selection

Peng Zhou [a,*], Xia Wang [a], Liang Du [b]

[a] Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University, Hefei 230601, China

[b] School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China

ARTICLE INFO

Keywords:
Ensemble learning
Feature selection
Clustering ensemble

ABSTRACT

Unsupervised feature selection is an important machine learning task and thus attracts increasingly more attention. However, due to the absence of labels, unsupervised feature selection often suffers from stability and robustness problems. To tackle these problems, some works try to ensemble multiple feature selection results to obtain a consensus result. Most of the existing methods do the ensemble on the feature level, i.e., they directly ensemble feature selection results by feature ranking or voting aggregation, without paying any attention to the following downstream tasks. In this paper, we take clustering as the downstream task and wish to ensemble the base results to select features which are appropriate for clustering. To this end, we propose a novel bi-level feature selection ensemble method, which ensembles on two levels: the feature level and the clustering level. Together with feature level ensemble, we also learn a consensus clustering result from base feature selection results with self-paced learning. Then, we apply the consensus clustering result to guide the feature selection in turn. Extensive experiments are conducted to demonstrate that the proposed method outperforms other state-of-the-art feature selection and feature selection ensemble methods in the clustering task. The codes of this paper are released in https://doctor-nobody.github.io/codes/BLFSE.zip.

1. Introduction

Feature selection is a fundamental and important problem in machine learning and attracts increasingly more attention [1–6]. It aims to select some informative features to facilitate the subsequent data analysis. According to the usage of data labels, feature selection can be roughly categorized into three classes: supervised feature selection [1,2,7], semi-supervised feature selection [8–11], and unsupervised feature selection [6,12]. Among them, since unsupervised feature selection does not need any label information of data, it is more attractive and also more challenging.

Since there are no labels of data, conventional unsupervised feature selection methods aim to select features to capture some intrinsic structure of the data itself. For example, some methods used the original data to construct a graph structure and selected features to preserve such graph structure [13,14]; Li et al. tried to preserve the sparse structure of data [15]; some methods generated some pseudo-labels and selected the features which are consistent with the pseudo-labels [16,17]. Although these methods have demonstrated promising performance, they still often suffer from robustness and stability problems due to the absence of labels [18]. The reasons are mainly two folds: first, different feature selection methods try to preserve different structures of data, whereas given a specific task, the ideal intrinsic structure of data is often unknown due to the absence of labels, and thus the selected features of one single method may not be those we really want; second, many unsupervised methods contain some hyper-parameters and random initializations, and different hyper-parameters and initializations may lead to very different selection results.

To address these issues, some feature selection ensemble methods are proposed [19–21]. They use one or multiple feature selection methods to generate multiple base feature selection results, and then aggregate them to generate a consensus feature selection result which is often more robust and stable. These methods often ensemble base results on feature level by feature ranking or voting aggregation. However, in many real applications, feature selection is not the terminal point of a machine learning workflow and is often followed by some downstream tasks. For example, sometimes we need to do clustering or detect the outliers of data, but the original data contain many uninformative features which may mislead the clustering or outlier detection, and thus the feature selection methods are applied to select the key features. In this scenario, clustering or outlier detection is the downstream task, and the final goal is to achieve good performance on the clustering or outlier detection instead of just feature selection. The existing feature level ensemble methods pay no attention to the

* Corresponding author.
E-mail addresses: zhoupeng@ahu.edu.cn (P. Zhou), e19201043@stu.ahu.edu.cn (X. Wang), duliang@sxu.edu.com (L. Du).

https://doi.org/10.1016/j.inffus.2023.101910
Received 26 November 2022; Received in revised form 13 April 2023; Accepted 2 July 2023
Available online 7 July 2023

downstream tasks and sometimes cannot guarantee that the selected features are suitable for the downstream tasks.

To tackle this problem, we redesign the feature selection ensemble framework by fully considering the specific downstream task. Since clustering is one of the most important unsupervised tasks, in this paper, we consider clustering as the downstream task and propose a novel Bi-Level Feature Selection Ensemble (BLFSE) method to select features suitable for clustering. Just as its name implies, it ensembles base feature selection results on two levels: feature level and clustering level. In more detail, for the feature level ensemble, given multiple base feature selection results or base feature scores, it aggregates them to learn a consensus feature score. For the clustering level, it first uses the features selected by multiple base methods to generate multiple clustering results and ensemble them to obtain a consensus clustering result to guide the feature selection. In the clustering level ensemble, since both the base feature selection methods and the clustering methods are imperfect, the base clustering results for the ensemble are also unreliable. To address this issue, we plug it into a self-paced learning framework, i.e., we use reliable data for the ensemble first, and with the learning process, the model becomes increasingly stronger and then we apply it to handle the unreliable data gradually. To make the two levels of the ensemble be boosted by each other, we integrate them into a unified feature selection objective function, and propose an iterative optimization algorithm to jointly do the bi-level ensembles and select features. Then we provide a strategy to automatically set the hyper-parameters, so that the proposed method is easy to use, especially in the unsupervised learning scenario. At last, we conduct extensive experiments by comparing with the state-of-the-art feature selection and feature selection ensemble methods. The experimental results demonstrate its effectiveness and superiority.

The main contributions are summarized as follows:

- To the best of our knowledge, we are the first to ensemble the unsupervised feature selection results in two levels, including feature and clustering levels. We seamlessly integrate the feature voting aggregation and self-paced clustering ensemble into a unified framework, which is more appropriate for the following clustering task.
- We develop an effective iterative algorithm to jointly do the bi-level ensemble. Our model can automatically adjust the hyper-parameters and does not involve any manually tuned hyper-parameters, and thus the model is easy to use.
- Extensive experiments on benchmark data sets show that the proposed bi-level ensemble method performs better than both the state-of-the-art feature selection methods and the state-of-the-art feature selection ensemble methods.

The remained parts of this paper are organized as follows. Section 2 briefly introduces some related work. Section 3 describes the proposed bi-level feature selection ensemble method in detail. Section 4 provides some experimental results. Section 5 concludes this paper.

2. Related work

In this section, we introduce some related work on unsupervised feature selection, clustering ensemble, and feature selection ensemble.

2.1. Unsupervised feature selection

Unsupervised feature selection aims to select features to preserve the intrinsic structure of data. Roughly speaking, it can be divided into three categories: filter methods, wrapper methods, and embedded methods [22].

Filter methods select the informative features based on the properties of the data itself, without using any clustering algorithms to guide the selection. They often evaluate each feature by some criteria to obtain the score of each feature, and then select features according to the scores. For example, Peng et al. used dependency, relevance, and redundancy to evaluate each feature [23]; He et al. defined a Laplacian score to indicate the importance of each feature [24]; Liu et al. selected features according to the dependency margin [25]; Yao et al. proposed a locally linear embedding score to select features [26]; Roffo et al. selected features on an infinite path among feature distributions based on the relevance and redundancy of each feature [27,28]. Since filter methods do not involve any downstream learning methods, they are often lightweight and efficient. However, also due to this, the selected features may sometimes be unsuitable for the downstream learning tasks.

Different from filter methods, wrapper, and embedded approaches apply a clustering algorithm to guide the feature selection. In wrapper methods, a specific clustering algorithm is used as a black box to assist the feature selection. For example, MacQueen et al. proposed a sequential method by applying kmeans to guide the feature subsets search [29]; Cai et al. used spectral clustering results to select features [30]; Luchian et al. selected features to minimize and maximize the intra-cluster and inter-cluster inertias, respectively [31]; Dutta et al. applied a multi-objective genetic algorithm to obtain the clusters of data and further used the clustering result to select features [32].

Embedded methods embed the feature selection procedure into a clustering algorithm and propose a unified optimization algorithm for clustering and feature selection. To preserve the graph structures, some methods embed the feature selection into spectral clustering. For example, Zhao et al. provided an efficient spectral feature selection by minimizing the redundancy [33]; Li et al. applied the nonnegative spectral analysis to embed unsupervised feature selection into the spectral clustering [34,35]; Shang et al. designed a non-negative spectral feature selection with dual graph regularization [36]; Tang et al. imposed manifold regularization on a spectral feature selection method in [37]. Subspace clustering, which is another famous clustering method, is also often used to guide the feature selection [38–41]. Zhou et al. integrated feature selection into a balanced clustering to select the features preserving the balanced structure [42,43]. Some works applied clustering ensemble to obtain a consensus clustering result and used it to select features [44,45]. In recent years, deep learning has achieved promising performance on many tasks. Hence, some works tried to apply deep neural networks to feature selection [46–49]. Since deep learning often needs a large number of labels of data, most deep feature selection methods focused on supervised learning. Few unsupervised feature selection methods often applied auto-encoder to select the features to reconstruct the data, such as [46,49]. Since wrapper and embedded methods consider the downstream clustering tasks, they often achieve a better clustering performance.

Due to the absence of labels in unsupervised feature selection, it often suffers from robustness and stability problems. To tackle these problems, this paper applies ensemble learning to feature selection, leading to the feature selection ensemble.

2.2. Clustering ensemble

Clustering ensemble takes multiple clustering results as inputs and learns a consensus clustering result from them by considering the consensus and diversity [50]. Since clustering ensemble can improve the reliability of the single clustering method via improving the robustness and stability, it has been widely studied [51–56].

Some clustering ensemble methods regard the multiple clustering results as the new features of data and directly apply some clustering methods to obtain the consensus results. For example, Topchy et al. proposed an expectation–maximization method to obtain the consensus result [57]; Nguyen et al. applied K-modes clustering method on the base results to obtain the final result [58]. Some methods applied the alignment method to do the ensemble. For example, Zhou et al. proposed a method to align the multiple results obtained by kmeans [53];

Table 1
Notations and descriptions used in our method.

Notation	Description
n, d, c	Number of instances, features, and clusters, respectively.
$\mathbf{X} \in \mathbb{R}^{d\times n}$	Data matrix, where each column represents an instance.
$\mathbf{v}^{(k)} \in [0,1]^d$	The kth base feature selection result.
$\mathbf{v} \in [0,1]^d$	Consensus feature selection result.
$\mathbf{Y}^{(k)} \in \{0,1\}^d$	The kth base clustering result.
$\mathbf{S}^{(k)} \in \{0,1\}^{n\times n}$	The connective matrix of the kth base clustering.
$\mathbf{S} \in [0,1]^{n\times n}$	The consensus matrix.
$\mathbf{L} \in \mathbb{R}^{n\times n}$	The Laplacian matrix of $\mathbf{S}$.
$\mathbf{W} \in [0,1]^{n\times n}$	The weight matrix.
$\mathbf{P} \in \mathbb{R}^{c\times d}$	The projection matrix.

Li et al. relabeled the data according to the Dempster–Shafer evidence theory to do the alignment [59].

One of the most popular clustering ensemble methods is the graph based or connective matrix based method. These methods construct the graph or connective matrices from the base results and obtain the consensus result from the graph or connective matrices. For example, Tao et al. proposed a robust spectral clustering ensemble method on the connective matrices [60]; Zhou et al. proposed a self-paced clustering ensemble method on the graph constructed from the connective matrices [61] and further designed an active clustering ensemble method [62]; Huang et al. designed an ultra-scalable spectral clustering method on the bipartite graph of the base results [63,64].

In this paper, we introduce the idea of clustering ensemble in the feature selection task and apply it to the clustering level ensemble for feature selection.

2.3. Feature selection ensemble

Inspired by the idea of ensemble learning [65–69], few works ensemble multiple base feature selection results to learn a more robust and stable result. The existing feature selection ensemble methods often ensemble multiple base selection results only on feature level, i.e., they directly aggregate the feature scores or feature rankings. For example, Hong et al. linearly combined the score of each base feature selection result to obtain a consensus one [19]; Zhang et al. applied the consensus affinity to evaluate the importance of features [20]; Seijo-Pardo et al. proposed feature selection methods by feature ranking combination [21]; Das et al. developed a bi-objective genetic algorithm to ensemble feature selection [70]; Chiew et al. provided a hybrid ensemble method to combine both the homogeneous and heterogeneous base feature selection results [71].

