Redundant features removal for unsupervised spectral feature selection algorithms: an empirical study based on nonparametric sparse feature graph

Pengfei Xu, Shuchu Han, Hao Huang & Hong Qin

International Journal of Data Science and Analytics

ISSN 2364-415X

Int J Data Sci Anal
DOI 10.1007/s41060-018-0167-1

International Journal of Data Science and Analytics
https://doi.org/10.1007/s41060-018-0167-1

REGULAR PAPER

Redundant features removal for unsupervised spectral feature selection algorithms: an empirical study based on nonparametric sparse feature graph

Pengfei Xu[1] · Shuchu Han[2] · Hao Huang[3] · Hong Qin[2]

Received: 17 December 2017 / Accepted: 27 November 2018

Abstract
For existing unsupervised spectral feature selection algorithms, the quality of the eigenvectors decides the performance. There eigenvectors are calculated from the Laplacian matrix of similarity graph which is built from samples. When applying these algorithms to high-dimensional data, we meet the very embarrassing *chicken-and-egg* problem: "the success of feature selection depends on the quality of indication vectors which are related to the structure of data. But the purpose of feature selection is to give more accurate data structure." To alleviate this problem, we propose a graph-based approach to reduce the dimension of data by searching and removing redundant features automatically. A sparse graph is generated at feature side and is used to learn the redundant relationship among features. We name this novel graph as *sparse feature graph* (SFG). To avoid the inaccurate distance information among high-dimensional vectors, the construction of SFG does not utilize the pairwise relationship among samples, which means the structure info of data is not used. Our proposed algorithm is also a nonparametric one as it does not make any assumption about the data distribution. We treat this proposed redundant feature removal algorithm as a data preprocessing approach for existing popular unsupervised spectral feature selection algorithms like multi-cluster feature selection (MCFS) which requires accurate cluster structure information based on samples. Our experimental results on benchmark datasets show that the proposed SFG and redundant feature remove algorithm can improve the performance of those unsupervised spectral feature selection algorithms consistently.

Keywords Sparse graph representation · Unsupervised spectral feature selection · Dense subgraph

1 Introduction

In unsupervised spectral feature selection algorithms, the structure information of data is used to generate indication vectors which guide the features selection process. The structure of data could be local manifold structure [9,10], global structure [15,33], discriminative information [14,17,28] and etc. To model the structure of data, existing methods like Gaussian similarity graph, or k-nearest neighbor graph are very popular in machine learning research. All these similarity graphs are built based on the pairwise distance like Euclidean distance (ℓ_2 norm) or Manhattan distance (ℓ_1 norm) among samples. As we can see, the pairwise distance is crucial to the quality of data's intrinsic structure. As a result, the success of unsupervised spectral feature selection depends on the modeling accuracy of the data structure.

When the dimension of feature vector becomes high, especially for those high-dimensional datasets, we will meet the well-known "curse of dimensionality" issue [2]. That means the differentiating ability of pairwise distance will be degraded rapidly when the dimension of feature vector goes high, and the nearest neighbor indexing will give inaccurate results [1,27]. As a result, the description of data structure by

✉ Shuchu Han
shuchu.han@gmail.com

Pengfei Xu
xupf@bnu.edu.cn

Hao Huang
haohuanghw@gmail.com

Hong Qin
qin@cs.stonybrook.edu

1 College of Information Science and Technology, Beijing Normal University, Beijing, China

2 Department of Computer Science, Stony Brook University, Stony Brook, USA

3 Machine Learning Laboratory, General Electric Global Research, San Ramon, CA, USA

using similarity graphs will be not precise and even wrong. This create an embarrassing *chicken-and-egg* problem [5] for unsupervised spectral feature selection algorithms: "the success of feature selection depends on the quality of indication vectors which are related to the structure of data. But the purpose of feature selection is to give more accurate data structure."

Most existing unsupervised spectral feature selection algorithms use all original features [5] to represent the (cluster) structure of data. As a result, the obtained data structure information will not as accurate as the intrinsic one it should be. To remedy this problem, dimensionality reduction techniques are required. For example, principal component analysis (PCA) and random projection (RP) are popular methods in machine learning research. However, most of them will project the data matrix into another (lower dimensional) space with the constraint to approximate the original pairwise similarities. As a result, we lose the physical meaning or original features and the meaning of projected features are unknown.

In this study, we proposed a graph-based approach to reduce the data dimension by removing redundant features. *We build a relation graph at feature side to represent the correlation among features, and assume that the dense subgraphs in this feature side graph represent the redundancy.* Without lose of generality, features can be categorized into three groups [4]: *relevant feature*, *irrelevant feature* and *redundant feature*. A feature $\boldsymbol{f}_i$ is relevant or irrelevant based on its correlation with indication vectors (or target vectors named in other articles): $\boldsymbol{Y} = \{\boldsymbol{y}_i, i \in [1, k]\}$. For supervised spectral feature selection algorithms like [20,23] and [19], these indication vectors usually relate to class labels. For unsupervised scenario [3,6], they follow the cluster structure of data. Redundant features are features that highly correlated to other features, and have no contribution or trivial contribution to the target learning task. The formal definition of redundant feature is by [30] based on the Markov blanket [12].

As mentioned in the multi-cluster feature selection (MCFS) work [3], a feature could be redundant to another single feature, or to a subset of features. As inspired by this famous work, we propose a graph-based approach to identify these two kind of redundancy at the same time. The first step is to build a sparse feature graph (SFG) at feature side based on sparse representation concept from subspace clustering theory [7]. In the last, we defined local compressible subgraphs (LCS) to represent those local feature groups which has high coherent inside. Moreover, a greedy local search algorithm is proposed to discover all those LCSs. Once we have all LCSs, we pick the feature which has the highest node in-degree (the "in" or "out" degree is decide by how you generate the sparse feature group) as the representative feature of each LCS and treat all other as redundant features. By this approach, we obtain a new data matrix with reduced dimensions, and the curse of dimensional issue is alleviated.

To be specific, the contribution of our study can be highlighted as:

– We propose sparse feature graph to model the feature redundancy among features. The sparse feature graph inherits the philosophy of sparse learning based unsupervised spectral feature selection framework. The sparse feature graph not only discloses the redundancy between two features but also show the redundancy between one feature and a subset of features.
– We propose the concept of local compressible subgraph to represent redundant feature groups. And also design a local greedy search algorithm to find all those subgraphs.
– The proposed algorithm avoids the using of pairwise distance (or similarity) among samples. As a result, the *chicken-and-egg* issue is not applicable to our approach. This is the elegant part of our solution.
– Abundant experiments and analyses over 12 datasets from three different domains are also presented in this study. The experiment results show the effectiveness of our proposed methods.

The rest of paper is organized as follows. Section 2 describes the math notations in this work. Section 3 introduces the background, motivation and preliminaries of our problem. In Sect. 4, we define the problem we are going to solve. In Sect. 5, we present our proposed sparse feature graph algorithm and discuss the sparse representation error problem. We also introduce the local compressible subgraph and related algorithm. In Sect. 6, we discuss the time complexity of our proposed algorithms. The experiment results are reported in Sect. 7, and a briefly reviewing of related works is given in Sect. 8. Finally, we conclude our study in last Sect. 9.

2 Math notation

Throughout this paper, matrices are written as boldface capital letters and vectors are represented as boldface lowercase letters. Let the data matrix be represented as $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, while each row is a sample (or instance), and each column represents a feature. If we view the data matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^{\mathrm{T}}, \boldsymbol{x}_i \in \mathbb{R}^{d \times 1}$ from feature side, it can be seen as $\boldsymbol{F} = \boldsymbol{X}^{\mathrm{T}} = [\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_d], \boldsymbol{f}_i \in \mathbb{R}^{n \times 1} (1 \leq i \leq d)$.

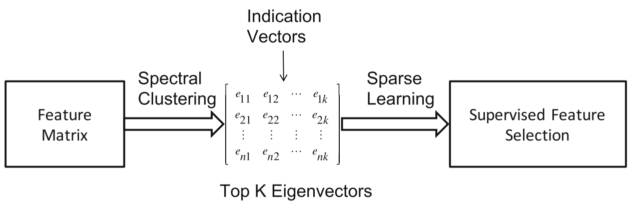

Fig. 1 The framework of sparse learning based unsupervised spectral feature selection

3 Background and preliminaries

3.1 Unsupervised spectral feature selection

In the unsupervised spectral feature selection framework, we do not have label information to determine the feature relevance. Instead, the data similarity or manifold structure constructed from the whole feature is used as criterion to select features. Among all those algorithms, there is a famous one named as MCFS. The MCFS algorithm is a sparse learning based unsupervised spectral feature selection method which can be illustrated as Fig. 1. The core idea of MCFS is to use the eigenvectors of graph Laplacian over similarity graph as indication vectors. And then find set of features that can approximate these eigenvectors through sparse linear regression. Let us assume the input data has number K clusters that is known beforehand (or an estimated K). The top K non-trivial eigenvectors, $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k]$, form the spectral embedding $\boldsymbol{Y}$ of the data. Each row of $\boldsymbol{Y}$ is the new coordinate in the embedding space. To select the relevant features, MCFS solves K sparse linear regression problems between $\boldsymbol{F}$ and $\boldsymbol{Y}$ as:

$$\min_{\alpha_i} \|\boldsymbol{y}_i - \boldsymbol{F}\boldsymbol{\alpha}_i\|^2 + \beta\|\boldsymbol{\alpha}_i\|_1, \tag{1}$$

where $\boldsymbol{\alpha}_i$ is a n-dimensional vector and it contains the combination coefficients for different features $\boldsymbol{f}_i$ in approximating $\boldsymbol{y}_i$. Once all coefficients $\boldsymbol{\alpha}_i$ are collected, features will be ranked by the absolute value of these coefficients and top features are selected. This can be shown by a weighted directed bipartite graph as following (Fig. 2):

3.2 Adaptive structure learning for high-dimensional data

As we can see, the MCFS uses whole features to model the structure of data. This may create troubles when the dimension of feature vector goes higher as indicated by the "curse of dimensionality". One consequence of using whole features is that the obtained structural information of data through similarity graph is not accuracy. This observation is the motivation of unsupervised algorithm: feature selection with adaptive structure learning (FSASL) algorithm which

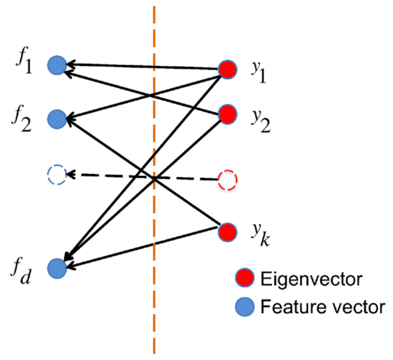

Fig. 2 Sparse learning bipartite graph for MCFS

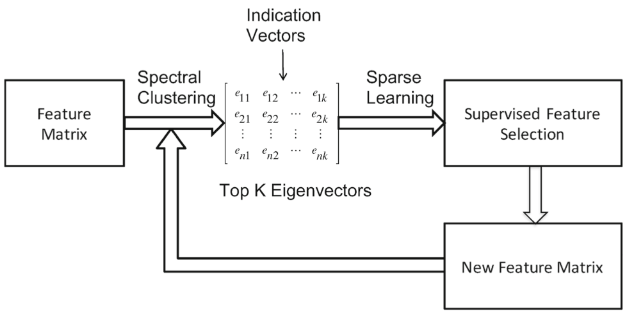

Fig. 3 Unsupervised feature selection with adaptive structure learning

is proposed by Du et al. [5]. The idea of FSASL is to repeat MCFS iteratively with updating selected feature sets. It can be illustrated as following: FASAL is an iterative algorithms which keeps pruning irrelevant and noisy features to obtain better manifold structure while improved structural info can help to search better relevant features. FASAL shows better performance in normalized mutual information and accuracy than MCFS generally. However, it is very time consuming as by repeating eigen-decomposition calculations (Fig. 3).

