Applied Intelligence (2019) 49:4033–4057
https://doi.org/10.1007/s10489-019-01470-z

A new hybrid feature selection based on multi-filter weights and multi-feature weights

Youwei Wang[1] · Lizhou Feng[2]

Published online: 21 May 2019

Abstract
A traditional feature selection of filters evaluates the importance of a feature by using a particular metric, deducing unstable performances when the dataset changes. In this paper, a new hybrid feature selection (called MFHFS) based on multi-filter weights and multi-feature weights is proposed. Concretely speaking, MFHFS includes the following three stages: Firstly, all samples are normalized and discretized, and the noises and the outliers are removed based on 10-folder cross validation. Secondly, the vector of multi-filter weights and the matrix of multi-feature weights are calculated and used to combine different feature subsets obtained by the optimal filters. Finally, a Q-range based feature relevance calculation method is proposed to measure the relationship of different features and the greedy searching policy is used to filter the redundant features of the temp feature subset to obtain the final feature subset. Experiments are carried out using two typical classifiers of support vector machine and random forest on six datasets (APS, Madelon, CNAE9, Gisette, DrivFace and Amazon). When the measurements of F_1^{macro} and F_1^{micro} are used, the experimental results show that the proposed method has great improvement on classification accuracy compared to the traditional filters, and it achieves significant improvements on running speed while guaranteeing the classification accuracy compared to typical hybrid feature selections.

Keywords Feature selection · Feature relevance · Greedy searching · Support vector machine · Random forest

1 Introduction

In machine learning fields, a sample often contains a large number of features, many of which are highly correlated or even logically redundant, leading to the problems of high computational complexity and low interpretability [1]. As a consequence, many dimension reduction methods such as feature selection, feature reduction and feature extraction have been studied deeply in recent years. Feature selection is concerned with identifying a small subset of relevant features that are sufficient for learning the target concept, while the aim of feature reduction is to eliminate the logically redundant features from the original feature space without sacrificing the classification accuracy [2]. Different with feature reduction and feature selection, feature extraction transforms a high dimensional feature space into a distinct low dimensional feature space through a transformation of the original feature space [3]. Typical feature extraction methods include: AutoEncoder (AE) [4], latent semantic indexing (LSI) [5], partial least square (PLS) [6], multidimensional scaling [7], principal component analysis (PCA) [8] and latent Dirichlet allocation (LDA) [9]. However, compared to the feature selections and feature reductions, feature extractions cannot be easily interpreted since the physical meaning of the original features cannot be retrieved, limiting its application in dimension reduction.

The methods of feature selections can be divided in to two types: filters and wrappers [10]. Filters evaluate the importance of the features separately based on some predefined metrics instead of using the classifiers. Features are measured and ranked according to their importance using simple measurements such as distance, dependency and information. The most widely used filters include: Chi-square (CHI) [11], improved gini index (IMGI) [12], distinguishing feature selector

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10489-019-01470-z) contains supplementary material, which is available to authorized users.

✉ Youwei Wang
ywwang15@126.com

[1] School of Information, Central University of Finance and Economics, Beijing 100081, China

[2] School of Science and Engineering, Tianjin University of Finance and Economics, Tianjin 300222, China

(DFS) [13], odds ratio (OR) [14], document frequency (DF) [11], Darmstadt indexing approach (DIA) [15], information gain (IG) [16], mutual information (MI) [17], F-score [18], orthogonal centroid feature selection (OCFS) [19], feature selection considering the imbalance problem in text categorization (CMFSX) [20], supervised meaning rank (SMR) [21], unsupervised meaning average (UMA) [21], balanced accuracy measure (AAC2) [22], normalized difference measure (NDM) [23], clustering based feature selection (CBFS) [24], improved mutual information feature selection (NDMI) [25], novel feature selection based on normalized mutual information (NNMI) [26], multi-label feature selection based on max-dependency and min-redundancy (MDMR) [27]. Among these methods, NDMI, NNMI and MDMR are MI based filters which use the greedy searching policy to select the best features by calculating the mutual information of feature-feature and feature-class. A wrapper estimates the accuracy of a classifier by adding each unselected feature to the feature subset until the accuracy is less than the estimated accuracy of the feature set already selected [28]. Typical feature selections of wrappers include: forward search (FS), backward search (BS), sequential floating search (SFS) and simplified silhouette filter (SSF) [29]. Compared to the wrapper methods, though filters may provide less precise results, they are particularly efficient when dealing with large datasets.

In order to combine the advantages of filters or wrappers, many hybrid methods are proposed in recent years. Typical hybrid feature selections include: cluster based hybrid feature selection approach (CBHFS) [30], improved global feature selection scheme (IGFSS) [31], variable global feature selection scheme (VGFSS) [32], hybrid dimension reduction by integrating feature selection with feature extraction method (TDPFS) [3], novel feature selection based on harmony search (HFS) [33], hybrid approach of differential evolution and artificial bee colony for feature selection (HDAFS) [34], particle swarm optimization for feature selection (PSOFS) [35], hybrid feature selection based on enhanced genetic algorithm (EGAFS) [36], hybrid feature selection using component co-occurrence based feature relevance measurement (HFSCC) [37], multi-measure multi-weight ranking approach (MMMW) [38], multi-filter based feature selection (EMFFS) [39], two-step based feature selection (TSFS) [40], etc. These methods have good performances on classification accuracy, thus have been widely used in data classification fields.

In this paper, a new hybrid feature selection (called MFHFS) based on the information of multi-filter weights and multi-feature weights is proposed. In our experiments, we applied the proposed hybrid feature selection method on SVM and RF using six benchmark datasets. We show the effectiveness of our method by demonstrating that it significantly outperforms typical existing feature selection methods on the aspects of classification accuracy or running speed. The remainder of this paper is organized as follows. Section 2 reviews the related work on feature selections. Section 3 gives the details of the proposed method. Section 4 shows the experimental results. Section 5 concludes the whole paper.

2 Related work

2.1 The filters

2.1.1 The IMGI method

In order to apply the feature selection to data classification tasks with multiple class labels, Shang [14] improved the traditional Gini index method [41] and proposed the IMGI method. Given a feature t_i, its IMGI value is defined as follows:

$$\mathrm{IMGI}(t_i) = \sum_{c_k} p(t_i|c_k)^2 \times p(c_k|t_i)^2 \tag{1}$$

where $p(t_i|c_k)$ represents the conditional probability that feature t_i occurs in category c_k, $p(c_k|t_i)$ represents the conditional probability that a sample belongs to c_k when it contains t_i.

2.1.2 The CHI method

Given a category c_k and a feature t_i, t_i has strong category discriminative ability if it has a high CHI value. The CHI method calculates the score of t_i as follows [42]:

$$\begin{cases} \mathrm{CHI}(t_i) = \max\limits_{c_k} \{\mathrm{CHI}(t_i, c_k)\} \\ \mathrm{CHI}(t_i, c_k) = \dfrac{N(a_{ik}d_{ik} - b_{ik}e_{ik})^2}{(a_{ik}+b_{ik})(a_{ik}+e_{ik})(b_{ik}+d_{ik})(e_{ik}+d_{ik})} \end{cases} \tag{2}$$

where N represents the total number of samples, a_{ik} represents the frequency that t_i and c_k co-occur, b_{ik} represents the frequency that t_i does not occur in c_k, e_{ik} represents the frequency that class c_k occurs and does not contain feature t_i, and d_{ik} represents the frequency that neither c_k nor t_i occurs.

2.1.3 The DFS method

DFS is one of the successful feature selections and is also a global feature selection metric [13]. The idea of DFS is to select a set of distinctive features while eliminating the uninformative ones. The formula of DFS is defined as follows [31]:

$$\mathrm{DFS}(t_i) = \sum_{c_k} \frac{p(c_k|t_i)}{p\left(\overline{t_i}|c_k\right) + p\left(t_i|\overline{c_k}\right) + 1} \tag{3}$$

where $p(\overline{t_i}|c_k)$ is the conditional probability of absence of term t_i given class c_k, and $p(t_i|\overline{c_k})$ is the conditional probability of term t_i given all the classes except c_k.

2.1.4 The CMFSX method

Yang proposed a new feature selection method (called CMFSX) which can weaken the adverse effect caused by the imbalance factor in the dataset. CMFSX calculates the score of a feature t_i as follows [20]:

$$\begin{cases} \mathrm{CMFSX}(t_i) = \max\limits_{c_k} \{\mathrm{CMFSX}(t_i, c_k)\} \\ \mathrm{CMFSX}(t_i, c_k) = \dfrac{p(t_i|c_k) \times p(c_k|t_i)}{p(c_k)} \end{cases} \tag{4}$$

2.1.5 The SMR method

SMR uses a class of documents as the basic unit or context in order to calculate the meaning scores for words. Assume that a feature t_i appears k times in the dataset S, and m times in the documents of class c_k, SMR calculates the score of t_i as follows [21]:

$$\begin{cases} \mathrm{SMR}(t_i) = \max\limits_{c_k} \{\mathrm{SMR}(t_i, c_k)\} \\ \mathrm{SMR}(t_i, c_k) = -\dfrac{1}{m} \log \binom{k}{m} - [(m-1)\log N] \\ N = L/B \end{cases} \tag{5}$$

where L and B are the feature frequency of t_i in the dataset and class c_k, respectively.

2.1.6 The NDM method

NDM is a modified form of ACC2 measure [22] which solves the problem of class imbalance by normalizing true positive and false positive rates by the respective class size. Mathematically, NDM is defined as follows [23]:

$$\begin{cases} \mathrm{NDM}(t_i) = \left| \dfrac{tpr(t_i) - fpr(t_i)}{\min(tpr(t_i), fpr(t_i))} \right| \\ tpr(t_i) = \dfrac{tp_i}{tp_i + fn_i} \\ fpr(t_i) = \dfrac{tn_i}{tn_i + fp_i} \end{cases} \tag{6}$$

Where tp_i is the number of samples belonging to the positive classes and containing the term t_i, fn_i is the number of samples not belonging to the positive classes and not containing the feature t_i; fp_i is the number of samples not belonging to the positive classes and not containing the feature t_i.

2.1.7 The NDMI method

NDMI uses the mutual information of feature-feature and feature-class to determine an optimal set of features [25]. It uses a greedy searching policy to select the most discriminative features and filters the redundant ones. Given the feature t_i, its NDMI score is defined as follows:

$$\begin{cases} \mathrm{NDMI}(t_i) = \max\left\{ \sum\limits_{c_k} \mathrm{MI}(t_i, c_k) - \dfrac{1}{|S|} \sum\limits_{t_j \in S} \mathrm{MI}(t_i, t_j) \right\} \\ \mathrm{MI}(t_i, c_k) = p(c_k, t_i) \log_2 \dfrac{p(c_k, t_i)}{p(c_k) p(t_i)} \\ \mathrm{MI}(t_i, t_j) = p(t_i, t_j) \log_2 \dfrac{p(t_i, t_j)}{p(t_i) p(t_j)} \end{cases} \tag{7}$$

where S is the set of selected features, $|S|$ is the number of features in S, $p(t_i)$ denotes the occurring probability that a sample contains the feature t_i, $p(c_k)$ denotes the probability of the samples in category c_k, $p(t_i, c_k)$ denotes the probability that t_i occurs in c_k, $p(t_i, t_j)$ denotes the probability that t_i and t_j both occurs in a sample.

2.1.8 The MDMR method

Different from traditional single-label feature selection, MDMR considers not only the redundancies between individual features or the redundancies between class label and the candidate features, but also the conditional dependencies between features and each class label. The objective function is described as follows [27]:

$$\max\left\{ \sum_{c_j \in C} \mathrm{MI}(t_i, c_j) - \frac{1}{|S|} \sum_{t_l \in S} \left(\mathrm{MI}(t_i, t_l) - \sum_{c_j \in C} \mathrm{MI}(t_i, c_j | t_l) \right) \right\} \tag{8}$$

where $\mathrm{MI}(t_i, c_j | t_l)$ is the mutual information between the candidate feature t_i and all categories when given the selected feature t_l.

2.2 The hybrid methods

As different filters use different metrics to evaluate the feature importance, they always output different feature subsets for a particular dataset. In recent years, hybrid methods which combine different feature evaluating metrics altogether have received considerable attentions.

2.2.1 The EGAFS method

This approach combines the advantages of the filters with an enhanced genetic algorithm (EGA) in a wrapper approach to handle the high dimensionality of the feature space [36]. EGA improves the crossover and mutation operators of traditional genetic algorithms. The crossover operation is performed based on feature subset partitioning, while the mutation is performed based

on the classifier performance of the original parents and feature importance. Moreover, this method combines six well-known filters with the EGA to obtain the final feature subset. Though EGAFS has high classification accuracy, it uses wrappers and deduces the problem of high computational complexity.

2.2.2 The TDPFS method

Based on two feature selections and a feature extraction method, Bharti and Singh proposed the TDPFS method for text clustering [3]. Firstly, a typical term frequency based feature selection and a typical document frequency based feature selection are used to obtain two feature subsets, respectively. Then, the modified union operation is proposed to merge the features of these feature subsets. Finally, the PCA algorithm is applied to further transform the features in an interpretable feature space to reduce the dimension further without losing much information.

2.2.3 The IGFSS method

Although the features selected by traditional feature selections scheme can represents some of the classes successfully, some of the classes still may not be represented. Uysal proposed an improved global feature selection scheme (IGFSS) where a traditional feature selection is modified to obtain a more representative feature set [31]. In other words, IGFSS aims to improve the performance of traditional feature selections by selecting the features which represents all classes almost equally.

