Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

An Unsupervised Feature Ranking Scheme by Discovering Biclusters

Qinghua Huang, Lianwen Jin
School of Electronic and Information Engineering
South China University of Technology
Guangzhou, China
qhhuang@scut.edu.cn

Dacheng Tao
School of Computer Engineering
Nanyang Technological University
Singapore 639798

***Abstract*—In this paper, we aim to propose an unsupervised feature ranking algorithm for evaluating features using discovered biclusters which are local patterns extracted from a data matrix. The biclusters can be expressed as sub-matrices which are used for scoring relevant features from two aspects, i.e. the interdependence of features and the separability of instances. The features are thereby ranked with respect to their accumulated scores from the total discovered biclusters before the pattern classification. Experimental results show that this proposed algorithm can yield comparable or even better performance in comparison with the well-known Fisher Score, Laplacian Score and Variance Score using several UCI data sets.**

***Keywords*—feature selection, Bicluster score, unsupervised learning**

I. INTRODUCTION

Feature selection is an important preprocessing step before recognizing meaningful patterns from a data set with a large number of features. Many studies have shown that features (dimensionality) can be reduced without degrading classification/clustering performance [1,2]. Selecting an appropriate subset of more representative features (or dimensions) can even improve the identification performance for patterns. Feature selection is therefore regarded as an important preprocessing step for analyzing various sorts of data analysis.

The methods of feature selection can be grouped into two categories, i.e. the filter [3] and wrapper [4] methods. Most of the filter and wrapper methods for feature selection can be regarded as supervised algorithms since the class labels are used. Even the presence of class label, it is a challenging problem. Because the class labels are often unavailable in real practices, we discuss unsupervised learning which is more challenging. Some unsupervised methods [11, 12] have been designed to find good features according to the separability of instances. Dy et al. [5] described an unsupervised wrapper method using an expectation-maximization (EM) algorithm. The quality of clusters obtained from different feature subsets are used for measuring cluster separability. In more recent work [13, 14], feature similarity was measured for detecting redundant features. Law et al. [15] proposed a concept of feature saliency estimated using an EM algorithm for simultaneously selecting features and clustering instances.

In both filter and wrapper methods, the optimal feature subset needs to be found. Accordingly, a number of methods including exhaustive search [1], sequential forward (backward) selection [5], sequential forward (backward) floating search [6], evolutionary search [7], etc. are performed to examine combinations of feature subsets. Because the computational complexity quickly increases with the number of features, it is always impractical to evaluate a large number of feature subsets. To overcome this problem, a number of filter methods adopt the ranking method [11, 12, 17], in which the original d features are individually assessed and the m ($<d$) best features can be selected for subsequent pattern analysis. Although these ranking methods are much faster than that of exhaustively (or heuristically) searching, it has been recognized that the subset of individually "good" features may not collectively provide good classification performance [8], mainly due to the lack of information about feature inter-relations.

2	7	8	4	17	26	3	9	5	1
4	18	**3**	13	**6**	12	**8**	**14**	9	10
24	3	9	11	4	16	3	11	20	4
11	6	**3**	10	**6**	17	**8**	**14**	13	5
5	8	**3**	7	**6**	2	**8**	**14**	6	10
10	22	8	2	10	16	9	7	1	13
21	13	8	5	15	27	14	2	3	7
14	18	2	6	9	4	12	11	23	1
1	4	**3**	8	**6**	19	**8**	**14**	11	5
7	3	11	24	3	2	8	16	9	6

Figure 1. An example of bicluster. A bicluster with constant columns is formed by the highlighted elements which are actually a sub-matrix with a local coherent pattern.

In recent years, more and more attentions have been paid to finding biclusters which are local coherent patterns with a subset of instances only under a certain subset of features in many data [9]. An example can be seen in Fig. 1, where the aggregated rows (instances) and columns (features) are not fully consecutive. The methods for detecting biclusters have been proposed for gene expression profiling in microarray data [10]. Because a bicluster contains a subset of experimental conditions and a subset of genes, the inter-relationships among the conditions and those among the genes can be revealed. In another words, due to the intrinsic idea of bi-dimensional clustering, the discovered biclusters are able to provide

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

important clues for extracting feature interdependencies and clusters of instances, and are therefore potentially useful for evaluating features by simultaneously considering both feature interdependencies and instance separability.