3.3 Definition of redundant features

For high-dimensional data $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, it exists information redundancy among features since $d \ll n$. Those redundant features cannot provide further performance improvement for ongoing learning task. Instead, they impair the efficiency of learning algorithms to find the intrinsic data structure. In this section, we describe our definition of feature redundancy. Comparing to the definition described in the famous feature selection work [30], where the redundant feature is categorized as redundant feature iff it is weakly relevant and has a Markov blanket in current feature set, our definition of feature redundancy is based on the linear correlation between two vectors (the "vector" we used here could be a feature vector or a linear combination of several feature vectors.) To measure the redundancy between two vectors $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$, squared cosine similarity [25] is used:

$$R_{ij} = \cos^2\left(\boldsymbol{f}_i, \boldsymbol{f}_j\right). \tag{2}$$

By the math definition of cosine similarity, it is straightforward to know that a higher value of $R_{i,j}$ means high redundancy existing between $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$. For example, feature vector $\boldsymbol{f}_i$ and its duplication $\boldsymbol{f}_i$ will have R_{ii} value equals to one. And two orthogonal feature vectors will have redundancy value equals to zero.

As a result of our definition of redundant features, not only the highly correlated relevant features will be removed by our proposed algorithm, but also those irrelevant features and noise features.

4 Problem statement

In this work, our goal is to detect those redundant features exist in high-dimensional data and obtain a more accurate intrinsic data structure. To be specific:

Problem 1 Given a high-dimensional data represented in the form of feature matrix $\boldsymbol{X}$, how to remove those redundant features $f_{(\cdot)} \in \boldsymbol{X}^{\mathrm{T}}$ for unsupervised spectral feature selection algorithms such as MCFS?

Technically, the MCFS algorithm does not involve redundant features. However, the performance of MCFS depends on the quality of indication vectors which are used to select features via sparse learning. And those indication vectors are highly related to the intrinsic structure of data distribution which is described by the selected features and given distance metric. The MCFS algorithm uses all features and Gaussian similarity to represent the intrinsic structure. The redundant features will lead to an inaccurate model of data structure. As a result, it is very demanding to remove those redundant features before the calculation of indication vectors.

5 Algorithm

In this section, we propose a graph-based algorithm to detect and remove redundant features existing in high-dimensional data. First, the sparse feature graph that modeling the redundancy among feature vectors is introduced. Secondly, the sparse representation error is discussed. In the last, the local compressible subgraphs concept is presented to extract redundant feature groups.

5.1 Sparse feature graph (SFG)

The most popular way to model the redundancy among feature vectors is correlation such as Pearson correlation coefficient (PCC). The correlation value is defined over two feature vectors, and it is a pairwise measurement. However,

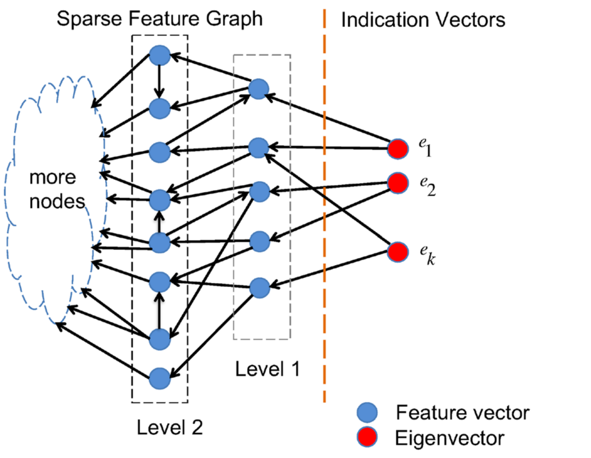

Fig. 4 Sparse feature graph and its relation with indication vectors. The level 1 features are direct sparse representation of those calculated indication vectors. The level 2 features only have representation relationship with level 1 features but not with indication vectors

there also existing redundancy between one feature vector and a set of feature vectors according to the philosophy of MCFS algorithm. In this section, we present SFG, which models the redundancy not only between two feature vectors but also one feature vector and a set of feature vectors (Fig. 4).

The basic idea of sparse feature graph follows our previous work [8] and is to look for a sparse linear representation for each feature vector while using all other feature vectors as dictionary. For each feature vector $\boldsymbol{f}_i$ in features set $\boldsymbol{F} = [\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_d]$, SFG solves the following optimization problem:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^{d-1}} \|\boldsymbol{f_i} - \boldsymbol{\Phi^i} \boldsymbol{\alpha_i}\|_2^2, \quad \text{s.t.} \quad \|\boldsymbol{\alpha_i}\|_0 < L, \tag{3}$$

where $\boldsymbol{\Phi}^i = [\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_{i-1}, \boldsymbol{f}_{i+1}, \ldots, \boldsymbol{f}_d]$ is the dictionary of f_i and each column of $\boldsymbol{\Phi}^i$ is a selected feature from data matrix $\boldsymbol{X}$. L is a constraint to limit the number of nonzero coefficients. In SFG, we set it to the number of features d. The $\boldsymbol{\alpha}_i$ is the coefficient of each atom of dictionary $\boldsymbol{\Phi}^i$. This coefficient vector not only decides the edge link to $\boldsymbol{f_i}$ but also indicates the weight of that connection. The resulted SFG is a weighted directed graph and may have multiple components.

To solve the optimization Problem 3, we use orthogonal matching pursuit (OMP) solver [29] here since the number of features in our datasets is larger than 1000. We modify the stop criterion of OMP by checking the value change of residual instead of residual itself or the maximum number of supports. The reason is that we want the number of supports (or say, the number of edge connections) to follow the raw data property. Real-world datasets are always noisy and messy. It is highly possible that several feature vectors may fail to find a correct sparse linear representation through

OMP. If we set residual or maximum of supports as criteria, we cannot differentiate the successful representations and the failed ones.

The OMP solver and SFG algorithm can be described as following.

```
Algorithm 1: Orthogonal Matching Pursuit (OMP)
  Input  : $\boldsymbol{\Phi} = [\boldsymbol{f_1}, \boldsymbol{f_2}, \ldots, \boldsymbol{f_{i-1}}, \boldsymbol{f_{i+1}}, \ldots, \boldsymbol{f_d}] \in \mathbb{R}^{n\times(d-1)}$, $\boldsymbol{f_i} \in \mathbb{R}^n$, $\epsilon$.
  Output: Coefficient $\boldsymbol{\alpha}_i$.
  Initialize residual difference threshold $r_0 = 1.0$, residual $\boldsymbol{q}_0 = \boldsymbol{f}_i$, support set $\Gamma_0 = \emptyset$, $k = 1$ ;
  while $k \le d - 1$ and $|r_k - r_{k-1}| > \epsilon$ do
      Search the atom which most reduces the objective:
        $j^* = \arg\min_{j\in\Gamma^C}\left\{\min_{\alpha} \|\boldsymbol{f}_i - \boldsymbol{\Phi}_{\Gamma\cup\{j\}}\boldsymbol{\alpha}\|_2^2\right\}$;
      Update the active set:
        $\Gamma_k = \Gamma_{k-1} \cup \{j^*\}$;
      Update the residual (orthogonal projection):
        $\boldsymbol{q}_k = (I - \boldsymbol{\Phi}_{\Gamma_k}(\boldsymbol{\Phi}_{\Gamma_k}^{\mathrm{T}}\boldsymbol{\Phi}_{\Gamma_k})^{-1}\boldsymbol{\Phi}_{\Gamma_k}^{\mathrm{T}})\boldsymbol{f}_i$;
      Update the coefficients:
        $\boldsymbol{\alpha}_{\Gamma_k} = (\boldsymbol{\Phi}_{\Gamma_k}^{\mathrm{T}}\boldsymbol{\Phi}_{\Gamma_k})^{-1}\boldsymbol{\Phi}_{\Gamma_k}^{\mathrm{T}}\boldsymbol{f}_i$;
      $r_k = \|\boldsymbol{q}_k\|_2^2$;
      $k \leftarrow k + 1$;
  end
```

```
Algorithm 2: Sparse Feature Graph
  Input  : Data matrix $\boldsymbol{F} = [\boldsymbol{f_1}, \boldsymbol{f_2}, \ldots, \boldsymbol{f_d}] \in \mathbb{R}^{n\times d}$;
  Output: Adjacent matrix $\boldsymbol{W}$ of Graph $\boldsymbol{G} \in \mathbb{R}^{d\times d}$;
  Normalize each feature vector $\boldsymbol{f_i}$ with $\|\boldsymbol{f_i}\|_2^2 = \mathbf{1}$;
  for $i = 1, \ldots, d$ do
      Compute $\boldsymbol{\alpha}_i$ from OMP($\boldsymbol{F}_{-i}$,$\boldsymbol{f}_i$) using Algorithm 1;
  end
  Set adjacent matrix $W_{ij} = \boldsymbol{\alpha_i(j)}$ if $i > j$, $W_{ij} = \boldsymbol{\alpha_i(j-1)}$, if $i < j$ and $W_{ij} = 0$ if $i == j$;
```

5.2 Sparse representation error

In our modified OMP Algorithm 1, we set a new stop criterion of searching sparse representation solution for each feature vector $\boldsymbol{f}_i$. Instead of keep searching until a minimization error arrives, we stop running while the solver could not reduce the length of residual vector anymore. To be specific, the 2-norm of residual vector is monitored and the solver will stop once the change of this value small than a user specified threshold.