2.2.4 The VGFSS method

Though IGFSS solves the problem that some of the classes may not be represented by selecting an equal number of representative features from all the classes. However, this method is not suitable for an unbalanced dataset which has many classes, deducing the problem that some important features of the class that contains a higher number of features may be ignored. On this basis, Agnihotri and Verma proposed the VGFSS method to select a variable number of features from each class based on the distribution of features [32]. The number of selected features in each class is defined as follows:

$$NV(C_j) = count(C_j) \times \frac{N}{TFC} \tag{9}$$

Where $count(C_j)$ is the number of features of class C_j, N is the number of final selected features, TFC is the number of all features in the dataset.

2.2.5 The CBHFS method

Jaskowiak and Campello presented a hybrid feature selection (CBHFS) tailored for data classification problems [30]. This method consists of two main stages: in the first stage, it uses a clustering algorithm to identify the best features and remove the redundant ones [18]; in the second stage, a wrapper is used to evaluate different feature subsets produced by the clustering processes, determining the one that produces the best classification performance in terms of accuracy. This method has high classification accuracy but faces the problems of parameter dependency and high computational complexity.

2.2.6 The HFSCC method

Wang and Feng proposed a hybrid feature selection which can achieve the best features effectively and efficiently [37]. HFSCC consists of three steps: firstly, the samples are preprocessed and two feature subsets are obtained by using two different optimal filters; secondly, a feature weight based union operation is proposed to merge the obtained feature subsets; finally, the hierarchical agglomerative clustering algorithm is applied to obtain the final feature subset by combining a component co-occurrence based feature relevance measurement and a predetermined threshold. Experimental results show that this method achieves high classification accuracy and execution speed.

2.2.7 The MMMW method

As the features selected by different feature selections are always different, it is not enough to evaluate the importance of a feature by using only one particular feature selection. Bhattacharya and Selvakumar proposed the MMMW method which combines a filter, a wrapper and a clustering based feature selection to select the best features [38]. Though MMMW uses novel metrics to assign weights to the features, it ignores the weights of different methods on different datasets.

2.2.8 The EMFFS method

Osanaiye and Cai proposed a multi-filter based feature selection that combines the results of four filters to achieve the best features [39]. This method uses a simple majority voting policy to merge the features selected by different filters. Moreover, a threshold is predetermined to select the frequently occurring features among the four filters to construct the set of final features. However, this method ignores the fact that different filters have different performances on different datasets, and the optimal value of the predetermined threshold is hard to tune.

As is shown in Table 1, the main advantages and limitations of the above feature selections can be described as follows: (1) The traditional filters have high running speed, but they cannot filter the redundant features and the output results

Table 1 Advantages and limitations of different feature selections

types of feature selections	representative methods	advantages	limitations
traditional filters	CHI [11], IMGI [12], DFS [13], OR [14], DIA [15], IG [16], F-score [18], OCFS [19], CMFSX [20], SMR [21], NDM [23]	high running speed	• cannot filter the redundant features • the results rely on the feature importance measurements
MI based filters	NDMI [25], NNMI [26], MDMR [27]	filter the redundant features	low running speed
hybrid methods	TDPFS [3], CBHFS [30], IGFSS [31], VGFSS [32], HDAFS [34], EGAFS [36], HFSCC [37], MMMW [38], EMFFS [39]	combine the advantages of different methods	• low running speed • simple majority voting or set union mechanism

rely on the feature importance measurements and datasets seriously; (2) The MI based filters and wrappers can filter the redundant features, but they generally have high computation complexities when dealing with the data sets with high numbers of features or samples [40]; (3) The hybrid methods can obtain higher classification accuracy than those of the other methods, but some are very time consuming as they use the wrappers to obtain the best feature subset. Moreover, some existing hybrid methods combine the results of different filters by using a simple majority voting or set union mechanism [37], ignoring the effects of feature weight and filter weight.

For solving these problems, we propose a new hybrid feature selection (called MFHFS) based on the information of multi-filter weights and multi-feature weights. The main contributions of this paper are given as follows: (1) A new hybrid feature selection frame which combines the advantage of traditional filters on running speed and the advantage of greedy searching policy on filtering the redundant features is proposed. (2) In order to avoid the problem that the performance of a filter is unstable when the dataset changes, the vector of multi-filter weights is introduced. Moreover, in order to distinguish the importance of different features from different datasets, the matrix of multi-feature weights is proposed to obtain the most discriminative features. (3) A Q-range based feature relevance calculation method is proposed to improve the running speed of achieving the feature relevance matrix when filtering the redundant features.

3 The proposed method - MFHFS

3.1 Description of MFHFS

Though the traditional filters have high executing speed, they fail to filter the redundant features and their performances rely on the feature importance measurements and the datasets seriously. Moreover, though the MI based filters or the hybrid methods can filter the redundant features, they always face the problem of high computational complexity, decreasing the executing speed when the number of features or the number of samples is high [37]. On this basis, we combine the advantages of the methods with different types, and propose a new multi-filter weights and multi-feature weights based feature selection (called MFHFS) of which the flowchart is shown in Fig. 1. MFHFS can be executed in the following three steps: (1) Data preprocessing: the samples are normalized and discretized by using the equal width interval binning (EWIB) algorithm, and the noises and

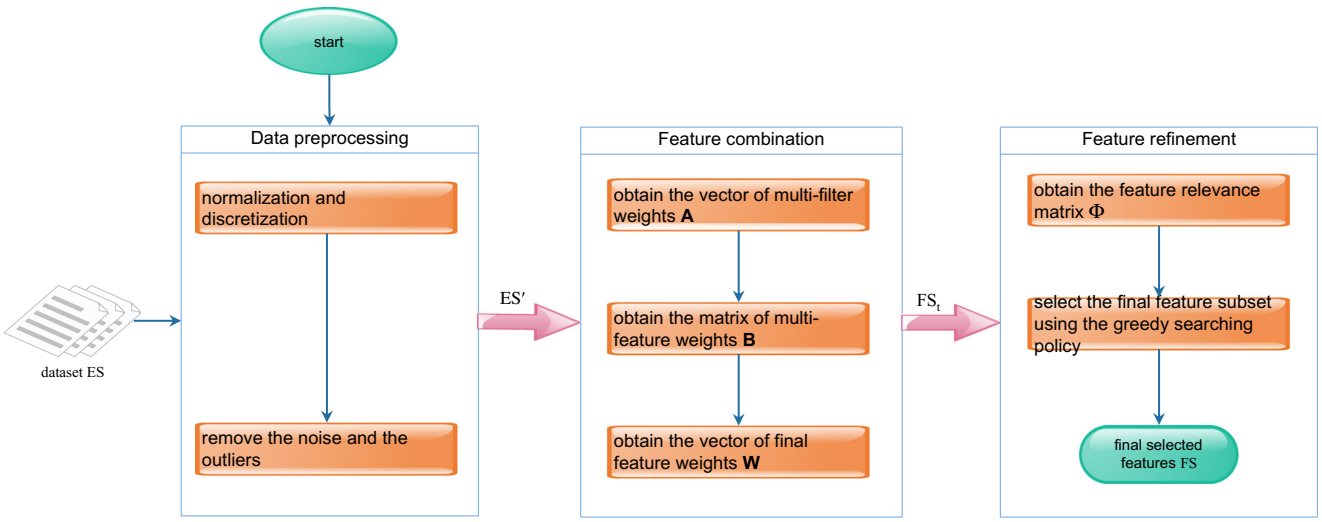


Fig. 1 Flowchart of MFHFS

outliers are filtered by combining the 10-folder cross validation; (2) Feature combination: several feature subsets are obtained by a set of optimal filters, and they are used to obtain a temp feature subset FS_t by considering the multi-filter weights and multi-feature weights; (3) Feature refinement: a Q-range based feature relevance calculation method is proposed to obtain the feature relevance matrix of FS_t, and the final features are selected from FS_t by using the greedy searching policy. The details of MFHFS are described as follows:

- Stage 1: Data preprocessing

Algorithm 1 Data preprocessing.

Input: the entire sample set: ES; minimum and maximum values in feature normalization: $f_{\min}$, $f_{\max}$; number of quantization levels in feature discretization: N_Q; number of selected features: N; cla: a classifier.

Output: the preprocessed sample set: ES'.

1. normalization and discretization:

1.1 **for** each feature f_i in ES

1.2 **for** each sample s_j in ES

1.3 normalize s_j's feature value f_{ij} in to f_{ij}' using the following formula:

$$f_{ij}' = f_{\min} + \frac{f_{ij} - f_{i,\min}}{f_{i,\max} - f_{i,\min}} \times (f_{\max} - f_{\min}) \tag{10}$$

where $f_{i,\min}$ and $f_{i,\max}$ are the minimum and maximum feature values of f_i, respectively.

1.4 apply the EWIB [43] algorithm to discretize f_{ij}' into a particular level by using N_Q.

1.5 **end for**

1.6 **end for**

2. removal of noise and outliers using 10-folder cross validation:

2.1 set a temporary array $C=\{c_0, c_1, \ldots, c_i \ldots, c_{|ES|-1}\}=\{0\}$ ($|ES|$ is the number of samples in ES).

2.2 **for** k=1:1:10

2.3 select $|ES|/10$ samples from ES randomly, and denote these samples as es_k.

2.4 $sp \leftarrow es_k$

2.5 $st \leftarrow ES - es_k$

2.6 **for each** filter of_i in OFS ($1 \le i \le L$)

2.7 apply of_i to st to obtain a feature subset fs_i which consists of N features.

2.8 train st using fs_i to obtain a classifier cla.

2.9 **for each** sample s_j in sp ($0 \le j < |sp|$)

2.10 predict the label of s_j using cla and denote it as pl_j.

2.11 set $c_{idx(s_j)} \leftarrow c_{idx(s_j)} + \mathrm{sgn}(pl_j == la_j)$

2.12 where la_j is the real class label of sample s_j, $idx(s_j)$ denotes the index number of s_j in the sample set ES, $\mathrm{sgn}(x)$ is a function which returns 1 when $x>0$ and returns -1 when $x \le 0$.

2.13 **end for**

2.14 **end for**

2.15 **end for**

2.16 **for** each sample s_j in ES

2.17 **if** $c_j \le 0$ remove s_j from ES.

2.18 **end if**

2.19 **end for**

In the pre-processing step, three operations: normalization, discretization and removal of noises and outliers are executed. Normalization aims to adjust values measured on different scales to a notionally common scale. It can reduce the computational cost and solve the problem that the classification accuracy may be unstable when the features have large ranges of values. Discretization is the process of transferring continuous data into discrete counterparts. It is used to reduce the computational complexity and improve the stability of feature selection in the preprocess of the proposed method. Moreover, as the noises and outliers may affect the distribution characteristic of the entire dataset, we will remove them to improve the robustness of the proposed method. The details of data preprocessing are given in Algorithm 1. In the normalization step (step 1.3), $f_{\min}$ and $f_{\max}$ are set to $f_{\min}=0$ and $f_{\max}=6$; in the discretization step (step 1.4), N_Q is set to $N_Q=f_{\max}+1$, and the interval width is set to $\delta=\frac{f_{\max}-f_{\min}}{N_Q}$ [37].

- Stage 2: Feature combination

Different traditional filters use different metrics to evaluate the feature importance, thus the performance of a filter may be unstable when the dataset changes. Osanaiye et al. proposed an ensemble-based multi-filter feature selection that combines the output of four filter methods to achieve an optimum selection [40]. However, this method uses a simple majority vote to determine the final selected features by combining a predetermined threshold, thus deduces the following problems: (1) It treats different filters equally thus ignores the filter weights, decreasing the classification accuracy when some of the filters have bad performances on particular datasets. (2) The features selected by a filter are treated equally, ignoring the fact that the category discriminative abilities of these features are always different. (3) The predetermined threshold is related to the number of selected features and is hard to tune. On this basis, we propose a novel feature combination method by introducing the vector of multi-filter weights and the matrix of multi-feature weights, and the executing flowchart of this stage is given in Fig. 2. Given the preprocessed entire sample set ES', we first obtain the vector of multi-filter weights and denote it as $\mathbf{A}=(\alpha_1, \alpha_2, \ldots, \alpha_L)$. As is shown in Algorithm 2, the details are given as follows:

Algorithm 2 obtain the vector of multi-filter weights.

Input: preprocessed dataset: ES'; optimal filters: OFS; number of selected features: N.

Output: vector of multi-filter weights: $\mathbf{A}=(\alpha_1, \alpha_2, \ldots, \alpha_L)$.

1. **for each** filter of_i in OFS $(1\le i\le L)$
2. apply of_i on ES' to obtain the top N features and denote them as fs_i.
3. obtain the accuracy (acc_i) of of_i on ES' by using classifier cla and fs_i.
4. **end for**
5. normalize the accuracies of the filters in OFS to obtain the vector of multi-filter weights $\mathbf{A}=(\alpha_1, \alpha_2, \ldots, \alpha_L)$:

$$\alpha_i = acc_i \Big/ \max_{of_j \in OFS}\left(acc_j\right) \tag{11}$$

Further, we obtain the matrix of multi-feature weights and denote it as $\mathbf{B}=\{\mathbf{B}_i\}=\{(\beta_{i1}, \beta_{i2}, \ldots, \beta_{ij}, \ldots, \beta_{iM})\}$. As is shown in Algorithm 3, the details are given as follows:

Further, we obtain the matrix of multi-feature weights and denote it as $\mathbf{B} = \{\mathbf{B}_i\} = \{(\beta_{i1}, \beta_{i2}, \ldots, \beta_{ij}, \ldots, \beta_{iM})\}$. As is shown in Algorithm 3, the details are given as follows:

On this basis, the $1 \times M$ size vector of final feature weights is obtained by calculating the dot product of the $1 \times L$ size vector of multi-filter weights and the $L \times M$ size matrix of

Algorithm 3 obtain the matrix of multi-feature weights.

Input: preprocessed dataset: ES'; optimal filters: OFS; number of selected features: N.

