# A bipartite matching-based feature selection for multi-label learning

**Article** *in* International Journal of Machine Learning and Cybernetics · February 2021

DOI: 10.1007/s13042-020-01180-w

Amin Hashemi
Yazd University

Mohammad Bagher Dowlatshahi
Lorestan University

Hossein Nezamabadi-pour
Shahid Bahonar University of Kerman

International Journal of Machine Learning and Cybernetics
https://doi.org/10.1007/s13042-020-01180-w

ORIGINAL ARTICLE

# A bipartite matching-based feature selection for multi-label learning

Amin Hashemi[1] · Mohammad Bagher Dowlatshahi[1] · Hossein Nezamabadi-Pour[2]

Received: 11 January 2020 / Accepted: 1 August 2020

## Abstract

Many real-world data have multiple class labels known as multi-label data, where the labels are correlated with each other, and as such, they are not independent. Since these data are usually high-dimensional, and the current multi-label feature selection methods have not been precise enough, then a new feature selection method is necessary. In this paper, for the first time, we have modeled the problem of multi-label feature selection to a bipartite graph matching process. The proposed method constructs a bipartite graph of features (as the left vertices) and labels (as the right vertices), called Feature-Label Graph (FLG), where each feature is connected to the set of labels, where the weight of the edge between each feature and label is equal to their correlation. Then, the Hungarian algorithm estimates the best matching in FLG. The selected features in each matching are sorted by weighted correlation distance and added to the ranking vector. To select the discriminative features, the proposed method considers both the redundancy of features and the relevancy of each feature to the class labels. The results indicate the superiority of the proposed method against the other methods in classification measures.

**Keywords** Multi-label learning · Bipartite graph matching · Hungarian algorithm · Weighted correlation distance

## 1 Introduction

Today, multi-label classification is widely used in many real-world fields, such as bioinformatics, text categorization, and image classification. Multi-label data are a type of data where a sample can be associated with multiple labels [36, 44]. The use of this type of classification is widespread in many issues today. For example, consider a natural landscape image which can simultaneously receive three different labels such as tree, mountain, and flower. Thus, using a single label classification for such data is very difficult. Another advantage of multi-label classification over single-label is that labels are correlated and not independent of each other. For example, if the natural landscape has a tree label, it is more likely that it will have a flower label than a car label as the flower is more related to the tree. This makes a classifier perform better [21].

Multi-label data often have a large number of features that contain redundant and irrelevant features. This can cause problems such as high computational cost, over-fitting, low classification accuracy, and long learning time during the learning procedure. Feature selection is a powerful tool to address the previously mentioned problems [1, 9, 11, 25]. In the feature selection process, the relevant features of the dataset will be chosen. It has a fundamental role in reducing the data processing scale by eliminating the redundant and irrelevant features in classification and regression tasks [3, 29].

Binary transformation and algorithm adaptation are two main groups of the multi-label feature selection algorithms. Binary transformation approaches with multi-label data can be treated the same as multiple independent single-label problems, where a state-of-the-art single-label method is employed for feature selection [21]. The main drawback of these methods is that the correlation between the labels is not considered. On the other hand, the algorithm adaptation methods generalize the single-label feature selection algorithms for multi-label problems [32].

✉ Mohammad Bagher Dowlatshahi
dowlatshahi.mb@lu.ac.ir

Amin Hashemi
hashemi.am@fe.lu.ac.ir

Hossein Nezamabadi-Pour
nezam@uk.ac.ir

[1] Department of Computer Engineering, Faculty of Engineering, Lorestan University, Khorramabad, Iran

[2] Department of Electrical Engineering, Shahid Bahonar University of Kerman, Kerman, Iran

Published online: 11 August 2020

In another categorization, there are three main groups of multi-label feature selection methods: filter, wrapper, and embedded [32]. Filter-based methods evaluate the features before the learning process and use some predefined metrics such as ReliefF [34], Mutual Information (MI) [24], Information Interest (II) [25], and Symmetric Uncertainty (SU) [22] instead of using the learning algorithms. These methods evaluate the feature subsets to determine the best values according to their predictive power [28]. Finally, embedded methods seek to address the weaknesses of the two previous methods and use the strengths of both [25].

Unlike meta-heuristic methods such as gravitational search algorithm [12–14, 33], ant colony optimization [8], and Particle Swarm Optimization [10], which require complex processes for optimization, the graphs are very useful in data mining and machine learning because of their representational power, flexibility, and less complex procedures. In feature selection, the features can also be represented more naturally using the concept of graphs rather than with feature vectors [35].

Nowadays, many data are multi-labeled, causing many challenges for learning algorithms with the rise of labels. Given the reasons stated at the beginning of this section for the necessity of feature selection and as the current feature selection algorithms for multi-label data have not yet achieved acceptable classification accuracy, so providing a new feature selection method for multi-label data is imperative. One of the reasons for the poor performance of current methods is that they do not consider the impact of the labels. Labels, as with features, may be redundant, or some labels may have a greater impact on the classification rate depends on their frequency. Thus, neglecting the impact of the labels can cause improper performance of the present methods. On the other hand, many of the existing methods focus on the correlation of features and labels, and Many FS methods consider the redundancy between features. The redundancy between labels generally are missed.

In this paper, we have proposed a new filter-based algorithm for a multi-label feature selection. In this method, we have used a two-step strategy to consider the relevancy to class labels and distribution of features. Unlike common multi-label feature selection methods, this method is not inspired by any single-label method, and to the best of our knowledge, it is the first time that graph matching is used for multi-label feature selection. Initially, as the relevancy measure, we have constructed a bipartite graph of features as the left vertices and labels as the right vertices, where each feature is connected to all of the labels, with the weight of the edge between each feature and labels being based on their correlation. Then, the Hungarian algorithm is used to find the best match between features and labels. Finally, the weighted correlation distance is applied to subsets obtained in each matching to sort the features.

An overview of the proposed method is as follows:

- The number of selected features is determined by the user
- This method would select the associated features and eliminate unrelated features according to their correlation with labels.
- The Hungarian algorithm is used as the first measure for finding the most relevant features
- The maximum correlation distance based on label weighting is used as the secondary measure for sorting the subset of features obtained by the Hungarian algorithm.

To illustrate the optimality and efficiency of the proposed method, we have compared the proposed approach with some similar methods, on different real-world datasets. The results indicate that the proposed method yields better results than the other methods in the classification measures.

The structure of this paper is organized as follows: Sect. 2 deals with reviewing related methods. In Sect. 3, the fundamental concepts used in the proposed method will be described, and Sect. 4 describes the proposed method in detail. Section 5 includes the experimental results, and Sect. 6 is the part that the results are shown, and Sect. 7 presents the conclusion as well as future works.

## 2 Related works

Reyes et al. [34] proposed a pruned problem transformation (PPT) method, which used the ReliefF filter measure to evaluate the features. In this method, the features are assessed multiple times according to the number of labels. Huang et al. [20] proposed an algorithm for multi-label learning. In this method, logical labels have been converted to numerical ones using manifold learning. These new labels show their degree of importance to the corresponding instances. Finally, the similarity between features and labels is calculated based on the Laplacian score Zhang et al. [41] offered a new multi-label feature selection algorithm that tries to find the features with the maximum predictive power based on manifold regularization. It calculated the label correlations locally on the original feature space and used $L_{2,1}$-norm regularization as the objective function. LRFS is a new multi-label feature selection which used conditional mutual information to calculate the correlation between features and labels to measure features relevancy and analyzed the differences between the two groups of labels. In this method, the labels are considered into two groups: dependent and independent labels [44].

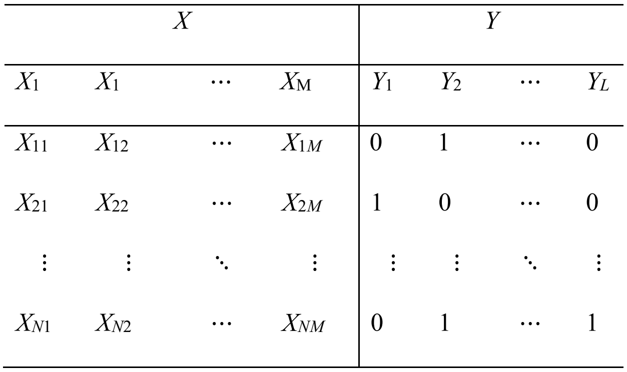
Recently, Paniri et al. [31] proposed a multi-label feature selection algorithm using Ant Colony Optimization (ACO). This method considers the maximum correlation between features and class labels as the relevancy metric and the minimum correlation between features as the redundancy measure. MLPSO [2] is a new approach which used Particle Swarm Optimization (PSO) to rank the features in multi-label data. MLCR [17] is a fast algorithm proposed for multi-label feature selection, which used a clustering ranking procedure to rank the features. In this method, the k-means algorithm and $L_2$-norm are considered as distribution and relevancy measures. Sun et al. [36] proposed a new multi-label feature selection through combining label correlations into the feature selection process and used a convex optimization to find an optimal feature set. Che et al. [4] applied a learning label strategy to categorize labels to relevant groups and then used local label correlation to select subsets of features for each label. Gonzalez-Lopez et al. (2020) proposed a distributed model to assign a score to each feature considering the aggregation of multiple labels based on mutual information. A multi-label feature selection method with missing labels was proposed by Wang et al. [37]. In data with missing labels, as the label space is not complete, some valuable features may not be considered. This method used the concept of feature interaction to address this problem.

In the field of graph theory, some efforts have been made for feature selection. Wang et al. [38] presented a factor graph model for unsupervised feature selection. In this method, the similarity between features was calculated and a message-passing algorithm was used to obtain the final importance score of features Zhou and Lin [45] offered a fine-grained recognition system constructed based on bipartite graph labels (BGL). They modeled the BGL via a convolutional neural network and performed the optimization by a backpropagation approach. Liu and Yang [26] proposed a bipartite edge prediction framework using the manifold structure of edges by a group product. Hashemi et al. [18] introduced a new feature selection algorithm for multi-label learning, called MGFS. This algorithm constructs a feature graph and then sets up a feature ranking system based on the PageRank algorithm.