In this paper, we propose a novel feature selection ensemble method. Different from the above-mentioned feature level ensemble methods, ours ensembles feature selection results on both the feature level and clustering level. Notice that, as introduced before, some methods apply clustering ensemble to feature selection [44,45]. Strictly speaking, they are embedded feature selection methods instead of feature selection ensemble methods, because they do not generate multiple base feature selection results and thus do not ensemble feature selection results. In our experiments, we also compare with them to show the superiority of the bi-level feature selection ensemble method.

3. Bi-level feature selection ensemble

In this section, we introduce our BLFSE in more detail. Firstly, we briefly introduce some notations. Boldface uppercase and lowercase letters are used to denote matrices and vectors, respectively. For a matrix $\mathbf{M}$, $\mathbf{M}_{i\cdot}$ and $\mathbf{M}_{\cdot i}$ denote the ith row and column of $\mathbf{M}$, respectively. The (i,j)-th element of $\mathbf{M}$ is denoted as M_{ij}. The main notations used in our method are summarized in Table 1.

Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$ denote a data set containing n instances and d features. We can use some lightweight filter feature selection methods, such as [2,24,27], to generate m base feature selection results $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(m)}$, where $\mathbf{v}^{(k)} \in [0,1]^d$ indicates the scores of all features computed by the kth base feature selection method (note that we normalize the score into the range $[0,1]$). For the methods which do not return the scores of features, we can easily set $v_i^{(k)} = 1$ if the ith feature is selected by the kth method, and $v_i^{(k)} = 0$ otherwise. Then, we will ensemble the m feature selection results to learn a consensus score $\mathbf{v} \in [0,1]^d$.

Fig. 1 shows the framework of BLFSE. It ensembles the results on two levels: feature and clustering levels. At the feature level, it directly ensembles the feature scores to learn a consensus score. At the clustering level, it uses the selected features to generate a clustering result, and ensembles them to learn a consensus clustering result to guide the feature level ensemble. In the following, we will introduce the two levels of the ensemble in more detail.

3.1. Feature level ensemble

In feature level ensemble, we directly ensemble $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(m)}$ to learn the consensus $\mathbf{v}$. The key idea is that we wish the consensus one to be similar to each base result, and this can be achieved by minimizing $\sum_{k=1}^{n} \|\mathbf{v} - \mathbf{v}^{(k)}\|_2^2$. However, the quality of each base result differs from each other, and we wish the consensus one to be more similar to the better results rather than the worse ones. Therefore, we impose a weight $\alpha_k \in [0,1]$ on each base result, and learn the consensus one by the following formula:

$$\min_{\mathbf{v},\boldsymbol{\alpha}} \quad \sum_{k=1}^{m} \alpha_k^2 \|\mathbf{v} - \mathbf{v}^{(k)}\|_2^2 \tag{1}$$

$$s.t. \quad 0 \le v_i \le 1, \ \sum_{i=1}^{d} v_i = 1, \ 0 \le \alpha_i \le 1, \ \sum_{k=1}^{m} \alpha_k = 1.$$

Note that we impose the constraint $\sum_{i=1}^{d} v_i = 1$ on $\mathbf{v}$, for the purpose that features will compete with each other and the worse ones will be more easily killed in the competition, leading to a sparse $\mathbf{v}$.

To see this, we can rewrite the objective function as $\sum_{k=1}^{m} \alpha_k^2 \|\mathbf{v} - \mathbf{v}^{(k)}\|_2^2 = \left\|\mathbf{v} - \frac{\sum_{k=1}^{m} \alpha_k^2 \mathbf{v}^{(k)}}{\sum_{k=1}^{m} \alpha_k^2}\right\|_2^2$. When considering optimizing $\mathbf{v}$, it is a Euclidean projection onto a simplex, which leads to a sparse solution. Small values in $\frac{\sum_{k=1}^{m} \alpha_k^2 \mathbf{v}^{(k)}}{\sum_{k=1}^{m} \alpha_k^2}$ (which is a weighted average score of $\mathbf{v}^{(k)}$) tends to be smaller. Intuitively, the worse features are those whose weighted average score is small, which means the worse features are ones that most base results agree that they are useless.

3.2. Clustering level ensemble

Since we often use the feature selection results for the clustering task, besides the feature level ensemble, we also ensemble on the clustering level for the purpose that the selected features could be more suitable for clustering. Specifically, for each base feature selection result $\mathbf{v}^{(k)}$, we run some off-the-shelf clustering methods on the data with the selected features to obtain a base clustering result $\mathbf{Y}^{(k)} \in \{0,1\}^{n\times c_k}$, where c_k is the number of clusters in the kth clustering result. $Y_{ij}^{(k)} = 1$ if the ith instance belongs to the jth cluster in the kth result, and $Y_{ij}^{(k)} = 0$ otherwise.

Since for two base clustering results $\mathbf{Y}^{(i)}$ and $\mathbf{Y}^{(j)}$ there may not exist a one-to-one match between their clusters, directly ensembling $\mathbf{Y}^{(k)}$ may be difficult. To address this issue, following [60,61,63,72], we generate the *connective matrix* $\mathbf{S}^{(k)} \in \{0,1\}^{n\times n}$ form $\mathbf{Y}^{(k)}$ by $\mathbf{S}^{(k)} = \mathbf{Y}^{(k)}\mathbf{Y}^{(k)T}$. $S_{ij}^{(k)} = 1$ if in the kth result $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same cluster, and $S_{ij}^{(k)} = 0$ otherwise. Then we ensemble $\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(m)}$ to learn a consensus matrix $\mathbf{S} \in [0,1]^{n\times n}$.

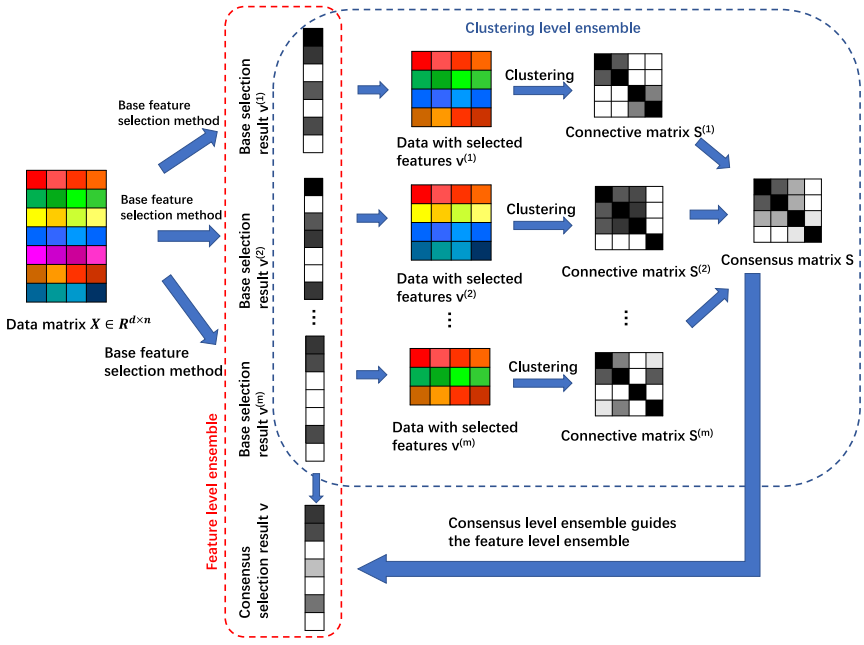


Fig. 1. The framework of BLFSE. Given a data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$, it applies base feature selection methods to generate multiple base results $\mathbf{v}^{(i)}$. In the feature level ensemble, denoted by the circle marked with a red-dotted line, it ensembles multiple $\mathbf{v}^{(i)}$ to a consensus result $\mathbf{v}$; in the clustering level ensemble, denoted by the circle marked with a blue-dotted line, with each base feature selection result, it applies off-the-shelf clustering methods on the data with selected features to generate multiple base connective matrices $\mathbf{S}^{(i)}$ and ensemble them to learn a consensus $\mathbf{S}$ to guide the feature level ensemble.

Intuitively, we can minimize $\sum_{k=1}^{m}\beta_k^2\|\mathbf{S}-\mathbf{S}^{(k)}\|_F^2$ to learn the consensus $\mathbf{S}$, where β_k is the weight of the kth clustering result to make better base results contribute more to the consensus one. However, unlike the feature level ensemble, directly optimizing it may not obtain the ideal result. The problem involves too many variables ($n\times n$ variables in $\mathbf{S}$) to be learned from $m\times n\times n$ data ($\mathbf{S}^{(1)},\ldots,\mathbf{S}^{(m)}$). What is worse, most of them are unreliable. Considering that the base unsupervised feature selection result $\mathbf{v}^{(k)}$ may be unreliable, the clustering result $\mathbf{Y}^{(k)}$ generated from it may also be unreliable and this leads to the low quality of the connective matrix $\mathbf{S}^{(k)}$. To alleviate this unreliability diffusion, we integrate ensemble learning into a self-paced learning framework. The key idea is that we gradually involve data in ensemble learning from more reliable ones to less reliable ones. In the beginning, the early model may be too weak to handle the unreliable data. Thus, we should use reliable data for learning. Then, during ensemble learning, the model becomes stronger and stronger, and thus it can handle some less reliable data.

To achieve this, we impose a weight matrix $\mathbf{W} \in [0,1]^{n\times n}$ on $\mathbf{S}$ to indicate the reliability of each instance pair and automatically determine it in the ensemble learning. Intuitively, for the instance pair $(\mathbf{x}_i,\mathbf{x}_j)$, if most of $\mathbf{S}^{(k)}$'s agree with each other that they belong to or do not belong to the same cluster, then the learned S_{ij} may be reliable, which leads to a large W_{ij}. More formally, we minimize the following problem:

$$\begin{aligned}\min_{\mathbf{W},\mathbf{S},\boldsymbol{\beta}}\ & \sum_{k=1}^{m}\beta_k^2\|\mathbf{W}\odot(\mathbf{S}-\mathbf{S}^{(k)})\|_F^2-\lambda\|\mathbf{W}\|_1,\\ s.t.\ & 0\le W_{ij}\le 1,\quad 0\le S_{ij}\le 1,\quad \mathbf{S}=\mathbf{S}^T,\\ & 0\le\beta_k\le 1,\quad \sum_{k=1}^{m}\beta_k=1,\end{aligned}\tag{2}$$

where $\odot$ denotes the Hadamard product, which is the element-wise production of two matrices. The second term is the self-paced regularized term and $\lambda>0$ is an adaptive parameter that grows in the process of the learning, as suggested by [61,73,74]. From the following optimization subsection, we can see that $\mathbf{W}$ will get larger and larger with λ growing, which means more and more data will be involved in the learning. The constraint $\mathbf{S}=\mathbf{S}^T$ makes sure $\mathbf{S}$ is symmetric as $\mathbf{S}^{(1)},\ldots,\mathbf{S}^{(m)}$ are.

Supposing in the clustering task, we want to partition the data into c clusters. To make $\mathbf{S}$ reflect such clustering structure more clearly, we wish that $\mathbf{S}$ contains just c connective components. To achieve this, we first compute its Laplacian matrix $\mathbf{L}=\mathbf{D}-\mathbf{S}$, where $\mathbf{D}$ is a diagonal matrix whose diagonal elements $D_{ii}=\sum_{j=1}^{n}S_{ij}$. Then, according to [75], if $\mathbf{S}$ is nonnegative and symmetric, the number of connective components in $\mathbf{S}$ is equal to n minus the rank of $\mathbf{L}$. Thus, we need a constraint $rank(\mathbf{L})=n-c$ in the objective function.