Output: matrix of multi-feature weights: $\mathbf{B}=\{\mathbf{B}_i\}=\{(\beta_{i1}, \beta_{i2}, \ldots, \beta_{ij}, \ldots, \beta_{iM})\}$.

1. set a $L \times M$ size matrix $\mathbf{B}=\{\beta_{ij}\}=\{0\}(1 \le i \le L, 1 \le j \le M)$.

2. **for each** filter of_i in OFS $(1 \le i \le L)$

3. **for each** feature f_j in ES'

apply of_i on f_j to obtain the output score $v(i, j)$.

4. **end for**

5. rank the all features in descending order according to their scores and select the top N features to form a feature subset S_i.

6. obtain the order indexes of the features in S_i and denote them as $OD=\{od_1, od_2, \ldots, od_N\}$:

6.1 $od_1=1$.

6.2 **for** $j=1:1:N-1$

6.3 **if** $v(i, j+1) > v(i, j)$

$od_{j+1}=j+1$.

6.4 **else**

6.5 $od_{j+1}=od_j$.

6.6 **end if**

6.7 **end for**

7. obtain the matrix of multi-feature weights $\mathbf{B}=\{\beta_{ij}\}$:

7.1 **for each** feature f_k in S_i $(1 \le k \le N)$

7.2 obtain the weight of f_k using formula (12) and denote it as τ_{ik}:

$$\tau_{ik} = \sqrt{1/od_k} \tag{12}$$

7.3 normalize τ_{ik} to interval [0, 1] and obtain the normalized feature weight λ_{ik}:

$$\lambda_{ik} = \frac{\tau_{ik} - \min\limits_{f_l \in S_i}(\tau_{il})}{\max\limits_{f_l \in S_i}(\tau_{il}) - \min\limits_{f_l \in S_i}(\tau_{il})} \tag{13}$$

7.4 obtain the index of f_k in ES' and denote it as j.

7.5 set $\beta_{ij}=\lambda_{ik}$.

7.6 **end for**

8. **end for**

multi-feature weights, and the temp feature subset FS_t is obtained by the steps described in Algorithm 4:

Algorithm 4 Obtain the temp feature subset FS_t.

Input: preprocessed dataset: ES'; optimal filters: OFS; number of selected features: N.

Output: temp feature subset: FS_t.

1. set a $1\times M$ size vector of final feature weights $\mathbf{W}=(w_1, w_2,\ldots, w_i,\ldots, w_M)=\{0\}(1\le i\le M)$.
2. obtain the vector of multi-filter weights and denote it as **A** using Algorithm 2.
3. obtain the matrix of multi-feature weights and denote it as **B** using Algorithm 3.
4. obtain **W** using the formula (14):

$$\mathbf{W}=\mathbf{A}\bullet\mathbf{B} \tag{14}$$

5. rank all features by their final feature weights in descending order.
6. select the top N features to form the temp feature subset FS_t:

set $FS_t \leftarrow \{f_i | \mathbf{W}(i) \ge TOP(\mathbf{W}, N)\}$, where $TOP(\mathbf{W}, N)$ is the N_{th} largest weight value in **W**.

From Algorithms 2–4 we know that we consider the effects of the multi-filter weights and the multi-feature weights by introducing the vector **W**, thus can avoid the problem that the optimal filters or the selected features are treated equally when searching the best features. For ease of computation, we set the number of optimal filters $L=4$ in this paper. Thus, the time complexity of obtaining the final feature subset FS_t is $O(4\times(M|C|+M\log_2 M+3\ N)+N\log_2 N)$.

- Stage 3: Feature refinement

In order to filter the redundant features in FS_t, TDPFS [3] uses the PCA algorithm to transform the high dimensional feature space into an un-interpreted low dimensional feature space. EGAFS [36] uses the enhanced GA based wrapper method to select the optimal features, thus has the problem of high computation cost. On this basis, we utilize the greedy searching policy [25–27] to obtain the optimal final feature subset from FS_t of which the feature number is much lower than that of entire dataset. The details of this stage are given as follows:

(1) obtain the feature relevance matrix

HFSCC [37] obtains the feature relevance matrix by calculating the feature relevance of each pair of the features in FS_t. This method ignores the fact that a feature is only redundant with the ones which have equal or approximate global feature weight values, thus deduces a high computation complexity of $O(N^2)$. In this paper, we suppose that the proportion of redundant features in FS_t is no higher than r ($0<r<1$), and propose a new feature relevance calculation method which only calculate the relevance between a feature f_i and the features which have similar normalized global feature weight values with f_i. As is shown in Fig. 3, given the temp feature subset FS_t obtained in stage 2, we first normalize the final feature weight values of FS_t, then calculate the feature relevance matrix $\Phi=\{\phi_{ij}\}$ ($1\le i,j\le N$) using formula (15):

$$\begin{cases}\phi_{ij}=\phi_{ji}=\begin{cases}c_{ij}, \text{if } |\mathbf{W}_N(f_i)-\mathbf{W}_N(f_j)|<Q\\ 0, \quad \text{else}\end{cases}\\ c_{ij}=\sum\limits_{f_{jl}\in\Omega}\sum\limits_{f_{ik}\in\Omega}\left(p(f_{ik}|f_{jl})\times p(f_{jl}|f_{ik})\times p(f_{jl},f_{ik})\right)\end{cases} \tag{15}$$

where f_i and f_j are the i_{th} and j_{th} features in the feature subset FS_t; Q is a factor which ranges in the interval (0, 1); $\mathbf{W}_N(f_i)$ and $\mathbf{W}_N(f_j)$ are the normalized final feature weight values of f_i and f_j, respectively; c_{ij} denotes the feature relevance of f_i and f_j using the CCFR algorithm of HFSCC method; Ω denotes the

corresponding component values (possible values) for each feature; f_{ik} denotes the k_{th} component value of f_i and f_{jl} denotes the l_{th} component value of f_j; $p(f_{ik}|f_{jl})$ denotes the conditional probability that f_{ik} occurs when f_{jl} occurs; $p(f_{jl}|f_{ik})$ denotes the conditional probability that f_{jl} occurs when f_{ik} occurs; $p(f_{ik},f_{jl})$ is the feature component based normalization coefficient which denotes the probability that f_{ik} and f_{jl} occur together in the dataset.

(2) select the final feature subset using the greedy searching policy

Though HFSCC [37] and TSFS [40] can filter the redundant features of FS_t, they both require a predetermined threshold *th* which has great influence on the number of selected features. Therefore, we apply the greedy searching policy of traditional MI filters on the sorted feature subset FS_t to obtain the final feature subset FS. A greedy algorithm is proposed for solving an optimization problem of finding the solution that maximises the measured fitness [44]. When dealing with an optimization problem, greedy searching policy selects the local optimal solution in each step, and hopes to generate a global optimal solution through a series of local optimal selections. As this policy measures the importance of the features by maximining the relevance between the features and the classes and minimizing the redundancies among the selected features [26, 27], in this paper, the objective function of selecting a feature f from FS_t is defined as follows:

$$\begin{cases} f = \arg\max\limits_{f_i \in FS_t} Score(f_i) \\ Score(f_i) = \theta \times \mathbf{W}_N(f_i) - \dfrac{\varphi}{|S|} \sum\limits_{f_j \in S} \phi_{ij} \end{cases} \tag{16}$$

Where S is the set of selected features; θ ($\theta > 0$) and φ ($\varphi > 0$) are two parameters balancing the feature category discriminative ability and the feature redundancy. In order to emphasize the importance of the feature category discriminative ability and weaken the data loss that may be brought by setting some elements in Φ to zero directly, the parameters of θ and φ are set to $\theta = 1$ and $\varphi = 0.5$ in this paper. On this basis, the details of stage 3 are shown in Algorithm 5:

From Algorithm 5 we know that the time complexity of steps 2–3 is $O(N + pN^2)$ ($0 < p < 1$, p is the probability that the expression $|\mathbf{W}_N(f_i) - \mathbf{W}_N(f_j)| < Q$ is true). As there are few redundant features in FS_t, Q is much lower than 1 in practical situations, which means that the running speed of the proposed Q-range based method is $1/p$ times of that of the HFSCC method. Moreover, we notice that the time complexity of step 4 is $O\left(\sum_{i=0}^{N_s-1}(N_1-i)\times i\right) = \frac{(N_s-1)N_sN_1}{2} - \frac{(N_s-1)N_s(2N_s-1)}{6}$, thus we conclude that the overall time complexity of Algorithm 5 is $O\left(N + pN^2 + \frac{(N_s-1)N_sN}{2} - \frac{(N_s-1)N_s(2N_s-1)}{6}\right)$.

For ease of understanding, we give an example to show the executing processes of MFHFS. As is shown in Table 2, the preprocessed samples are denoted as $ES' = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$ and the set of class labels are denoted as $La = \{1,2,3\}$. Moreover, we suppose that the set of optimal filters used in Algorithms 2–4 is $OFS = \{\text{IG, IMGI, CHI, DFS}\}$, the corresponding vector of multi-filter weights is set to $\mathbf{A} = (1, 1, 1, 1)$, the feature number of FS_t is set to $N = 8$, the factor Q is set to $Q = 0.1$, and the feature number of FS is set to $N_s = 6$. On this basis, Table 3 shows the feature scores when different methods are used and the corresponding feature orders are given in the brackets. Moreover, Tables 4 and 5 give the feature relevance matrices of the features in FS_t when HFSCC and the proposed Q-range based method are used, respectively. Obviously, we know from Table 4 that the features f_1 and f_2 are redundant with each other as their corresponding feature relevance equals to 1. Further, it can be seen from Table 5 that the feature relevance between f_1 and f_2 can also be achieved by the proposed Q-range based method accurately and the computational cost of our method is 25% (16 of 64 cases) of that of HFSCC method. Moreover, Table 6 gives the *Score* values of the features in FS_t and the changes of the final feature subset FS when the number of selected feature increases. Obviously, we know from Table 6 that the redundant feature f_2 is filtered and the final selected feature subset is $FS = \{f_4, f_5, f_1, f_8, f_9, f_7\}$.

3.2 Time complexity analysis of MFHFS

The time complexity of the proposed feature selection consists of three stages:

(1) The stage of data processing: we know from Algorithm 1 that the complexity of this stage is $T_1 = O(MD) + L\times(O(flt) + O(cla_t) + |ES| \times O(cla_p))$, where $O(flt)$, $O(cla_t)$ and $O(cla_p)$ are the time complexities of the filter *flt*, the training and the predicting processes of classifier *cla*. However, as this stage can be applied to all feature selections, we do not consider this part when calculating the time complexity of MFHFS.

(2) The stage of feature combination: we know from Algorithms 2–4 that the complexity of this stage is $T_2 = O(4 \times (M|\,C| + M\log_2 M + 3N) + N\log_2 N)$, where M is the number of all features and N is the number of selected features in Algorithm 2.

(3) The stage of feature refinement: we know from Algorithm 5 that the time complexity of this stage is $T_3 = O\left(N + pN^2 + \frac{(N_s-1)N_sN}{2} - \frac{(N_s-1)N_s(2N_s-1)}{6}\right)$, where N_s is the number of features in FS. Generally, there exists $N_s \approx N >> 1$, thus we have:

Algorithm 5 Feature refinement.

Input: temp feature subset obtained in Algorithm 4: FS_t; number of selected features in Algorithms 2-4: N; number of final selected features: N_s; predetermined factor: Q.

Output: final feature subset: FS.

1. initialization: set the final feature subset FS to $FS \leftarrow null$, set a $N \times N$ size matrix $\Phi = \{\phi_{ij}\} \leftarrow \{0\}$.

2. normalize the final feature weight values of FS_t:

   2.1 **for each** feature f_i in FS_t

   normalize f_i's final feature weight value $\mathbf{W}(f_i)$ into $\mathbf{W}_N(f_i)$ using formula (17):

$$\mathbf{W}_N(f_i) = \frac{\mathbf{W}(f_i) - \min_{f_j \in FS_t}\left(\mathbf{W}(f_j)\right)}{\max_{f_j \in FS_t}\left(\mathbf{W}(f_j)\right) - \min_{f_j \in FS_t}\left(\mathbf{W}(f_j)\right)} \tag{17}$$

   2.2 **end for**

3. obtain the feature relevance matrix $\Phi = \{\phi_{ij}\}$ $(1 \le i, j \le N)$:

   3.1 **for** i=1 **to** N **step** 1

   3.2 **for** j=i+1 **to** N **step** 1

   obtain the feature relevance ϕ_{ij} between f_i and f_j by using formula (15).

   3.3 **end for**

   3.4 **end for**

4. select the final feature subset using greedy searching policy:

   4.1 select the feature f such that:

$$f = \operatorname{agr\,max}_{f_i \in FS_t}\{\mathbf{W}_N(f_i)\} \tag{18}$$

   4.2 add the feature f into FS.