## 3 Fundamental concepts

### 3.1 Multi-label classification

In a multi-label data, each training instance contains a feature vector $X_i = \left(X_{i1}, X_{i2}, \dots \dots, X_{iM}\right)$, where $M$ is the number of features and a binary label vector $Y_i = \left(Y_{i1}, Y_{i2}, \dots \dots, Y_{iL}\right)$ where $L$ refers to the number of labels. The multi-label learning is a process that learns a model of $N$ training samples and predicts the labels for new instances. Figure 1 illustrates a multi-label dataset structure.



| $X$ | | | | $Y$ | | | |
|---|---|---|---|---|---|---|---|
| $X_1$ | $X_1$ | $\cdots$ | $X_M$ | $Y_1$ | $Y_2$ | $\cdots$ | $Y_L$ |
| $X_{11}$ | $X_{12}$ | $\cdots$ | $X_{1M}$ | 0 | 1 | $\cdots$ | 0 |
| $X_{21}$ | $X_{22}$ | $\cdots$ | $X_{2M}$ | 1 | 0 | $\cdots$ | 0 |
| $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ |
| $X_{N1}$ | $X_{N2}$ | $\cdots$ | $X_{NM}$ | 0 | 1 | $\cdots$ | 1 |

**Fig. 1** Multi-label data structure

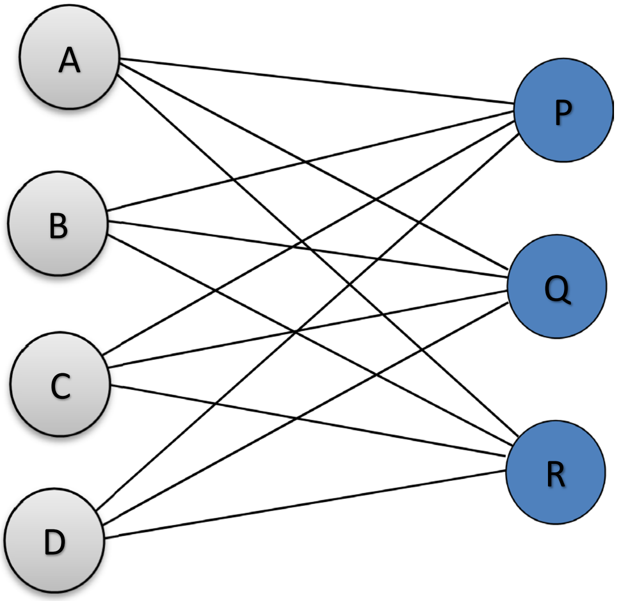


**Fig. 2** An example of a complete bipartite graph

### 3.2 Bipartite graph matching

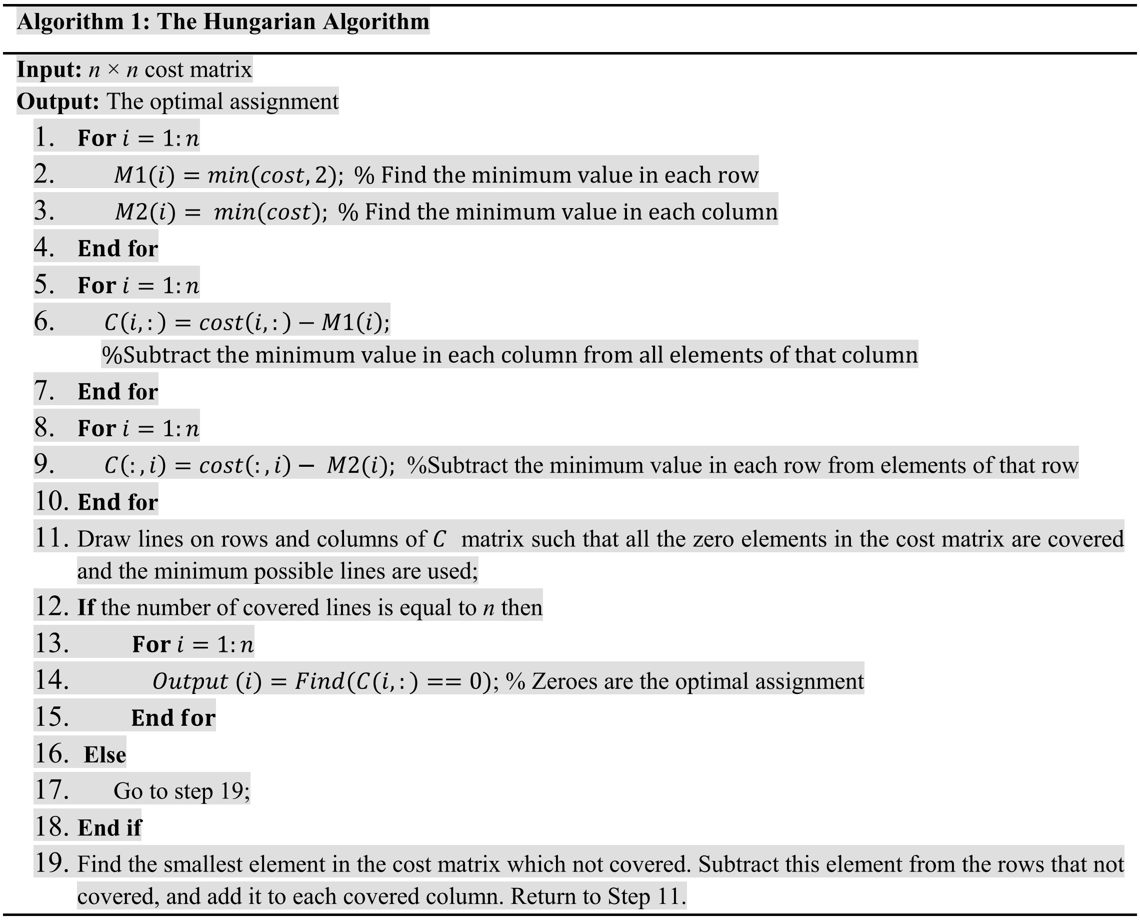
A bipartite graph is a type of graph whose vertices, $S$, can be decomposed into two independent sets $S_1$ and $S_2$. There is no edge between the vertices of a set, but every vertex of $S_1$ can connect to a vertex in $S_2$. If each vertex from $S_1$ is connected to all of the vertices from $S_2$ and vice versus, the graph is called a complete bipartite graph [40].

Figure 2 displays an example of a complete bipartite graph where $S_1 = \{A, B, C, D\}$ and $S_2 = \{P, Q, R\}$ represent the sets of vertices. Many real-world problems can be modeled as a bipartite graph, where graph matching is widely used to find an optimal solution for these problems. Graph matching is an optimization technique in the graph theory which is used to find an optimal correspondence between graphs vertices to minimize (maximize) their nodes and edge discrepancy (dependency) [39]. It tries to find a subset of edges such that no two edges in a set share a vertex. In pattern recognition problems, the graph matching is used to check the similarity between two graphs [27]. The Hungarian algorithm is a powerful method for bipartite graph matching.

Bipartite graph modeling is a powerful technique in machine learning and data mining applications. For example, in recommendation systems, user-item interactions could be modeled in a bipartite graph where the users and items are on the left and right of this graph. Another example is the citation network analysis, where the publications are the vertices on both sides of the graph. Many other applications of machine learning can be modeled as a bipartite graph including multi-label text classification, prerequisite linkage, question–answer mapping, and more [26].

### 3.3 The Hungarian algorithm

Kuhn [23] and Munkres [30] developed the Hungarian algorithm to solve the graph matching problem in polynomial time [15]. This algorithm is widely used for assignment problems such as machines to tasks, workers to jobs, soccer players to positions, etc. The Hungarian algorithm consists of the following four steps. The first two steps of the algorithm are executed only once, while the next steps are repeated until an optimal matching is found. The following algorithm applies to a given $n \times n$ cost matrix. Algorithm 1 shows the step-by-step procedure of the Hungarian algorithm.



**Algorithm 1: The Hungarian Algorithm**

```
Input: $n \times n$ cost matrix
Output: The optimal assignment
1.  For $i = 1:n$
2.      $M1(i) = min(cost, 2);$ % Find the minimum value in each row
3.      $M2(i) = min(cost);$ % Find the minimum value in each column
4.  End for
5.  For $i = 1:n$
6.      $C(i,:) = cost(i,:) - M1(i);$
        %Subtract the minimum value in each column from all elements of that column
7.  End for
8.  For $i = 1:n$
9.      $C(:,i) = cost(:,i) - M2(i);$ %Subtract the minimum value in each row from elements of that row
10. End for
11. Draw lines on rows and columns of $C$ matrix such that all the zero elements in the cost matrix are covered
    and the minimum possible lines are used;
12. If the number of covered lines is equal to $n$ then
13.         For $i = 1:n$
14.             $Output\ (i) = Find(C(i,:) == 0);$ % Zeroes are the optimal assignment
15.         End for
16.  Else
17.      Go to step 19;
18. End if
19. Find the smallest element in the cost matrix which not covered. Subtract this element from the rows that not
    covered, and add it to each covered column. Return to Step 11.
```

# 4 Proposed method

In this section, we discuss the proposed algorithm in detail. This algorithm is filter-based and has been designed specifically for multi-label data via a two-step procedure. Hence, as a first measure, a bipartite graph-based model is used where the subset of features is selected based on the Hungarian algorithm, and the members of each subgroup are sorted based on the weighted correlation distance with the other features in that subgroup.

## 4.1 Motivation

Graph-based modeling is a powerful tool in pattern recognition and machine learning as graphs have great presentation power. In multi-label data, each training sample corresponds to two vectors of feature and labels. Hence, we have imagined this multi-label data as a complete bipartite graph, where the first part is the feature set, and the second part is the label set. In this model, each feature is connected to a label with a weighted edge. Hence, to select the features in the following model, we used a matching algorithm to obtain the best subset of features concerning the correlation distance with the labels, which overall yields the lowest correlation distance with the label set, compared to all possible subsets of features. This creates a balance between the selected features. The Hungarian algorithm selects a feature for each label. Since the number of features and labels in a data set are not usually equal, we consider the features obtained by the Hungarian algorithm as the first subset of the selected features and remove them from the main feature set. Again, we form a bipartite graph with the remaining features to obtain the second subset of the attributes with the minimum correlation distance to the set of labels. This procedure continues until there is no feature left in the feature space. On the other hand, the features obtained at each stage of matching must be compared against each other to achieve the feature ranks, with these features selected based on their relevancy with class labels. Thus, as the redundancy measure, we consider a secondary measure which sorts the features obtained at each step according to their weighted correlation distance with the labels concerning each other. In this part, we applied the weight of labels to the correlation distance matrix. This means that the feature with the maximum correlation distance to other features is the most important in that subset. This secondary measure will cause a proper distribution between selected features, while the redundant features will obtain fewer ranks. In this secondary measure, we have also applied the weight of labels on the correlation distance matrix to consider the impact of the labels on the feature selection process.