3.3. Objective function

After obtaining the consensus matrix $\mathbf{S}$, we wish to select the features to preserve the clustering structure $\mathbf{S}$. To this end, we first generate the weighted instance matrix $diag(\mathbf{v})\mathbf{X}$, where $diag(\mathbf{v})$ is a diagonal matrix whose diagonal vector is the consensus feature scoew $\mathbf{v}$. Then we define an orthogonal transformation matrix $\mathbf{P}\in\mathbb{R}^{c\times d}$ to project the weighted instances into a new subspace to preserve $\mathbf{S}$.

In more detail, for each weighted instance pair $diag(\mathbf{v})\mathbf{x}_i$ and $diag(\mathbf{v})\mathbf{x}_j$, if $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same cluster according to $\mathbf{S}$, we wish the projected instance pair $\mathbf{P}diag(\mathbf{v})\mathbf{x}_i$ and $\mathbf{P}diag(\mathbf{v})\mathbf{x}_j$ could be close to each other. To achieve this, we can minimize $\sum_{i,j=1}^{n}\|\mathbf{P}diag(\mathbf{v})\mathbf{x}_i-\mathbf{P}diag(\mathbf{v})\mathbf{x}_j\|_2^2S_{ij}$. Integrating this term into the feature level ensemble and clustering level ensemble, we can obtain the final objective function of the proposed method:

$$\begin{aligned}\min_{\boldsymbol{\theta}}\ & \sum_{i,j=1}^{n}\|\mathbf{P}diag(\mathbf{v})\mathbf{x}_i-\mathbf{P}diag(\mathbf{v})\mathbf{x}_j\|_2^2S_{ij}+\sum_{k=1}^{m}\beta_k^2\|\mathbf{W}\odot(\mathbf{S}-\mathbf{S}^{(k)})\|_F^2\\ & -\lambda\|\mathbf{W}\|_1+\sum_{k=1}^{m}\alpha_k^2\|\mathbf{v}-\mathbf{v}^{(k)}\|_2^2,\\ s.t.\ & 0\le v_i\le 1,\ \sum_{i=1}^{d}v_i=1,\ 0\le\alpha_i\le 1,\ \sum_{k=1}^{m}\alpha_k=1,\ 0\le\beta_k\le 1,\ \sum_{k=1}^{m}\beta_k=1,\\ & 0\le W_{ij}\le 1,\ 0\le S_{ij}\le 1,\ \mathbf{S}=\mathbf{S}^T,\ rank(\mathbf{L})=n-c,\ \mathbf{P}\mathbf{P}^T=\mathbf{I},\end{aligned}\tag{3}$$

where $\boldsymbol{\theta}$ is the set of all learned parameters, i.e., $\boldsymbol{\theta}=\{\mathbf{v},\mathbf{S},\mathbf{W},\mathbf{P},\boldsymbol{\alpha},\boldsymbol{\beta}\}$. By Eq. (3), we seamlessly integrate the feature level ensemble and clustering level ensemble into a unified feature selection framework. In

this framework, we jointly ensemble the feature scores and clustering results and use the ensemble results to guide the feature selection, so that the ensemble and selection can be boosted by each other.

3.4. Optimization

We first handle the rank constraint. According to Ky Fan Theorem [76], by introducing an orthogonal matrix $\mathbf{Y} \in \mathbb{R}^{n\times c}$, Eq. (3) can be rewritten as:

$$\begin{aligned}
&\min_{\theta} \sum_{i,j=1}^{n} \|\mathbf{P}diag(\mathbf{v})\mathbf{x}_i - \mathbf{P}diag(\mathbf{v})\mathbf{x}_j\|_2^2 S_{ij} - \lambda\|\mathbf{W}\|_1 \\
&\quad + \sum_{k=1}^{m} \beta_k^2 \|\mathbf{W}\odot(\mathbf{S}-\mathbf{S}^{(k)})\|_F^2 + \sum_{k=1}^{m} \alpha_k^2 \|\mathbf{v}-\mathbf{v}^{(k)}\|_2^2 + 2\rho tr(\mathbf{Y}^T\mathbf{L}\mathbf{Y}), \\
&s.t.\ 0 \le v_i \le 1, \quad \sum_{i=1}^{d} v_i = 1, \quad 0 \le \alpha_i \le 1, \quad \sum_{k=1}^{m} \alpha_k = 1, \\
&\quad 0 \le W_{ij} \le 1,\ 0 \le S_{ij} \le 1,\ \mathbf{S} = \mathbf{S}^T,\ \mathbf{Y}^T\mathbf{Y} = \mathbf{I}, \\
&\quad 0 \le \beta_k \le 1, \quad \sum_{k=1}^{m} \beta_k = 1, \quad \mathbf{P}\mathbf{P}^T = \mathbf{I},
\end{aligned} \tag{4}$$

where $\theta = \{\mathbf{v}, \mathbf{S}, \mathbf{W}, \mathbf{P}, \mathbf{Y}, \boldsymbol{\alpha}, \boldsymbol{\beta}\}$, and ρ is a large enough hyper-parameter to make $rank(\mathbf{L}) = n-c$. Then, we optimize Eq. (4) by a block coordinate descent method.

3.4.1. Optimizing W

When fixing other variables, the subproblem w.r.t. $\mathbf{W}$ can be decoupled into $n\times n$ independent subproblems, and we consider the (i,j)-th subproblem:

$$\min_{0\le W_{ij}\le 1} \quad W_{ij}^2 \sum_{k=1}^{m} (S_{ij} - S_{ij}^{(k)})^2 - \lambda W_{ij}. \tag{5}$$

By setting its derivative w.r.t. W_{ij} to zero, we obtain $W_{ij} = \frac{\lambda}{2A_{ij}}$, where $A_{ij} = \sum_{k=1}^{m}(S_{ij} - S_{ij}^{(k)})^2$. Since $A_{ij} \ge 0$, $W_{ij} \ge 0$. If $\frac{\lambda}{2A_{ij}} > 1$, then in the range $[0,1]$, Eq. (5) is a monotone decreasing function, and thus the solution is 1. To sum up, the solution of W_{ij} is:

$$W_{ij} = \min\left(\frac{\lambda}{2A_{ij}}, 1\right), \tag{6}$$

Note that, a small A_{ij} means most $\mathbf{S}^{(i)}$'s agree with each other, and thus leads to a large W_{ij} which means the pair $(\mathbf{x}_i, \mathbf{x}_j)$ is reliable. Moreover, λ represents the "age" of the model. $\mathbf{W}$ is proportional to λ, which means at the early stage (λ is small), most pairs have a small weight and only a few reliable ones (where A_{ij} is small) will have the large weights. With λ growing, more pairs with large A_{ij} will have large weights and affect the model. This is consistent with the motivation of self-paced learning.

3.4.2. Optimizing P

When fixing the other variables, we obtain the following subproblem:

$$\min_{\mathbf{P}\mathbf{P}^T=\mathbf{I}} tr(\mathbf{P}diag(\mathbf{v})\mathbf{X}\mathbf{L}\mathbf{X}^T diag(\mathbf{v})\mathbf{P}^T). \tag{7}$$

According to Ky Fan Theorem [76], the closed-form solution of $\mathbf{P}$ is the c eigenvectors of $diag(\mathbf{v})\mathbf{X}\mathbf{L}\mathbf{X}^T diag(\mathbf{v})$ corresponding to the c smallest eigenvalues.

3.4.3. Optimizing S

Note that $\mathbf{L}$ is relative with $\mathbf{S}$, and thus the subproblem w.r.t $\mathbf{S}$ are complicated. Fortunately, the following Theorem provides its closed-form solution:

Theorem 1. *Denoting* $B_{ij} = \|\mathbf{P}diag(\mathbf{v})\mathbf{x}_i - \mathbf{P}diag(\mathbf{v})\mathbf{x}_j\|_2^2$ *and* $C_{ij} = \|\mathbf{Y}_{i\cdot} - \mathbf{Y}_{j\cdot}\|_2^2$, *the closed-form solution of the subproblem w.r.t.* $\mathbf{S}$ *is*

$$S_{ij} = \max\left(\min\left(\frac{\sum_{k=1}^{m}\beta_k^2 S_{ij}^{(k)} - \frac{B_{ij}+\rho C_{ij}}{2W_{ij}^2}}{\sum_{k=1}^{m}\beta_k^2}, 1\right), 0\right). \tag{8}$$

Proof. See Appendix. □

3.4.4. Optimizing v

When fixing the other variables, we can rewrite Eq. (4) as follows:

$$\begin{aligned}
&\min_{\mathbf{v}} \sum_{i,j=1}^{n} \|\mathbf{P}diag(\mathbf{v})\mathbf{x}_i - \mathbf{P}diag(\mathbf{v})\mathbf{x}_j\|_2^2 S_{ij} + \sum_{k=1}^{m}\alpha_k^2\|\mathbf{v}-\mathbf{v}^{(k)}\|_2^2 \\
&s.t. \quad 0 \le v_i \le 1, \quad \sum_{i=1}^{d} v_i = 1.
\end{aligned} \tag{9}$$

Although Eq. (9) seems complicated, the following Theorem shows that it is strictly convex w.r.t. $\mathbf{v}$.

Theorem 2. *Eq. (9) is strictly convex quadratic programming.*

Proof. See Appendix. □

Since Eq. (9) is strictly convex quadratic programming, we can use standard convex optimization, such as the accelerated penalty method [77], to find the global solution to this subproblem.

3.4.5. Optimizing Y

The subproblem which involves $\mathbf{Y}$ is

$$\min_{\mathbf{Y}^T\mathbf{Y}=\mathbf{I}} tr(\mathbf{Y}^T\mathbf{L}\mathbf{Y}). \tag{10}$$

Similar to the optimization of $\mathbf{P}$, it can also be solved by Ky Fan Theorem. The closed-form solution is the c eigenvectors of $\mathbf{L}$ corresponding to the c smallest eigenvalues.

3.4.6. Optimizing α

When fixing the other variables, we rewrite Eq. (4) as:

$$\min_{\boldsymbol{\alpha}} \sum_{k=1}^{m}\alpha_k^2\|\mathbf{v}-\mathbf{v}^{(k)}\|_2^2, \quad s.t.\ 0 \le \alpha_i \le 1, \ \sum_{k=1}^{m}\alpha_k = 1. \tag{11}$$

According to Cauchy–Schwarz Inequality, we can obtain its closed-form solution:

$$\alpha_k = \frac{\|\mathbf{v}-\mathbf{v}^{(k)}\|_2^{-2}}{\sum_{j=1}^{m}\|\mathbf{v}-\mathbf{v}^{(j)}\|_2^{-2}}. \tag{12}$$

3.4.7. Optimizing β

The optimization of $\boldsymbol{\beta}$ is similar to $\boldsymbol{\alpha}$. The closed-form solution of $\boldsymbol{\beta}$ is:

$$\beta_k = \frac{\|\mathbf{W}\odot(\mathbf{S}-\mathbf{S}^{(k)})\|_F^{-2}}{\sum_{j=1}^{m}\|\mathbf{W}\odot(\mathbf{S}-\mathbf{S}^{(j)})\|_F^{-2}}. \tag{13}$$

3.5. Discussion and algorithm

We first introduce some initialization of the proposed method. We initialize $\mathbf{v} = \frac{1}{m}\sum_{k=1}^{m}\mathbf{v}^{(k)}$, $\mathbf{S} = \frac{1}{m}\sum_{k=1}^{m}\mathbf{S}^{(k)}$, $\alpha_k = \frac{1}{m}$ and $\beta_k = \frac{1}{m}$.