   4.3 **while** $(|FS| < N_s)$

   select a feature f from FS_t using formula (16).

   set $FS \leftarrow FS \cup \{f\}$

   4.4 **end while**

$$\begin{aligned} T_3 &\approx \mathrm{O}\left(N + pN^2 + \frac{(N_s-1)N_sN}{2} - \frac{(N_s-1)N_s(2N_s-1)}{6}\right) \\ &\approx \mathrm{O}\left(N_s + pN_s^2 + \frac{N_s^3}{2} - \frac{N_s^3}{3}\right) = \mathrm{O}\left(N_s + pN_s^2 + \frac{N_s^3}{6}\right) \end{aligned} \tag{19}$$

On this basis, we combine the results of T_2 and T_3 and remove the constant coefficients, then obtain the overall time complexity T_{MFHFS} (constant coefficients are removed) as follows:

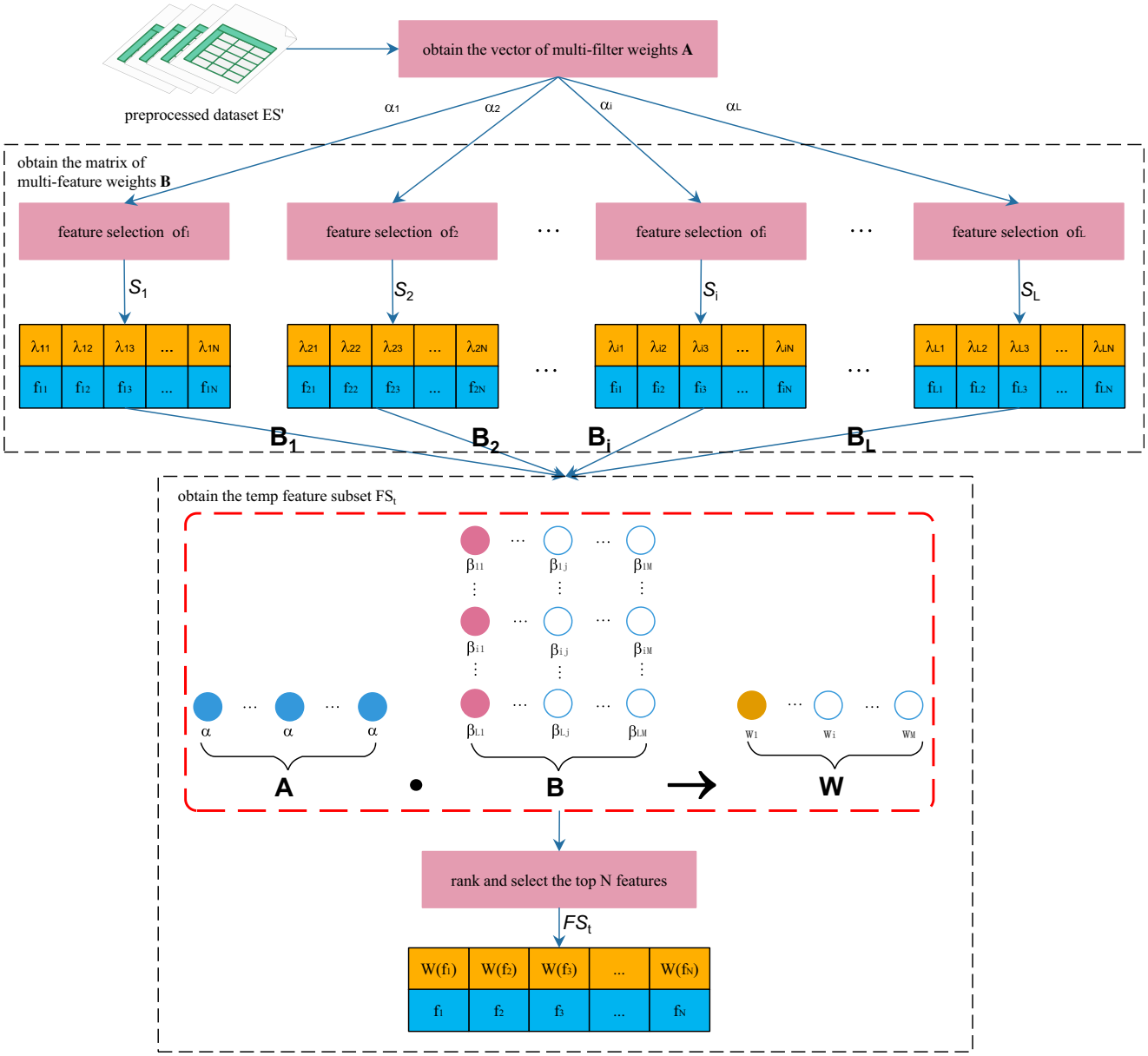
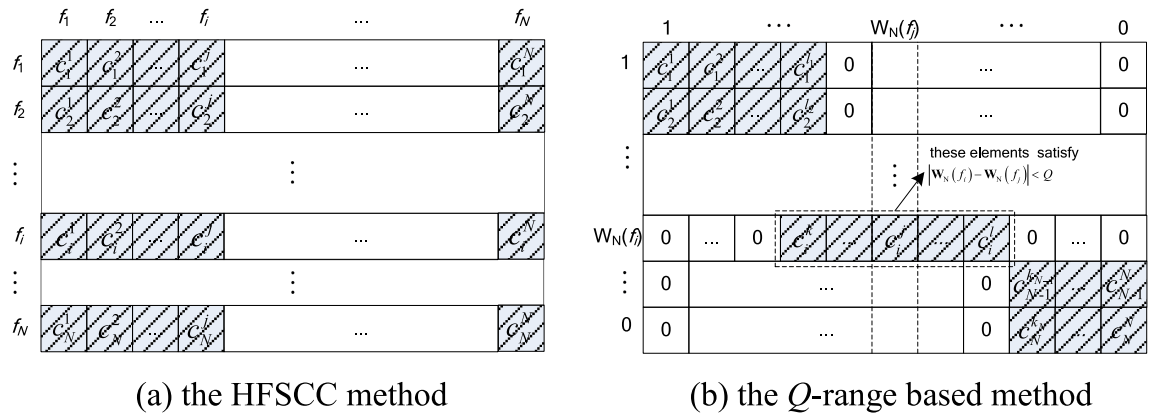


Fig. 2 Flowchart of feature combination stage



(a) the HFSCC method (b) the *Q*-range based method

Fig. 3 Graphical representation of different feature relevance calculation methods

Table 2 Example of the preprocessed sample set ES'

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8
f_1	2	1	2	1	1	4	2	2
f_2	2	1	2	1	1	4	2	2
f_3	3	3	2	1	2	1	1	1
f_4	2	2	1	2	1	3	1	4
f_5	2	1	5	1	1	2	2	3
f_6	1	5	1	1	1	1	4	5
f_7	4	2	4	5	3	2	2	5
f_8	3	5	5	5	3	5	1	2
f_9	2	4	1	2	3	2	4	2
f_{10}	1	3	4	1	2	1	3	1
class	1	1	2	2	3	3	2	1

$$T_{\mathrm{MFHFS}} \approx \mathrm{O}\left(M|C| + M\log_2 M + {N_s}^3\right) \tag{20}$$

When considering the traditional MI based filters (like NDMI and NNMI), according to [35] we know that their computational complexities are all:

$$\begin{aligned} T_{\mathrm{MI}} &= \mathrm{O}\left(\frac{(N_s-1)N_s M}{2} - \frac{(N_s-1)N_s(2N_s-1)}{6}\right) \\ &\approx \mathrm{O}\left({N_s}^2\left(\frac{M}{2}-\frac{1}{3}\right)\right) \end{aligned} \tag{21}$$

When the value of M is large enough, there generally exists $M/2 >> 1/3$, thus we remove the constant coefficients of formula (21) and transform T_{MI} to the following formula:

$$T_{\mathrm{MI}} = O\left({N_s}^2 M\right) \tag{22}$$

Then, we have:

$$\begin{aligned} \frac{T_{\mathrm{MFHFS}}}{T_{\mathrm{MI}}} &= \frac{M|C| + M\log_2 M + {N_s}^3}{{N_s}^2 M} \\ &= \frac{|C|}{{N_s}^2} + \frac{\log_2 M}{{N_s}^2} + \frac{N_s}{M} \end{aligned} \tag{23}$$

Table 3 Scores of the features when different methods are used

	IG	IMGI	CHI	DFS	MFHFS
f_1	**0.074(3)**	**0.361(6)**	**7.289(6)**	**2.408(1)**	**2.364(3)**
f_2	**0.074(3)**	**0.361(6)**	**7.289(6)**	**2.408(1)**	**2.364(3)**
f_3	0.037(10)	**0.336(8)**	**7.771(8)**	**0.903(6)**	**1.170(8)**
f_4	**0.074(3)**	**0.528(1)**	**11.987(1)**	**1.204(3)**	**3.155(1)**
f_5	**0.056(8)**	**0.528(1)**	**11.987(1)**	**1.204(3)**	**2.930(2)**
f_6	0.042(9)	0.210(9)	3.378(9)	**0.778(7)**	0.378(9)
f_7	**0.074(3)**	**0.446(4)**	**9.676(3)**	**0.347(8)**	**2.008(7)**
f_8	**0.197(1)**	**0.458(3)**	**9.321(4)**	−1.732(10)	**2.077(5)**
f_9	**0.100(2)**	**0.365(5)**	**8.858(5)**	**1.051(5)**	**2.049(6)**
f_{10}	**0.063(7)**	0.164(10)	3.220(10)	−0.250(9)	0.378(9)

(the scores of the selected features are denoted in bold)

Table 4 Feature relevance matrix of FS_t using HFSCC method

	f_1	f_2	f_3	f_4	f_5	f_7	f_8	f_9
f_1	1.000	1.000	0.073	0.194	0.625	0.219	0.063	0.146
f_2	1.000	1.000	0.073	0.194	0.625	0.219	0.063	0.146
f_3	0.073	0.073	1.000	0.135	0.198	0.000	0.073	0.406
f_4	0.194	0.194	0.135	1.000	0.076	0.031	0.042	0.333
f_5	0.625	0.625	0.198	0.076	1.000	0.125	0.025	0.208
f_7	0.219	0.219	0.000	0.031	0.125	1.000	0.313	0.177
f_8	0.063	0.063	0.073	0.042	0.025	0.313	1.000	0.192
f_9	0.146	0.146	0.406	0.333	0.208	0.177	0.192	1.000

As there exists $|C| << {N_s}^2$, $\log_2 M << {N_s}^2$ and $N_s << M$, then we have $\frac{T_{\mathrm{MFHFS}}}{T_{\mathrm{MI}}} << 1$, which means that the computational complexity of MFHFS is obviously lower than that of the traditional MI based filters.

4 Experimental results and analysis

We developed our hybrid method on a computer having Windows 10 Home operating system, 8 GB of RAM, and Intel Core i7–7500 processor. In this section, to verify the classification performance of the selected feature subsets, a series of experiments are conducted to compare the proposed method with typical feature selections. All experiments are implemented with matlab 2015a which is popular on machine learning and data mining. Moreover, the 10-folder cross validation is used to test the performances of different methods.

4.1 Datasets

In this section, we select six datasets which contain more than 100 features from UCI machine learning repository [45, 46]. Table 7 represents the characteristics of these datasets including the numbers of features, the numbers of instances, and the numbers of classes. These datasets included APS, Madelon, CNAE9, Gisette, DrivFace and Amazon. As Amazon contains

Table 5 Feature relevance matrix of FS_t using the proposed Q-range based method

	f_4	f_5	f_1	f_2	f_8	f_9	f_7	f_3
f_4	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
f_5	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
f_1	0.000	0.000	1.000	1.000	0.000	0.000	0.000	0.000
f_2	0.000	0.000	1.000	1.000	0.000	0.000	0.000	0.000
f_8	0.000	0.000	0.000	0.000	1.000	0.192	0.200	0.000
f_9	0.000	0.000	0.000	0.000	0.192	1.000	0.177	0.000
f_7	0.000	0.000	0.000	0.000	0.200	0.177	1.000	0.000
f_3	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000

Table 6 Changes of the final feature subset *FS* and *Score* values of the features in FS_t

	$\|FS\|=1$	$\|FS\|=2$	$\|FS\|=3$	$\|FS\|=4$	$\|FS\|=5$	$\|FS\|=6$
f_4	*	*	*	*	*	*
f_5	0.887	*	*	*	*	*
f_1	0.601	**0.601**	*	*	*	*
f_2	0.601	0.601	0.101	0.101	0.101	–
f_8	0.457	0.457	**0.457**	*	*	*
f_9	0.443	0.443	0.443	**0.347**	*	*
f_7	0.422	0.422	0.422	0.322	**0.322**	*
f_3	0.000	0.000	0.000	0.000	0.000	–

(the symbol '*' denotes that the corresponding feature is selected; the symbol '-' denotes that the algorithm terminates and the corresponding *Score* value is not calculated; the float values denote the corresponding *Score* values with the highest one denoted in bold)

too many classes, for ease of computation, we only consider the top six classes in which each class consists of 30 samples.

4.2 Classifiers

In order to investigate the performance of the proposed algorithm, two well-known classifiers: support vector machine (SVM) [47] and random forest (RF) [48] are used. SVM is a discriminative classifier which is formally defined by a separating hyper-plane. In SVM, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data point of any class. RF is a classification method which combines multiple tree predictors in a way that each tree depends on a value of randomly chosen vector distributed among all trees in forest in the same way [49]. The parameters of these classifiers are given as follows: the number of trees in RF classifier is set to $n_t = 5$; the libSVM library [50] is used for SVM classifier with the parameters $c = 15{,}000$ and $gamma = 0.07$; the classifier used in Algorithms 1–2 is SVM.