## 4.2 Proposed algorithm

Algorithm 2 shows the step-by-step procedure of the proposed method. Now we aim to describe the steps of the algorithm. In the first step, we define an empty vector as a feature ranking vector to add the features into it. A multi-label data includes two features and label matrix, as with the structure below:

$$features = X = \begin{bmatrix} X_{11} & X_{12} & \cdots X_{1M} \\ X_{21} & X_{22} & \cdots X_{2M} \\ \vdots & \vdots & \ddots \vdots \\ X_{N1} & X_{N2} & \dots \; X_{NM} \end{bmatrix} labels = Y = \begin{bmatrix} Y_{11} & Y_{12} & \cdots Y_{1L} \\ Y_{21} & Y_{22} & \cdots Y_{2L} \\ \vdots & \vdots & \ddots \vdots \\ Y_{N1} & Y_{N2} & \dots \; Y_{NL} \end{bmatrix} \tag{1}$$

The rows of both matrices refer to the instances, while the columns are features and labels in that corresponding matrix.

In multi-label feature selection, the features with a less correlation distance with labels are better. So, in steps 2–7, we calculated the correlation distance between features and labels using Eq. 2 and obtained the Cordis matrix.

$$CD(X, Y) = 1 - \frac{cov(X, Y)}{\sqrt{var(X)var(Y)}}, \tag{2}$$

where, X refers to features and Y represents the labels; the $cov(X, Y)$ calculates the covariance of feature $X$ and label $Y$, and $var(X)$ determines the variance of the feature x. Finally, $var(Y)$ shows the variance of label Y. Thus, the correlation distance matrix (Cordis) will be obtained as follows:

$$Cordis = \begin{bmatrix} CD(X_1, Y_1) & CD(X_1, Y_2) & \cdots CD(X_1, Y_L) \\ CD(X_2, Y_1) & CD(X_2, Y_2) & \cdots CD(X_2, Y_L) \\ \vdots & \vdots & \ddots \vdots \\ CD(X_M, Y_1) & CD(X_M, Y_2) & \dots \; CD(X_M, Y_L) \end{bmatrix} \tag{3}$$

where,$CD(X_i, Y_j)$, is the correlation distance between feature $i$ and label $j$.

**Algorithm 2: BMFS - The proposed bipartite matching-based feature selection for multi-label learning**

**Input:** Feature data matrix $\boldsymbol{X} = \{\boldsymbol{X_1}, \boldsymbol{X_2}, \ldots\ldots, \boldsymbol{X_M}\}$ , label data matrix $\boldsymbol{Y} = \{\boldsymbol{Y_1}, \boldsymbol{Y_2}, \ldots\ldots, \boldsymbol{Y_L}\}$, $N$ samples

**Output:** Feature ranking vector $w$

```
1. $w = \emptyset$;
2. // Calculate correlation distance between each feature and label
3. For $i = 1{:}M$
4.     For $j = 1{:}L$
5.         $Cordis(i,j) = correlation\_distance(i,j)$;
6.     End for
7. End for
8. // Calculate the weight of labels
9.  For $i = 1{:}L$
10.        $ff = length\big(find(Y(:,i) == 1)\big)$;
11.        $WL(i) = ff(i)/N$;
12. End for
13. $WCD = WL.*\ Cordis$; //calculate the weighted correlation distance
14. EDM = calculate the Euclidean distance between features according to WCD matrix
15. CDV = sum(EDM); //calculate the sum of correlation distance for each feature
16. Build a bipartite graph of features and labels considering Cordis as edge weights;
17. While $length\ (X) > L$ Do
18.         $Square\ Cordis\ matrix$;
19.         $C = Hungarian(Cordis);//using\ algorithm$ 1
20.         $eliminate\ the\ columns\ that\ add\ to\ make\ matrix\ square$;
21.         $C = Sort\ vector\ C\ based\ on\ CDV\ in\ descend\ order$;
22.         $w = w \cup C$;
23.         $X = X - C$;
24.         C=$\emptyset$;
25. End while
26. If $isempty(X) == 0$
27.     Sort X based on CDV in descend order;
28.     $w = w \cup X$   ;
29. End if
30. $w$ = feature ranking vector
```

In steps 8–12, we calculated the weight of labels. To do this, for each label, we calculated the number of instances in the label matrix, which equals to 1, and then divided it by the number of instances. The vector $WL$ is obtained as follows:

$$WL = \begin{bmatrix} WL_1 \\ WL_2 \\ \vdots \\ \vdots \\ WL_L \end{bmatrix} \tag{4}$$

**Fig. 3** An Example for steps of the proposed method

$$features = \begin{bmatrix} 0.0347 & 0.0897 & 0.0912 & 73.3024 & 6.2152 \\ 0.0814 & 0.2727 & 0.0857 & 62.5844 & 3.1832 \\ 0.1105 & 0.2736 & 0.0844 & 65.2353 & 2.7950 \end{bmatrix} \quad labels = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

a Sample dataset

$$Cordis = \begin{bmatrix} 0.8671 & 1.1329 & 1.9248 \\ 0.5037 & 1.4963 & 2.0000 \\ 1.3359 & 0.6641 & 0.0163 \\ 1.6913 & 0.3087 & 0.0286 \\ 1.4075 & 0.5925 & 0.0054 \end{bmatrix}$$

b Correlation Distance Matrix

$$WL = \begin{bmatrix} 0.3333 \\ 0.6666 \\ 0.3333 \end{bmatrix}$$

c Weight of Labels

$$WCD = \begin{bmatrix} 0.2890 & 0.7552 & 0.6416 \\ 0.1679 & 0.9975 & 0.6667 \\ 0.4453 & 0.4427 & 0.0054 \\ 0.5638 & 0.2058 & 0.0095 \\ 0.4692 & 0.3950 & 0.0018 \end{bmatrix}$$

d Weighted Correlation Distance

$$EDM = \begin{bmatrix} 0 & 0.2721 & 0.7258 & 0.8814 & 0.7560 \\ 0.2721 & 0 & 0.9066 & 1.1025 & 0.9465 \\ 0.7258 & 0.9066 & 0 & 0.2650 & 0.0535 \\ 0.8814 & 1.1025 & 0.2650 & 0 & 0.2117 \\ 0.7560 & 0.9465 & 0.0535 & 0.2117 & 0 \end{bmatrix}$$

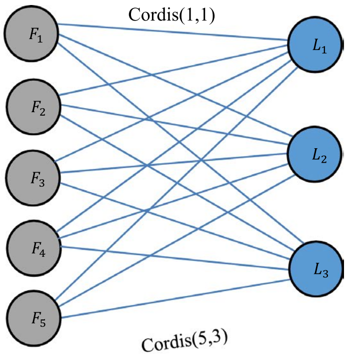
e Euclidean Distance Matrix

$$CDV = \begin{bmatrix} 2.6353 \\ 3.2277 \\ 1.9509 \\ 2.4605 \\ 1.9678 \end{bmatrix}$$

f Correlation Distance Vector

$$\begin{bmatrix} 0.3634 & 0.8242 & 1.9194 & 0.0000 & 0.0000 \\ 0.0000 & 1.1876 & 1.9946 & 0.0000 & 0.0000 \\ 0.8322 & 0.3554 & 0.0109 & 0.0000 & 0.0000 \\ 1.1876 & 0.0000 & 0.2806 & 0.0000 & 0.0000 \\ 0.9038 & 0.2838 & 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$$

**g** Sample Cost Matrix Subtract column minima



h FLG Graph

$$\begin{bmatrix} 0.3634 & 0.8242 & 1.9194 & 0.0000 & 0.0000 \\ 0.0000 & 1.1876 & 1.9946 & 0.0000 & 0.0000 \\ 0.8322 & 0.3554 & 0.0109 & 0.0000 & 0.0000 \\ 1.1876 & 0.0000 & 0.2806 & 0.0000 & 0.0000 \\ 0.9038 & 0.2838 & 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$$

**i** Cost Matrix Cover Zeroes

$$\begin{bmatrix} 0.3634 & 0.8242 & 1.9194 & 0.0000 & 0.0000 \\ 0.0000 & 1.1876 & 1.9946 & 0.0000 & 0.0000 \\ 0.8322 & 0.3554 & 0.0109 & 0.0000 & 0.0000 \\ 1.1876 & 0.0000 & 0.2806 & 0.0000 & 0.0000 \\ 0.9038 & 0.2838 & 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$$

j Optimal Solution

$$\begin{bmatrix} 0.8671 & 1.1329 & 1.9248 \\ 0.5037 & 1.4963 & 2.0000 \\ 1.3359 & 0.6641 & 0.0163 \\ 1.6913 & 0.3087 & 0.0286 \\ 1.4075 & 0.5925 & 0.0054 \end{bmatrix}$$

k Features with maximum matching

$$W = [F2, F4, F5]$$

l) Sort the selected features in the first match

$$W = [F2, F4, F5, F1, F3]$$

m Feature ranking vector

In step 13, we applied the weight of labels to the correlation distance matrix and obtained a weighted correlation distance (WCD) matrix. Each value in *WCD* is the product of the corresponding value in Cordis and WL.

$$WCD = \begin{bmatrix} Cordis\left(X_1, Y_1\right) * WL_1 & Cordis\left(X_1, Y_2\right) * WL_2 & \cdots & Cordis(X_1, Y_L) * WL_L \\ Cordis(X_2, Y_1) * WL_1 & Cordis(X_2, Y_2) * WL_2 & \cdots & Cordis(X_2, Y_L) * WL_L \\ \vdots & \vdots & \ddots & \vdots \\ Cordis(X_M, Y_1) * WL_1 & Cordis(X_M, Y_2) * WL_2 & \dots & Cordis(X_M, Y_L) * WL_L \end{bmatrix} \tag{5}$$