The reason we use this new stop criterion is that several feature vectors may not find correct sparse representation in current dataset, and the ordinary OMP solver will return a meaningless sparse representation when the maximum iteration threshold arrived. Since the goal of SFG is not to find a correct sparse representation for every feature vectors, we utilize the new stop criterion and add a filter process in our algorithm to identify those failed sparse representation.

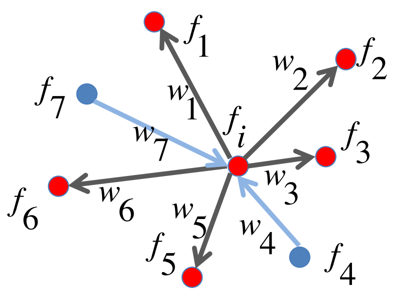

Fig. 5 Illustration of sparse representation error. SFG is a weighted directed graph

To identify those failed sparse representation, we check the angle between the original vector and the linear combination of its sparse representation. In the language of SFG, we check the angle between a node (a feature vector) and the weighted combination of its one-ring neighbor. Only the neighbors of out edges will be considered. This can be illustrated by following Fig. 5. As the example in Fig. 5, node $\boldsymbol{f}_i$ has seven one-ring neighbors. But only $\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3, \boldsymbol{f}_5, \boldsymbol{f}_6$ are its sparse representation and $\boldsymbol{f}_4$ and $\boldsymbol{f}_7$ are not. Then, the sparse representation error ζ is calculated by:

$$\boldsymbol{f}_i^* = w_1\boldsymbol{f}_1 + w_2\boldsymbol{f}_2 + w_3\boldsymbol{f}_3 + w_5\boldsymbol{f}_5 + w_6\boldsymbol{f}_6,$$
$$\zeta = \arccos\left(\boldsymbol{f}_i, \boldsymbol{f}_i^*\right).$$

Once we have the SFG, we calculate the sparse representation errors for all nodes. A sparse representation is treated as fail if the angle ζ less than a user specified value. We will filter out these node which has failed representation by removing its out edges.

5.3 Local compressible subgraph

We group high correlated features through *local compressible subgraphs*. The proposed SFG $\boldsymbol{G}$ is a weighted directed graph. With it, we need to find all feature subsets which are dense subgraphs that has very high redundancy. Dense subgraph searching [13] is a classical problem in graph mining and theory research. Recently, several elegant algorithms [24,31] were proposed to find subgraphs with specific property efficiently. However, most of them focus on the structure of subgraph. For our specific redundant problem, the edge weight is the most important factor. By this observation, we propose a local search algorithm with seed nodes to group those highly correlated features into many subgraphs which are named as *local compressible subgraphs* in this study. Our local search algorithm involves two steps, the first step is to sort all nodes by the in-degree. By the definition of SFG, the node with higher in-degree means it appears more frequently in other nodes' sparse representation. The second step is a local breadth-first search approach to grow the subgraph by finding all nodes that has higher weight connections (in and out) to the current subgraph. The details of subgraphs searching algorithm can be described by Algorithm 3.

Algorithm 3: Local Compressible Subgraphs.

```
Input  : Weighted directed graph $G = (V, E)$, edge weight
         threshold $\theta$;
Output: Local compressible subgraphs $C$ .

Tag all nodes with initial label 0;
Sort the nodes by its in-degree decreasingly;
$current\_label = 1$;
for $n = 1 : |V|$ do
    if $label(n) \,! = 0$ then
        continue;
    end
    set label of node $n$ to $current\_label$;
    $BFS(n, \theta, current\_label)$;
    $current\_label += 1$;
end
/* current_label now has the maximum value of
   labels.                                      */
for $i = 1 : current\_label$ do
    Extract subgraph $c_i$ which all nodes have label $i$;
    if $|c_i| > 1$ then
        add $c_i$ to $C$;
    end
end
```

In Algorithm 3, function $label(n)$ check the current label of node n, and $BFS(n, \theta, current\_label)$ function runs a local breadth-first search for subgraph that has edge weight large than θ.

5.4 Redundant feature removal

The last step of our algorithm is to remove the redundant features. For each local compressible subgraph we found, we pick up the node which has the highest in-degree as the representative node of that local compressible subgraph. So the number of final feature vectors equals to the number of local compressible subgraphs.

6 Complexity

The complexity of sparse feature graph (2) is $\mathcal{O}\left(n^2 d + ndk + ndk^2 + nk^3\right)$ approximately, where n equals to the number of samples, d equals to the size of features and k equals to the number of iterations of OMP solver. This complexity value is calculated based on the fact that each sample needs to search for a sparse representation w.r.t other samples. In our work, we use OMP solver to find those sparse representations. Its native version has time complexity $\mathcal{O}\left(nd + dk + dk^2 + k^3\right)$ in general [22].

Table 1 Details of selected twelve datasets

Name	#Sample	#Feature	#Class	Type
ORL	400	1024	40	Image
Yale	165	1024	15	Image
PIE10P	210	2420	10	Image
ORL10P	100	10,304	10	Image
BASEHOCK	1993	4862	2	Text
RELATHE	1427	4322	2	Text
PCMAC	1943	3289	2	Text
Reuters	8293	18,933	65	Text
Lymphoma	96	4026	9	Biology
LUNG	203	3312	5	Biology
Carcinom	174	9182	11	Biology
CLL-SUB-111	111	11,340	3	Biology

We use a modified version of OMP to improve the computational efficiency. During our experiments, we observe that the convergence rate of searching sparse representation for several samples is slow because the OMP solver cannot reach the regression error threshold which is set by the user. To avoid this situation, we modify the stop of criterion of OMP as we mentioned in Sect. 5.2. This modification does not change the time complexity of original OMP solver in theory. It is more like a practical remedy for our proposed algorithm.

The complexity of local compressible subgraphs Algorithm 3 is $|\mathcal{E}|$, where $|E|$ equals to the number of sparse feature graph $G = (V, E)$.

7 Experiments

In this section, we present experimental results to demonstrate the effectiveness of our proposed algorithms. We first evaluate the spectral clustering performance before and after applying our algorithms. Secondly, we show the performance of MCFS with or without our algorithm. In the last, the properties of generated sparse graphs and sensitivity of parameters are discussed.

7.1 Experiment setup

Datasets We select twelve real-world high-dimensional datasets [11] from three different domains: image, text and biology. The detail of each dataset is as in Table 1. These datasets have sample size different from 96 to 8293 and feature size range from 1024 to 18,933.

Normalization The features of each dataset are normalized to have unit length: $\|\boldsymbol{f_i}\|_2 = 1$.

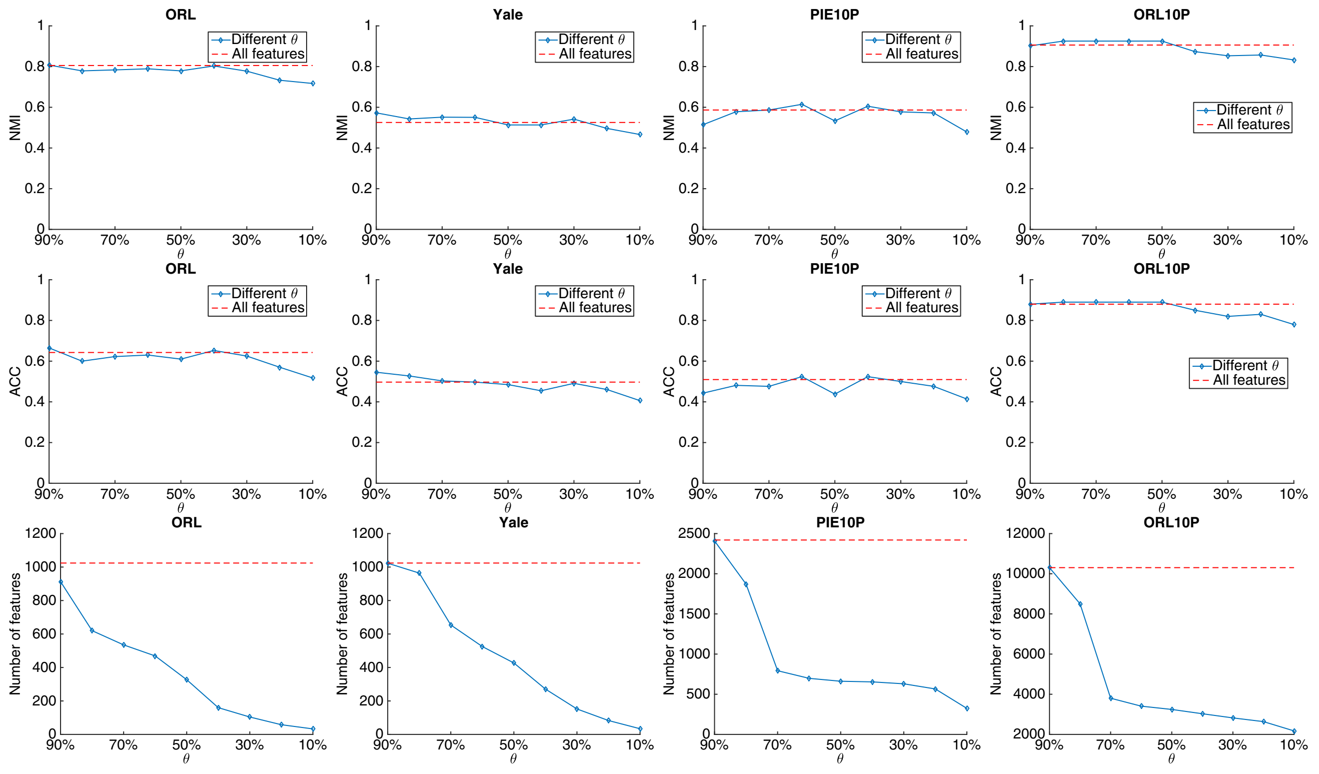

Fig. 6 Spectral clustering performance of *image* datasets with different parameter θ. Top row: NMI; middle row: ACC; bottom row: number of features, the red dash line means the size of raw dataset