4.3 Evaluation measurements

The macro-averaged F_1 measurement and the micro-averaged F_1 measurement are used to evaluate the performances of different feature selections. Given class c, the F_1 measurement

Table 7 Details of the datasets used in this paper

datasets	number of features	number of instances	number of classes
APS	170	60,000	2
Madelon	500	4000	2
CNAE9	857	1080	9
Gisette	5000	13,500	2
DrivFace	6400	606	3
Amazon	10,000	1500	50

Table 8 F_a values of different filters when SVM is used on different datasets

datasets	IG	IMGI	CHI	DFS	CMFSX	DIA	CDM	OR
APS	0.978	0.984	0.985	0.986	**0.987**	0.981	0.977	0.980
Madelon	0.482	0.501	**0.667**	0.655	0.558	0.486	0.609	0.498
CNAE9	0.406	0.831	0.834	0.788	**0.926**	0.669	0.488	0.437
Gisette	0.901	0.947	0.949	**0.958**	0.947	0.950	0.956	0.803
DrivFace	0.952	0.962	**0.966**	0.965	0.878	0.954	0.960	0.960
Amazon	0.699	0.833	**0.879**	0.777	0.774	0.754	0.654	0.701

(F_1^c) which is defined in formula (24) considers both the precision p_c and the recall r_c [21]:

$$\begin{cases} F_1^c = \dfrac{2 \times p_c \times r_c}{p_c + r_c} \\ p_c = \dfrac{tp_c}{tp_c + fp_c} \\ r_c = \dfrac{tp_c}{tp_c + fn_c} \end{cases} \tag{24}$$

Where tp_c is the number of the samples which are correctly classified into c, fp_c is the number of the samples which are wrongly classified into c, fn_c is the number of the samples which belong to c but are wrongly classified. Obviously, the F_1^c measurement is a harmonic mean of precision and recall whose best value is 1 and worst value is 0 [15]. On this basis, the definition of Macro-averaged F_1 measurement (F_1^{macro}) is given as follows:

$$F_1^{macro} = \frac{\sum_{c \in C} F_1^c}{|C|} \tag{25}$$

Where $|C|$ is the number of classes in the dataset.

Different with Macro-averaged F_1 measurement, Micro-averaged F_1 measurement sums up all classification decisions for all classes of a dataset to calculate global precision and recall. The calculation for Micro-averaged F_1 measurement is shown as follows:

Table 9 F_a values of different filters when RF is used on different datasets

datasets	IG	IMGI	CHI	DFS	CMFSX	DIA	CDM	OR
APS	0.984	0.984	0.984	0.987	**0.990**	0.985	0.984	0.983
Madelon	0.497	0.500	0.696	**0.707**	0.554	0.501	0.629	0.499
CNAE9	0.411	0.831	0.836	0.798	**0.894**	0.677	0.499	0.342
Gisette	0.815	0.940	**0.956**	0.953	0.951	0.912	0.955	0.646
DrivFace	0.947	0.953	**0.957**	**0.957**	0.951	0.940	0.945	0.945
Amazon	0.502	0.701	**0.742**	0.603	0.581	0.573	0.477	0.524

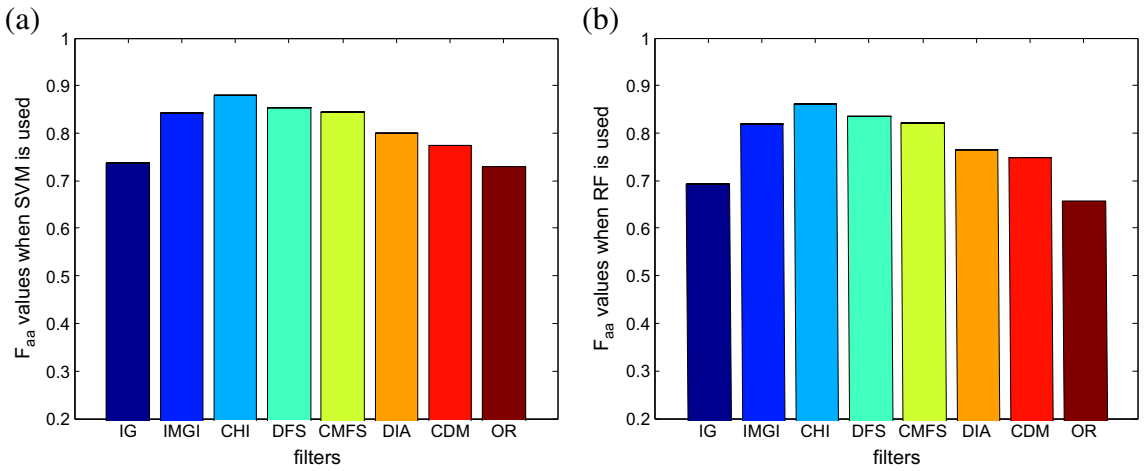


Fig. 4 F_{aa} values of different filters when SVM and RF are used, respectively

$$F_1^{\text{micro}} = \frac{2 \times p \times r}{p + r} \tag{26}$$

Where p is the global precision for all classes and r is the global recall for all classes. The definitions of p and r are shown in formula (27) [51, 52]:

$$p = \frac{\sum_{c \in |C|} tp_c}{\sum_{c \in |C|} (tp_c + fp_c)}, r = \frac{\sum_{c \in |C|} tp_c}{\sum_{c \in |C|} (tp_c + fn_c)} \tag{27}$$

4.4 Selection of the optimal filters in *OFS*

From Algorithms 2–4 we know that, a good selection of the optimal filters may deduce a feature subset FS_t with high quality. On this basis, eight traditional filters (IG, IMGI, CHI, CMFSX, DIA, DFS, CDM [51] and OR) are used for experiments to select the set of four optimal filters $OFS = \{of_1, of_2, of_3, of_4\}$. When the classifiers of SVM and RF are used on different datasets, we calculate the average ${F_1}^{\text{macro}}$ values ($F_{\text{mac\_a}}$) and average ${F_1}^{\text{micro}}$ values ($F_{\text{mic\_a}}$) of different filters as the ratio of selected features ranges from 2% to 10% with a step of 2%. Further, the average values (called F_a) of $F_{\text{mac\_a}}$ values and $F_{\text{mic\_a}}$ values with respect to each method are calculated and the results are shown in Tables 8 and 9. For ease of understanding, the highest F_a values with respect to each dataset are denoted in bold. We know from Tables 8 and 9 that CHI, CMFSX and DFS perform generally better than the other methods as they obtain the highest F_a values for 6, 4 and 3 times, respectively. Moreover, Fig. 4 give the average F_a values (F_{aa}) of each method with respect to all datasets when SVM and RF are used, respectively. As the performances of CHI, CMFSX, DFS and IMGI are obviously better than those of the other filters, they are selected to form the set of optimal filters $OFS = \{\text{CHI, CMFSX, DFS, IMGI}\}$. Moreover, according to Algorithm 2, we obtain the vectors of multi-filter weights of different datasets and the results are shown in Table 10 and 11.

4.5 Sensitivity analysis of the parameter *Q*

From Algorithm 5 we know that the parameter Q affects the running speed of calculating the feature relevance matrix. In this section, we evaluate the performances of the Q-range based method when Q ranges from 0.1 to 1.0 with a step of 0.1. With respect to each value of Q, we calculate the average consuming time (denoted as t_a and expressed in seconds) of

Table 10 Vectors of multi-filter weights of different datasets when SVM is used

datasets	α_1	α_2	α_3	α_4
APS	0.996	0.997	0.998	1.000
Madelon	0.821	0.988	1.000	0.969
CNAE9	0.964	0.949	0.928	1.000
Gisette	0.988	0.990	1.000	0.988
DrivFace	0.982	0.999	1.000	0.983
Amazon	0.788	1.000	0.827	0.840

Table 11 Vectors of multi-filter weights of different datasets when RF is used

datasets	α_1	α_2	α_3	α_4
APS	0.998	0.998	1.000	0.997
Madelon	0.823	0.980	1.000	0.973
CNAE9	1.000	0.994	0.988	0.979
Gisette	0.987	1.000	0.998	0.981
DrivFace	0.993	0.984	1.000	0.991
Amazon	1.000	0.993	0.719	0.939

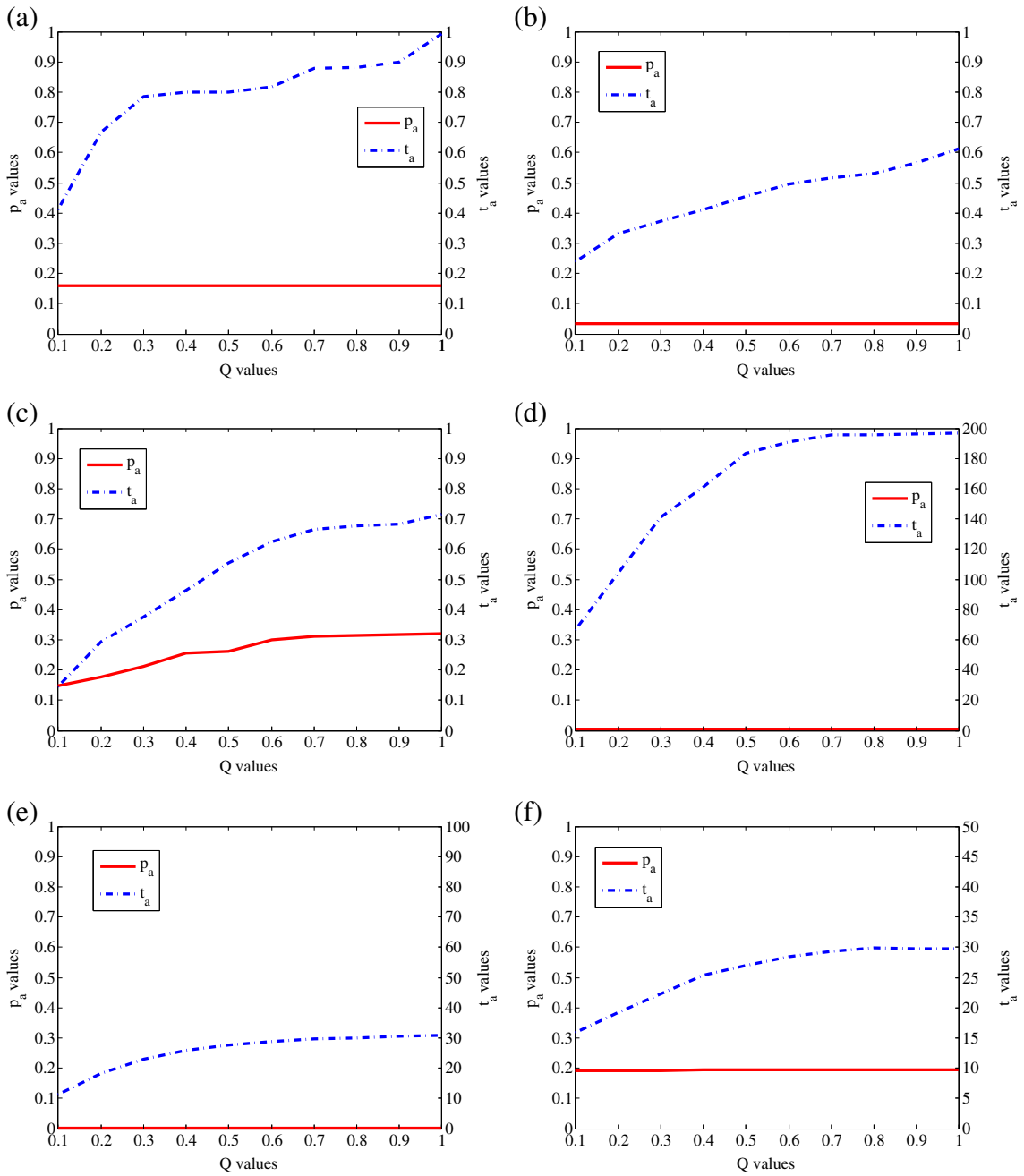


Fig. 5 t_a and p_a values of MFHFS on each dataset when Q ranges from 0.1 to 1

step 3 in Algorithm 5 when the ratio of selected features ranges from 1% to 10% on each dataset, and the results are shown in Fig. 5. Further, given a predetermined threshold th ($th = 0.9$ in this paper) which is approaching 1, we calculate the average probability (denoted as p_a) that the feature relevance is higher than th when the ratio of selected features ranges from 1% to 10% to test the ability of the Q-range based method on achieving the redundant features, and the results are also shown in Fig. 5. Obviously, when considering the datasets of APS, Madelon, Gisette, DrivFace and Amazon, we can see from these figures that the p_a values remain unchanged, but the t_a values increase gradually with the highest increment of about 120 s when Gisette is used. When considering the dataset of CNAE9, we notice that the p_a values have slight increments but the t_a values have significant increments when value of Q increases. Therefore, we conclude that, when Q is greater than 0.1, it has great effect on the running speed of calculating the feature relevance matrix but little effect on the performance of filtering the redundant features. On this basis, in order to improve the efficiency of the proposed method while guaranteeing the classification accuracy, Q is set to $Q = 0.1$ in this paper.