In this paper, we propose a new unsupervised feature ranking algorithm based on the discovery of biclusters. Because this method incorporates a biclustering algorithm to discover biclusters and ranks the features, it has some characteristics of both feature ranking and wrapper methods and can therefore be viewed as a hybrid model. Its main distinction from conventional wrapper methods is that it ranks the features instead of searching for optimal feature subsets without the determination of the number of clusters, hence reducing the computational complexity. Meanwhile, unlike some ranking filter methods that score the relevance between an individual feature and the class labels, the feature interdependencies can be well considered in this method according to the feature subsets extracted from these biclusters. As a result, we make use of the discovered biclusters to evaluate features from two aspects, i.e. the interdependencies among features and the separability of instances. By considering both the feature correlations and instance separability in evaluating the features, we propose a scoring scheme to rank each of the features and test its performance using several often used UCI data sets [16]. This feature selection algorithm based on the discovered biclusters is named as Bicluster score in this paper.

This paper is organized as follows. Section II introduces the proposed algorithm in detail. Section III presents the experimental results and the last section draws conclusions for the proposed algorithm.

II. THE UNSUPERVISED FEATURE RANKING ALGORITHM

A. Basic Idea

As illustrated in Fig. 1, a bicluster including a subset of rows (instances) and a subset of columns (features) indicates a sub-matrix which can be viewed as a local coherent pattern. In such a pattern, all of the features contained in the sub-matrix have the same contribution to the identification of the clustered instances, indicating an inter-correlation among them. Similarly, the correlation among these instances can also be revealed, and they can be represented as a cluster discovered under the feature subset, indicating a successful separation from the other instances. Thus, it is observed that a well discovered bicluster can provide useful information about both the inter-correlations among features in the feature subset and the separability of the instance subset from the others under the feature subset. In this paper, we make use of the biclusters found in a data matrix to score the features and this new scoring scheme is named as Bicluster Score.

In order to use the intrinsic information contained in a bicluster for evaluating features, we firstly propose an effective biclustering algorithm which converts the problem of searching for biclusters into two easy-to-apply procedures: conventional hierarchical clustering of instances for each feature and heuristic search for the biclusters (sub-matrices) associated with the clustered instances. From the discovered biclusters, two factors (i.e. the feature interdependencies and the instance separabilities) are thereafter considered and incorporated into the computation of Bicluster Score for each feature. Finally, the features are ranked according to their Bicluster Scores.

1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0

(a)

2.0	2.0	2.0	2.0
1.5	1.5	1.5	1.5
2.5	2.5	2.5	2.5
0.5	0.5	0.5	0.5

(b)

1.2	1.8	0.6	2.0
1.2	1.8	0.6	2.0
1.2	1.8	0.6	2.0
1.2	1.8	0.6	2.0

(c)

1.5	2.5	0.5	5.5
3.5	4.5	2.5	7.5
2.5	3.5	1.5	6.5
4.5	5.5	3.5	8.5

(d)

1.0	2.0	3.0	4.0
0.5	1.0	1.5	2.0
2.0	4.0	6.0	8.0
1.5	3.0	4.5	6.0

(e)

1.0	4.0	5.0	2.0
2.0	3.0	7.0	2.5
4.0	6.0	8.0	5.0
3.0	8.0	9.0	4.0

(f)

Figure 2. Different bicluster patterns. (a) Constant bicluster, (b) constant rows, (c) constant columns, (d) coherent values with an additive model, (e) coherent values with a multiplicative model, and (f) coherent evolution values in columns.

B. Biclustering Method

As shown in Fig. 2, biclusters have several different models, i.e. the constant, additive, multiplicative, and coherent evolutionary models. In recent years, a large number of biclustering algorithms have been proposed and successfully applied to analysis of microarray gene expression data. Details for the biclustering algorithms can be found in [10]. However, most of those algorithms are specifically designed for analyzing gene expression profiles, where the genes may be co-regulated in a scaling, shifting, or even hybrid manner, hence cannot be directly used to solve a generalized classification/clustering problem. According to the Euclidean distance, only the models shown in Figs. 2(a) and (c) can be regarded as a group of points being able to form a compact cluster in a multi-dimensional space, which can be easily recognized using conventional clustering algorithms. Consequently, we try to find the biclusters with constant columns (Figs. 2(a) and (c)) from a data matrix, where the rows denote instances and the columns denote the features.