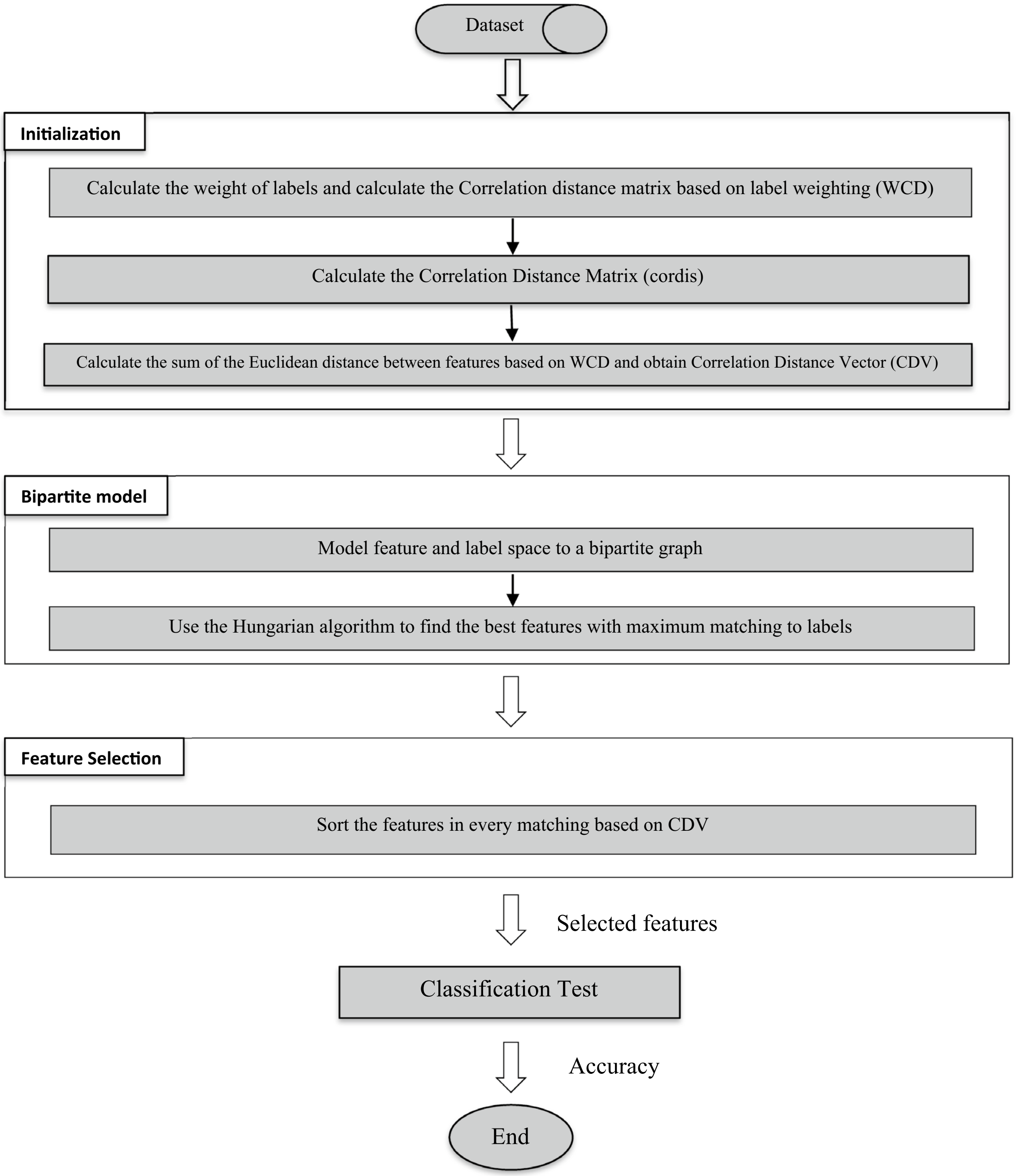

Dataset
Initialization
Calculate the weight of labels and calculate the Correlation distance matrix based on label weighting (WCD)
Calculate the Correlation Distance Matrix (cordis)
Calculate the sum of the Euclidean distance between features based on WCD and obtain Correlation Distance Vector (CDV)
Bipartite model
Model feature and label space to a bipartite graph
Use the Hungarian algorithm to find the best features with maximum matching to labels
Feature Selection
Sort the features in every matching based on CDV
Selected features
Classification Test
Accuracy
End

**Fig. 4** Graphical abstract of the proposed method

**Table 1** Description of the datasets

| Dataset | Samples | Features | Labels | Type | Domain |
|---|---|---|---|---|---|
| Yeast | 2417 | 130 | 14 | Numeric | Biology |
| Corel5k | 5000 | 499 | 374 | Nominal | Image |
| Medical | 978 | 1449 | 45 | Nominal | Text |
| Enron | 1702 | 1001 | 53 | Nominal | Text |
| Image | 2000 | 294 | 5 | Numeric | Image |
| Scene | 2407 | 294 | 6 | Numeric | Image |
| Bibtex | 7395 | 1836 | 159 | Nominal | Text |

$$ED(p,q) = \sqrt{\sum_{k=1}^{L} \left(p_k - q_k\right)^2}, \tag{6}$$

where $ED(p,q)$ calculates the Euclidean distance between features p and q, and $L$ is the number of labels. Using Eq. 6, the Euclidean Distance Matrix (EDM) is obtained as follows:

$$EDM = \begin{bmatrix} ED(WCD_1, WCD_1) & ED(WCD_1, WCD_2) & \cdots & ED(WCD_1, WCD_M) \\ ED(WCD_2, WCD_1) & ED(WCD_2, WCD_2) & \cdots & ED(WCD_2, WCD_M) \\ \vdots & \vdots & \ddots & \vdots \\ ED(WCD_M, WCD_1) & ED(WCD_M, WCD_2) & \ldots & ED(WCD_M, WCD_M) \end{bmatrix} \tag{7}$$

For example, $ED(WCD_i, WCD_j)$ refers to the Euclidean distance between features *i* and *j* of *WCD*. At step 15, we calculate the Correlation Distance Vector (CDV). Each element of this vector is obtained by the sum of the rows of EDM and shows the overall difference of each feature comparing to reminder of features based on the correlation distance metric. Equation 8 shows the structure of CDV.

$$CDV = \begin{bmatrix} Sum(EDM(1,1), EDM(1,2), \ldots ., EDM(1,M)) \\ Sum(EDM(2,1), EDM(2,2), \ldots ., EDM(2,M)) \\ \vdots \\ \vdots \\ Sum(EDM(M,1), EDM(M,2), \ldots ., EDM(M,M)) \end{bmatrix} \tag{8}$$

Now we construct our bipartite graph called Feature-Label Graph (FLG) according to CDM, as step 16 of the algorithm. Then, in steps 17–25, the Hungarian algorithm will be used to find the first matching subset of features, where these features will be sorted according to their value in CDV in descending order. The sorting features will be added in a vector called w and then eliminate from the feature set. This procedure will continue until the feature set is empty.

In Fig. 3 we used a sample data set with 5 features, 3 labels, and 3 instances to show the steps of the proposed algorithm with an example. The block diagram of the proposed method is depicted in Fig. 4.

## 5 Experimental studies

In this section, the performance of the proposed method is compared with those of five multi-label feature selection algorithms: PPT-MI [7], LRFS [44], MDFS [41], PPT-ReliefF [34] and MCLS [20]. To this end, we ran all the algorithms on 7 real-world datasets.

### 5.1 Datasets

To measure the performance of all methods, we used 7 real-world datasets obtained from the Mulan[1] repository. Table 1 describes these datasets.

### 5.2 Performance evaluation criteria

- The performance of BMFS and comparison of methods are evaluated based on the run-time of algorithms on four multi-label evaluation metrics: hamming loss, accuracy, average-precision, and one-error.

  If we consider $\boldsymbol{T} = \left\{ \left(\boldsymbol{x_i}, \boldsymbol{Y_i}\right), \boldsymbol{i} = 1, \ldots \ldots ., \boldsymbol{p} \right\}$ as the test set, $\boldsymbol{Y_i} \subseteq \boldsymbol{L}$ as the actual label subset, and $\boldsymbol{Z_i} \subseteq \boldsymbol{L}$ as the predicted label set to $\boldsymbol{x_i}$. Also, let $f(x, y)$ denotes the score assigned to label $y$ for sample $x$. Thus, the evaluation metrics are defined as follows [42]:
- Hamming Loss: It measures how many times an instance-label pair is not truly classified [5].

$$\boldsymbol{Hammingloss(T)} = \frac{1}{\boldsymbol{p}} \sum_{i=1}^{\boldsymbol{p}} \frac{|\boldsymbol{Y_i \Delta Z_i}|}{|\boldsymbol{L}|}, \tag{9}$$

  where, Δ is the symmetric difference between two sets.
- Accuracy: Accuracy calculates the number of correctly predicted labels among all actual and predicted labels.

$$\boldsymbol{Accuracy(T)} = \frac{1}{\boldsymbol{p}} \sum_{i=1}^{\boldsymbol{p}} \frac{|\boldsymbol{Y_i} \cap \boldsymbol{Z_i}|}{|\boldsymbol{Y_i} \cup \boldsymbol{Z_i}|}. \tag{10}$$

- Average_Precision: Average-Precision (Avg-Pre) calculates the average fraction of relevant labels which are ranked higher than a specific label.

$$\boldsymbol{Avg - pre} = \frac{1}{p} \sum_{i=1}^{p} \frac{1}{|Y_i|} \sum_{y \in Y_i} \frac{\left\{ y' | rank_f\left(x_i, y'\right) \le rank_f\left(x_i, y\right) . y' \in Y_i \right\}}{rank_f\left(x_i, y\right)} \tag{11}$$

- One_Error: One-Error calculates how many times the top-ranked label is irrelevant.