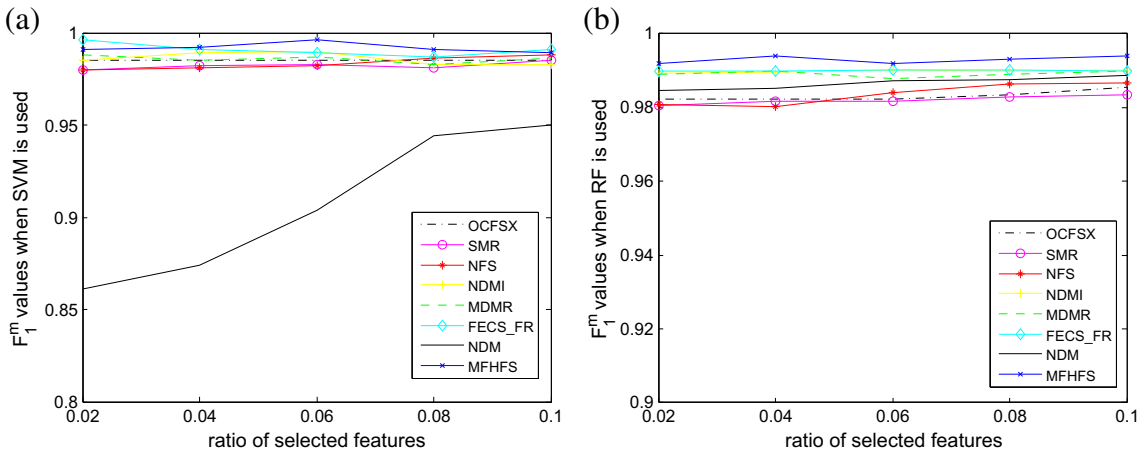


Fig. 6 APS dataset

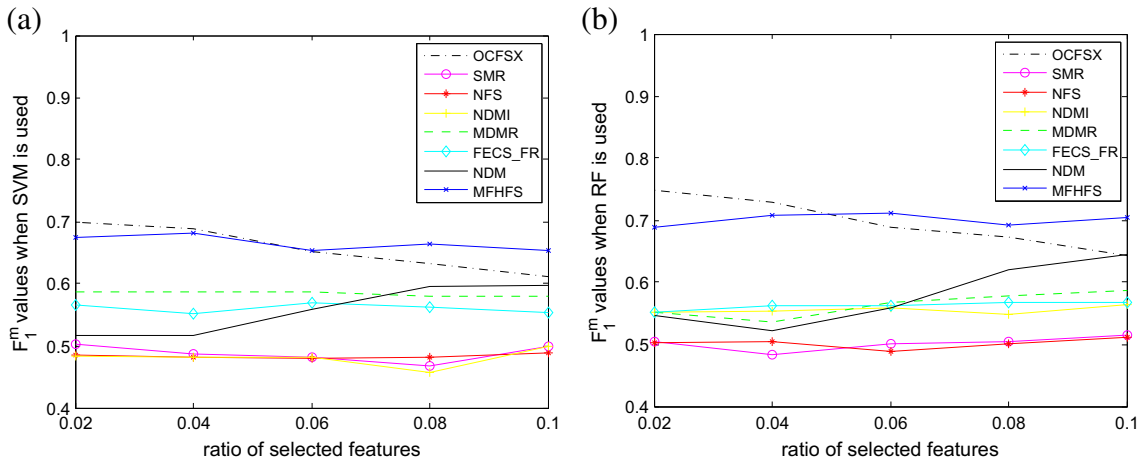


Fig. 7 Madelon dataset

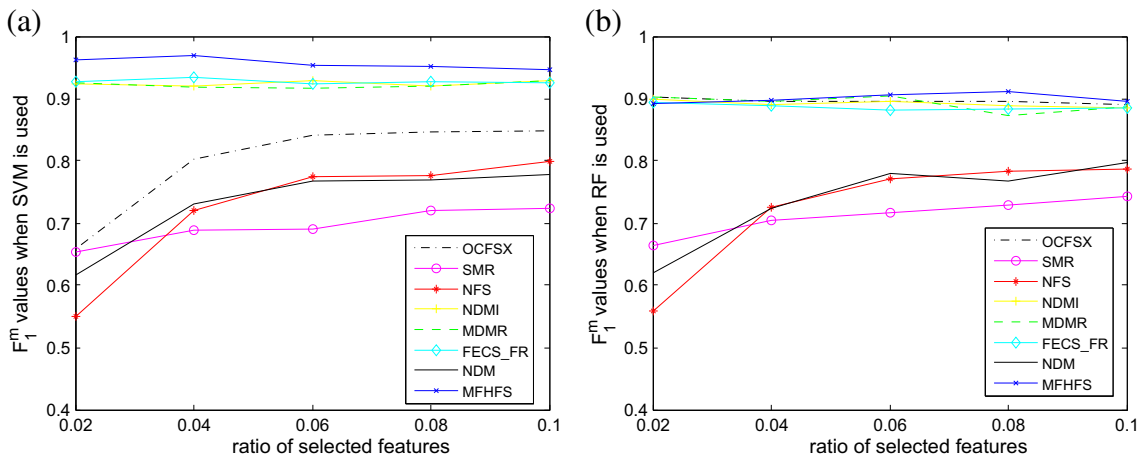


Fig. 8 CNAE9 dataset

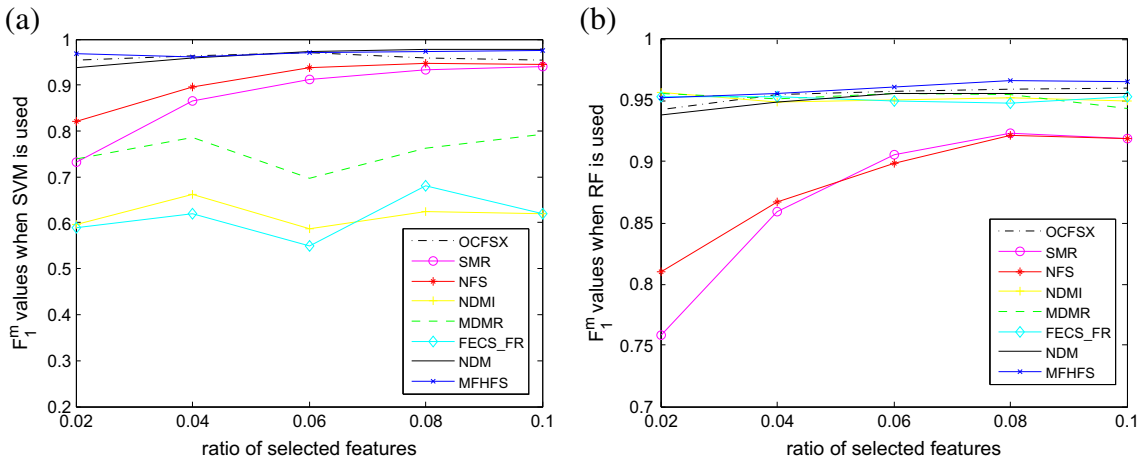


Fig. 9 Gisette dataset

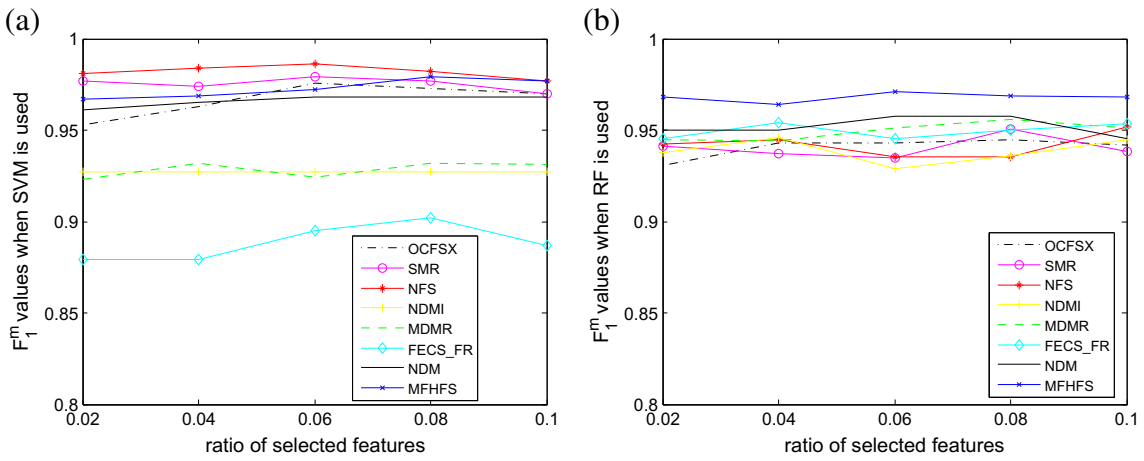


Fig. 10 DrivFace dataset

4.6 Comparisons of MFHFS with typical filters

In order to validate the performances of different methods, we compare typical methods of filters (SMR [21], NDM [23], NDMI [25], MDMR [27], NFS [52], OCFSX [53] and FECS_FR [54]) on the aspects of classification accuracy and running speed. Figure 6, 7, 8, 9, 10 and 11 gives the average values (called $F_1{}^m$) of $F_1{}^{macro}$ values and $F_1{}^{micro}$ values of SVM and RF classifiers when the ratio of selected features ranges from 2% to 10% with a step of 2%. Obviously, when

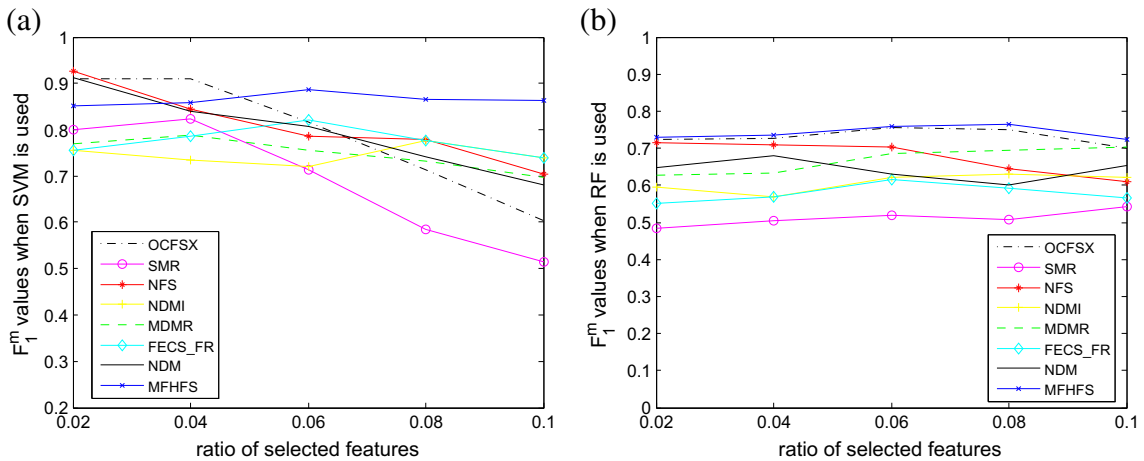


Fig. 11 Amazon dataset

SVM is used, the F_1^m values of MFHFS are greater than those of the other filters in most of the cases when the ratio of selected features is no less than 0.06. As the ratio of selected features increases, the performances of MFHFS are much more stable than those of the other filters, showing the robustness of the proposed method on different datasets. When RF is used, the performances of SMR and NFS are less stable than those of the other filters as the ratio of selected features increases. MFHFS obtains the highest F_1^m values in all cases with respect to the datasets of APS, DrivFace and Amazon, with the highest F_1^m value (0.994) when 0.04 features are selected from the dataset of APS.

Further, Tables 12, 13, 14 and 15 given the average F_1^{macro} values (F_{mac_a}) and average F_1^{micro} values (F_{mic_a}) with the ratio of selected features ranging from 2% to 10% when the classifiers of SVM and RF are used, respectively. In these tables, the highest values of each dataset are denoted in bold. Obviously, we know from Tables 12 and 13 that when SVM is used, MFHFS obtains the highest F_{mac_a} values except for the DrivFace dataset, and it obtains the highest F_{mic_a} values except for the Madelon and DrivFace datasets. Moreover, it can be seen from Tables 14 and 15 that, when RF is used, MFHFS outperforms the other methods for six and five times on the measurements of F_{mac_a} and F_{mic_a}, respectively, illustrating the effectiveness of the proposed method on selecting the best features. Therefore, we conclude that the performance of MFHFS is generally better than those of the typical filters on classification accuracy. This may due to the following two reasons: 1) MFHFS filters the noises and outliers in the preprocessing process, avoiding the overfit problem and improving the robustness of the proposed method; 2) MFHFS achieves the best features of different optimal filters by combining the multi-filter weights and the multi-feature weights, differentiating the importance of different features and solving the problem that the classification performance is not stable when the dataset changes.

Furthermore, Table 16 shows the average running time (called rt_a, expressed in seconds) of different methods when the ratio of selected features ranging from 2% to 10% with a step of 2%. It can be seen from Table 16 that the filters of OCFSX, SMR, NFS, FECS_FR and NDM run obviously faster than MFHFS. This is due to the fact that MFHFS combines multiple filters and executes redundant feature filtering process which is time-consuming on calculating the feature relevance matrix. However, when comparing MFHFS with NDMI and MDMR, we notice that the rt_a values of the former method are obviously lower than those of the later methods. For example, when Gisette dataset is used, the rt_a values of the NDMI and MDMR are 1260.371 and 4387.189, respectively, while the corresponding rt_a value of MFHFS is 87.162. Therefore, by combining Tables 12, 13, 14 and 15 we conclude that: 1) for the aspect of classification accuracy, the performances of MFHFS are generally better than those of typical filters; 2) for the aspect of running speed, MFHFS has acceptable decrements compared to the traditional filters (OCFSX, SMR, NFS, FECS_FR and NDM) and has great increments compared to the MI based filters (NDMI and MDMR).

4.7 Comparisons of MFHFS with typical feature extractions and hybrid feature selections

In this section, we conduct a series of experiments to compare MFHFS with several typical feature extractions and hybrid feature selections: AE [4], PCA [8], CBHFS [30], VGFSS [32], EGAFS [36], EMFFS [39], MFHFS1 and MFHFS2. Among these methods, MFHFS1 is modified from MFHFS and calculates the feature relevance matrix using HFSCC method [37]; MFHFS2 is modified from MFHFS without filtering the noises and outliers in the data preprocessing stage. Here, the parameters of each method are given as follows: (1) AE: the maximum epochs: $M_e = 100$; (2) EMFFS: the optimal filters are CMFSX, CHI, IMGI and DFS; (3) EGAFS: number of particles: $N_p = 20$, number of iterations: $N_{it} = 10$; (4) CBHFS: number of the K values: $N_k = 10$, number of repetitions for each K: $N_r = 10$. Moreover, the classifiers used in the wrappers of these methods are SVM and RF. On this basis, when the ratio of selected features ranges from 2% to 10% with a step of 2%, the average F_1^{macro} values (F_{mac_a}) and average F_1^{micro} values (F_{mic_a}) are shown in Tables 17, 18, 19 and 20, and the average F_{mac_a} (F_{mic_a}) values of all datasets for each method when SVM and RF are used are given in Fig. 12.