In order to extract the biclusters with constant columns, we propose a new biclustering algorithm involving three main procedures, i.e. (1) discovery of bicluster seeds, (2) heuristic formation of biclusters and (3) removal of redundant biclusters. In the first procedure, we detect clusters of elements in each of the columns. It has been mentioned that the rows of a bicluster can be simply extracted using a conventional clustering method when the feature subset is determined. However, it is not easy to find the feature subset from the full size of features for a specific bicluster. Instead of exhaustively searching for feature subsets, we attempt to detect a feature subset by detecting each of its members in this paper. As demonstrated in Figs. 1 and 2(c), the elements under a single column in the sub-matrix of a bicluster are approximately the same with a small variance, and hence can be found by a directly clustering method. As a result, the clustered elements under a single column can be thought of as being potentially associated with a single or multiple biclusters. A cluster detected in a single column is called a bicluster seed. Thus, given a data matrix M with n_r rows and n_c columns, we firstly apply an conventional

agglomerative hierarchical clustering (HC) method [1] using the average linkage for clustering all of the elements under each of the columns in the original data matrix, and then obtain a preliminary set of bicluster seeds, as formulated by

$$[C_s(i,j), N_{cl}(j)] = HC(j, T_d),\ j = 1 \cdots n_c \quad (1)$$

$$BS\_set = \{C_s(i,j) \mid i = 1 \cdots N_{cl}(j),\ j = 1 \cdots n_c\} \quad (2)$$

where $HC(j,T_d)$ is the HC algorithm applied to the elements under the jth colomn with a pre-set distance threshold T_d, $N_{cl}(j)$ denotes the number of clusters for the jth column, $C_s(i,j)$ the ith bicluster seed under the jth column, and *BS_set* is the aggregation of the bicluster seeds detected from all of columns. The time complexity of this procedure is $O(n_c n_r^2)$.

As aforementioned, each of the detected bicluster seeds in *BS_set* is regarded as a potential part of some unknown biclusters. We need to form larger biclusters from these small bicluster seeds in the second procedure. The details are described as follows. First, according to the number of rows, the bicluster seeds in *BS_set* are sorted in an ascending order. Beginning with the bicluster seed with the lowest row number, each of the bicluster seeds in *BS_set* is then expanded along the column dimension. Given a bicluster seed with R_j rows, a new sub-matrix ***Ms*** can be formed with R_j rows and all of the n_c columns. An optimization procedure is finally required to find the largest bicluster that meets a certain homogeneity criterion in ***Ms***.

In this paper, the mean-square-residue (MSR) score [9] which has been widely used as a metrics for measuring the homogeneity of a bicluster is employed. Given a sub-matrix with R rows and C columns, its MSR score is defined by

$$h(R,C) = \frac{1}{|R| \cdot |C|} \sum_{i \in R, j \in C} \left(e_{ij} - e_{iC} - e_{Rj} + e_{RC} \right)^2$$

$$e_{iC} = \frac{1}{|C|} \sum_{j \in C} e_{ij} \quad , \quad e_{Gj} = \frac{1}{|R|} \sum_{i \in R} e_{ij} \quad (3)$$

$$e_{RC} = \frac{1}{|R| \cdot |C|} \sum_{i \in R, j \in C} e_{ij}$$

If $h(R,C) \le \delta$, *accept it as a valid bicluster*,

where e_{ij} denotes the element value at the ith row and jth column in the bicluster, δ a homogeneity threshold defining the maximum allowable dissimilarity within the elements of the bicluster, and $h(R,C)$ the value of MSR score for the bicluster. The homogeneity threshold is set by users according to their respective applications.

A local search algorithm is conducted to find the largest bicluster in ***Ms***. For a submatrix, defining a group of nodes denoting its rows and columns, the search is performed by iteratively deleting the node that mostly increases the MSR score until the score of the refined submatrix is no larger than a predefined homogeneity threshold T_m. The algorithm starts with every ***Ms*** associated with the clusters in *BS_set* and consists of the following steps:

(**i**) Input a submatrix M

(**ii**) Set an array of nodes denoting all of the rows and columns of M

(**iii**) For every node, calculate the MSR score for a new submatrix where this node is deleted from M.