For the adaptive parameter λ which will influence the reliability matrix $\mathbf{W}$, we should initialize and adjust it more carefully. At the beginning, $\mathbf{S}$ is initialized as the mean of $\mathbf{S}^{(k)}$ and $\alpha_k = \frac{1}{m}$, and then we take a closer look at $\mathbf{A}$ in Eq. (6). Given any pair $(\mathbf{x}_i, \mathbf{x}_j)$, we suppose that m_{ij} clustering results agree that they belong to the same cluster and

Algorithm 1 BLFSE Algorithm

Input: Instance matrix $\mathbf{X}$, and m feature score vectors $\mathbf{v}^{(1)}, \cdots, \mathbf{v}^{(m)}$.
Output: Selected features.

```
On each base feature selection result, run off-the-shelf clustering methods to obtain base clustering results, and further obtain connective matrices S^(1), ..., S^(m).
Initialize the parameters as introduced before.
for ψ = 0.9, 0.8, ..., 0.5 do
    Compute λ by Eq. (14), and compute W by Eq. (6).
    while not converge do
        Compute P by solving Eq. (7).
        Compute S by Eq. (8).
        Compute v by solving Eq. (9).
        Compute Y by solving Eq. (10).
        Compute α and β by Eqs. (12) and (13), respectively.
        Adjust ρ as introduced before.
    end while
end for
Select the top features according to v.
```

Table 2
Information of the data sets.

	#instances	#features	#classes
20NG	3970	1000	4
BBC	737	1000	5
CSTR	475	1000	4
Isolet	1560	617	26
PIE	1428	1024	68
WEBACE	2340	1000	20
Tr11	414	6429	9
Tr12	313	5804	8

the other $m - m_{ij}$ results believe that they belong to different clusters. Then A_{ij} can be calculated as:

$$A_{ij} = \sum_{k=1}^{m} \alpha_k^2 (S_{ij} - S_{ij}^{(k)})^2 = \left(\left(\frac{m_{ij}}{m} - 1 \right)^2 \frac{m_{ij}}{m} + \left(\frac{m_{ij}}{m} \right)^2 \left(1 - \frac{m_{ij}}{m} \right) \right) \frac{1}{m}.$$

Define $\psi = m_{ij}/m$, which indicates the ratio of the results that reach an agreement. For instance, $\psi = 0.7$ indicates that 70% results agree that $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same cluster. Therefore, when $\psi > 0.5$, the larger ψ is, the more reliable the pair is. To this end, we initialize $\psi = 0.9$ and compute λ as:

$$\lambda = 2((\psi - 1)^2 \psi + \psi^2 (1 - \psi))/m = 2(\psi(1 - \psi))/m. \tag{14}$$

Taking it back to Eq. (6), we find that, for $\mathbf{x}_i$ and $\mathbf{x}_j$, if more than 90% clustering results reach an agreement, then $W_{ij} = 1$, i.e., this pair is used completely. In the following process, λ is increased gradually by decreasing ψ from 0.9 to 0.5 with a step size of 0.1.

When considering ρ, it is initialized as 1. Then, it is adjusted according to the rank of $\mathbf{L}$. In more detail, if $rank(\mathbf{L}) > n - c$, which means the rank regularization is not strong enough, we update $\rho \leftarrow 2\rho$. If $rank(\mathbf{L}) < n - c$, which means the constraint is too strong, we update it by $\rho \leftarrow \rho/2$.

Note that λ and ρ are adjusted automatically, and the number of clusters c is often given by the downstream clustering task. We do not need any other manually tuned hyper-parameters, which is practical in unsupervised learning scenarios. The whole algorithm is summarized in Algorithm 1. The most expensive steps of Algorithm 1 are the eigenvalue decompositions (optimizing $\mathbf{P}$ and $\mathbf{Y}$) and the optimization of the quadratic programming (optimizing $\mathbf{v}$). When solving $\mathbf{P}$, it costs $O(n^2 d + d^2 n)$ time to compute the matrix multiplication $\mathbf{X}^T \mathbf{L} \mathbf{X}$, and $O(d^2 c)$ time for eigenvalue decomposition. When solving $\mathbf{Y}$, it costs $O(n^2 c)$ time for eigenvalue decomposition. When solving $\mathbf{v}$, since Eq. (9) is strictly convex quadratic programming according to Theorem 2, it has a faster convergence rate when applying accelerated penalty method [77]. In each iteration to solve $\mathbf{v}$, it costs $O(d^2)$ time to compute the gradients. Since it is strictly convex, given a predefined tolerance ϵ, it needs $K = O(\frac{1}{\sqrt{\epsilon}})$ step to converge. Therefore, the time complexity of updating $\mathbf{v}$ is $O(d^2 K)$. To sum up, the whole time complexity is $O(n^2 d + n d^2 + d^2 K)$ which is not worse than the conventional embedded feature selection methods (e.g. [13,78]) whose time complexity is often cubic in d. In the future, we will further study how to reduce its time complexity.

4. Experiments

In this section, we compare BLFSE with some state-of-the-art unsupervised feature selection and ensemble methods on benchmark data sets.

4.1. Data sets

We conduct experiments on 8 benchmark data sets, including 20NG [84], BBC [85], CSTR [86], Isolet [87], PIE [88], WEBACE [89], Tr11 [90], and Tr12 [90].

Table 2 provides detailed information of these data sets.

4.2. Compared methods

We compare with the following feature selection and ensemble methods:

- **RCE** [44], which first applies clustering ensemble to generate the consensus clustering result and then selects features to preserve it.
- **FSASL** [79], which adaptively learns the local and global structure when selecting features.
- **SOGFS** [13], which learns an optimal graph structure for feature selection.
- **LRPFS** [80], which selects features to preserve the low rank structure.
- **URAFS** [14], which applies generalized uncorrelated regression to select features.
- **CGUFS** [45], which applies clustering ensemble to guide the feature selection.
- **NSSLFS** [38], which uses sparse subspace learning to guide the feature selection.
- **FSE** [18], which is a feature selection ensemble method with ranking aggregation of features.
- **HEFS** [71], which is a hybrid ensemble feature selection method to aggregate the feature scores.
- **AEFS** [46], which is an unsupervised feature selection method with auto-encoder.
- **LDSSL** [81], which is a feature selection method with local discriminative based sparse subspace learning.
- **SLSDR** [82], which is a robust feature selection method with sparse and low-redundant subspace learning.
- **DUFS** [83], which is an unsupervised feature selection method by considering pairwise dependence.
- **UFSTAE** [49], which is an unsupervised feature selection method via transformed auto-encoder.
- **BSFS** [43], which is a balanced spectral feature selection method.

Among them, RCE and CGUFS are feature selection methods based on the clustering ensemble; and FSE and HEFS are feature selection ensemble methods on the feature level.

4.3. Experimental setup

To evaluate the quality of the features selected by each method, we apply kmeans clustering on the selected features and report the

Table 3
ACC results compared with other feature selection methods. The bold font means that the difference is statically significant, i.e., the p-value of the t-test is smaller than 0.05. The numbers in the parentheses are the p-values.

Methods	20NG	BBC	CSTR	Isolet	PIE	WEBACE	Tr11	TR12
RCE [44]	0.4696 (0.0043)	0.5793 (0.0000)	0.5695 (0.0002)	0.4548 (0.0013)	0.4500 (0.0000)	0.3861 (0.0000)	0.3598 (0.0000)	0.3701 (0.0000)
FSASL [79]	0.4708 (0.0047)	0.6168 (0.0014)	0.5621 (0.0000)	0.4838 (0.0229)	0.4996 (0.0000)	0.3807 (0.0000)	**0.4903** (0.4484)	0.4858 (0.0000)
SOGFS [13]	0.3606 (0.0000)	0.4453 (0.0000)	0.3843 (0.0000)	0.3750 (0.0000)	0.3435 (0.0000)	0.2931 (0.0000)	0.3021 (0.0000)	0.2855 (0.0000)
LRPFS [80]	0.3159 (0.0000)	0.4603 (0.0000)	0.5186 (0.0000)	0.4587 (0.0007)	0.3345 (0.0000)	0.2447 (0.0000)	0.2950 (0.0000)	0.2369 (0.0000)
URAFS [14]	0.4712 (0.0041)	0.4749 (0.0000)	0.5299 (0.0000)	0.4257 (0.0000)	0.4853 (0.0000)	0.3832 (0.0000)	0.4719 (0.0252)	0.4815 (0.0000)
CGUFS [45]	0.3087 (0.0000)	0.4764 (0.0000)	0.5830 (0.0001)	0.4651 (0.0032)	0.3450 (0.0000)	0.3405 (0.0000)	0.4045 (0.0000)	0.4459 (0.0000)
NSSLFS [38]	0.2809 (0.0000)	0.3768 (0.0000)	0.3792 (0.0000)	0.4485 (0.0000)	0.3127 (0.0000)	0.3549 (0.0000)	0.3221 (0.0000)	0.2910 (0.0000)
FSE [18]	0.4632 (0.0004)	0.5874 (0.0000)	0.5723 (0.0000)	0.4617 (0.0000)	0.5857 (0.0000)	0.3898 (0.0000)	**0.4975** (0.7416)	0.4809 (0.0000)
HEFS [71]	0.4748 (0.0158)	0.6159 (0.0004)	0.6117 (0.0192)	0.4291 (0.0000)	0.5571 (0.0000)	0.3936 (0.0000)	0.3710 (0.0000)	0.4234 (0.0000)
AEFS [46]	0.2790 (0.0000)	0.4064 (0.0000)	0.4337 (0.0000)	0.4585 (0.0010)	0.3476 (0.0000)	0.3236 (0.0000)	0.2092 (0.0000)	0.2474 (0.0000)
LDSSL [81]	0.4417 (0.0003)	0.6203 (0.0025)	0.5921 (0.0019)	0.4794 (0.0157)	0.4886 (0.0000)	0.4178 (0.0000)	**0.4893** (0.3136)	0.4843 (0.0000)
SLSDR [82]	0.3917 (0.0000)	0.3464 (0.0000)	**0.6223** (0.1174)	**0.4843** (0.2032)	0.4426 (0.0000)	0.3061 (0.0000)	0.3077 (0.0000)	0.2890 (0.0000)
DUFS [83]	0.4645 (0.0011)	0.6755 (0.0420)	0.5631 (0.0000)	0.4677 (0.0010)	0.3336 (0.0000)	0.3446 (0.0000)	**0.4875** (0.2586)	0.4748 (0.0000)
UFSTAE [49]	– –	0.4275 (0.0000)	0.4243 (0.0000)	0.4629 (0.0074)	0.5056 (0.0000)	**0.4631** –	0.3717 (0.0136)	0.3860 (0.0000)
BSFS [43]	0.4780 (0.0108)	0.6200 (0.0010)	**0.6293** (0.2785)	0.4185 (0.0000)	0.5269 (0.0000)	0.3877 (0.0000)	0.4728 (0.0408)	0.4710 (0.0000)
BLFSE	**0.5239** –	**0.6936** –	**0.6451** –	**0.5006** –	**0.6325** –	0.4075 (0.0000)	**0.5013** –	**0.5450** –

clustering performance. Two clustering metrics including Accuracy (ACC) and Normalized Mutual Information (NMI) are used. Since it is often difficult to know the optimal number of selected features in advance, the results with $10, 20, \ldots, 200$ selected features are reported.

In our method, we use a lightweight filter feature selection method Inf-FS (Infinite Feature Selection) proposed in [27] to generate the base feature selection results. In more detail, to generate diverse base feature selection results, we randomly split the data set into 10 subsets, with $1/10$ instances in each one. Then, in each subset, we run Inf-FS to generate a base feature score vector, and thus we get 10 score vectors $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(10)}$. Then, we apply kmeans to generate the base clustering results for simplicity. We run our BLFSE to ensemble the 10 base score vectors to select features. For other feature selection ensemble methods (i.e., FSE and HEFS), since we aim to compare the performance of ensemble learning, we use the same base feature selection results as used in ours for a fair comparison. For the RCE and CGUFS, which are clustering ensemble based feature selection methods, we use the same base clustering results as ours uses for a fair comparison.