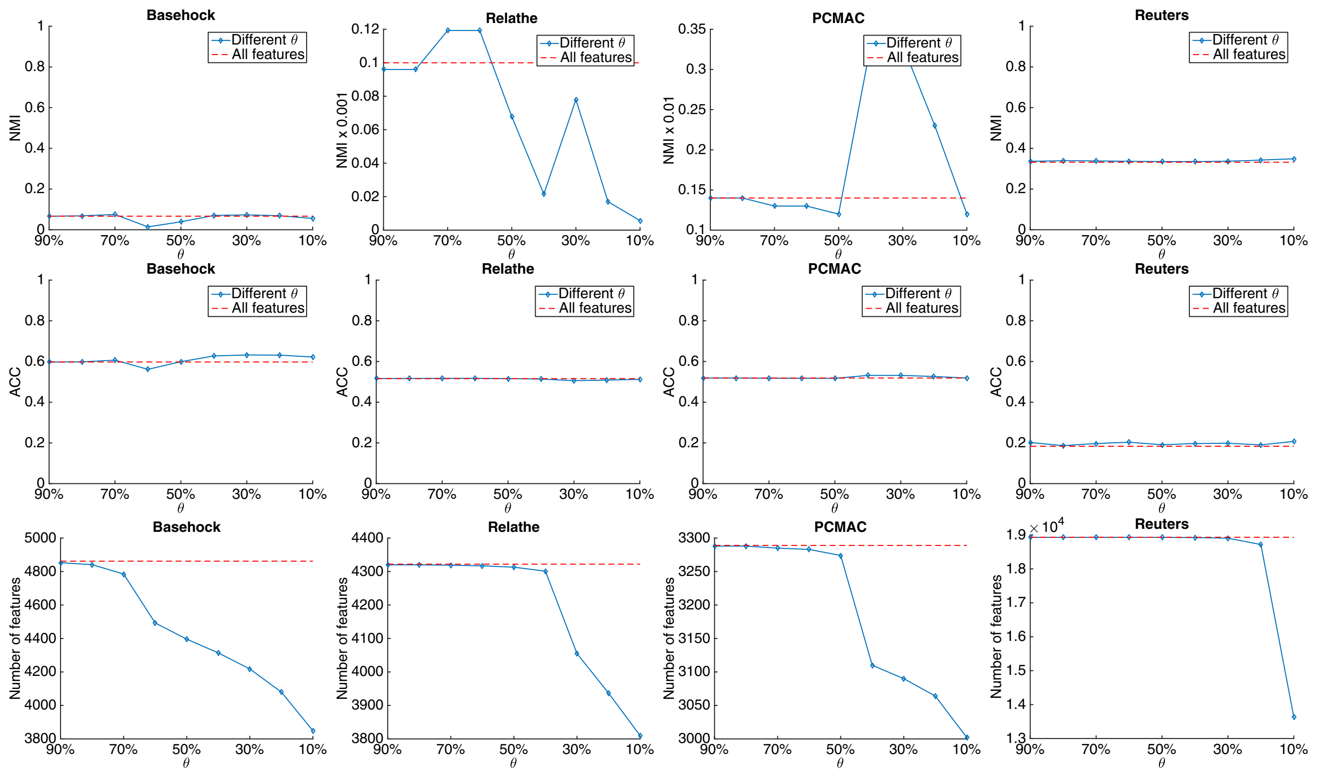

Fig. 7 Spectral clustering performance of *text* datasets with different parameter θ. Top row: NMI; middle row: ACC; bottom row: number of features, the red dash line means the size of raw dataset

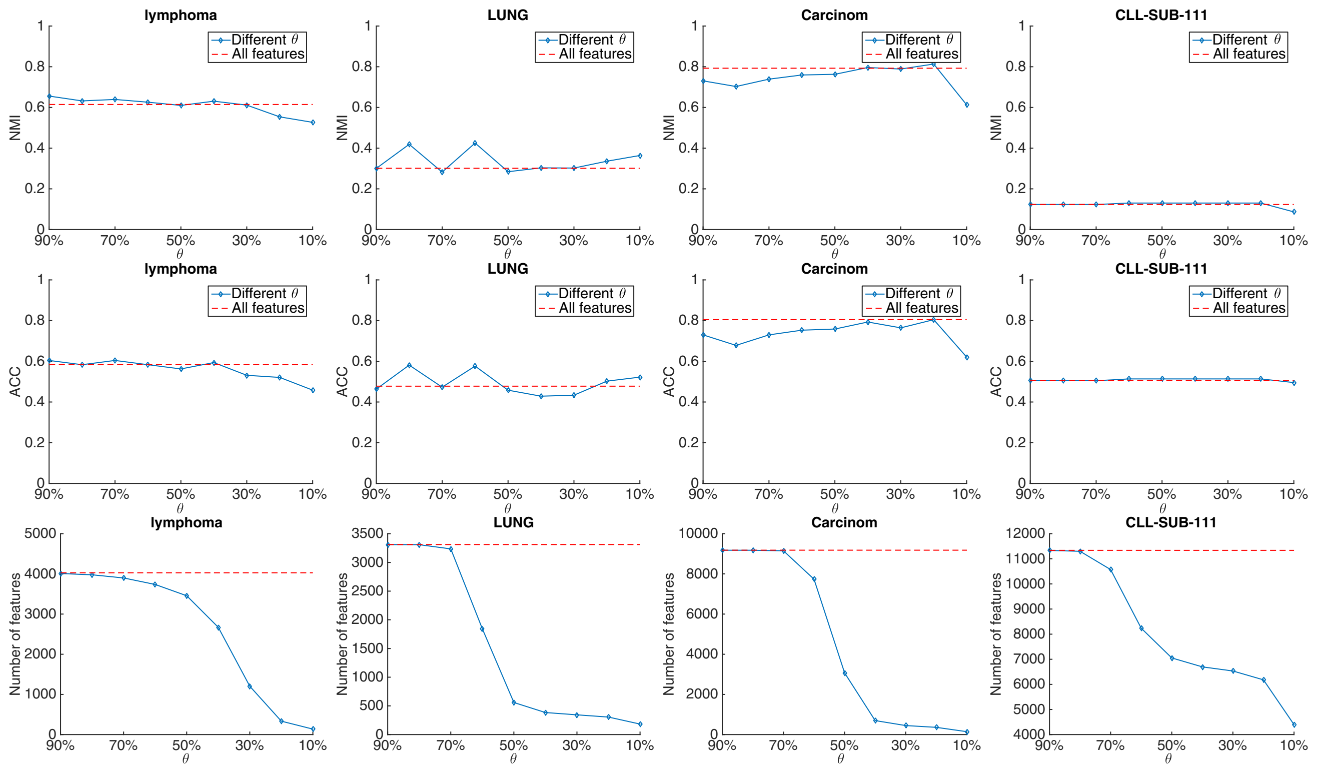

Fig. 8 Spectral clustering performance of *Biology* datasets with different parameter θ. Top row: NMI; middle row: ACC; bottom row: number of features, the red dash line means the size of raw dataset

Table 2 NMI results of "ORL" dataset

# *f*	1024	913	620	535	469	327	160	104	58	33
10	0.63	0.51	0.60	0.56	0.53	0.62	0.61	**0.65**	0.60	0.62
15	0.66	0.56	0.63	0.60	0.58	**0.67**	0.62	0.60	0.63	0.58
20	0.67	0.59	0.65	0.64	0.59	0.64	0.63	0.61	0.64	0.56
25	0.67	0.59	0.66	0.64	0.63	0.65	0.66	0.64	0.65	0.58
30	0.68	0.63	0.66	0.65	0.66	0.67	0.65	0.67	0.65	0.59
35	0.69	0.64	**0.70**	0.66	0.65	0.67	0.67	0.68	0.65	–
40	0.70	0.67	**0.71**	0.68	0.67	0.68	**0.70**	**0.70**	0.66	–
45	0.70	0.69	**0.70**	0.69	0.66	0.69	**0.70**	0.69	0.65	–
50	0.73	0.71	0.72	0.68	0.66	0.70	0.72	0.69	0.66	–
55	0.71	**0.74**	0.70	0.68	0.67	0.71	0.71	0.71	0.66	–
60	0.71	**0.74**	**0.71**	**0.72**	**0.71**	0.69	**0.72**	0.71	–	–

Bold values indicate better result

Evaluation metric Our proposed algorithms are under the framework of unsupervised learning. Without loss of generality, the cluster structure of data is used for evaluation. To be specific, we measure the spectral clustering performance with normalized mutual information (NMI) and accuracy (ACC). NMI value ranges from 0.0 to 1.0, with higher value means better clustering performance. ACC is another metric to evaluate the clustering performance by measuring the fraction of its clustering result that are correct. Similar to NMI, its values range from 0 to 1 and higher value indicates better algorithm performance.

Suppose A is the clustering result and B is the known sample label vector. Let $p(a)$ and $p(b)$ denote the marginal probability mass function of A and B, and let $p(a, b)$ be the joint probability mass function of A and B. Suppose $H(A)$, $H(B)$ and $H(A, B)$ denote the entropy of $p(a)$, $p(b)$ and $p(a, b)$, respectively. Then, the normalized mutual information NMI is defined as:

Table 3 ACC results of "ORL" dataset

# *f*	1024	913	620	535	469	327	160	104	58	33
10	0.38	0.28	0.36	0.31	0.28	**0.39**	**0.39**	**0.46**	**0.39**	**0.41**
15	0.45	0.33	0.41	0.40	0.34	0.43	0.40	0.38	0.42	0.36
20	0.47	0.34	0.43	0.43	0.35	0.43	0.41	0.39	0.43	0.32
25	0.48	0.35	0.45	0.44	0.37	0.42	0.47	0.41	0.45	0.34
30	0.47	0.40	0.42	0.42	0.43	0.47	0.43	0.45	0.42	0.35
35	0.49	0.41	0.48	0.46	0.44	0.44	0.47	0.47	0.42	–
40	0.51	0.46	**0.53**	0.48	0.46	0.45	0.48	**0.51**	0.43	–
45	0.49	0.47	**0.51**	**0.51**	0.44	0.48	**0.49**	**0.49**	0.43	–
50	0.55	0.51	0.52	0.47	0.47	0.50	0.52	0.48	0.46	–
55	0.53	**0.53**	0.51	0.46	0.45	0.48	0.50	**0.53**	0.46	–
60	0.51	**0.55**	**0.52**	**0.54**	0.51	0.47	**0.54**	0.51	–	–