(**iv**) Delete the node which mostly increases the MSR score of M and set the new submatrix as M.

(**v**) If the MSR score for M is larger than a predefined value T_m, repeat step (**ii**). Otherwise, output M as the largest bicluster.

The algorithm is applied to each of the bicluster seeds in *BS_set* and the output biclusters are put into a new bicluster set, *BC_set*. This procedure is illustrated in Fig. 3. The complexity of this local search algorithm is $O(dn^2)$, where d is the number of clusters in *BS_set*, and n the number of both rows and columns.

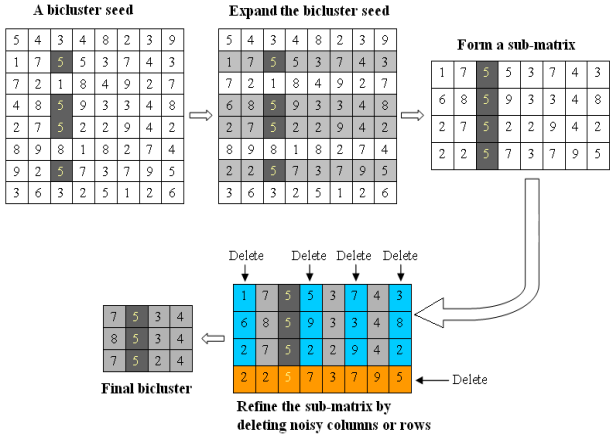


Figure 3. An example for illustrating the procedure of expanding a bicluster seed and refining the expanded sub-matrix into a real bicluster.

The third procedure gets rid of all redundant biclusters which are fully overlapped by larger ones. We first rank the biclusters in *BC_set* with respect to their column numbers in an ascending order. With the sorted biclusters, a bicluster is deleted from *BC_set* if it is mostly contained by one ranked at a lower position. Thereafter, the biclusters reserved in *BC_set* are the final output and can be used for feature evaluation. The complexity of this procedure is $O(n^2)$. This biclustering algorithm is summarized in Fig. 4.

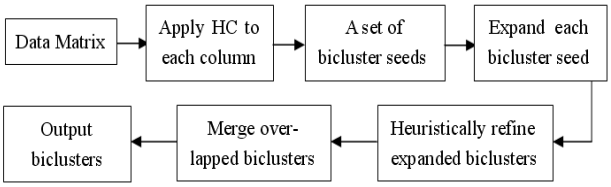


Figure 4. Diagram of the proposed biclustering algorithm.

Instead of exhaustively or heuristically searching for feature subsets in conventional wrapper methods, the procedure of searching for the column combination of a bicluster in our method is converted into a heuristic refining a sub-matrix to output a bicluster. As a result, the proposed algorithm is relatively more efficient without the need to repeatedly perform a clustering algorithm to evaluate every new column combination.

It is noteworthy that the values of an instance may vary greatly under different features. Therefore, we use the

following method to normalize each column to ensure that most of the values in each column fall into a limited range.

$$e_n(i,j)=\frac{e(i,j)-mean(e(\cdot,j))}{2\cdot std(e(\cdot,j))},\qquad i=1\ldots n_r,\ j=1\ldots n_c \tag{4}$$

where $e(i,j)$ is the element value at the ith row and jth column, $mean(e(\cdot,j))$ denotes the mean of the elements under the jth column, $std(e(\cdot,j))$ the standard deviation of the jth column, and $e_n(i,j)$ the normalized element value. After the normalization of data values, the distance threshold T_d and the homogeneity threshold T_m are fixedly set to 0.01 and 0.02, respectively, in this study.