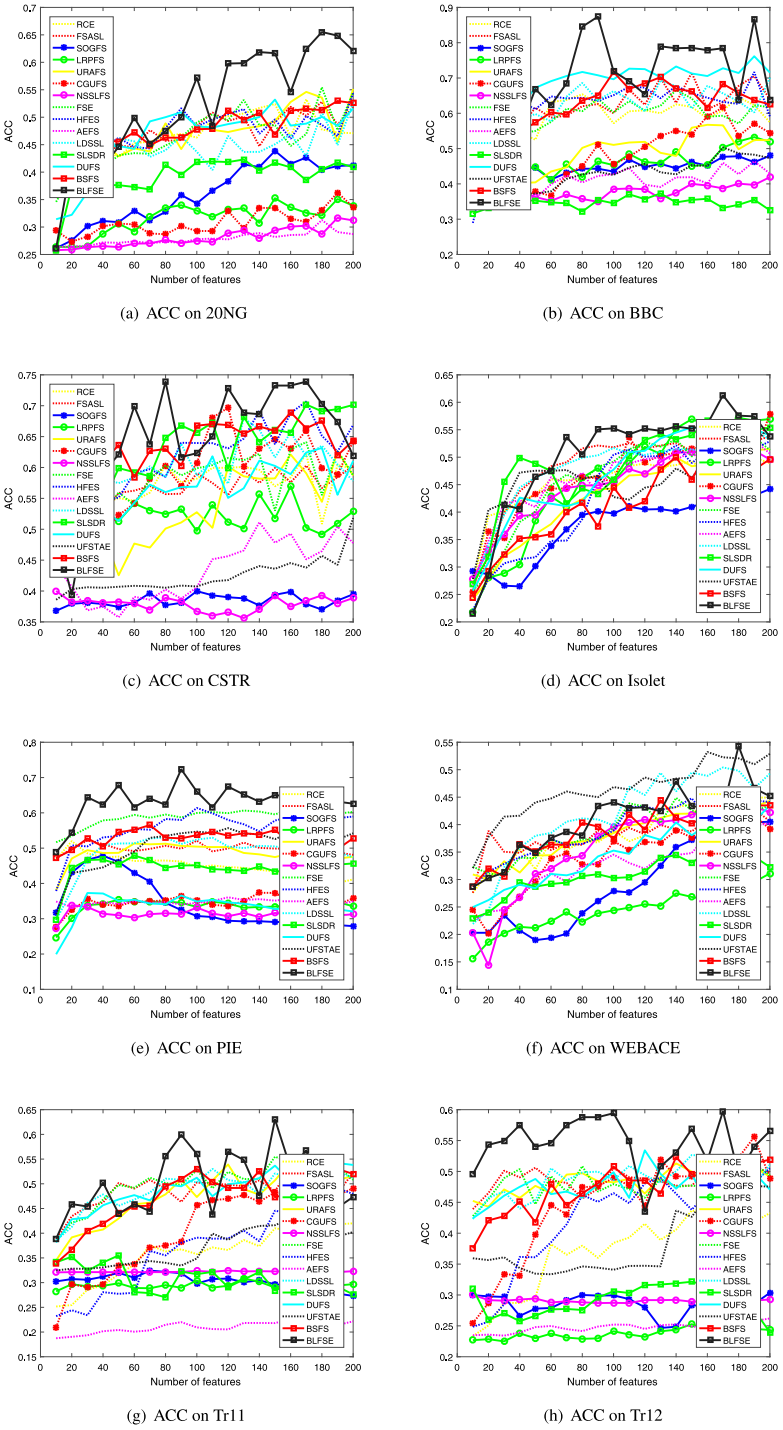
4.4. Experimental results

Tables 3 and 4 show the average ACC and NMI of BLFSE and other methods over the range of selected features (with $10, 20, \ldots, 200$ selected features). We also report the t-test results. The bold font means that the difference is statically significant according to t-test, i.e., the p-value of the t-test is smaller than 0.05. The numbers in the parentheses are the p-values. Notice that UFSTAE has no results on the 20NG data set because it runs out of memory on this data set. From Tables 3 and 4, we find that, although the base feature selection results we used are imperfect because we just use a subset of instances to generate them, BLFSE can still outperform other state-of-the-art feature selection methods even the deep feature selection methods AEFS and UFSTAE, which shows the effectiveness of the proposed ensemble method. Even compared with clustering ensemble based feature selection methods (RCE and CGUFS) and feature level ensemble methods (FSE and HEFS), ours also achieves better performance. It well demonstrates that the proposed bi-level ensemble method can outperform single-level ensemble methods.

Figs. 2 and 3 show the detailed ACC and NMI results w.r.t. the different numbers of selected features of all methods, respectively. We find that BLFSE outperforms both the feature selection and feature selection ensemble methods most time. Moreover, among the $10, 20, \ldots, 200$ selected features, when comparing on the best performance all methods can achieve, ours also outperforms other compared methods on most data sets. It means that if we select a suitable number of features for all methods, ours performs better.

To further demonstrate the effectiveness of the ensemble approach, we compare our method BLFSE with the 10 base feature selection results. We use R1, ..., R10 to denote the clustering results on the 10 base feature selection results, respectively. Table 5 shows the average results of all methods over the range of $10, 20, \ldots, 200$ selected features, and Figs. 4 and 5 show the detailed ACC and NMI results w.r.t. the different numbers of selected features. We find that BLFSE can outperform the 10 base results or at least are comparable with the best one most time. It shows that our ensemble schema can indeed improve the performance of the base results or at least provide a stable good consensus result. It well demonstrates our motivation of ensemble.

Fig. 6 shows the convergence curves of our method on 20NG, Isolet, WEBACE, and Tr11. The results on other data sets are similar. From Fig. 6, we find that our method can converge very fast, i.e., it often converges within ten iterations.



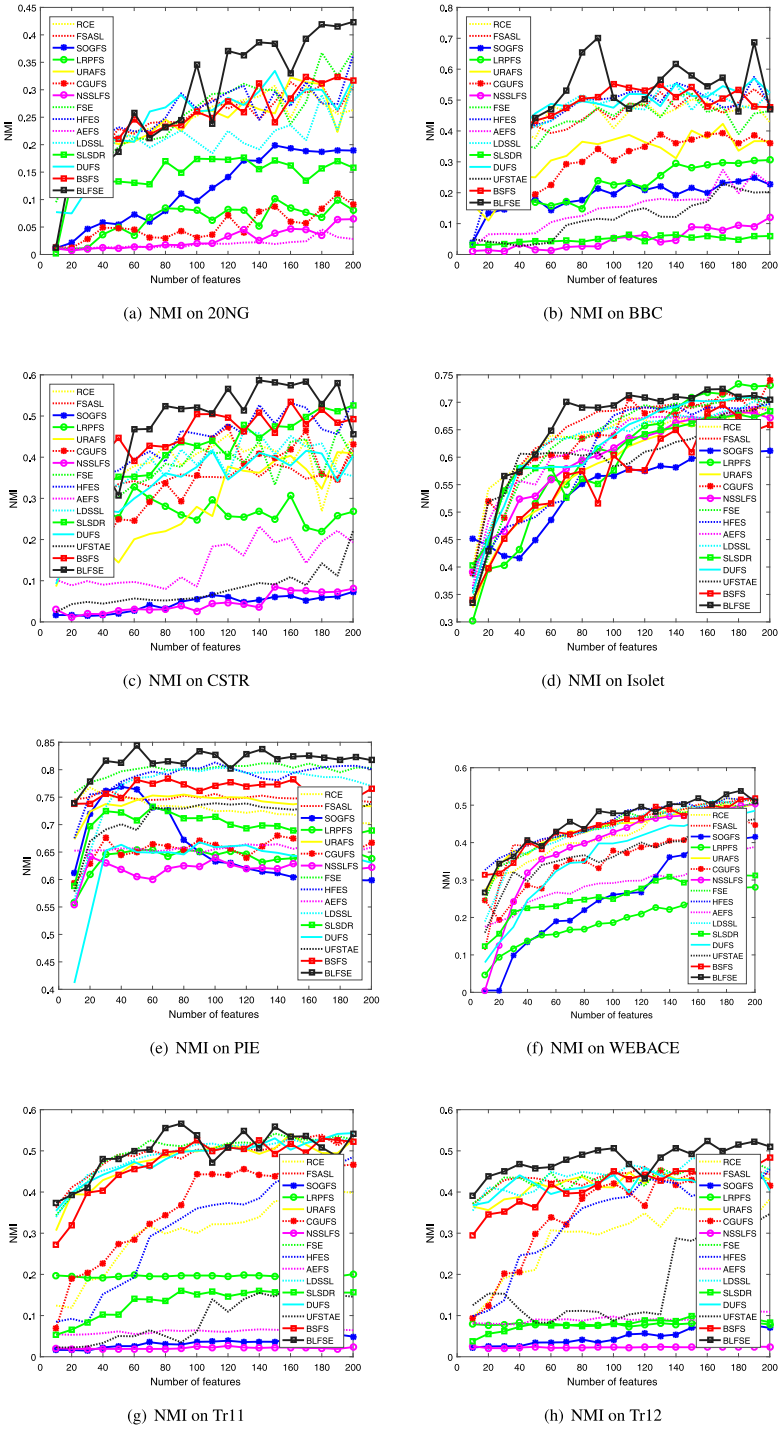
(a) ACC on 20NG (b) ACC on BBC

(c) ACC on CSTR (d) ACC on Isolet

(e) ACC on PIE (f) ACC on WEBACE

(g) ACC on Tr11 (h) ACC on Tr12

Fig. 2. ACC with different numbers of features for all methods.



(a) NMI on 20NG

(b) NMI on BBC

(c) NMI on CSTR

(d) NMI on Isolet

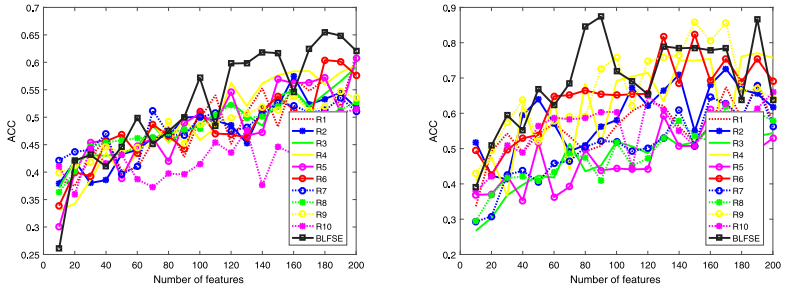
(e) NMI on PIE

(f) NMI on WEBACE

(g) NMI on Tr11

(h) NMI on Tr12

Fig. 3. NMI with different numbers of features for all methods.



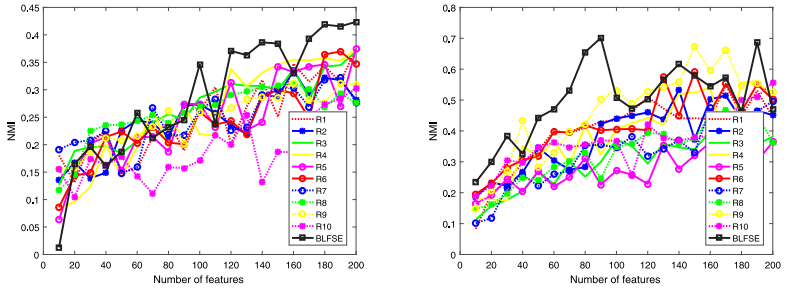
(a) ACC on 20NG (b) ACC on BBC

(c) ACC on CSTR (d) ACC on Isolet

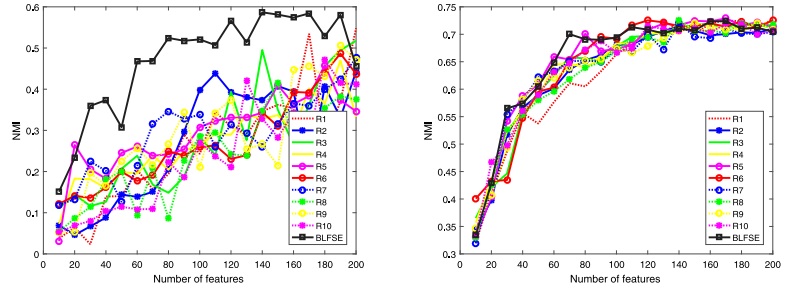
(e) ACC on PIE (f) ACC on WEBACE

(g) ACC on Tr11 (h) ACC on Tr12

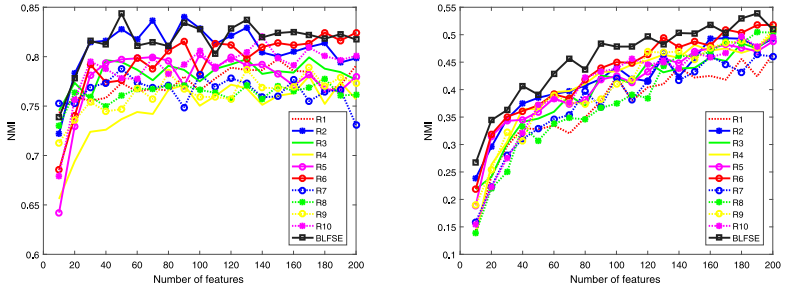
Fig. 4. ACC with different numbers of features for all base feature selection results.