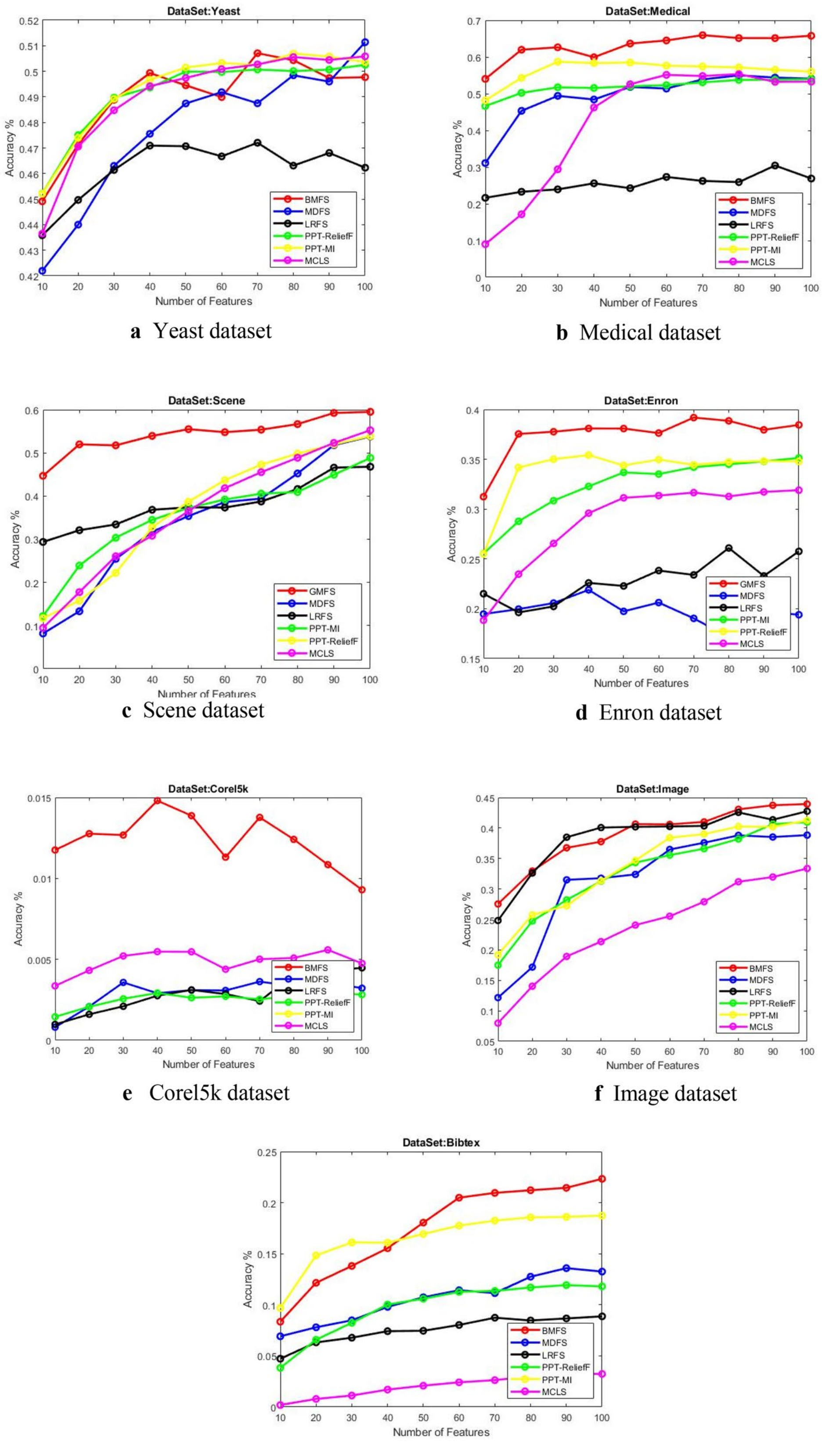
$$\boldsymbol{One - error(f)} = \frac{1}{p} \sum_{i=1}^{p} \left[ \left[ \underset{y \in Y}{\operatorname{argmax}} f\left(x_i, y\right) \right] \notin Y_i \right] \tag{12}$$

## 6 Results

As mentioned earlier, the performance of the proposed method is compared with five multi-label feature selection algorithms, including PPT-MI [7], LRFS [44], MDFS [41], PPT-ReliefF [34], and MCLS [20]. All the parameter

[1] https://mulan.sourceforge.net/datasets.html

**Fig. 5** The results based on accuracy metric



**a** Yeast dataset

**b** Medical dataset

**c** Scene dataset

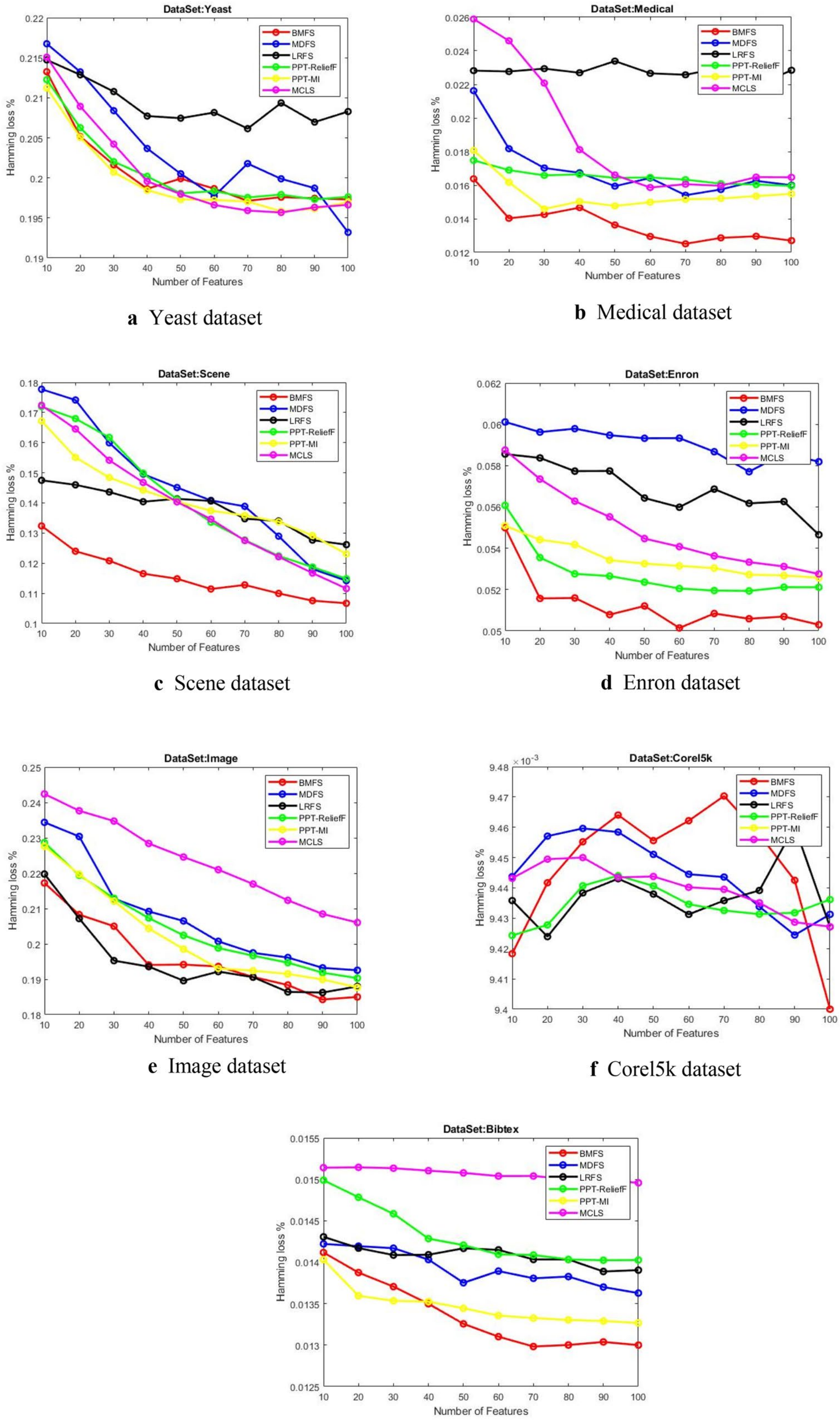
**d** Enron dataset

**e** Corel5k dataset

**f** Image dataset

**g** Bibtex dataset

**Fig. 6** The results based on hamming loss metric



**a** Yeast dataset

**b** Medical dataset

**c** Scene dataset

**d** Enron dataset

**e** Image dataset

**f** Corel5k dataset

**g** Bibtex dataset

**Table 2** Average-precision and one-error for yeast dataset

| M/feat | Average-precision | | | | | | One-error | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS |
| 10 | 0.7333 | 0.7192 | 0.7347 | 0.7360 | **0.7398** | 0.7393 | 0.2529 | 0.2544 | 0.2440 | 0.2490 | 0.2422 | **0.2381** |
| 20 | 0.7406 | 0.7278 | 0.7344 | 0.7457 | 0.7456 | **0.7481** | 0.2536 | 0.2587 | 0.2517 | 0.2406 | 0.2455 | **0.2344** |
| 30 | 0.7490 | 0.7376 | 0.7338 | 0.7530 | 0.7515 | **0.7538** | 0.2431 | 0.2451 | 0.2578 | 0.2346 | 0.2408 | **0.2331** |
| 40 | **0.7570** | 0.7426 | 0.7410 | 0.7554 | 0.7540 | 0.7491 | **0.2230** | 0.2443 | 0.2459 | 0.2347 | 0.2373 | 0.2396 |
| 50 | 0.7541 | 0.7450 | 0.7444 | 0.7582 | 0.7551 | **0.7607** | 0.2375 | 0.2446 | 0.2386 | 0.2325 | 0.2396 | **0.2277** |
| 60 | 0.7545 | 0.7558 | 0.7364 | 0.7589 | **0.7665** | 0.7607 | 0.2377 | 0.2397 | 0.2511 | 0.2348 | 0.2379 | **0.2225** |
| 70 | 0.7588 | 0.7438 | 0.7407 | 0.7592 | 0.7559 | **0.7620** | 0.2353 | 0.2539 | 0.2524 | 0.2345 | 0.2399 | **0.2314** |
| 80 | 0.7553 | 0.7539 | 0.7389 | 0.7609 | 0.7547 | **0.7623** | 0.2413 | 0.2371 | 0.2551 | 0.2343 | 0.2436 | **0.2302** |
| 90 | 0.7576 | 0.7449 | 0.7385 | 0.7601 | 0.7543 | **0.7605** | 0.2394 | 0.2516 | 0.2532 | 0.2367 | 0.2435 | **0.2346** |
| 100 | 0.7494 | **0.7622** | 0.7380 | 0.7594 | 0.7561 | 0.7602 | **0.2332** | 0.2360 | 0.2511 | 0.2348 | 0.2410 | 0.2351 |
| Mean | 0.7509 | 0.7433 | 0.7380 | 0.7547 | 0.7523 | **0.7567** | 0.2407 | 0.2464 | 0.2501 | 0.2366 | 0.2411 | **0.2327** |