C. Feature Ranking Scheme

Once the biclusters have been found from the data matrix, we need to extract information from them which can be used to evaluate each of the features. As motivated by the two factors (i.e. feature correlation and instance separability) mentioned above, a scoring scheme (called Bicluster score) is proposed by considering both factors in this study. We define two subsidiary scores that stand for the two factors, respectively, i.e. the *correlation score* which measures the correlations among features in a feature subset, and the *separability score* which measures the separability of a feature. For the kth feature, suppose that it is included by any one of biclusters from a bicluster subset Z_k, the two scores (denoted as *Cor_Score* and *Sep_Score*, respectively) are defined as follows:

$$Cor\_Score(k)=\sum_{i=1}^{n_{b,k}}\frac{n_{f,k}(i)}{n_c} \tag{5}$$

$$Sep\_Score(k)=\frac{n_{s,k}}{n_r}\sqrt{\sum_{i=1}^{n_{b,k}}\left(\mu_{i,k}-\mu_{a,k}\right)^2/n_{b,k}} \tag{6}$$

where $n_{b,k}$ denotes the number of biclusters in Z_k, $n_{f,k}(i)$ the number of features for the ith bicluster in Z_k, $n_{s,k}$ the number of the rows enumerated from all of the biclusters in Z_k, $\mu_{i,k}$ the element average for the ith bicluster in Z_k under the kth feature, and $\mu_{a,k}$ the average of $\mu_{i,k}$, $i=1\ldots n_{b,k}$. It is observed that in *Cor_Score*, $n_{f,k}(i)/n_c$ is the ratio of the number of columns for the ith bicluster in Z_k to the full length of columns. The *Cor_Score* actually equals the summation of the ratios. If a feature is associated with a larger number of biclusters, and/or the column dimensions of these biclusters cover a larger portion of the full size of dimension, the corresponding *Cor_Score* is larger and vice versa. In *Sep_Score*, $n_{s,k}/n_r$ denotes the ratio of the instances which can be clustered by the biclustering algorithm to the full number of instances, and $\sqrt{\sum_{i=1}^{n_{b,k}}\left(\mu_{i,k}-\mu_{a,k}\right)^2/n_{b,k}}$ the squared variance of the cluster centers for the kth feature. The larger the ratio and/or the variance are, the larger *Sep_Score* for the feature is.

Finally, the Bicluster score (denoted as *Bic_Score*) for the kth feature is obtained by considering both of the two subsidiary scores, and is expressed as:

$$Bic\_Score(k)=\alpha\cdot\overline{Cor\_Score(k)}+\overline{Sep\_Score(k)} \tag{7}$$

where $\overline{Cor\_Score(k)}$ and $\overline{Sep\_Score(k)}$ denote the normalized values for *Cor_Score*(k) and *Sep_Score*(k), $k=1\ldots n_c$, respectively, and α is a regulation coefficient for balancing the contributions of the *Cor_Score* and the *Sep_Score* to the final *Bic_Score*. The features with higher *Bic_Score* are viewed as being better at characterizing the data clusters and linking with other features.

III. Experiments

In order to evaluate the performance of the proposed feature ranking algorithm, we conduct experiments using several standard data sets and make comparisons with three popular feature selection algorithms: Variance Score [11], Laplacian Score [12] and Fisher Score [11]. The former two methods are unsupervised, while Fisher Score is supervised.

A. UCI Data sets and the Classifier

We use 3 real world data sets downloaded from UCI database [16]. They are wine data, Wisconsin diagnostic breast cancer (wdbc) data and congressional voting records (House-Votes-84) data. The wine data set has 13 features and 178 instances categorized into 3 groups. The instances are wines and the features are chemical components. The wdbc data has 569 instances and 30 features. It contains two groups, i.e. benign and malignant breast tumors. The House-Votes-84 data has 435 instances which are congressmen and grouped into two parties, i.e. republican and democrat. The features are the votes for 16 topics. An affirmative vote is denoted as 1, a negative vote is denoted as -1, and an abstaining vote is denoted as 0.

In the experiments, we can generate a pair of training and testing sets by randomly selecting half of instances from all classes as the training set and setting the remaining half as the testing set. For each UCI data, 20 pairs of training and testing sets are generated. The feature selection algorithms are then applied to the testing sets. The features are ranked according to their scores computed by each algorithm. The feature number can be preset by users. With a pre-determined feature number, the nearest neighborhood (1-NN) method with Euclidean distance is used as a classifier to obtain the classification accuracy. Following the experimental method used in [17], we evaluate our algorithm by comparing the classification accuracies obtained by different feature selection algorithms. For Bicluster Score, we set the parameter α in (7) to 1.0 when comparing with the other three algorithms.