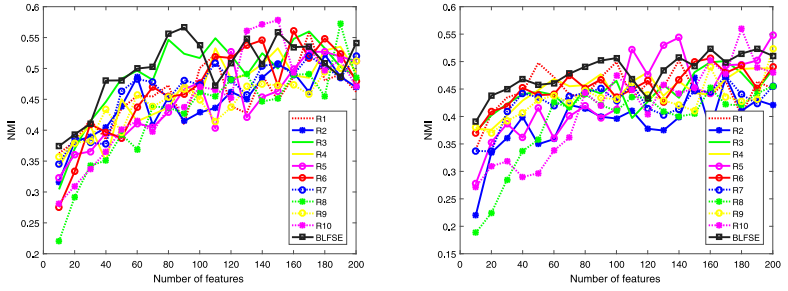
(a) NMI on 20NG (b) NMI on BBC



(c) NMI on CSTR (d) NMI on Isolet



(e) NMI on PIE (f) NMI on WEBACE



(g) NMI on Tr11 (h) NMI on Tr12

Fig. 5. NMI with different numbers of features for all base feature selection results.

Table 4
NMI results compared with other feature selection methods. The bold font means that the difference is statically significant, i.e., the p-value of the t-test is smaller than 0.05. The numbers in the parentheses are the p-values.

Methods	20NG	BBC	CSTR	Isolet	PIE	WEBACE	Tr11	TR12
RCE [44]	0.2453 (0.0240)	0.4334 (0.0007)	0.3837 (0.0006)	0.6304 (0.0387)	0.7276 (0.0000)	0.4247 (0.0000)	0.3019 (0.0000)	0.2932 (0.0000)
FSASL [79]	0.2462 (0.0232)	0.4406 (0.0042)	0.3390 (0.0000)	0.6427 (0.0203)	0.7489 (0.0000)	0.4332 (0.0124)	**0.4902** (0.1933)	0.4350 (0.0000)
SOGFS [13]	0.1181 (0.0000)	0.1880 (0.0000)	0.0444 (0.0000)	0.5414 (0.0000)	0.6563 (0.0000)	0.2551 (0.0000)	0.0348 (0.0000)	0.0484 (0.0000)
LRPFS [80]	0.0629 (0.0000)	0.2320 (0.0000)	0.2641 (0.0000)	0.5976 (0.0004)	0.6386 (0.0000)	0.1908 (0.0000)	0.1959 (0.0000)	0.0785 (0.0000)
URAFS [14]	0.2427 (0.0155)	0.3178 (0.0000)	0.2808 (0.0000)	0.5805 (0.0000)	0.7387 (0.0000)	0.4355 (0.0000)	0.4751 (0.0009)	0.4187 (0.0000)
CGUFS [45]	0.0523 (0.0000)	0.3005 (0.0000)	0.3288 (0.0000)	0.6339 (0.0448)	0.6559 (0.0000)	0.3578 (0.0000)	0.3609 (0.0000)	0.3524 (0.0000)
NSSLFS [38]	0.0275 (0.0000)	0.0496 (0.0000)	0.0448 (0.0000)	0.5967 (0.0000)	0.6187 (0.0000)	0.3901 (0.0003)	0.0211 (0.0000)	0.0230 (0.0000)
FSE [18]	0.2512 (0.0064)	0.4227 (0.0002)	0.3566 (0.0000)	0.6274 (0.0001)	0.7993 (0.0000)	0.4368 (0.0000)	**0.4973** (0.6360)	0.4355 (0.0000)
HEFS [71]	0.2597 (0.0406)	0.4528 (0.0177)	0.4343 (0.0336)	0.6026 (0.0005)	0.7869 (0.0000)	**0.4531** (0.5787)	0.3165 (0.0000)	0.3399 (0.0000)
AEFS [46]	0.0194 (0.0000)	0.1472 (0.0000)	0.1425 (0.0000)	0.6167 (0.0002)	0.6561 (0.0000)	0.2888 (0.0000)	0.0610 (0.0000)	0.0910 (0.0000)
LDSSL [81]	0.2075 (0.0004)	0.4592 (0.0293)	0.3699 (0.0000)	0.6265 (0.0200)	0.7695 (0.0000)	0.4334 (0.0006)	**0.4885** (0.1040)	0.4372 (0.0000)
SLSDR [82]	0.1458 (0.0000)	0.0489 (0.0000)	0.4105 (0.0010)	0.5977 (0.0004)	0.6976 (0.0000)	0.2584 (0.0000)	0.1346 (0.0000)	0.0798 (0.0000)
DUFS [83]	0.2444 (0.0109)	0.4714 (0.0450)	0.3337 (0.0000)	0.6214 (0.0100)	0.6321 (0.0000)	0.3601 (0.0000)	0.4836 (0.0382)	0.4200 (0.0000)
UFSTAE [49]	– –	0.1182 (0.0000)	0.0786 (0.0000)	0.6067 (0.0000)	0.7154 (0.0000)	0.3695 (0.0000)	0.0952 (0.0000)	0.1836 (0.0000)
BSFS [43]	0.2538 (0.0431)	0.4691 (0.0460)	0.4470 (0.1964)	0.5683 (0.0000)	0.7645 (0.0000)	0.4422 (0.0089)	0.4701 (0.0027)	0.4147 (0.0000)
BLFSE	**0.2870** –	**0.5012** –	**0.4701** –	**0.6523** –	**0.8152** –	**0.4559** –	**0.5000** –	**0.4797** –

Table 5
Clustering results compared with single base feature selection result.

Data sets	Metrics	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	BLFSE
20NG	ACC	0.4837	0.4764	0.4879	0.4915	0.4840	0.4816	0.4804	0.4813	0.4809	0.4327	**0.5239**
	NMI	0.2495	0.2447	0.2663	0.2561	0.2483	0.2412	0.2461	0.2571	0.2478	0.1899	**0.2870**
BBC	ACC	0.5548	0.5936	0.4685	0.6222	0.4646	0.6485	0.5040	0.4823	0.6643	0.5568	**0.6936**
	NMI	0.3977	0.3800	0.2999	0.3918	0.2737	0.4152	0.3246	0.3249	0.4599	0.3664	**0.5012**
CSTR	ACC	0.5644	0.5106	0.5418	0.4744	0.5243	0.4907	0.4935	0.4995	0.5583	0.4572	**0.6451**
	NMI	0.2611	0.2685	0.2792	0.2701	0.2948	0.2706	0.2907	0.2376	0.2874	0.2359	**0.4701**
Isolet	ACC	0.4715	0.4840	0.4832	0.4866	0.4927	0.4918	0.4856	0.4710	0.4778	0.4848	**0.5006**
	NMI	0.6257	0.6333	0.6376	0.6402	0.6492	0.6455	0.6363	0.6328	0.6354	0.6409	**0.6523**
PIE	ACC	0.5499	0.5830	0.5642	0.4929	0.5346	0.5877	0.5350	0.5312	0.5425	0.5827	**0.6325**
	NMI	0.7733	0.8092	0.7830	0.7475	0.7764	0.7947	0.7656	0.7629	0.7626	0.7881	**0.8152**
WEBACE	ACC	0.3313	0.3982	0.3722	0.3750	0.3599	0.3946	0.3601	0.3324	0.3619	0.3519	**0.4075**
	NMI	0.3695	0.4162	0.3976	0.4096	0.4072	0.4302	0.3782	0.3806	0.4046	0.4024	**0.4559**
Tr11	ACC	0.4496	0.4460	0.4832	0.4762	0.4612	0.4632	0.4673	0.4256	0.4565	0.4467	**0.5013**
	NMI	0.4540	0.4448	0.4898	0.4606	0.4436	0.4659	0.4649	0.4275	0.4460	0.4494	**0.5000**
Tr12	ACC	0.5250	0.4617	0.5137	0.5045	0.4974	0.5187	0.4850	0.4401	0.4973	0.4674	**0.5450**
	NMI	0.4476	0.3898	0.4452	0.4489	0.4394	0.4541	0.4256	0.3891	0.4312	0.4118	**0.4797**

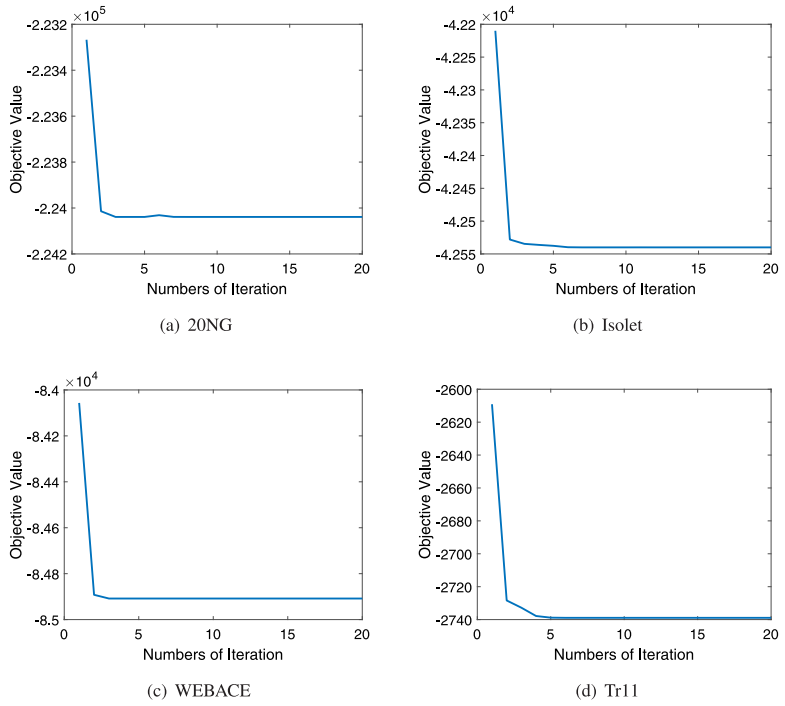


Fig. 6. Convergence curves of BLFSE.

Table 6
Clustering results compared with three degenerated versions of BLFSE.

Data sets	Metrics	Fea	Clu	woSP	BLFSE
20NG	ACC	0.4565	0.3546	0.4585	**0.5239**
	NMI	0.2421	0.1101	0.2311	**0.2870**
BBC	ACC	0.5785	0.4155	0.6335	**0.6936**
	NMI	0.4225	0.1607	0.4752	**0.5012**
CSTR	ACC	0.5605	0.4686	0.6000	**0.6451**
	NMI	0.3464	0.2381	0.4134	**0.4701**
Isolet	ACC	0.4608	0.4853	0.4886	**0.5006**
	NMI	0.6238	0.6187	0.6479	**0.6523**
PIE	ACC	0.5928	0.4025	0.6100	**0.6325**
	NMI	0.8013	0.6735	0.8076	**0.8152**
WEBACE	ACC	0.3875	0.3374	0.4026	**0.4075**
	NMI	0.4401	0.3580	0.4461	**0.4559**
Tr11	ACC	0.4822	0.2838	0.4874	**0.5013**
	NMI	0.4828	0.1235	0.4928	**0.5000**
Tr12	ACC	0.5022	0.2757	0.5315	**0.5450**
	NMI	0.4416	0.1253	**0.4825**	0.4797

4.5. Results of running time

In this subsection, we report the running time of our method and other compared methods. The results are shown in Fig. 7. The rightmost black bar indicates our proposed method. Although the proposed method involves two levels of the ensemble, which increases the computation costs, it is still comparable with other compared methods on many data sets. The proposed one is even faster than some methods, such as SOGFS, LRPFS, NSSFLS, and AEFS, on most data sets.