We can see from Tables 17 and 18 that when SVM is used, MFHFS obtains the highest values for five times, which is obviously higher than those of the other methods. The performances of MFHFS and MFHFS1 are similar and generally better than that of NFHFS2, showing the efficiency of filtering noises and outliers in the preprocessing stage on improving the classification accuracy. Moreover, we can see from Fig. 12a and b that the average F_{mac_a} values and average F_{mic_a} values of the proposed method are generally higher than those of the other methods, with the highest improvement of about 0.04 over that of PCA when SVM is used. We know from Tables 19 and 20 that when RF is used, MFHFS obtains the highest F_{mac_a} value when the datasets of APS, Gisette and DrivFace are used, illustrating the efficacy of MFHFS on dealing with the datasets those have high numbers of features or samples. Moreover, we know from Fig. 12c and d that MFHFS and MFHFS1 outperforms the other method significantly, with the average improvements of about 0.05 and 0.04 over those of EGAFS and EMFFS when the measurements of average F_{mac_a} values and average F_{mic_a} values are used, respectively. This may be due to the following reasons: (1)

Table 12 F_{mac_a} values of typical filters and MFHFS when SVM is used

datasets	OCFSX	SMR	NFS	NDMI	MDMR	FECS_FR	NDM	MFHFS
APS	0.985	0.982	0.983	0.985	0.986	0.991	0.907	**0.992**
Madelon	0.657	0.488	0.483	0.481	0.584	0.560	0.557	**0.666**
CNAE9	0.799	0.695	0.724	0.925	0.922	0.928	0.732	**0.957**
Gisette	0.961	0.877	0.909	0.618	0.755	0.612	0.965	**0.970**
DrivFace	0.967	0.975	**0.982**	0.927	0.928	0.888	0.966	0.973
Amazon	0.791	0.687	0.808	0.746	0.748	0.776	0.796	**0.865**

Table 13 F_{mic_a} values of typical filters and MFHFS when SVM is used

datasets	OCFSX	SMR	NFS	NDMI	MDMR	FECS_FR	NDM	MFHFS
APS	0.984	0.986	0.979	0.985	0.985	**0.994**	0.911	0.992
Madelon	0.651	0.488	0.481	0.485	0.582	0.564	0.563	**0.672**
CNAE9	0.802	0.691	0.731	0.932	0.926	0.931	0.733	**0.955**
Gisette	0.964	0.879	0.905	0.622	0.757	0.619	0.966	**0.968**
DrivFace	0.971	0.972	**0.981**	0.927	0.933	0.892	0.964	0.971
Amazon	0.788	0.685	0.810	0.749	0.751	0.778	0.798	**0.867**

Table 14 F_{mac_a} values of typical filters and MFHFS when RF is used

datasets	OCFSX	SMR	NFS	NDMI	MDMR	FECS_FR	NDM	MFHFS
APS	0.983	0.982	0.984	0.990	0.989	0.990	0.987	**0.993**
Madelon	0.697	0.502	0.502	0.555	0.564	0.562	0.579	**0.701**
CNAE9	0.896	0.711	0.725	0.892	0.892	0.887	0.738	**0.900**
Gisette	0.941	0.940	0.942	0.939	0.949	0.950	0.952	**0.968**
DrivFace	0.955	0.873	0.883	0.951	0.952	0.951	0.951	**0.960**
Amazon	0.731	0.511	0.677	0.607	0.668	0.578	0.642	**0.742**

Table 15 F_{mic_a} values of typical filters and MFHFS when RF is used

datasets	OCFSX	SMR	NFS	NDMI	MDMR	FECS_FR	NDM	MFHFS
APS	0.981	0.978	0.982	0.989	0.986	**0.992**	0.986	0.991
Madelon	0.696	0.508	0.506	0.552	0.565	0.562	0.576	**0.705**
CNAE9	0.895	0.715	0.723	0.886	0.889	0.885	0.742	**0.904**
Gisette	0.939	0.940	0.941	0.944	0.952	0.953	0.957	**0.963**
DrivFace	0.953	0.871	0.883	0.953	0.952	0.951	0.956	**0.962**
Amazon	0.735	0.516	0.679	0.611	0.669	0.575	0.642	**0.745**

Table 16 rt_a values of typical filters and MFHFS on each dataset (unit: second)

datasets	OCFSX	SMR	NFS	NDMI	MDMR	FECS_FR	NDM	MFHFS
APS	0.124	0.560	1.134	6.596	296.785	0.783	0.491	2.372
Madelon	0.207	0.177	0.304	4.678	141.920	0.353	0.135	0.876
CNAE9	0.313	0.482	0.496	4.787	837.124	0.556	0.244	1.1432
Gisette	1.268	3.407	6.683	1260.371	4387.189	40.556	2.328	87.162
DrivFace	0.159	1.926	1.267	313.774	1255.475	7.932	1.386	14.962
Amazon	0.193	2.735	1.457	425.286	3273.763	5.106	0.775	24.236

Table 17 F_{mac_a} values of typical hybrid methods and MFHFS when SVM is used

datasets	PCA	AE	EMFFS	EGAFS	VGFSS	CBHFS	MFHFS1	MFHFS2	MFHFS
APS	0.980	0.978	0.982	0.985	**0.989**	0.981	0.985	0.981	**0.989**
Madelon	0.691	0.659	**0.741**	0.621	0.638	0.627	0.677	0.646	0.678
CNAE9	0.904	0.692	0.843	**0.942**	0.925	0.926	0.887	0.861	0.885
Gisette	0.968	0.971	0.961	0.975	0.958	0.966	**0.983**	0.976	0.982
DrivFace	0.972	0.976	0.796	0.826	0.801	0.832	0.989	0.969	**0.990**
Amazon	0.605	0.892	0.836	0.862	**0.915**	0.912	0.871	0.863	0.870

Table 18 F_{mic_a} values of typical hybrid methods and MFHFS when SVM is used

datasets	PCA	AE	EMFFS	EGAFS	VGFSS	CBHFS	MFHFS1	MFHFS2	MFHFS
APS	0.985	0.983	0.985	0.989	0.991	0.983	0.989	0.981	**0.992**
Madelon	0.682	0.658	**0.741**	0.623	0.638	0.629	0.677	0.647	0.677
CNAE9	0.904	0.687	0.841	**0.947**	0.936	0.928	0.883	0.859	0.883
Gisette	0.971	0.973	0.963	0.972	0.961	0.969	**0.981**	0.978	**0.981**
DrivFace	0.972	0.977	0.955	0.935	0.881	0.907	0.989	0.972	**0.990**
Amazon	0.632	0.895	0.832	0.871	**0.927**	0.942	0.875	0.865	0.876

Table 19 F_{mac_a} values of typical hybrid methods and MFHFS when RF is used

datasets	PCA	AE	EMFFS	EGAFS	VGFSS	CBHFS	MFHFS1	MFHFS2	MFHFS
APS	0.984	0.983	0.984	0.990	0.989	0.985	0.988	0.983	**0.991**
Madelon	0.672	0.695	**0.752**	0.614	0.621	0.592	0.701	0.681	0.703
CNAE9	0.767	0.547	0.843	0.882	**0.899**	0.847	0.898	0.855	0.898
Gisette	0.789	0.812	0.956	0.946	0.946	0.952	0.965	0.958	**0.967**
DrivFace	0.919	0.967	0.681	0.852	0.813	0.643	0.969	0.959	**0.972**
Amazon	0.334	0.695	0.662	0.683	0.551	0.698	**0.762**	0.743	0.758

Feature extractions like PCA and AE ignore the category information of the samples, losing some information which may be helpful for data classification; (2) EMFFS and VGFSS evaluate the importance of the features separately, ignoring the effect of the redundant information contained in the selected features; (3) The results of EGAFS and CBHFS rely on some important parameters (such as the numbers of iterations or particles) in the genetic algorithm and the clustering algorithm, deducing unstable results when dealing with high dimensional datasets; (4) The samples of noise or outliers are removed in MFHFS, improving the robustness and generalization of the proposed method.

Table 21 gives the average running time (called rt_a, expressed in seconds) of these methods on each dataset when SVM and RF are used, respectively. We observe that CBHFS is the slowest one in these methods, and it obtains the highest

Table 20 F_{mic_a} values of typical hybrid methods and MFHFS when RF is used

datasets	PCA	AE	EMFFS	EGAFS	VGFSS	CBHFS	MFHFS1	MFHFS2	MFHFS
APS	0.985	0.984	0.985	0.991	0.991	0.987	**0.996**	0.991	**0.996**
Madelon	0.681	0.691	**0.755**	0.597	0.603	0.597	0.706	0.680	0.705
CNAE9	0.765	0.556	0.847	0.883	0.906	0.852	0.894	0.858	**0.896**
Gisette	0.795	0.809	0.956	0.949	0.949	0.958	0.962	0.955	**0.967**
DrivFace	0.926	0.968	0.928	0.926	0.967	0.952	0.971	0.961	**0.974**
Amazon	0.332	0.712	0.689	0.672	0.587	0.711	**0.769**	0.752	0.767

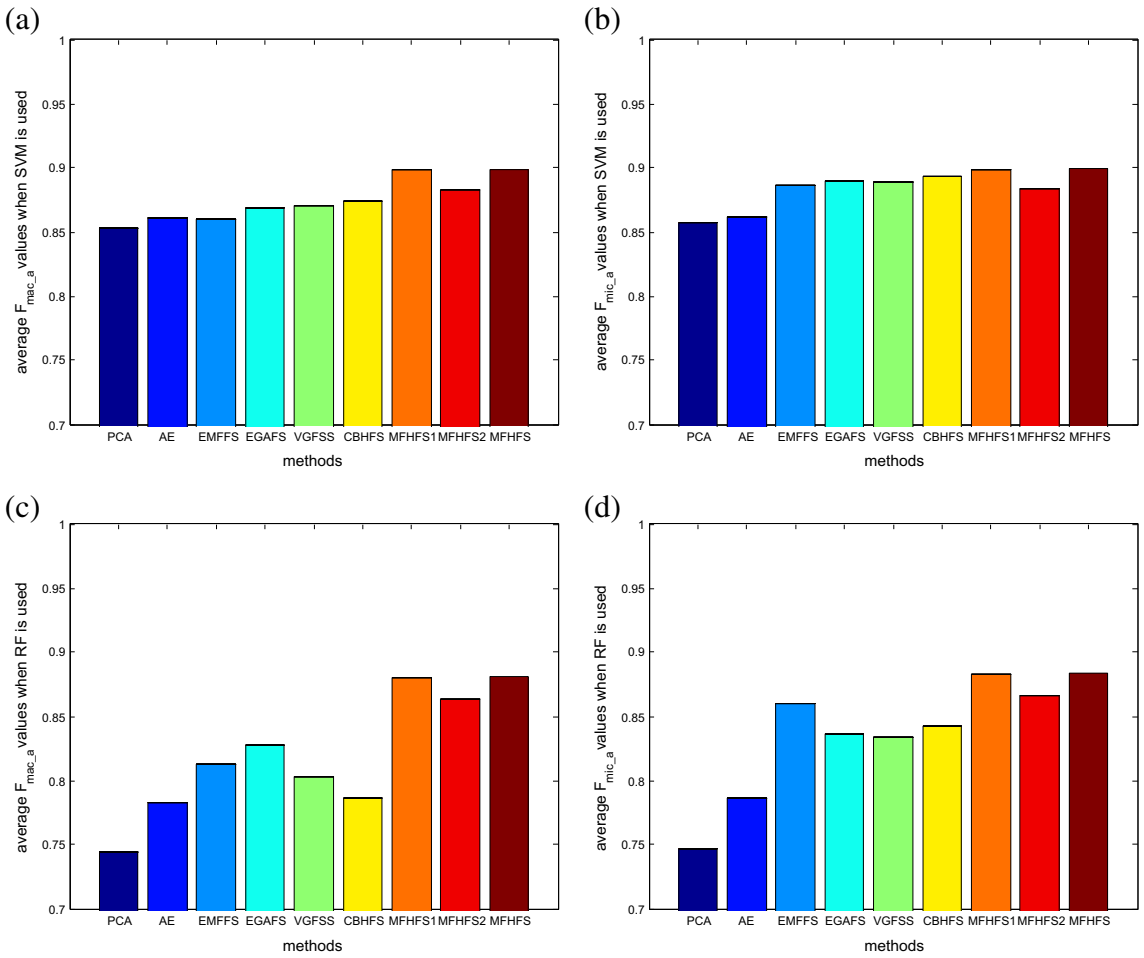


Fig. 12 Average F_{mac_a} values and average F_{mic_a} values of different methods when SVM and RF are used, respectively

Table 21 rt_a values of typical hybrid methods and MFHFS (unit: second)

datasets	PCA	AE	EMFFS	EGAFS	VGFSS	CBHFS	MFHFS1	MFHFS
APS	0.919	72.335	2.568	7720.625	1.236	27,630.129	28.664	1.1432
Madelon	0.216	9.731	1.663	893.326	0.933	4028.185	3.519	2.372
CNAE9	0.528	10.223	2.289	1923.996	1.527	2497.662	3.058	0.876
Gisette	69.325	1125.764	30.228	3689.342	22.728	335,926.365	237.862	87.162
DrivFace	3.552	301.513	5.335	904.173	3.426	56,267.124	32.753	14.962
Amazon	3.025	512.637	6.983	1029.637	5.321	49,693.356	46.565	24.236

Table 22 p values of different methods when SVM is used (the cases of $p > \alpha$ are denoted in bold)

datasets	CMFSX	IMGI	OCFSX	NFS	NDMI	FECS_FR	AE	EMFFS	VGFSS	MFHFS2
APS	0.007	0.002	0.004	0.004	0.004	**0.355**	0.002	0.002	0.004	0.002
Madelon	0.002	0.002	**0.359**	0.004	0.004	0.004	0.000	0.040	0.002	0.002
CNAE9	0.002	0.002	0.004	0.004	0.004	0.004	0.002	0.002	0.002	0.002
Gisette	0.002	0.002	0.031	0.004	0.004	0.004	0.002	0.002	0.002	0.002
DrivFace	0.002	0.002	0.020	0.008	0.004	0.004	0.002	0.002	0.002	0.002
Amazon	0.004	**0.132**	**0.128**	**0.055**	0.004	0.004	**0.078**	**0.852**	**0.625**	0.002

running time (335,926.365 s) on Gisette dataset. Moreover, EGAFS runs much slower than the other methods except for CBHFS. The reason may be that CBHFS and EGAFS both use the wrapper methods which take a lot of time on training and classification to obtain the best features. In addition, we notice that the increments of MFHFS over PCA, EMFFS and VGFSS on rt_a are all lower than 70 s, which is acceptable in actual situations. Moreover, compared to the MFHFS1 methods, MFHFS runs much faster, illustrating the efficiency of the Q-range based feature relevance calculation method on improve the running speed while guaranteeing the qualities of selected features.