Bold values indicate better result

Table 4 NMI results of "Yale" dataset

# *f*	1024	1023	964	654	525	427	271	152	83	34
10	0.48	0.43	0.43	0.45	0.42	0.46	0.45	0.46	0.47	0.44
15	0.49	0.47	0.46	**0.51**	**0.49**	0.48	0.45	0.47	**0.50**	0.43
20	0.49	0.48	0.46	**0.55**	0.48	**0.51**	0.47	0.47	**0.51**	0.41
25	0.51	0.49	0.49	**0.52**	**0.52**	**0.52**	0.45	0.49	**0.54**	0.41
30	0.51	**0.51**	0.49	**0.54**	0.50	**0.51**	**0.51**	0.49	**0.50**	0.39
35	0.53	0.49	0.50	**0.54**	**0.53**	0.52	0.52	0.48	0.50	–
40	0.49	**0.50**	**0.51**	**0.53**	**0.58**	**0.55**	**0.55**	0.48	**0.51**	–
45	0.48	**0.51**	**0.51**	**0.56**	**0.59**	**0.57**	**0.52**	**0.52**	**0.49**	–
50	0.52	0.50	0.47	**0.53**	**0.59**	**0.53**	**0.53**	**0.56**	0.49	–
55	0.54	0.51	0.52	**0.55**	0.50	0.51	0.51	0.51	0.49	–
60	0.54	0.49	0.51	0.49	**0.54**	0.50	0.51	0.46	0.52	–

Bold values indicate better result

$$NMI(A, B) = \frac{H(A) + H(B) - H(A, B)}{\max(H(A), H(B))}. \tag{4}$$

Assume A is the clustering result label vector, and B is the known ground truth label vector, ACC is defined as:

$$ACC = \frac{\sum_{i=1}^{N} \delta(B(i), Map_{(A,B)}(i))}{N}, \tag{5}$$

where N denotes the length of label vector, $\delta(a, b)$ equals to 1 if only if a and b are equal. $Map_{A,B}$ is the best mapping function that permutes A to match B.

7.2 Effectiveness of redundant features removal

Our proposed algorithm removes many features to reduce the dimension size of all data vectors. As a consequence, the pairwise Euclidean distance is changed and the cluster structure will be affected. To measure the effectiveness of our proposed algorithm, we check the spectral clustering performance before and after redundant feature removal. If the NMI and ACC values are not changed to much and stay in the same level, the experiment results show that our proposed algorithm is correct and effective.

The spectral clustering algorithm we used in our experiments is the Ng–Jordan–Weiss (NJW) algorithm [18]. The Gaussian similarity graph is applied here as the input and parameter σ is set to the mean value of pairwise Euclidean distance among all vectors.

Our proposed LCS algorithm includes a parameter θ which is the threshold of redundancy. It decides the number of redundant features implicitly and affects the cluster structure of data consequently. In our experiment design, we test different θ values ranging from 90 to 10% with step size equal to 10%: $\theta = [0.9, 0.8, 0.7, \ldots, 0.1]$.

We present our experiment results for image datasets, text datasets, and biological datasets in Figs. 6, 7 and 8, respectively. For each dataset, we show the NMI, ACC performance with different θ and compare with original spectral clustering performance by using all features. From the experimental

Table 5 ACC results of "Yale" dataset

# *f*	1024	1023	964	654	525	427	271	152	83	34
10	0.39	0.36	0.37	0.36	0.33	0.38	**0.41**	**0.40**	**0.41**	0.36
15	0.43	0.41	0.42	**0.44**	0.41	0.41	0.39	0.41	**0.46**	0.39
20	0.44	0.42	0.41	**0.48**	**0.44**	**0.44**	0.43	0.42	**0.44**	0.35
25	0.45	**0.45**	0.44	**0.46**	**0.47**	**0.45**	0.41	0.43	**0.49**	0.33
30	0.48	0.44	0.42	0.47	0.47	0.45	0.45	0.40	0.47	0.33
35	0.48	**0.48**	0.44	**0.50**	0.47	0.46	0.47	0.41	0.44	–
40	0.42	**0.44**	**0.45**	**0.50**	**0.55**	**0.48**	**0.53**	0.41	**0.44**	–
45	0.41	**0.48**	**0.46**	**0.51**	**0.53**	**0.54**	**0.49**	**0.47**	**0.42**	–
50	0.46	0.41	0.42	**0.48**	**0.56**	**0.50**	**0.46**	**0.52**	0.41	–
55	0.48	0.44	**0.48**	**0.48**	0.43	0.45	**0.49**	0.47	0.42	–
60	0.50	0.42	0.44	0.40	**0.50**	0.41	0.46	0.42	0.43	–

Bold values indicate better result

Table 6 NMI results of "PIE10P" dataset

# *f*	2420	2409	1871	793	698	662	654	630	566	324
10	0.44	**0.48**	**0.55**	**0.53**	**0.58**	**0.56**	**0.54**	**0.61**	**0.50**	0.38
15	0.44	**0.61**	**0.57**	**0.50**	**0.58**	**0.58**	**0.55**	**0.59**	**0.53**	0.39
20	0.43	**0.56**	**0.61**	**0.59**	**0.60**	**0.56**	**0.62**	**0.59**	**0.56**	0.41
25	0.52	**0.61**	**0.61**	**0.64**	**0.61**	**0.60**	**0.58**	**0.58**	**0.54**	**0.43**
30	0.53	**0.61**	**0.62**	**0.57**	**0.62**	**0.62**	**0.60**	**0.53**	**0.63**	0.41
35	0.59	**0.60**	**0.59**	**0.60**	**0.63**	**0.61**	**0.60**	**0.62**	**0.64**	0.43
40	0.53	**0.60**	**0.58**	**0.57**	**0.66**	**0.62**	**0.59**	**0.62**	**0.69**	0.42
45	0.55	**0.61**	**0.61**	**0.62**	**0.60**	**0.64**	0.60	**0.64**	**0.65**	0.43
50	0.56	**0.63**	**0.62**	**0.68**	**0.64**	**0.62**	**0.58**	**0.63**	**0.66**	0.37
55	0.61	0.60	**0.62**	**0.69**	**0.62**	0.60	0.57	**0.65**	0.58	0.39
60	0.55	**0.64**	**0.63**	**0.64**	**0.60**	**0.63**	0.54	**0.63**	0.51	0.39

Bold values indicate better result

Table 7 ACC results of "PIE10P" dataset

# *f*	2420	2409	1871	793	698	662	654	630	566	324
10	0.39	**0.45**	**0.48**	**0.50**	**0.56**	**0.50**	**0.53**	**0.59**	**0.46**	0.39
15	0.39	**0.58**	**0.51**	**0.49**	**0.51**	**0.55**	**0.56**	**0.60**	**0.50**	**0.41**
20	0.36	**0.51**	**0.53**	**0.53**	**0.55**	**0.56**	**0.60**	**0.54**	**0.50**	**0.38**
25	0.45	**0.59**	**0.53**	**0.60**	**0.54**	**0.59**	**0.60**	**0.56**	**0.52**	0.40
30	0.50	**0.58**	**0.56**	**0.58**	**0.59**	**0.60**	**0.59**	**0.49**	**0.60**	0.40
35	0.48	**0.57**	**0.51**	**0.59**	**0.61**	**0.53**	**0.54**	**0.62**	**0.61**	**0.37**
40	0.42	**0.52**	**0.53**	**0.56**	**0.63**	**0.59**	**0.53**	**0.60**	**0.64**	0.38
45	0.44	**0.52**	**0.52**	**0.58**	**0.51**	**0.63**	**0.54**	**0.62**	**0.60**	**0.41**
50	0.44	**0.61**	**0.52**	**0.64**	**0.60**	**0.59**	**0.55**	**0.62**	**0.61**	0.37
55	0.46	**0.54**	**0.53**	**0.67**	**0.58**	**0.57**	**0.57**	**0.63**	**0.54**	0.37
60	0.49	**0.60**	**0.61**	**0.61**	**0.57**	**0.61**	**0.51**	**0.61**	0.46	0.35

Bold values indicate better result

Table 8 NMI results of "ORL10P" dataset

# *f*	10,304	10,302	8503	3803	3408	3244	3030	2822	2638	2175
10	0.65	**0.78**	**0.77**	**0.76**	**0.77**	**0.80**	**0.74**	**0.72**	**0.75**	**0.73**
15	0.72	**0.82**	**0.79**	**0.78**	**0.81**	**0.83**	**0.79**	**0.81**	**0.75**	**0.79**
20	0.76	**0.81**	0.74	**0.78**	**0.84**	**0.83**	**0.81**	**0.76**	**0.80**	**0.78**
25	0.79	**0.84**	0.74	0.73	**0.82**	**0.86**	**0.88**	**0.83**	**0.86**	**0.81**
30	0.75	**0.77**	**0.82**	0.74	**0.88**	**0.82**	**0.83**	**0.83**	**0.86**	**0.86**
35	0.81	**0.81**	0.80	**0.83**	**0.85**	**0.83**	0.80	**0.82**	**0.85**	**0.85**
40	0.83	**0.88**	**0.84**	**0.84**	**0.90**	**0.86**	0.81	**0.93**	**0.84**	**0.87**
45	0.84	0.93	0.83	**0.85**	**0.91**	**0.86**	0.83	**0.88**	**0.84**	**0.86**
50	0.78	**0.88**	**0.88**	**0.87**	**0.89**	**0.86**	**0.82**	**0.90**	**0.84**	**0.83**
55	0.84	**0.89**	**0.86**	**0.89**	**0.91**	**0.89**	**0.88**	**0.86**	**0.84**	**0.86**
60	0.85	**0.88**	**0.86**	0.84	**0.85**	**0.91**	**0.85**	**0.88**	**0.86**	**0.85**