4.6. Ablation study

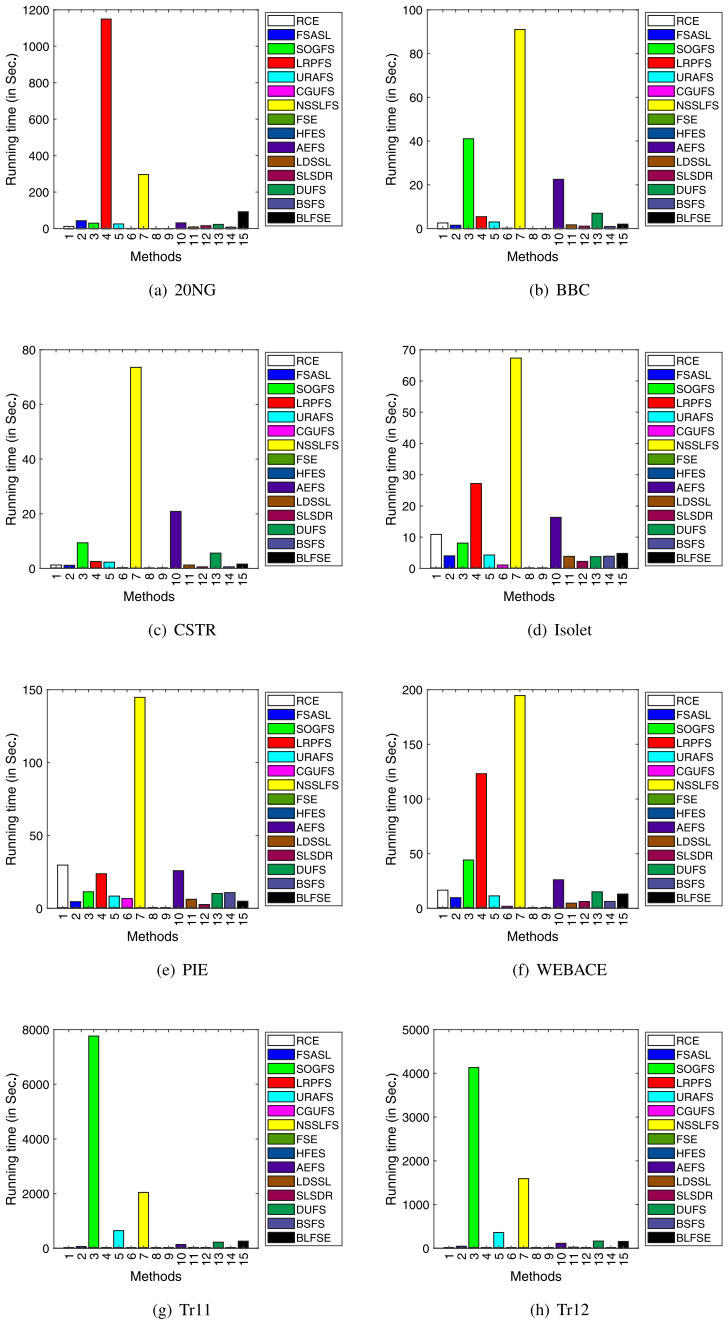
Since our method involves two levels of the ensemble, in this subsection, we compare with two degenerated versions which only use one level of the ensemble. Moreover, we also compare with the degenerated version without the self-paced learning approach to show the effectiveness of self-paced learning. In more detail, we compare with the following three degenerated versions:

- **Fea**, which is our method only ensembles on the feature level.
- **Clu**, which is our method only ensembles on the clustering level.
- **woSP**, which is our method without the self-paced learning, i.e., it fixes the weight $\mathbf{W}$ as an all-ones matrix.

Table 6 shows the ACC and NMI results of the three degenerated versions. From Table 6, we find that Fea often outperforms Clu, which means feature level ensemble is more important than the clustering level. The reason may be that the clustering level ensemble discards the original feature selection information, and is only based on the multiple clustering results. Notice that the base clustering results come from unreliable base feature selection methods and unreliable base clustering methods, and the unreliability diffuses in the process, which leads to poor performance of clustering level ensemble. That can also explain why most of the existing feature selection ensemble methods are on the feature level ensemble. Compared with Fea and Clu, the bi-level method BLFSE can outperform both single-level ensemble methods, which demonstrates the effectiveness of the bi-level ensemble. When compared with woSP, which is the one without self-paced learning, our BLFSE often achieves better performance, demonstrating that self-paced learning can indeed further improve performance.

5. Conclusion

In this paper, we proposed a novel bi-level feature selection ensemble method, which ensembled on both the feature and clustering levels by fully considering the downstream clustering task. On the clustering level ensemble, to alleviate the unreliability diffusion in the unsupervised model, we plugged it into a self-paced learning framework that used data for consensus learning in order of reliability. Then we proposed an iterative algorithm to optimize it and provided some theoretical analysis of hyper-parameters to make the model easy to use. At last, extensive experiments were conducted on benchmark data sets, and the results demonstrated that the proposed ensemble method outperformed all base results or at least was comparable with the best



(a) 20NG (b) BBC

(c) CSTR (d) Isolet

(e) PIE (f) WEBACE

(g) Tr11 (h) Tr12

Fig. 7. Running time of our method and other compared methods.

one. Moreover, when compared with the state-of-the-art unsupervised feature selection methods, the proposed BLFSE also performed better. Even compared with feature selection ensemble methods, ours could still achieve better performance.

Although the ensemble schema improves the feature selection performance, since it involves two levels of the ensemble, it increases the computation costs. In the future, we will study some approximation and accelerated methods to reduce the time complexity and speed up the method. In addition, there are many other unsupervised learning downstream tasks like outlier detection and so on. In this paper, we use the clustering assumption in the BLFSE, and thus it is appropriate for the clustering task. If we handle some other downstream tasks such as outlier detection and visualization, we should redesign other strategies by fully considering the specific tasks. Therefore, in the future, we will design new algorithms to tackle other specific downstream tasks.

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 is supported by the National Natural Science Foundation of China grants 62176001, 61806003, and 61976129.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.inffus.2023.101910.

References

[1] F. Nie, H. Huang, X. Cai, C.H. Ding, Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization, in: Advances in Neural Information Processing Systems, 2010, pp. 1813–1821.

[2] Q. Gu, Z. Li, J. Han, Generalized Fisher score for feature selection, in: UAI-11, 2011, pp. 266–273.

[3] X. Liu, L. Wang, J. Zhang, J. Yin, H. Liu, Global and local structure preservation for feature selection, IEEE Trans. Neural Networks Learn. Syst. 25 (6) (2014) 1083–1095.

[4] D. Ming, C. Ding, Robust flexible feature selection via exclusive L21 regularization, in: IJCAI-19, 2019, pp. 3158–3164.

[5] C. Tang, X. Liu, X. Zhu, J. Xiong, M. Li, J. Xia, X. Wang, L. Wang, Feature selective projection with low-rank embedding and dual Laplacian regularization, IEEE Trans. Knowl. Data Eng. 32 (9) (2020) 1747–1760.

[6] P. Zhou, L. Du, X. Li, Y. Shen, Y. Qian, Unsupervised feature selection with adaptive multiple graph learning, Pattern Recognit. 105 (2020) 107375.

[7] Z. Ling, Y. Li, Y. Zhang, K. Yu, P. Zhou, B. Li, X. Wu, A light causal feature selection approach to high-dimensional data, IEEE Trans. Knowl. Data Eng. (2022) 1–13, http://dx.doi.org/10.1109/TKDE.2022.3218786.

[8] X. Chang, F. Nie, Y. Yang, H. Huang, A convex formulation for semi-supervised multi-label feature selection, in: AAAI-14, 2014, pp. 1171–1177.

[9] M. Luo, X. Chang, L. Nie, Y. Yang, A.G. Hauptmann, Q. Zheng, An adaptive semisupervised feature analysis for video semantic recognition, IEEE Trans. Cybern. 48 (2) (2018) 648–660.

[10] E. Yu, J. Sun, J. Li, X. Chang, X. Han, A.G. Hauptmann, Adaptive semi-supervised feature selection for cross-modal retrieval, IEEE Trans. Multim. 21 (5) (2019) 1276–1288.

[11] Z. Li, J. Tang, Semi-supervised local feature selection for data classification, Sci. China Inf. Sci. 64 (9) (2021).

[12] P. Zhu, W. Zuo, L. Zhang, Q. Hu, S.C. Shiu, Unsupervised feature selection by regularized self-representation, Pattern Recognit. 48 (2) (2015) 438–446.

[13] F. Nie, W. Zhu, X. Li, et al., Unsupervised feature selection with structured graph optimization, in: AAAI, 2016, pp. 1302–1308.

[14] X. Li, H. Zhang, R. Zhang, Y. Liu, F. Nie, Generalized uncorrelated regression with adaptive graph for unsupervised feature selection, IEEE TNNLS 30 (5) (2019) 1587–1595.

[15] Z. Li, J. Liu, Y. Yang, X. Zhou, H. Lu, Clustering-guided sparse structural learning for unsupervised feature selection, IEEE Trans. Knowl. Data Eng. 26 (9) (2014) 2138–2150.

[16] C. Hou, F. Nie, X. Li, D. Yi, Y. Wu, Joint embedding learning and sparse regression: A framework for unsupervised feature selection, IEEE Trans. Cybern. 44 (6) (2013) 793–804.

[17] S. Wang, J. Tang, H. Liu, Embedded unsupervised feature selection, in: Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.

[18] P. Drotár, M. Gazda, L. Vokorokos, Ensemble feature selection using election methods and ranker clustering, Inform. Sci. 480 (2019) 365–380.

[19] Y. Hong, S. Kwong, Y. Chang, Q. Ren, Consensus unsupervised feature ranking from multiple views, Pattern Recognit. Lett. 29 (5) (2008) 595–602.

[20] S. Zhang, H. Wong, Y. Shen, D. Xie, A new unsupervised feature ranking method for gene expression data based on consensus affinity, IEEE/ACM TCBB 9 (4) (2012) 1257–1263.

[21] B. Seijo-Pardo, I. Porto-Díaz, V. Bolón-Canedo, A. Alonso-Betanzos, Ensemble feature selection: Homogeneous and heterogeneous approaches, Knowl.-Based Syst. 118 (2017) 124–139.

[22] 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.

[23] H. Peng, F. Long, C.H.Q. Ding, Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1226–1238.

[24] X. He, D. Cai, P. Niyogi, Laplacian score for feature selection, in: Advances in Neural Information Processing Systems, 2006, pp. 507–514.

[25] Y. Liu, F. Tang, Z. Zeng, Feature selection based on dependency margin, IEEE Trans. Cybern. 45 (6) (2015) 1209–1221.

[26] C. Yao, Y. Liu, B. Jiang, J. Han, J. Han, LLE score: A new filter-based unsupervised feature selection method based on nonlinear manifold embedding and its application to image recognition, IEEE TIP 26 (11) (2017) 5257–5269.

[27] G. Roffo, S. Melzi, M. Cristani, Infinite feature selection, in: 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015, 2015, pp. 4202–4210.

[28] G. ROFFO, s. melzi, U. Castellani, A. Vinciarelli, M. Cristani, Infinite feature selection: A graph-based feature filtering approach, IEEE TPAMI (2020) 1.