**Table 3** Average-precision and one-error for medical dataset

| M/feat | Average-precision | | | | | | One-error | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS |
| 10 | **0.7302** | 0.6431 | 0.5189 | 0.7034 | 0.7060 | 0.4895 | **0.3294** | 0.4367 | 0.5957 | 0.3579 | 0.3520 | 0.6284 |
| 20 | **0.8160** | 0.7099 | 0.5386 | 0.7578 | 0.7430 | 0.5768 | **0.2402** | 0.3587 | 0.5652 | 0.2977 | 0.3109 | 0.5307 |
| 30 | **0.8202** | 0.7245 | 0.5388 | 0.7919 | 0.7545 | 0.6733 | **0.2327** | 0.3395 | 0.5652 | 0.2577 | 0.2976 | 0.4196 |
| 40 | **0.8208** | 0.7328 | 0.5545 | 0.7945 | 0.7583 | 0.7588 | **0.2304** | 0.3344 | 0.5512 | 0.2513 | 0.2945 | 0.3120 |
| 50 | **0.8360** | 0.7506 | 0.5529 | 0.7968 | 0.7616 | 0.7863 | **0.2066** | 0.3127 | 0.5471 | 0.2496 | 0.2922 | 0.2762 |
| 60 | **0.8428** | 0.7560 | 0.5606 | 0.7939 | 0.7681 | 0.7990 | **0.1995** | 0.2992 | 0.5345 | 0.2540 | 0.2863 | 0.2556 |
| 70 | **0.8554** | 0.7735 | 0.5585 | 0.7916 | 0.7702 | 0.7995 | **0.1818** | 0.2864 | 0.5366 | 0.2570 | 0.2835 | 0.2565 |
| 80 | **0.8462** | 0.7763 | 0.5659 | 0.7898 | 0.7729 | 0.7934 | **0.1923** | 0.2730 | 0.5327 | 0.2577 | 0.2811 | 0.2606 |
| 90 | **0.8352** | 0.7828 | 0.5869 | 0.7878 | 0.7731 | 0.7897 | **0.2054** | 0.2705 | 0.5033 | 0.2595 | 0.2815 | 0.2692 |
| 100 | **0.8471** | 0.7776 | 0.5752 | 0.7839 | 0.7795 | 0.7892 | **0.1916** | 0.2717 | 0.5182 | 0.2698 | 0.2699 | 0.2687 |
| Mean | **0.8250** | 0.7427 | 0.5551 | 0.7792 | 0.7587 | 0.7256 | **0.2210** | 0.3183 | 0.5450 | 0.2707 | 0.2949 | 0.3477 |

**Table 4** Average-precision and one-error for scene dataset

| M/feat | Average-precision | | | | | | One-error | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS |
| 10 | **0.7668** | 0.5995 | 0.7136 | 0.6098 | 0.5920 | 0.5814 | **0.3810** | 0.6244 | 0.4727 | 0.5918 | 0.6188 | 0.6286 |
| 20 | **0.7883** | 0.6317 | 0.7284 | 0.6925 | 0.6230 | 0.6400 | **0.3493** | 0.5744 | 0.4513 | 0.4962 | 0.5777 | 0.5522 |
| 30 | **0.7952** | 0.6880 | 0.7383 | 0.7229 | 0.6580 | 0.6776 | **0.3378** | 0.5046 | 0.4348 | 0.4552 | 0.5281 | 0.5004 |
| 40 | **0.8062** | 0.7233 | 0.7435 | 0.7382 | 0.7079 | 0.7096 | **0.3224** | 0.5558 | 0.4236 | 0.4324 | 0.4580 | 0.4591 |
| 50 | **0.8074** | 0.7359 | 0.7486 | 0.7471 | 0.7355 | 0.7335 | **0.3221** | 0.4372 | 0.4169 | 0.4190 | 0.4191 | 0.4274 |
| 60 | **0.8122** | 0.7499 | 0.7496 | 0.7538 | 0.7547 | 0.7526 | **0.3104** | 0.4155 | 0.4151 | 0.4088 | 0.3915 | 0.3963 |
| 70 | **0.8124** | 0.7532 | 0.7592 | 0.7590 | 0.7679 | 0.7694 | **0.3101** | 0.4093 | 0.4006 | 0.4010 | 0.3713 | 0.3728 |
| 80 | **0.8194** | 0.7772 | 0.7664 | 0.7669 | 0.7836 | 0.7844 | **0.2996** | 0.3725 | 0.3896 | 0.3886 | 0.3486 | 0.3510 |
| 90 | **0.8233** | 0.8043 | 0.7765 | 0.7773 | 0.7925 | 0.7991 | **0.2964** | 0.3298 | 0.3734 | 0.3750 | 0.3358 | 0.3280 |
| 100 | **0.8237** | 0.8124 | 0.7851 | 0.7927 | 0.8006 | 0.8119 | **0.2924** | 0.3149 | 0.3583 | 0.3498 | 0.3226 | 0.3096 |
| Mean | **0.8053** | 07275 | 0.7509 | 0.7360 | 0.7216 | 0.7260 | **0.3221** | 0.4438 | 0.4136 | 0.4318 | 0.4372 | 0.4326 |

**Table 5** Average-precision and one-error for Enron dataset

| M/feat | Average-precision | | | | | | One-error | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS |
| 10 | **0.6001** | 0.5300 | 0.5477 | 0.6063 | 0.5895 | 0.5459 | 0.3310 | 0.4137 | 0.4040 | **0.3179** | 0.3426 | 0.4182 |
| 20 | **0.6452** | 0.5368 | 0.5587 | 0.6107 | 0.6129 | 0.5761 | **0.2680** | 0.4086 | 0.4051 | 0.3098 | 0.3022 | 0.3777 |
| 30 | **0.6454** | 0.5343 | 0.5664 | 0.6140 | 0.6172 | 0.5883 | **0.2772** | 0.4115 | 0.4088 | 0.2993 | 0.2948 | 0.3515 |
| 40 | **0.6532** | 0.5364 | 0.5785 | 0.6184 | 0.6196 | 0.5962 | **0.2681** | 0.4123 | 0.3938 | 0.2927 | 0.2943 | 0.3326 |
| 50 | **0.6510** | 0.5495 | 0.5854 | 0.6197 | 0.6230 | 0.6051 | **0.2681** | 0.4148 | 0.3793 | 0.2879 | 0.2923 | 0.3243 |
| 60 | **0.6525** | 0.5460 | 0.5898 | 0.6201 | 0.6260 | 0.6085 | **0.2661** | 0.4148 | 0.3742 | 0.2873 | 0.2924 | 0.3168 |
| 70 | **0.6538** | 0.5623 | 0.5920 | 0.6224 | 0.6291 | 0.6129 | **0.2661** | 0.4009 | 0.3769 | 0.2870 | 0.2885 | 0.3116 |
| 80 | **0.6574** | 0.5507 | 0.5958 | 0.6245 | 0.6301 | 0.6157 | **0.2601** | 0.4207 | 0.3722 | 0.2807 | 0.2887 | 0.3184 |
| 90 | **0.6514** | 0.5644 | 0.6024 | 0.6257 | 0.6294 | 0.6172 | **0.2680** | 0.4038 | 0.3571 | 0.2835 | 0.2918 | 0.3058 |
| 100 | **0.6563** | 0.5558 | 0.5998 | 0.6264 | 0.6285 | 0.6192 | **0.2609** | 0.4192 | 0.3532 | 0.2836 | 0.2901 | 0.3001 |
| Mean | **0.6466** | 0.5476 | 0.5817 | 0.6188 | 0.6205 | 0.5986 | **0.2734** | 0.4120 | 0.3825 | 0.2930 | 0.2978 | 0.3347 |

**Table 6** Average-precision and one-error for Image dataset

| M/feat | Average-precision | | | | | | One-error | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS |
| 10 | **0.7015** | 0.6500 | 0.6955 | 0.6616 | 0.6546 | 0.6077 | **0.4571** | 0.5391 | 0.4716 | 0.5196 | 0.5281 | 0.5963 |
| 20 | 0.7215 | 0.6716 | **0.7251** | 0.6927 | 0.6914 | 0.6470 | 0.4291 | 0.5090 | **0.4268** | 0.4736 | 0.4747 | 0.5404 |
| 30 | 0.7329 | 0.7186 | **0.7465** | 0.7092 | 0.7092 | 0.6585 | 0.4116 | 0.4365 | **0.3907** | 0.4508 | 0.4474 | 0.5222 |
| 40 | **0.7488** | 0.7270 | 0.7482 | 0.7272 | 0.7192 | 0.6768 | **0.3848** | 0.4230 | 0.3880 | 0.4215 | 0.4331 | 0.4954 |
| 50 | 0.7487 | 0.7322 | **0.7611** | 0.7376 | 0.7312 | 0.6845 | 0.3841 | 0.4157 | **0.3685** | 0.4063 | 0.4138 | 0.4849 |
| 60 | 0.7504 | 0.7398 | **0.7592** | 0.7468 | 0.7400 | 0.6958 | 0.3823 | 0.4016 | **0.3710** | 0.3895 | 0.3996 | 0.4675 |
| 70 | **0.7617** | 0.7443 | 0.7616 | 0.7516 | 0.7452 | 0.7029 | **0.3665** | 0.3985 | 0.3674 | 0.3832 | 0.3909 | 0.4588 |
| 80 | 0.7594 | 0.7467 | **0.7649** | 0.7531 | 0.7499 | 0.7119 | 0.3714 | 0.3926 | **0.3631** | 0.3813 | 0.3854 | 0.4408 |
| 90 | 0.7620 | 0.7505 | **0.7670** | 0.7546 | 0.7558 | 0.7180 | 0.3638 | 0.3857 | **0.3566** | 0.3789 | 0.3741 | 0.4320 |
| 100 | **0.7661** | 0.7529 | 0.7642 | 0.7611 | 0.7601 | 0.7256 | **0.3556** | 0.3814 | 0.3658 | 0.3673 | 0.3677 | 0.4227 |
| Mean | 0.7453 | 0.7234 | **0.7493** | 0.7295 | 0.7257 | 0.6829 | 0.3906 | 0.4283 | **0.3869** | 0.4172 | 0.4215 | 0.4861 |