Bold values indicate better result

Table 9 ACC results of "ORL10P" dataset

# *f*	10,304	10,302	8503	3803	3408	3244	3030	2822	2638	2175
10	0.66	**0.74**	**0.81**	**0.75**	**0.75**	**0.69**	**0.72**	**0.70**	**0.69**	**0.67**
15	0.69	**0.85**	**0.76**	**0.78**	**0.78**	**0.86**	**0.80**	**0.75**	**0.73**	**0.75**
20	0.77	**0.84**	0.74	0.76	**0.80**	**0.80**	**0.78**	0.69	0.75	0.74
25	0.71	**0.79**	0.68	**0.74**	**0.78**	**0.86**	**0.84**	**0.82**	**0.82**	**0.74**
30	0.71	**0.71**	**0.77**	0.68	**0.86**	**0.77**	**0.81**	**0.77**	**0.82**	**0.81**
35	0.74	**0.74**	**0.74**	**0.76**	**0.81**	**0.77**	0.73	**0.76**	**0.82**	**0.78**
40	0.80	**0.85**	0.74	0.77	**0.87**	**0.80**	0.75	**0.89**	**0.80**	**0.83**
45	0.82	**0.89**	0.73	0.81	**0.88**	0.78	0.77	**0.86**	0.80	0.79
50	0.73	**0.80**	**0.80**	**0.74**	**0.86**	**0.79**	**0.74**	**0.88**	**0.81**	**0.77**
55	0.79	**0.85**	**0.82**	**0.86**	**0.89**	**0.87**	**0.80**	**0.82**	**0.81**	**0.79**
60	0.82	**0.84**	0.77	0.75	**0.82**	**0.89**	0.77	**0.84**	**0.82**	**0.82**

Bold values indicate better result

Table 10 NMI results of "Lymphoma" dataset

# *f*	4026	4009	3978	3899	3737	3456	2671	1203	334	136
10	0.51	**0.59**	**0.58**	**0.52**	0.50	0.50	**0.51**	0.50	0.50	0.49
15	0.55	**0.60**	**0.62**	**0.56**	**0.58**	**0.58**	**0.58**	**0.56**	0.47	0.52
20	0.60	**0.61**	**0.60**	0.57	**0.62**	**0.62**	**0.64**	0.58	0.58	**0.60**
25	0.63	0.59	**0.64**	0.60	**0.63**	0.58	**0.66**	0.57	0.56	0.53
30	0.59	**0.61**	**0.62**	**0.60**	**0.62**	**0.64**	**0.65**	**0.60**	**0.60**	**0.59**
35	0.61	**0.66**	**0.62**	0.60	**0.65**	**0.62**	**0.61**	**0.62**	0.56	0.53
40	0.64	0.60	**0.66**	0.63	0.61	0.63	**0.66**	0.61	0.58	0.55
45	0.58	**0.63**	**0.62**	**0.62**	**0.58**	**0.61**	**0.63**	**0.64**	**0.60**	0.57
50	0.65	0.60	0.61	0.61	0.56	0.63	0.61	0.63	0.58	0.54
55	0.63	0.60	0.61	0.62	0.60	0.60	**0.63**	0.60	0.58	0.58
60	0.60	**0.60**	**0.63**	**0.61**	**0.63**	0.59	**0.65**	0.59	0.57	0.57

Bold values indicate better result

Table 11 ACC results of "Lymphoma" dataset

# *f*	4026	4009	3978	3899	3737	3456	2671	1203	334	136
10	0.50	**0.57**	**0.56**	**0.53**	0.49	**0.51**	**0.51**	0.48	**0.50**	**0.50**
15	0.53	**0.62**	**0.59**	**0.58**	**0.56**	**0.59**	**0.58**	**0.55**	0.50	**0.53**
20	0.59	0.56	0.55	0.56	0.56	**0.59**	**0.59**	0.54	0.55	**0.59**
25	0.60	0.57	**0.62**	0.56	**0.62**	0.58	**0.64**	0.56	0.52	0.50
30	0.56	**0.60**	**0.58**	**0.58**	**0.59**	**0.61**	**0.65**	**0.59**	**0.57**	0.55
35	0.55	**0.62**	**0.59**	**0.58**	**0.61**	**0.60**	**0.57**	**0.59**	**0.55**	0.53
40	0.66	0.57	0.61	0.61	0.61	0.59	0.60	0.58	0.59	0.54
45	0.54	**0.60**	**0.60**	**0.58**	**0.55**	**0.60**	**0.62**	**0.59**	**0.56**	**0.54**
50	0.65	0.62	0.58	0.64	0.52	0.59	0.56	0.59	0.53	0.53
55	0.57	**0.60**	**0.65**	**0.60**	0.54	**0.57**	**0.65**	**0.59**	0.54	**0.59**
60	0.56	**0.58**	**0.64**	**0.58**	**0.61**	**0.57**	**0.67**	**0.56**	0.53	**0.57**

Bold values indicate better result

Table 12 NMI results of "LUNG" dataset

# *f*	3312	3311	3309	3236	1844	559	384	344	305	183
10	0.42	**0.42**	**0.43**	**0.49**	**0.52**	**0.53**	**0.43**	**0.46**	**0.43**	0.25
15	0.54	**0.54**	0.53	0.51	0.51	0.51	0.45	0.52	0.38	0.21
20	0.51	**0.51**	**0.52**	**0.53**	0.41	0.49	0.36	**0.52**	0.38	0.20
25	0.51	**0.51**	**0.53**	0.48	0.42	**0.52**	0.40	0.48	0.35	0.26
30	0.47	**0.48**	**0.52**	**0.49**	0.41	0.37	**0.49**	**0.48**	0.41	0.24
35	0.46	0.38	**0.46**	**0.48**	0.39	**0.52**	**0.49**	0.38	0.35	0.27
40	0.49	**0.49**	**0.50**	0.46	0.43	0.40	0.38	0.35	0.40	0.29
45	0.36	**0.42**	0.33	**0.47**	**0.40**	0.33	**0.38**	0.35	0.35	0.31
50	0.45	**0.45**	**0.47**	**0.49**	**0.52**	0.32	0.40	0.36	0.35	0.31
55	0.44	**0.44**	**0.44**	**0.49**	**0.51**	0.33	**0.49**	0.31	0.30	0.31
60	0.47	0.46	0.45	**0.51**	**0.49**	0.33	0.39	0.32	0.31	0.35

Bold values indicate better result

Table 13 ACC results of "LUNG" dataset

# *f*	3312	3311	3309	3236	1844	559	384	344	305	183
10	0.71	**0.72**	**0.73**	**0.77**	**0.77**	**0.75**	0.68	0.65	0.66	0.56
15	0.81	**0.81**	0.79	0.72	0.73	0.72	0.67	0.65	0.58	0.48
20	0.71	**0.73**	**0.74**	**0.72**	0.69	0.69	0.61	0.60	0.58	0.39
25	0.71	**0.71**	**0.74**	0.67	0.69	0.68	0.59	0.61	0.56	0.49
30	0.66	**0.66**	**0.67**	**0.71**	**0.68**	0.56	0.59	0.59	0.61	0.43
35	0.64	0.60	0.63	**0.68**	**0.66**	0.60	0.58	0.56	0.53	0.49
40	0.65	**0.65**	**0.66**	**0.65**	0.64	0.57	0.54	0.54	0.56	0.46
45	0.60	**0.63**	0.57	**0.65**	**0.61**	0.52	0.54	0.52	0.52	0.49
50	0.65	**0.65**	0.63	**0.65**	**0.65**	0.48	0.57	0.53	0.53	0.52
55	0.61	**0.61**	0.59	**0.65**	**0.62**	0.48	0.59	0.48	0.49	0.49
60	0.64	0.63	0.63	**0.64**	0.62	0.51	0.55	0.49	0.48	0.51

Bold values indicate better result

Table 14 NMI results of "Carcinom" dataset

# f	9182	9180	9179	9150	7736	3072	697	449	360	144
10	0.70	**0.70**	**0.70**	0.69	0.67	0.64	0.66	0.65	0.66	0.47
15	0.71	0.70	**0.73**	**0.73**	**0.74**	0.66	0.67	0.70	0.66	0.52
20	0.77	**0.78**	**0.77**	0.72	0.75	0.72	0.73	0.71	0.73	0.54
25	0.74	**0.77**	**0.77**	**0.75**	**0.74**	0.71	**0.79**	**0.75**	**0.74**	0.53
30	0.69	**0.71**	**0.72**	**0.70**	**0.74**	**0.75**	**0.77**	**0.79**	**0.73**	0.54
35	0.77	0.76	0.76	0.76	0.74	**0.77**	**0.78**	**0.78**	**0.78**	0.60
40	0.75	0.74	**0.76**	**0.77**	0.74	**0.79**	**0.76**	**0.78**	**0.75**	0.59
45	0.77	0.76	0.74	**0.78**	0.74	**0.82**	**0.78**	**0.80**	**0.79**	0.57
50	0.79	0.76	0.75	0.75	**0.79**	0.76	**0.79**	**0.84**	**0.83**	0.58
55	0.75	**0.76**	**0.76**	0.74	**0.75**	**0.79**	**0.79**	**0.83**	**0.83**	0.59
60	0.74	0.72	**0.76**	0.73	**0.76**	**0.82**	**0.84**	**0.82**	**0.78**	0.62

Bold values indicate better result

Table 15 ACC results of "Carcinom" dataset

# f	9182	9180	9179	9150	7736	3072	697	449	360	144
10	0.63	**0.66**	0.62	0.61	**0.67**	0.60	0.60	0.59	**0.64**	0.48
15	0.67	0.57	**0.70**	0.66	**0.68**	0.63	0.57	**0.67**	0.64	0.53
20	0.70	0.68	**0.74**	0.66	**0.71**	**0.71**	0.64	**0.73**	**0.74**	0.56
25	0.70	**0.72**	**0.75**	0.69	**0.75**	0.64	**0.75**	**0.72**	**0.76**	0.51
30	0.61	**0.64**	**0.70**	**0.69**	**0.67**	**0.71**	**0.74**	**0.76**	**0.71**	0.52
35	0.76	0.74	0.74	0.74	0.70	0.75	0.70	**0.76**	**0.77**	0.57
40	0.72	**0.72**	**0.73**	**0.75**	0.69	**0.76**	0.66	**0.78**	0.71	0.56
45	0.75	0.74	0.70	**0.75**	0.74	**0.79**	0.72	**0.79**	**0.76**	0.55
50	0.74	**0.74**	0.70	0.72	**0.74**	0.66	**0.74**	**0.83**	**0.79**	0.56
55	0.73	**0.74**	**0.74**	0.72	0.71	0.72	0.72	**0.82**	**0.80**	0.56
60	0.70	0.61	**0.71**	0.66	**0.72**	**0.75**	**0.82**	**0.80**	**0.77**	0.55

Bold values indicate better result

Table 16 NMI results of "CLL-SUB-111" dataset

# f	11,340	11,335	11,301	10,573	8238	7053	6697	6533	6180	4396
10	0.16	**0.16**	0.15	**0.26**	**0.18**	**0.22**	**0.20**	**0.20**	**0.20**	**0.21**
15	0.14	**0.14**	**0.15**	**0.26**	**0.18**	**0.28**	0.09	**0.24**	0.07	0.06
20	0.16	**0.16**	0.15	0.08	0.14	**0.21**	0.04	**0.31**	**0.16**	0.11
25	0.14	**0.14**	**0.15**	0.09	0.08	**0.22**	**0.23**	0.10	0.09	0.11
30	0.13	**0.13**	**0.13**	0.08	0.07	**0.18**	0.03	**0.14**	0.10	0.11
35	0.17	**0.17**	0.13	0.03	0.07	0.12	0.10	0.01	0.08	0.10
40	0.14	**0.14**	**0.14**	0.07	0.08	0.13	0.12	0.05	**0.14**	0.09
45	0.09	**0.09**	**0.18**	0.08	**0.11**	**0.10**	**0.13**	0.07	**0.12**	**0.09**
50	0.15	0.14	**0.15**	0.08	0.11	0.11	0.12	0.12	0.13	0.09
55	0.15	**0.15**	0.14	**0.21**	0.08	0.13	0.13	0.12	0.13	0.07
60	0.10	**0.10**	**0.14**	**0.15**	0.08	**0.10**	**0.12**	**0.12**	**0.14**	0.07

Bold values indicate better result

results, we can read that: Even when θ is reduced to 30%, the NMI and ACC values are staying in same level as original data. When θ equals to 30%, it means that the edges of SFG with weights (absolute value) in the highest 70% value range are removed. (It does not mean that 70% of top weights edges are removed.) This observation validates the correctness of our proposed algorithm.