[29] J. Macqueen, Some methods for classification and analysis of multivariate observations, in: 5-Th Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.

[30] D. Cai, C. Zhang, X. He, Unsupervised feature selection for multi-cluster data, in: KDD-10, ACM, 2010, pp. 333–342.

[31] M. Breaban, H. Luchian, A unifying criterion for unsupervised clustering and feature selection, Pattern Recognit. 44 (4) (2011) 854–865.

[32] D. Dutta, P. Dutta, J. Sil, Simultaneous feature selection and clustering with mixed features by multi objective genetic algorithm, Int. J. Hybrid Intell. Syst. 11 (1) (2014) 41–54.

[33] Z. Zhao, L. Wang, H. Liu, Efficient spectral feature selection with minimum redundancy, in: AAAI-10, AAAI Press, 2010.

[34] Z. Li, Y. Yang, J. Liu, X. Zhou, H. Lu, Unsupervised feature selection using nonnegative spectral analysis, in: J. Hoffmann, B. Selman (Eds.), Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada, AAAI Press, 2012.

[35] Z. Li, J. Tang, Unsupervised feature selection via nonnegative spectral analysis and redundancy control, IEEE Trans. Image Process. 24 (12) (2015) 5343–5355.

[36] R. Shang, W. Wang, R. Stolkin, L. Jiao, Non-negative spectral learning and sparse regression-based dual-graph regularized feature selection, IEEE Trans. Cybern. 48 (2) (2018) 793–806.

[37] C. Tang, M. Bian, X. Liu, M. Li, H. Zhou, P. Wang, H. Yin, Unsupervised feature selection via latent representation learning and manifold regularization, Neural Netw. 117 (2019) 163–178.

[38] W. Zheng, H. Yan, J. Yang, Robust unsupervised feature selection by nonnegative sparse subspace learning, Neurocomputing 334 (2019) 156–171.

[39] F. Nie, Z. Wang, L. Tian, R. Wang, X. Li, Subspace sparse discriminative feature selection, IEEE TCYB (2020) 1–13.

[40] D. Huang, X. Cai, C. Wang, Unsupervised feature selection with multi-subspace randomization and collaboration, Knowl.-Based Syst. 182 (2019) 104856.

[41] X. Zhang, M. Fan, D. Wang, P. Zhou, D. Tao, Top-<italic>k</italic> feature selection framework using robust 0–1 integer programming, IEEE TNNLS 32 (7) (2021) 3005–3019.

[42] P. Zhou, J. Chen, M. Fan, L. Du, Y.-D. Shen, X. Li, Unsupervised feature selection for balanced clustering, KBS (2020) 105417.

[43] P. Zhou, J. Chen, L. Du, X. Li, Balanced spectral feature selection, IEEE Trans. Cybern. (2022) 1–13, http://dx.doi.org/10.1109/TCYB.2022.3160244.

[44] H. Elghazel, A. Aussem, Unsupervised feature selection with ensemble learning, Mach. Learn. 98 (1–2) (2015) 157–180.

[45] H. Liu, M. Shao, Y. Fu, Feature selection with unsupervised consensus guidance, IEEE Trans. Knowl. Data Eng. 31 (12) (2019) 2319–2331.

[46] K. Han, Y. Wang, C. Zhang, C. Li, C. Xu, Autoencoder inspired unsupervised feature selection, in: ICASSP-18, 2018, pp. 2941–2945.

[47] Y. Huang, W. Jin, Z. Yu, B. Li, Supervised feature selection through deep neural networks with pairwise connected structure, Knowl.-Based Syst. 204 (2020) 106202.

[48] A. Mirzaei, V. Pourahmadi, M. Soltani, H. Sheikhzadeh, Deep feature selection using a teacher-student network, Neurocomputing 383 (2020) 396–408.

[49] Y. Zhang, Z. Lu, S. Wang, Unsupervised feature selection via transformed auto-encoder, Knowl.-Based Syst. 215 (2021) 106748.

[50] A. Strehl, J. Ghosh, Cluster ensembles — A knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (3) (2003) 583–617.

[51] L. Bai, J. Liang, F. Cao, A multiple k-means clustering ensemble algorithm to find nonlinearly separable clusters, Inf. Fusion 61 (2020) 36–47.

[52] P. Zhou, X. Wang, L. Du, X. Li, Clustering ensemble via structured hypergraph learning, Inf. Fusion 78 (2022) 171–179.

[53] Z. Zhou, W. Tang, Clusterer ensemble, Knowl. Based Syst. 19 (1) (2006) 77–83.

[54] P. Zhou, L. Du, H. Wang, L. Shi, Y. Shen, Learning a robust consensus matrix for clustering ensemble via Kullback-Leibler divergence minimization, in: IJCAI, 2015, pp. 4112–4118.

[55] P. Zhou, L. Du, X. Li, Self-paced consensus clustering with bipartite graph, in: C. Bessiere (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, ijcai.org, 2020, pp. 2133–2139.

[56] P. Zhou, L. Du, X. Li, Adaptive consensus clustering for multiple K-means via base results refining, IEEE Trans. Knowl. Data Eng. (2023) 1–14.

[57] A.P. Topchy, A.K. Jain, W.F. Punch, Clustering ensembles: Models of consensus and weak partitions, IEEE Trans. Pattern Anal. Mach. Intell. 27 (12) (2005) 1866–1881.

[58] N. Nguyen, R. Caruana, Consensus clusterings, in: Proceedings of the 7th IEEE International Conference on Data Mining, ICDM 2007, October 28-31, 2007, Omaha, Nebraska, USA, IEEE Computer Society, 2007, pp. 607–612.

[59] F. Li, Y. Qian, J. Wang, J. Liang, Multigranulation information fusion: A Dempster-Shafer evidence theory-based clustering ensemble method, Inform. Sci. 378 (2017) 389–409.

[60] Z. Tao, H. Liu, S. Li, Y. Fu, Robust spectral ensemble clustering, in: CIKM, 2016, pp. 367–376.

[61] P. Zhou, L. Du, X. Liu, Y.-D. Shen, M. Fan, X. Li, Self-paced clustering ensemble, IEEE Trans. Neural Netw. Learn. Syst. 32 (4) (2021) 1497–1511.

[62] P. Zhou, B. Sun, X. Liu, L. Du, X. Li, Active clustering ensemble with self-paced learning, IEEE Trans. Neural Netw. Learn. Syst. (2023) 1–15, http://dx.doi.org/10.1109/TNNLS.2023.3252586.

[63] D. Huang, C.-D. Wang, J. Wu, J.-H. Lai, C.K. Kwoh, Ultra-scalable spectral clustering and ensemble clustering, IEEE TKDE 32 (6) (2020) 1212–1226.
[64] D. Huang, C.-D. Wang, J.-H. Lai, Fast multi-view clustering via ensembles: Towards scalability, superiority, and simplicity, IEEE Trans. Knowl. Data Eng. (2023) 1–16, http://dx.doi.org/10.1109/TKDE.2023.3236698.
[65] Z. Yu, L. Li, Y. Gao, J. You, J. Liu, H. Wong, G. Han, Hybrid clustering solution selection strategy, Pattern Recognit. 47 (10) (2014) 3362–3375.
[66] D. Huang, J. Lai, C. Wang, Ensemble clustering using factor graph, Pattern Recognit. 50 (2016) 131–142.
[67] Z. Tao, H. Liu, S. Li, Z. Ding, Y. Fu, Robust spectral ensemble clustering via rank minimization, ACM Trans. Knowl. Discov. Data 13 (1) (2019) 1–25.
[68] D. Huang, C.-D. Wang, H. Peng, J. Lai, C.-K. Kwoh, Enhanced ensemble clustering via fast propagation of cluster-wise similarities, IEEE Trans. Syst., Man, Cybern.: Syst. 51 (1) (2021) 508–520, http://dx.doi.org/10.1109/TSMC.2018.2876202.
[69] P. Zhou, X. Liu, L. Du, X. Li, Self-paced adaptive bipartite graph learning for consensus clustering, ACM Trans. Knowl. Discov. Data 17 (5) (2023).
[70] A.K. Das, S. Das, A. Ghosh, Ensemble feature selection using bi-objective genetic algorithm, KBS 123 (2017) 116–127.
[71] K. Chiew, C.L. Tan, K. Wong, K.S.C. Yong, W.K. Tiong, A new hybrid ensemble feature selection framework for machine learning-based phishing detection system, Inform. Sci. 484 (2019) 153–166.
[72] P. Zhou, L. Du, Y. Shen, X. Li, Tri-level robust clustering ensemble with multiple graph learning, in: AAAI-21, AAAI Press, 2021, pp. 11125–11133.
[73] M.P. Kumar, B. Packer, D. Koller, Self-paced learning for latent variable models, in: NIPS, 2010, pp. 1189–1197.
[74] Q. Zhao, D. Meng, L. Jiang, Q. Xie, Z. Xu, A.G. Hauptmann, Self-paced learning for matrix factorization, in: AAAI, 2015, pp. 3196–3202.
[75] F. Nie, X. Wang, H. Huang, Clustering and projected clustering with adaptive neighbors, in: SIGKDD, 2014, pp. 977–986.
[76] K. Fan, On a theorem of Weyl concerning eigenvalues of linear transformations: II*, Proc. Natl. Acad. Sci. USA 36 (1) (1949) 31–35.
[77] H. Li, C. Fang, Z. Lin, Convergence rates analysis of the quadratic penalty method and its applications to decentralized distributed optimization, 2017, http://dx.doi.org/10.48550/ARXIV.1711.10802, URL https://arxiv.org/abs/1711.10802.
[78] J. Li, J. Tang, H. Liu, Reconstruction-based unsupervised feature selection: An embedded approach, in: C. Sierra (Ed.), IJCAI-17, 2017, pp. 2159–2165.
[79] L. Du, Y.-D. Shen, Unsupervised feature selection with adaptive structure learning, in: KDD-15, ACM, 2015, pp. 209–218.
[80] W. Zheng, C. Xu, J. Yang, J. Gao, F. Zhu, Low-rank structure preserving for unsupervised feature selection, Neurocomputing 314 (2018) 360–370.
[81] R. Shang, Y. Meng, W. Wang, F. Shang, L. Jiao, Local discriminative based sparse subspace learning for feature selection, Pattern Recognit. 92 (2019) 219–230.
[82] R. Shang, K. Xu, F. Shang, L. Jiao, Sparse and low-redundant subspace learning-based dual-graph regularized robust feature selection, Knowl.-Based Syst. 187 (2020).
[83] H. Lim, D. Kim, Pairwise dependence-based unsupervised feature selection, Pattern Recognit. 111 (2021) 107663.
[84] D. Cai, X. He, W.V. Zhang, J. Han, Regularized locality preserving indexing via spectral regression, in: CIKM-07, 2007, pp. 741–750.
[85] D. Greene, P. Cunningham, Producing accurate interpretable clusters from high-dimensional data, in: PKDD-05, in: Lecture Notes in Computer Science, vol. 3721, Springer, 2005, pp. 486–494.
[86] P. Zhou, L. Du, L. Shi, H. Wang, Y. Shen, Recovery of corrupted multiple kernels for clustering, in: IJCAI-15, 2015, pp. 4105–4111.
[87] M.A. Fanty, R.A. Cole, Spoken letter recognition, in: NIPS-90, Morgan Kaufmann, 1990, pp. 220–226.
[88] T. Sim, S. Baker, M. Bsat, The CMU pose, illumination, and expression (PIE) database, in: Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition, 2002, pp. 53–58.
[89] L. Du, X. Li, Y.-D. Shen, Cluster ensembles via weighted graph regularized nonnegative matrix factorization, in: ADMA-11, Springer, 2011, pp. 215–228.
[90] J. Cao, Z. Wu, J. Wu, H. Xiong, SAIL: Summation-based incremental learning for information-theoretic text clustering, IEEE Trans. Cybern. 43 (2) (2013) 570–584.