4.8 Statistical comparisons

The nonparametric tests are widely used in statistical learning. The Wilcoxon signed rank test is usually powerful in detecting the difference between two populations [55]. In this section, the Wilcoxon signed rank test is used, and the null hypothesis is that the two methods are equivalent. If the null hypothesis is rejected (p value is less than or equal to the significance level α), the differences between the methods are significant. Given the significance level $\alpha = 0.05$, the average ${F_1}^{macro}$ and ${F_1}^{micro}$ values (called F_a) of different typical methods are compared using the Wilcoxon signed rank test when the ratio of selected feature ranges from 0.01 to 0.1 with a step of 0.01. Moreover, experiments are carried out 10 times and the average p values are shown in Tables 22 and 23 when SVM and RF are used, respectively. We can see from Table 22 that the p values are lower than α in 52 of 60 cases when SVM is used, showing that the proposed method outperforms the other methods significantly in 86.7% of the cases on classification accuracy. Moreover, it is obvious that the proposed method outputs the other methods significantly on the datasets of CNAE9, Gisette and DrivFace, though its superiority on Amazon dataset is not remarkable when compared to IMGI, OCFSX, EMFFS and VGFSS. Further, we can see from Table 23 that the proposed method outputs the other methods significantly in 91.7% cases (55 of 60 cases) when RF is used, illustrating the effectiveness of the proposed method on selecting the best features and ensuring the classification accuracy.

5 Conclusions

Many traditional feature selections of filters rely on the datasets and cannot deal with the redundant information effectively. Moreover, many MI based filters or hybrid methods have high time complexities when the numbers of features or the numbers of samples are high. In this paper, a multi-filter weights and multi-feature weights based hybrid feature selection (called MFHFS) is proposed. In the data preprocessing stage, the samples are normalized and discretized by using the equal width interval binning (EWIB) algorithm. Moreover, the 10-folder cross validation is combined to remove the samples of noises and outliers. In the feature combination stage, several optimal filters are chosen and used to obtain the feature subsets, and these feature subsets are merged into a temp feature subset by considering the multi-filter weights and multi-feature weights. In the feature refinement stage, the redundant information of the temp feature subset is filtered to obtain the final feature subset, and a Q-range based feature relevance calculation method is proposed to improve the running speed of redundant information filtering. The efficiency of MFHFS is examined through the classification experiments with SVM and RF classifiers on six datasets: APS, Madelon, CNAE9, Gisette, DrivFace and Amazon. Experimental results show that: (1) When compared to the traditional filters, MFHFS has great improvement on classification accuracy at the cost of acceptable decrement of running speed; (2) MFHFS achieves obvious improvements on classification accuracy over typical feature extractions and hybrid feature selections, and it has significant improvements on running speed over typical methods like AE, EGAFS and CBHFS, illustrating its efficacy on obtaining the best features in data classification field.

In the future, we will study deeper in the following two aspects: (1) Learn from different datasets of multiple fields and investigate the multi-task based feature selection methods; (2) Investigate parallel computing methods to improve the running speed and extend the proposed method to real applications.

Table 23 p values of different methods when RF is used (the cases of $p > \alpha$ are denoted in bold)

datasets	CMFSX	IMGI	OCFSX	NFS	NDMI	FECS_FR	AE	EMFFS	VGFSS	MFHFS2
APS	0.002	0.002	0.004	0.004	0.004	0.004	0.002	0.002	0.002	0.002
Madelon	0.002	0.002	**0.632**	0.004	0.004	0.004	0.004	**0.102**	0.002	0.002
CNAE9	0.044	0.002	**0.102**	0.004	0.012	0.008	0.002	0.002	**0.088**	0.002
Gisette	0.002	0.002	0.004	0.004	0.002	0.002	0.002	0.002	0.002	0.002
DrivFace	0.002	0.002	0.004	0.004	0.004	0.004	0.002	0.002	0.002	0.002
Amazon	0.002	0.002	0.004	0.004	0.004	0.004	**0.052**	0.002	0.035	0.002

Acknowledgements This research is supported by the Beijing Natural Science Foundation, China (No. 4174105), the Key Projects of National Bureau of Statistics of China (No. 2017LZ05), the National Key R&D Program of China (2017YFB1400700), the Joint Funds of the National Natural Science Foundation of China (No. U1509214).

References

1. Hancer E, Xue B, Zhang M (2018) Differential evolution for filter feature selection based on information theory and feature ranking. Knowledge-Based Systems
2. Rawles S, Flach P (2004) Redundant feature elimination for multi-class problems. International Conference on Machine Learning ACM
3. Bharti KK, Singh PK (2015) Hybrid dimension reduction by integrating feature selection with feature extraction method for text clustering. Expert Syst Appl 42(6):3105–3114
4. Zabalza J, Ren J, Zheng J et al (2016) Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. Neurocomputing 185(C):1–10
5. Quispe O, Ocsa A, Coronado R (2017) Latent semantic indexing and convolutional neural network for multi-label and multi-class text classification. IEEE Latin American Conference on Computational Intelligence. IEEE, 1–6
6. Marquetti I, Link JV, Lemes ALG et al (2016) Partial least square with discriminant analysis and near infrared spectroscopy for evaluation of geographic and genotypic origin of arabica coffee. Comput Electron Agric 121(C):313–319
7. Okada K, Lee MD (2016) A Bayesian approach to modeling group and individual differences in multidimensional scaling. J Math Psychol 70:35–44
8. Fan Z, Xu Y, Zuo W et al (2017) Modified principal component analysis: an integration of multiple similarity subspace models. IEEE Transactions on Neural Networks & Learning Systems 25(8):1538–1552
9. Prihatini PM, Putra IKGD, Giriantari IAD et al (2017) Fuzzy-Gibbs latent Dirichlet allocation model for feature extraction on Indonesian documents. Contemporary Engineering Sciences 10: 403–421
10. Zhang Y, Zhang Z (2012) Feature subset selection with cumulate conditional mutual information minimization. Expert Syst Appl 39(5):6078–6088
11. Yang Y, Pedersen J (1997) A comparative study on feature set selection in text categorization. In: Fisher DH (ed) Proceedings of the 14th International Conference on Machine Learning. Morgan Kaufmann, San Francisco, pp 412–420
12. Shang W, Huang H, Zhu H et al (2007) A novel feature selection algorithm for text classification. Expert Syst Appl 33(1):1–5
13. Uysal AK, Gunal S A novel probabilistic feature selection for text classification. Knowl-Based Syst 36:226–235
14. Mengle SSR, Goharian N (2009) Ambiguity measure feature-selection algorithm. J Am Soc Inf Sci Technol 60:1037–1050
15. Sebastiani F (2002) Machine learning in automated text categorization. ACM Computing Surveys (CSUR) 34(1):1–47
16. Shi JT, Liu HL, Xu Y et al (2014) Chinese sentiment classifier machine learning based on optimized information gain feature selection. Adv Mater Res 988:511–516
17. Peng H, Long F, Ding C (2005) 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
18. Moradi P, Rostami M (2015) Integration of graph clustering with ant colony optimization for feature selection. Knowl-Based Syst 84(C):144–161
19. Yan J, Liu N, Zhang B (2009) OCFS: optimal orthogonal centroid feature selection for text categorization. International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM: 122–129
20. Yang J, Qu Z, Liu Z (2014) Improved feature-selection method considering the imbalance problem in text categorization. Sci World J:1–17
21. Tutkan M, Ganiz MC, Akyokuş S (2016) Helmholtz principle based supervised and unsupervised feature selection methods for text mining. Inf Process Manag 52(5):885–910
22. Forman G (2003) An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 3:1289–1305
23. Rehman A, Javed K, Babri HA (2017) Feature selection based on a normalized difference measure for text classification. Inf Process Manag 53(2):473–489
24. Zhou X, Hu Y, Guo L (2014) Text categorization based on clustering feature selection. Procedia Computer Science 31(31):398–405
25. Hoque N, Bhattacharyya DK, Kalita JK (2014) MIFS-ND: A mutual information-based feature selection. Expert Syst Appl 41(14): 6371–6385
26. Vinh LT, Lee S, Park YT et al (2012) A novel feature selection based on normalized mutual information. Appl Intell 37(1):100–120
27. Lin Y, Hu Q, Liu J et al (2015) Multi-label feature selection based on max-dependency and min-redundancy. Neurocomputing 168: 92–103
28. Das S (2001) Wrappers and a boosting-based hybrid for feature selection. International Conference on Machine Learning 74–81
29. Es TF, Hruschka ER, Castro LN et al (2009) A cluster-based feature selection approach. Hybrid Artificial Intelligence Systems, International Conference, Salamanca, Spain, Proceedings DBLP: 169–176
30. Jaskowiak PA, Campello RJGB (2015) A cluster based hybrid feature selection approach. Intelligent Systems. IEEE, 43–48
31. Uysal AK (2016) An improved global feature selection scheme for text classification. Expert Syst Appl 43:82–92
32. Agnihotri D (2017) Variable global feature selection scheme for automatic classification of text documents. Expert Syst Appl 81(C):268–281
33. Wang Y, Liu Y, Feng L et al (2015) Novel feature selection based on harmony search for email classification. Knowl-Based Syst 73(1): 311–323
34. Zorarpacı E, Özel SA (2016) A hybrid approach of differential evolution and artificial bee colony for feature selection. Expert Syst Appl 62:91–103
35. Xue B, Zhang M, Browne WN (2014) Particle swarm optimization for feature selection in classification: novel initialization and updating mechanisms. Appl Soft Comput 18:261–276
36. Ghareb AS, Bakar AA, Hamdan AR (2016) Hybrid feature selection based on enhanced genetic algorithm for text categorization. Expert Syst Appl 49:31–47
37. Wang Y, Feng L (2018) Hybrid feature selection using component co-occurrence based feature relevance measurement. Expert Syst Appl 102:83–99
38. Bhattacharya S, Selvakumar S (2016) Multi-measure multi-weight ranking approach for the identification of the network features for the detection of DoS and Probe attacks. Comput J 59(6):bxv078
39. Osanaiye O, Cai H, Choo KKR et al (2016) Ensemble-based multi-filter feature selection for DDoS detection in cloud computing. EURASIP J Wirel Commun Netw 2016(1):130
40. Wang Y, Feng L, Li Y (2017) Two-step based feature selection for filtering redundant information. J Intell Fuzzy Syst 33(4):2059–2073
41. Breiman L, Friedman JH, Olshen RA (1984) Classification and regression trees. Wadsworth International Group, Montery

42. Wang Y, Feng L, Zhu J (2017) Novel artificial bee colony based feature selection for filtering redundant information. Appl Intell 3: 1–18
43. Duda J (1995) Supervised and unsupervised discretization of continuous Features. Machine Learning Proceedings (2):194–202
44. Paulus J, Klapuri A (2009) Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. IEEE Trans Audio Speech Lang Process 17(6):1159–1170
45. Dadaneh BZ, Markid HY, Zakerolhosseini A (2016) Unsupervised probabilistic feature selection using ant colony optimization. Expert Syst Appl 53:27–42
46. Asuncion A, Newman DJ (2007) UCI machine learning repository. University of California, Department of Information and Computer Science, Irvine
47. Shan S (2016) Support vector machine. Machine Learning Models and Algorithms for Big Data Classification. Springer US, 24–52
48. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
49. Masetic Z, Subasi A (2016) Congestive heart failure detection using random forest classifier. Comput Methods Prog Biomed 130(C): 54–64
50. Chang CC, Lin CJLIBSVM (2001) A library for support vector machines. ACM Trans Intell Syst Technol 2(27):1–27
51. Chen J, Huang H, Tian S, Qu Y (2009) Feature selection for text classification with Naïve Bayes. Expert Syst Appl 36(3):5432–5435
52. Chang F, Guo J, Xu W et al (2015) A feature selection to handle imbalanced data in text classification. J Digit Inf Manag 13(3):169–175
53. Yang J, Qu Z, Liu Z (2014) Improved feature-selection method considering the imbalance problem in text categorization. Sci World J 3:625342
54. Liu WS, Chen X, Gu Q (2018) A noise tolerable feature selection framework for software defect prediction. Chinese Journal of Computers 41(3):506–520
55. Wang YW, Feng LZ (2018) A new feature selection for handling redundant information in text classification. Frontiers of Information Technology & Electronic Engineering 19(2):221–234

Youwei Wang Associate professor of School of Information, Central University of Finance and Economics, Beijing, China. He received the Doctor degree in computer science and technology college from Jilin University of China in 2015. He received the master degree from Jilin University of China in 2011. His current research interests include data mining, machine learning, watermarking, etc.

Lizhou Feng Lecturer of School of science and engineering, Tianjin University of Finance and Economics, Tianjin, China. She received the Doctor degree in computer science and technology college from Jilin University of China in 2015. She received the master degree from Jilin University of China in 2011. Her current research interests include data mining, text classification, Web Intelligence, etc.