**Table 7** Average-precision and one-error for Corel5k dataset

| M/feat | Average-precision | | | | | | One-error | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS |
| 10 | **0.2148** | 0.2095 | 0.2147 | 0.2162 | 0.2091 | 0.2162 | **0.7587** | 0.7753 | 0.7722 | 0.7681 | 0.7759 | 0.7681 |
| 20 | **0.2250** | 0.2132 | 0.2163 | 0.2172 | 0.2102 | 0.2172 | **0.7395** | 0.7728 | 0.7710 | 0.7677 | 0.7764 | 0.7677 |
| 30 | **0.2304** | 0.2150 | 0.2172 | 0.2179 | 0.2104 | 0.2179 | **0.7311** | 0.7718 | 0.7681 | 0.7651 | 0.7787 | 0.7651 |
| 40 | **0.2320** | 0.2164 | 0.2174 | 0.2186 | 0.2111 | 0.2186 | **0.7325** | 0.7724 | 0.7789 | 0.7659 | 0.7785 | 0.7659 |
| 50 | **0.2345** | 0.2182 | 0.2199 | 0.2185 | 0.2113 | 0.2185 | **0.7312** | 0.7731 | 0.7692 | 0.7683 | 0.7793 | 0.7683 |
| 60 | **0.2363** | 0.2192 | 0.2192 | 0.2193 | 0.2117 | 0.2193 | **0.7212** | 0.7701 | 0.7744 | 0.7679 | 0.7823 | 0.7679 |
| 70 | **0.2372** | 0.2211 | 0.2214 | 0.2208 | 0.2128 | 0.2208 | **0.7240** | 0.7682 | 0.7694 | 0.7691 | 0.7778 | 0.7691 |
| 80 | **0.2385** | 0.2216 | 0.2225 | 0.2205 | 0.2141 | 0.2205 | **0.7279** | 0.7731 | 0.7686 | 0.7711 | 0.7797 | 0.7711 |
| 90 | **0.2358** | 0.2233 | 0.2233 | 0.2215 | 0.2145 | 0.2215 | **0.7374** | 0.7706 | 0.7685 | 0.7697 | 0.7790 | 0.7697 |
| 100 | **0.2387** | 0.2250 | 0.2233 | 0.2222 | 0.2167 | 0.2222 | **0.7304** | 0.7681 | 0.7662 | 0.7669 | 0.7756 | 0.7669 |
| Mean | **0.2323** | 0.2183 | 0.2195 | 0.2193 | 0.2122 | 0.2193 | **0.7335** | 0.7716 | 0.7707 | 0.7680 | 0.7783 | 0.7680 |

**Table 8** Average-precision and one-error for Bibtex dataset

| M/feat | Average-precision | | | | | | One-error | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS | BMFS | MDFS | LRFS | PPT-MI | PPT-reliefF | MCLS |
| 10 | 0.2200 | 0.2231 | 0.2091 | **0.2345** | 0.1922 | 0.1575 | 0.7356 | 0.7406 | 0.7601 | **0.7166** | 0.7879 | 0.8333 |
| 20 | 0.2841 | 0.2442 | 0.2270 | **0.3006** | 0.2162 | 0.1686 | 0.6667 | 0.7165 | 0.7333 | **0.6398** | 0.7610 | 0.8257 |
| 30 | **0.3360** | 0.2701 | 0.2400 | 0.3189 | 0.2380 | 0.1771 | **0.6049** | 0.6853 | 0.7203 | 0.6230 | 0.7366 | 0.8181 |
| 40 | **0.3593** | 0.2869 | 0.2441 | 0.3406 | 0.2540 | 0.1858 | **0.5848** | 0.6636 | 0.7104 | 0.5984 | 0.7136 | 0.8082 |
| 50 | **0.3927** | 0.2983 | 0.2427 | 0.3560 | 0.2609 | 0.1947 | **0.5428** | 0.6526 | 0.7126 | 0.5798 | 0.7031 | 0.7985 |
| 60 | **0.4245** | 0.3073 | 0.2486 | 0.3684 | 0.2675 | 0.2035 | **0.5055** | 0.6415 | 0.7090 | 0.5666 | 0.6923 | 0.7884 |
| 70 | **0.4401** | 0.3008 | 0.2584 | 0.3761 | 0.2711 | 0.2057 | **0.4924** | 0.6500 | 0.6916 | 0.5603 | 0.6874 | 0.7829 |
| 80 | **0.4225** | 0.3100 | 0.2575 | 0.3818 | 0.2746 | 0.2097 | **0.4884** | 0.6383 | 0.6942 | 0.5546 | 0.6818 | 0.7787 |
| 90 | **0.4674** | 0.3238 | 0.2675 | 0.3873 | 0.2787 | 0.2143 | **0.4711** | 0.6198 | 0.6864 | 0.5484 | 0.6752 | 0.7722 |
| 100 | **0.4730** | 0.3227 | 0.2698 | 0.3908 | 0.2789 | 0.2170 | **0.4666** | 0.6291 | 0.6857 | 0.5458 | 0.6757 | 0.7706 |
| Mean | **0.3850** | 0.2887 | 0.2465 | 0.3455 | 0.2532 | 0.1934 | **0.5559** | 0.6637 | 0.7104 | 0.5933 | 0.7114 | 0.7977 |

**Table 9** The win/tie/loss results of BMFS against the other methods based on the Friedman test

| BMFS against | Metrics | | | |
|---|---|---|---|---|
| | Accuracy | Hamming loss | Average-precision | One-error |
| MDFS | 7/0/0 | 5/2/0 | 7/0/0 | 7/0/0 |
| LRFS | 6/1/0 | 5/2/0 | 5/2/0 | 6/1/0 |
| PPT-MI | 5/2/0 | 5/1/1 | 6/0/1 | 6/0/1 |
| PPT-ReliefF | 6/1/0 | 5/1/1 | 6/1/0 | 6/1/0 |
| MCLS | 6/1/0 | 5/1/1 | 6/0/1 | 6/0/1 |
| Total | 30/5/0 | 25/7/3 | 30/3/2 | 31/2/2 |

values for comparing algorithms are set based on the recommendations by that corresponding paper.

We used the well-known multi-label classifier, ML-KNN [43], for the learning process. The number of neighbors is set to 10 for each classification procedure. For each test, randomly 60% of the samples are chosen as training data, and the remaining 40% considered as test data. The reported results are averaged results achieved by 10 separate runs for each method. For testing each method, as the number of selected features is determined by the user, we change the size of features subset from 10 to 100, which results in 100 different runs on each dataset. In the proposed method, the number of features is determined by the user. Figures 5 and 6 report the obtained results of the proposed method and other methods in terms of accuracy, hamming loss. Tables 2, 3, 4,  5,6, 7, 8 report the results in terms of Average-precision and One-error. In these tables, the bold values are the best value among all algorithms based on the classification metrics. The results show the superiority of the proposed method in most cases and based on all four metrics.

The obtained results of BMFS and all comparing methods are also compared statistically. For this purpose, The Friedman test [19] is applied to the obtained results. The desired significance level for the post-hoc test is set to 0.05. If the result obtained by the Friedman test is less than the significance level, we perform an analysis for pairwise comparison of variables according to Coakley and Conover [6]. Table 9 outlines the number of win/tie/loss of the proposed method against others based on the Friedman test. Also, Table 10 reports the average runtime of each algorithm on different datasets.

**Table 10** Average run-time of methods on different datasets

| Method/dataset | BMFS | PPT-MI | MCLS | MDFS | LRFS | PPT-ReliefF |
|---|---|---|---|---|---|---|
| Yeast | 0.0468 | 0.725 | 60.2076 | 11.0152 | 7.3683 | 1.1691 |
| Medical | 10.0841 | 0.2460 | 2.1806 | 3.2240 | 3.6579 | 3.6189 |
| Scene | 0.6859 | 0.0478 | 6.8064 | 12.1621 | 0.7541 | 8.2587 |
| Enron | 6.7311 | 2.5371 | 7.8378 | 17.3045 | 5.5223 | 3.5694 |
| Image | 0.7508 | 0.0439 | 7.0390 | 13.0178 | 0.6096 | 4.7497 |
| Bibtex | 47.4972 | 78.7431 | 259.4514 | 511.6161 | 800.5607 | 129.0513 |
| Corel5k | 2.8433 | 9.55788 | 239.0202 | 218.8183 | 38.9907 | 10.1358 |

**Table 11** The computational complexity of BMFS and other methods

| Method | Computational complexity (big omicron) |
|---|---|
| MDFS | $\mathrm{O}\left(N^2k + \left(N^2M + NM^2 + NML\right) \times t\right)$ |
| LRFS | $\mathrm{O}\left(NML + NM^2\right)$ |
| PPT-MI | $\mathrm{O}\left(N^2 + NM\right)$ |
| PPT-ReliefF | $\mathrm{O}\left(N^2 + NMn\right)$ |
| MCLS | $\mathrm{O}\left(N^2(M + k) + N^2M\right)$ |
| BMFS | $\mathrm{O}\left(NML + \left[\frac{M}{L}\right] \times L^3\right)$ |

Description of symbols:

*M* number of features

*N* number of instances

*L* number of labels

*t* number of iterations until convergence

*k* number of neighbors

*n* number of subsampling instances

### 6.1 Discussion

Based on the obtained results, the BMFS is superior to similar algorithms. Figures 5 and 6 display the result of the BMFS algorithm and the other methods in terms of accuracy and hamming loss. Also, we have recorded the results of the method in terms of Average-precision and one-error metrics in Tables 2, 3, 4, 5, 6, 7, 8. The results show the superiority of the proposed method in all evaluated criteria. In Table 9, we have shown the win/tie/loss results of the proposed method based on the Friedman test. We can see that the BMFS algorithm is superior to other similar methods in all four criteria. Table 10 reports the obtained average run-time of BMFS and comparing methods on 7 datasets. We can see that the proposed method functions at a proper speed.

Table 11 presents the computational complexity of the proposed method and comparison of methods through "Big Omicron" notation. We can see that the proposed method does not have much computational complexity compared to other similar methods.

In the end, we intend to discuss the main advantages and disadvantages of the proposed methods as compared to the above-mentioned methods. The most significant advantage of the proposed method over the other methods is better performance in increasing the classification accuracy. The proposed method outperformed most other methods in terms of classification criteria and had a significantly higher accuracy. The reason for this better performance is that unlike other methods, it considers the redundancy of features in addition to considering the relevancy to class labels, which leads to the selection of discriminative features. Another reason is the consideration of the weight of the labels, which helps improve the performance. Another advantage of the proposed method is the use of a graph-based model that provides a better understanding of the problem. Another positive note of the proposed method is that it functions better for a dataset with a high number of labels. This performance involves classification accuracy and execution time. This is evident in the results obtained for Corel5k and Bibtex, as two datasets with a high number of labels. The results obtained for these two datasets show the best performance according to the classification criteria and the lowest execution time among all methods. The disadvantage of the proposed method is that it is slower than the PPT-MI method for datasets with a small number of labels. The main reason for this is the slowness of the Hungarian algorithm, which can be solved by replacing the Hungarian algorithm with a more efficient method. Note that this slower execution speed is not usually very lower than that of the PPT-MI method. Thus, it can be concluded that in general, the proposed method has an acceptable performance compared to other methods and has established a tradeoff between accuracy and execution time.