7.3 Performance of MCFS

Our proposed algorithm is targeting for unsupervised feature selection. And the quality of indication vectors (or the spectral clustering performance based on eigenvectors) is an important factor evaluate the effectiveness of our proposed algorithm. In this section, we evaluate the MCFS perfor-

Table 17 ACC results of "CLL-SUB-111" dataset

# *f*	11,340	11,335	11,301	10,573	8238	7053	6697	6533	6180	4396
10	0.51	**0.51**	0.50	**0.54**	**0.59**	**0.57**	**0.58**	**0.55**	**0.51**	0.50
15	0.51	**0.51**	0.50	**0.57**	**0.55**	**0.62**	0.47	**0.59**	0.45	0.43
20	0.50	**0.50**	0.48	0.46	**0.50**	**0.54**	0.40	**0.59**	**0.54**	**0.50**
25	0.48	**0.48**	**0.51**	0.44	0.46	**0.54**	**0.57**	**0.50**	0.46	**0.50**
30	0.49	**0.49**	**0.49**	0.44	0.44	**0.53**	0.42	**0.51**	0.48	0.48
35	0.51	**0.51**	0.49	0.42	0.44	0.49	0.49	0.41	0.44	0.48
40	0.51	**0.51**	0.50	0.43	0.45	0.50	0.49	0.43	0.48	0.47
45	0.46	0.45	**0.52**	0.44	**0.46**	**0.47**	**0.51**	0.45	**0.47**	**0.47**
50	0.51	0.50	**0.51**	0.45	0.46	0.50	0.49	0.49	0.49	0.48
55	0.49	**0.49**	**0.50**	**0.54**	0.46	**0.50**	**0.50**	**0.49**	**0.49**	0.45
60	0.49	**0.49**	**0.50**	**0.53**	0.43	0.48	**0.49**	**0.49**	**0.50**	0.44

Bold values indicate better result

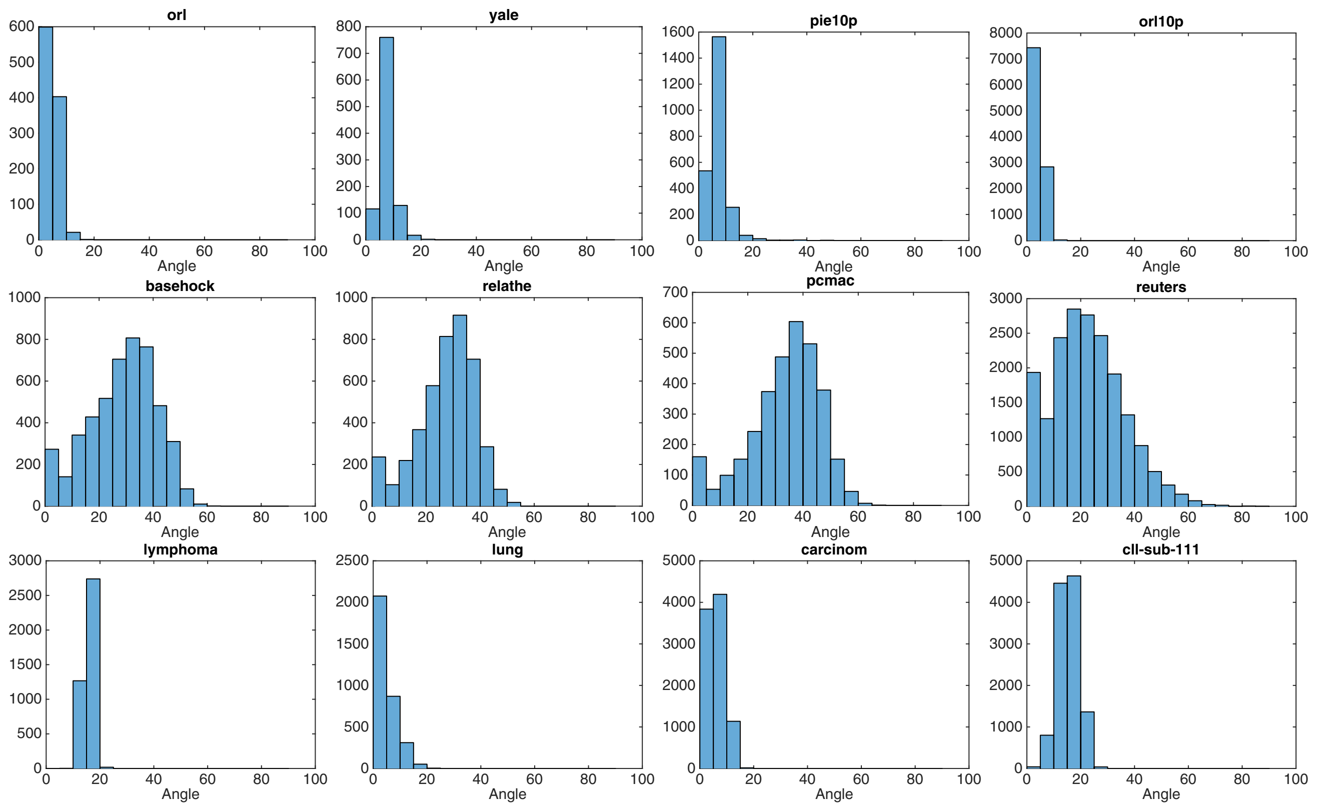

Fig. 9 The distribution of angle between original feature vector and its sparse representation

mance over the redundant feature removed data and compare with the raw data that without any feature removal.

The spectral clustering performance is measured for different input data from original whole feature data to processed ones by our proposed algorithm with different θ. We report the experiment results over image datasets and biological datasets in this section. For text datasets, the feature vectors of them are very sparse, and our eigen-decomposition process is always failed and we only can collect partial results. For fair evaluation, we omit the experiment results of text datasets in this section. The result of MCFS performance shows from Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 and 17.

For each dataset, we set the number of selected features ranging from [10, 15, . . . , 60], which has 11 different sizes in total. The parameter θ is configured from 0.9 to 0.1 with step size equals to 0.1.

We report the experimental results in tables (from Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 and 17). For each table, the first row means the number of features that used as input of MCFS. The first column is the number of selected features by MCFS algorithm. The baseline is in the second

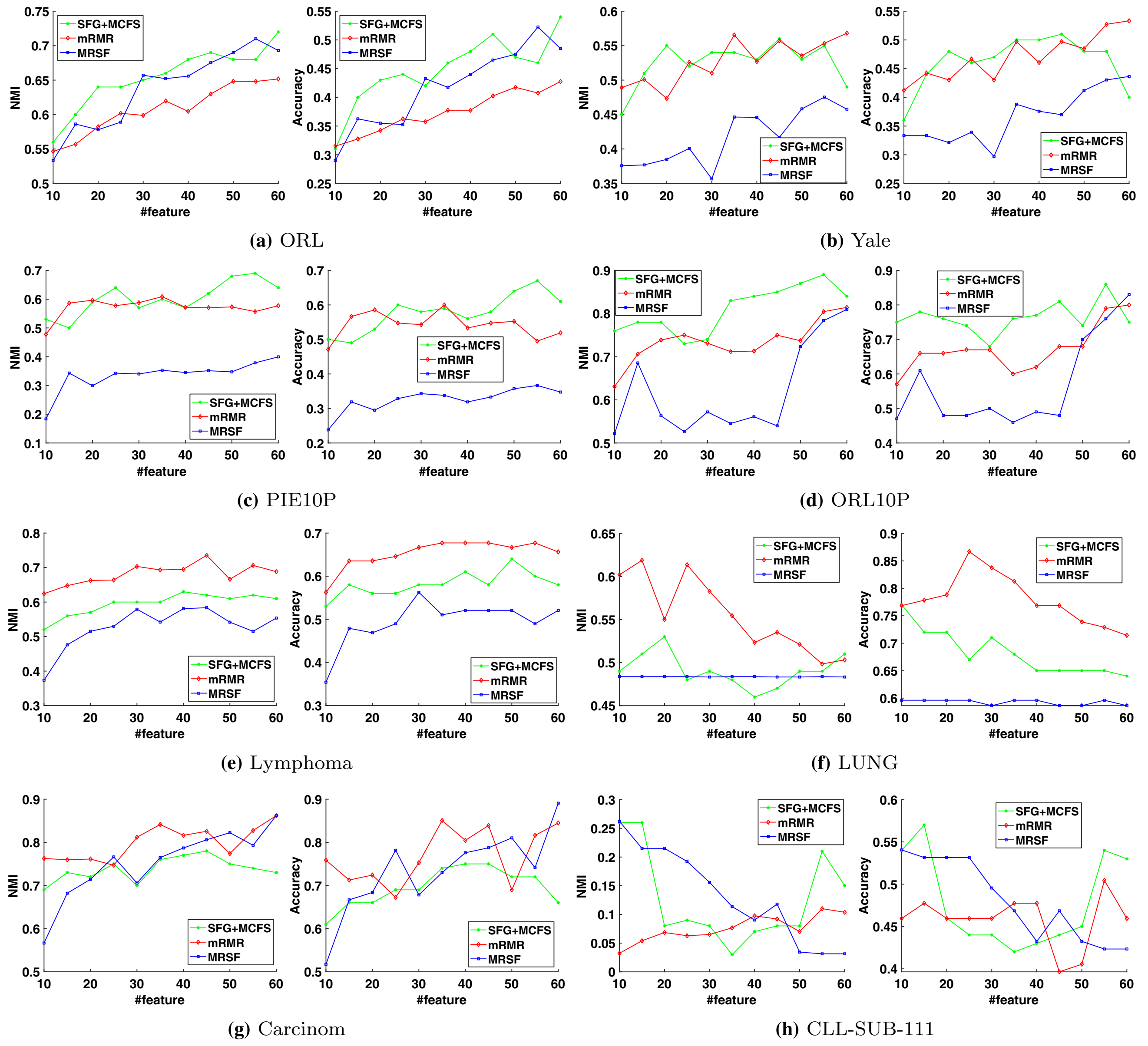

Fig. 10 The comparison results with mRMR algorithm and MRSF algorithm, **a** ORL, **b** Yale, **c** PIE10P, **d** ORL10P, **e** Lymphoma, **f** LUNG, **g** Carcinom, **h** CLL-SUB-111