## 7 Conclusions

In this work, we proposed a new graph-based feature selection algorithm for multi-label learning called the BMFS which mapped the features and labels space to a bipartite graph. The Hungarian algorithm was employed to select the best subset of features matching the labels. This method was based on a filter-based strategy and specifically designed for multi-label data. In this method, we considered the correlation distance between the set of features and labels as the cost matrix of the matching process. Next, the Hungarian algorithm was used to find the best subset of features, after which the matching features were sorted based on the maximum correlation distance obtained by applying the weight of labels to the correlation distance matrix. The results of different datasets showed the optimality and efficiency of the proposed method. We intend to use this model for another type of feature selection, such as semi-supervised feature selection as well as other graph matching algorithms to improve the presented method. We also try to generalize the proposed algorithm to other feature selection types such as semi-supervised learning and which to apply the feature selection in medical sciences.

## References

1. Arslan S, Ozturk C (2019) Multi hive artificial bee colony programming for high dimensional symbolic regression with feature selection. Appl Soft Comput J 78:515–527. https://doi.org/10.1016/j.asoc.2019.03.014

2. Bayati H, Dowlatshahi MB, Paniri M (2020) MLPSO: a filter multi-label feature selection based on particle swarm optimization. In: 2020 25th International Computer Conference, Computer Society of Iran (CSICC). IEEE, pp 1–6
3. Cai J, Luo J, Wang S, Yang S (2018) Feature selection in machine learning: A new perspective. Neurocomputing 300:70–79. https://doi.org/10.1016/j.neucom.2017.11.077
4. Che X, Chen D, Mi J (2020) A novel approach for learning label correlation with application to feature selection of multi-label data. Inf Sci (Ny) 512:795–812. https://doi.org/10.1016/j.ins.2019.10.022
5. Cherman EA, Spolaôr N, Valverde-Rebaza J, Monard MC (2015) Lazy Multi-label learning algorithms based on mutuality strategies. J Intell Robot Syst Theory Appl 80:261–276. https://doi.org/10.1007/s10846-014-0144-4
6. Coakley CW, Conover WJ (2000) Practical nonparametric statistics. J Am Stat Assoc 95:332. https://doi.org/10.2307/2669565
7. Doquire G, Verleysen M (2011) Feature selection for multi-label classification problems. In: lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). pp 9–16
8. Dowlatshahi MB, Derhami V (2019) Winner determination in combinatorial auctions using hybrid ant colony optimization and multi-neighborhood local search. J AI Data Min 5:169–181. https://doi.org/10.22044/jadm.2017.880
9. Dowlatshahi MB, Derhami V, Nezamabadi-pour H (2018) A novel three-stage filter-wrapper framework for miRNA subset selection in cancer classification. Informatics. https://doi.org/10.3390/informatics5010013
10. Dowlatshahi MB, Derhami V, Nezamabadi-Pour H (2020) Fuzzy particle swarm optimization with nearest-better neighborhood for multimodal optimization. Iran J Fuzzy Syst 17:7–24. https://doi.org/10.22111/ijfs.2020.5403
11. Dowlatshahi MB, Derhami V, Nezamabadi-Pour H (2017) Ensemble of filter-based rankers to guide an epsilon-greedy swarm optimizer for high-dimensional feature subset selection. Inf. https://doi.org/10.3390/info8040152
12. Dowlatshahi MB, Nezamabadi-Pour H (2014) GGSA: a grouping gravitational search algorithm for data clustering. Eng Appl Artif Intell 36:114–121. https://doi.org/10.1016/j.engappai.2014.07.016
13. Dowlatshahi MB, Nezamabadi-Pour H, Mashinchi M (2014) A discrete gravitational search algorithm for solving combinatorial optimization problems. Inf Sci (Ny) 258:94–107. https://doi.org/10.1016/j.ins.2013.09.034
14. Dowlatshahi MB, Rezaeian M (2016) Training spiking neurons with gravitational search algorithm for data classification. In: 1st conference on swarm intelligence and evolutionary computation, CSIEC 2016—Proceedings. pp 53–58
15. Duan R, Su HH (2012) A scaling algorithm for maximum weight matching in bipartite graphs. In: proceedings of the annual ACM-SIAM symposium on discrete algorithms, pp 1413–1424
16. Ventura JS, Cano A (2020) Distributed multi-label feature selection using individual mutual information measures. Knowl Based Syst. https://doi.org/10.1016/j.knosys.2019.105052
17. Hashemi A, Dowlatshahi MB (2020) MLCR: a fast multi-label feature selection method based on K-means and L2-norm. In: 2020 25th international computer conference, Computer Society of Iran (CSICC). IEEE, pp 1–7
18. Hashemi A, Dowlatshahi MB, Nezamabadi-pour H (2020) MGFS: a multi-label graph-based feature selection algorithm via pagerank centrality. Expert Syst Appl 142:113024. https://doi.org/10.1016/j.eswa.2019.113024
19. Hastie T, Tibshirani R, Friedman J, Franklin J (2017) The elements of statistical learning: data mining, inference, and prediction. Math Intell. https://doi.org/10.1007/BF02985802
20. Huang R, Jiang W, Sun G (2018) Manifold-based constraint Laplacian score for multi-label feature selection. Pattern Recognit Lett 112:346–352. https://doi.org/10.1016/j.patrec.2018.08.021
21. Kashef S, Nezamabadi-pour H, Nikpour B (2018) Multilabel feature selection: a comprehensive review and guiding experiments. Wiley Interdiscip Rev Data Min Knowl Discov 8:e1240. https://doi.org/10.1002/widm.1240
22. Kashef S, Nezamabadi-Pour H, Nikpour B (2018b) FCBF3Rules: a feature selection method for multi-label datasets. In: 3rd conference on swarm intelligence and evolutionary computation (CSIEC). IEEE, pp 1–5
23. Kuhn HW (2010) The hungarian method for the assignment problem. In: 50 years of integer programming 1958–2008: From the early years to the state-of-the-art. Springer, Berlin, pp 29–47
24. Lee J, Kim D-W (2015) Mutual Information-based multi-label feature selection using interaction information. Expert Syst Appl. https://doi.org/10.1016/j.eswa.2014.09.063
25. Li J, Cheng K, Wang S et al (2017) Feature selection: a data perspective. ACM Comput Surv. https://doi.org/10.1145/3136625
26. Liu H, Yang Y (2015) Bipartite edge prediction via transductive learning over product graphs. In: 32nd International Conference on Machine Learning, ICML 2015. pp 1880–1888
27. Livi L, Rizzi A (2013) The graph matching problem. Pattern Anal Appl 16:253–283. https://doi.org/10.1007/s10044-012-0284-8
28. Miao J, Niu L (2016) A survey on feature selection. Procedia Comput Sci 91:919–926. https://doi.org/10.1016/j.procs.2016.07.111
29. Momeni E, Dowlatshahi MB, Omidinasab F et al (2020) Gaussian process regression technique to estimate the pile bearing capacity. Arab J Sci Eng. https://doi.org/10.1007/s13369-020-04683-4
30. Munkres J (1957) Algorithms for the assignment and transportation problems. J Soc Ind Appl Math 5:32–38. https://doi.org/10.1137/0105003
31. Paniri M, Dowlatshahi MB, Nezamabadi-pour H (2020) MLACO: a multi-label feature selection algorithm based on ant colony optimization. Knowl Based Syst. https://doi.org/10.1016/j.knosys.2019.105285
32. Pereira RB, Plastino A, Zadrozny B, Merschmann LHC (2018) Categorizing feature selection methods for multi-label classification. Artif Intell Rev 49:57–78. https://doi.org/10.1007/s10462-016-9516-4
33. Rafsanjani MK, Dowlatshahi MB (2012) Using gravitational search algorithm for finding near-optimal base station location in two-tiered WSNs. Int J Mach Learn Comput. https://doi.org/10.7763/ijmlc.2012.v2.148
34. Reyes O, Morell C, Ventura S (2015) Scalable extensions of the ReliefF algorithm for weighting and selecting features on the multi-label learning context. Neurocomputing. https://doi.org/10.1016/j.neucom.2015.02.045
35. Stauffer M, Tschachtli T, Fischer A, Riesen K (2017) A survey on applications of bipartite graph edit distance. In: lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp 242–252
36. Sun Z, Zhang J, Dai L et al (2019) Mutual information based multi-label feature selection via constrained convex optimization. Neurocomputing. https://doi.org/10.1016/j.neucom.2018.10.047
37. Wang C, Lin Y, Liu J (2019) Feature selection for multi-label learning with missing labels. Appl Intell 49:3027–3042. https://doi.org/10.1007/s10489-019-01431-6
38. Wang H, Zhang Y, Zhang J et al (2019) A factor graph model for unsupervised feature selection. Inf Sci (Ny) 480:144–159. https://doi.org/10.1016/j.ins.2018.12.034

39. Yan J, Yin XC, Lin W, et al (2016) A short survey of recent advances in graph matching. In: ICMR 2016—proceedings of the 2016 ACM International Conference on Multimedia Retrieval, pp 167–174
40. Zepeda-Mendoza ML, Resendis-Antonio O (2013) Bipartite Graph. Encyclopedia of Systems Biology. Springer, New York, pp 147–148
41. Zhang J, Luo Z, Li C et al (2019) Manifold regularized discriminative feature selection for multi-label learning. Pattern Recognit 95:136–150. https://doi.org/10.1016/j.patcog.2019.06.003
42. Zhang L, Hu Q, Zhou Y, Wang X (2014) Multi-label attribute evaluation based on fuzzy rough sets, pp 100–108
43. Zhang M-L, Zhou Z-H (2007) ML-KNN: a lazy learning approach to multi-label leaming. Pattern Recognit 40:2038–2048. https://doi.org/10.1016/j.patcog.2006.12.019
44. Zhang P, Liu G, Gao W (2019) Distinguishing two types of labels for multi-label feature selection. Pattern Recognit 95:72–82. https://doi.org/10.1016/j.patcog.2019.06.004
45. Zhou F, Lin Y (2016) Fine-grained image classification by exploring bipartite-graph labels. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp 1124–1133

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.