column, which is the testing result of MCFS algorithm with raw data. The hyphens in the tables mean the number of selected features is larger than the feature size of input data, which means invalid test. To show the effectiveness of our algorithm, we also mark those NMI and ACC scores that larger or equals to baseline.

7.4 Sparse representation errors

With the design of our modified OMP solvers, there will be failed/wrong sparse representations existing in generated sparse feature graph. The meaning of these edge connections and edge weights are invalid. And they should be removed from the SFG since wrong connections will deteriorate the accuracy of feature redundancy relationship. To validate the sparse representation, we check the angle between original feature vector and the linear weighted summation resulted vector (or recover signal from sparse coding point of view) from its sparse representation. If the angle lower than a threshold, we remove all out-edges from the generated sparse feature graph. To specify the threshold, we learn it from the empirical results of our selected twelve datasets. The distribution (or histogram) result of angle values is presented in Fig. 9.

7.5 Comparison with mRMR and MRSF

In this section, we present the comparison results between SFG and two famous feature selection methods regarding to the feature redundancy. They are the "minimum Redundancy Maximum Relevance Feature Selection" (mRMR) method [19] (http://home.penglab.com/proj/mRMR/) and the "Efcient Spectral Feature Selection with Minimum Redundancy" (MRSF) method [32] (https://sites.google.com/site/alanzhao/Home).

During the comparison experiments, we fix the hyperparameter θ of our algorithm to 0.7. The reason we choose this specific value is that the number of features after removal operation is significant reduced, especially for "image" datasets. For "biology" datasets, a lower θ may have more obvious decrease in the number of features. But for the fair comparison, we fixed θ to 0.7 for all datasets. The parameters of mRMR and MRSF algorithms are the default settings by the original author themselves.

The comparison results are visualized in Fig. 10. From the results, we can see that the our algorithm has better or equivalent performance on "image" datasets comparing to mRMR and MRSF. For "biology" datasets, the performance of our algorithm is between mRMR and MRSF. By exam the results from Tables 10, 11, 12, 13, 14, 15, 16 and 17, we find out that setting the θ value to 0.7 may not a good choice for "biology" datasets as the number of removed features is less comparing to "image" datasets. The best performance of our algorithm almost happens around $\theta = 0.4$. But to be fair with other algorithms, we only report the results of same θ value which is 0.7.

8 Related works

Remove redundant features is an important step for feature selection algorithms. Prestigious works include [30] which gives a formal definition of redundant features. Peng et al. [19] propose a greedy algorithm (named as mRMR) to select features with minimum redundancy and maximum dependency. Zhao et al. [32] develop an efficient spectral feature selection algorithm to minimize the redundancy within the selected feature subset through $\ell_{2,1}$ norm. Recently, researchers pay attention to unsupervised feature selection with global minimized redundancy [25,26]. Several graph-based approaches are proposed in [16,21].

9 Conclusion

In this study, we propose sparse feature graph to model both one-to-one feature redundancy and one-to-many features redundancy at the feature side of given feature matrix. By separating whole features into different redundancy feature group through local compressible subgraphs, we reduce the dimension of feature vector by only select one representative feature from each group. One advantage of our algorithm is that it does not need to calculate the pairwise distances among samples which may lead to the "chicken-and-egg" trouble. The experiment results show that our algorithm is an effective way to obtain accurate data structure information which is demanding for unsupervised spectral feature selection algorithms.

Acknowledgements This work is supported by NSF funding IIS-1715985.

References

1. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional space. In: International Conference on Database Theory, pp. 420–434. Springer, Berlin (2001)
2. Beyer, K., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: International Conference on Database Theory, pp. 217–235. Springer, Berlin (1999)
3. Cai, D., Zhang, C., He, X.: Unsupervised feature selection for multi-cluster data. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 333–342. ACM, New York (2010)
4. Dash, M., Liu, H.: Feature selection for classification. Intell. Data Anal. **1**(3), 131–156 (1997)
5. Du, L., Shen, Y.D.: Unsupervised feature selection with adaptive structure learning. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 209–218. ACM, New York (2015)
6. Dy, J.G., Brodley, C.E.: Feature selection for unsupervised learning. J. Mach. Learn. Res. **5**, 845–889 (2004)
7. Elhamifar, E., Vidal, R.: Sparse subspace clustering: algorithm, theory, and applications. IEEE Trans. Pattern Anal. Mach. Intell. **35**(11), 2765–2781 (2013)
8. Han, S., Qin, H.: A greedy algorithm to construct sparse graph by using ranked dictionary. Int. J. Data Sci. Anal. **2**(3), 131–143 (2016). https://doi.org/10.1007/s41060-016-0020-3
9. He, X., Ji, M., Zhang, C., Bao, H.: A variance minimization criterion to feature selection using Laplacian regularization. IEEE Trans. Pattern Anal. Mach. Intell. **33**(10), 2013–2025 (2011)
10. Hou, C., Nie, F., Li, X., Yi, D., Wu, Y.: Joint embedding learning and sparse regression: a framework for unsupervised feature selection. IEEE Trans. Cybern. **44**(6), 793–804 (2014)
11. Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R.P., Tang, J., Liu, H.: Feature selection: a data perspective (2016). http://featureselection.asu.edu/
12. Koller, D.: Toward optimal feature selection. In: Proceeding of the 13th International Conference on Machine Learning, pp. 284–292. Morgan Kaufmann, Los Altos (1996)
13. Lee, V.E., Ruan, N., Jin, R., Aggarwal, C.: A survey of algorithms for dense subgraph discovery. In: Managing and Mining Graph Data, pp. 303–336. Springer, Berlin (2010)
14. Li, Z., Yi, Y., Liu, J., Zhou, X., Lu, H.: Unsupervised feature selection using nonnegative spectral analysis. In: AAAI (2012)
15. Liu, X., Wang, L., Zhang, J., Yin, J., Liu, H.: Global and local structure preservation for feature selection. IEEE Trans. Neural Netw. Learn. Syst. **25**(6), 1083–1095 (2014)

16. Mairal, J., Yu, B.: Supervised feature selection in graphs with path coding penalties and network flows. J. Mach. Learn. Res. **14**(1), 2449–2485 (2013)
17. Moujahid, A., Dornaika, F.: Feature selection for spatially enhanced lbp: application to face recognition. Int. J. Data Sci. Anal. **5**(1), 11–18 (2018). https://doi.org/10.1007/s41060-017-0083-9
18. Ng, A.Y., Jordan, M.I., Weiss, Y., et al.: On spectral clustering: analysis and an algorithm. In: NIPS, vol. 14, pp. 849–856 (2001)
19. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans. Pattern Anal. Mach. Intell. **27**(8), 1226–1238 (2005)
20. Robnik-Šikonja, M., Kononenko, I.: Theoretical and empirical analysis of relieff and rrelieff. Mach. Learn. **53**(1–2), 23–69 (2003)
21. Song, Q., Ni, J., Wang, G.: A fast clustering-based feature subset selection algorithm for high-dimensional data. IEEE Trans. Knowl. Data Eng. **25**(1), 1–14 (2013)
22. Sturm, B.L., Christensen, M.G.: Comparison of orthogonal matching pursuit implementations. In: 2012 Proceedings of the 20th European on Signal Processing Conference (EUSIPCO), pp. 220–224. IEEE (2012)
23. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B (Methodol.) **58**, 267–288 (1996)
24. Tsourakakis, C., Bonchi, F., Gionis, A., Gullo, F., Tsiarli, M.: Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 104–112. ACM, New York (2013)
25. Wang, X., Mccallum, A., Wei, X.: Feature selection with integrated relevance and redundancy optimization. In: ICDM 2015. 15th IEEE International Conference on Data Mining, 2015, pp. 697–702. IEEE (2015)
26. Wang, D., Nie, F., Huang, H.: Feature selection via global redundancy minimization. IEEE Trans. Knowl. Data Eng. **27**(10), 2743–2755 (2015)
27. Weber, R., Schek, H.J., Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: VLDB, vol. 98, pp. 194–205 (1998)
28. Yang, Y., Shen, H.T., Ma, Z., Huang, Z., Zhou, X.: L2, 1-norm regularized discriminative feature selection for unsupervised learning. In: IJCAI Proceedings-International Joint Conference on Artificial Intelligence, vol. 22, p. 1589, Citeseer (2011)
29. You, C., Robinson, D.P., Vidal, R.: Scalable sparse subspace clustering by orthogonal matching pursuit. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3918–3927 (2016)
30. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy. J. Mach. Learn. Res. **5**, 1205–1224 (2004)
31. Zhai, H., Haraguchi, M., Okubo, Y., Tomita, E.: A fast and complete algorithm for enumerating pseudo-cliques in large graphs. Int. J. Data Sci. Anal. **2**(3), 145–158 (2016). https://doi.org/10.1007/s41060-016-0022-1
32. Zhao, Z., Wang, L., Liu, H.: Efficient spectral feature selection with minimum redundancy. In: AAAI (2010)
33. Zhao, Z., Wang, L., Liu, H., Ye, J.: On similarity preserving feature selection. IEEE Trans. Knowl. Data Eng. **25**(3), 619–632 (2013)