B. Results

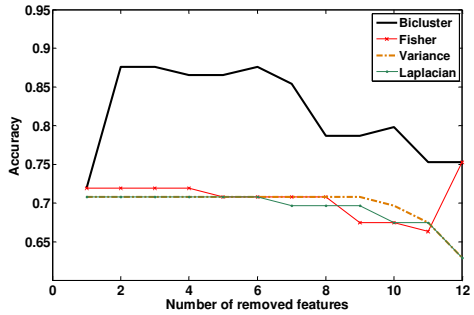
We summarize the averaged classification accuracies using the 20 pairs of data sets for each UCI data in Tables 1, and illustrate the simulation results using an example data set in Figs. 5-7, for the three UCI data, respectively.

The accuracies vs. the number of removed features for the wine data can be seen in Fig. 5. It is obvious that Bicluster score significantly outperforms the others. The reason can be explained by the intrinsic properties of the features in wine data. Because the features are chemical components contained in the wines, the density of one component can influence those of the other components. Thus, it is concluded that there are strong interdependencies among the features. As stated above, our algorithm is good at discovering feature interdependencies

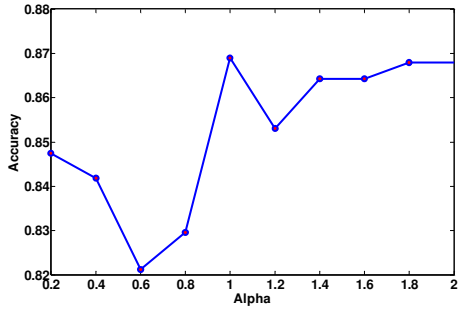
and hence can achieve the best results for the wine data. From Fig.5 (b), the performance of Bicluster score is approximately improved as the balancing parameter α is increasing. It implies that considering inter-dependencies of features is able to improve the feature selection performance.

Table 1. Averaged accuracies (in percentage) of different algorithms using the UCI datasets.

Data	Bicluster	Fisher	Variance	Laplacian
wine	80.71±5.77	70.6±2.48	69.76±2.37	69.29±2.36
wdbc	80.88±8.74	88.73±3.98	75.02±2.21	74.99±2.21
House-votes-84	95.17±0.88	93.92±1.57	87.16±15.22	92.97±3.51



(a)

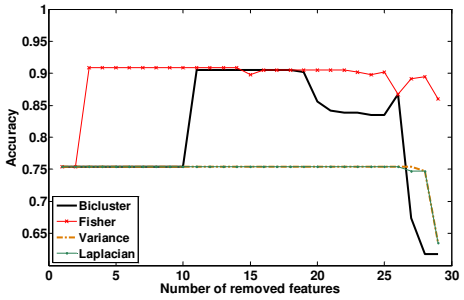


(b)

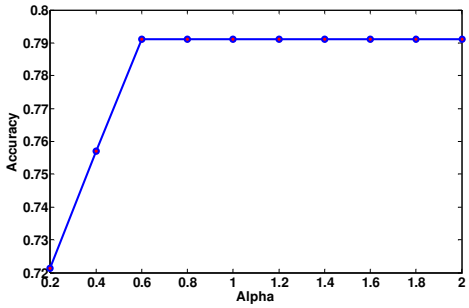
Figure 5. Feature selection performance on the wine data sets. (a) The classification accuracy, and (b) accuracies against α for Bicluster score.

The simulation results for the wdbc data can be seen in Fig. 6(a). According to the results, the Fisher score outperforms the others. Nevertheless, our algorithm performs better than the Variance and Laplacian scores, indicating an improved performance of feature selection for the data without class labels. According to the description for wdbc data, the features are some quantities (such as area, smoothness and dimensions) measured from breast tumor regions. Because these measures have no significant inter-dependencies, the *Cor_Score* cannot effectively influence the feature orders. This explanation can be further proved as shown in Fig. 6(b). It is noted that the varying values for the balancing parameter α cannot significantly change the classification accuracies, indicating its relatively weak effect on the feature selection.

For the house-votes-84 data, our algorithm outperforms the others. As illustrated in Fig. 7(a), Bicluster score achieves comparable (even better) classification accuracies than Fisher score, and much better results than the other two unsupervised methods. We can conclude that there exist inter-relationships among the features which are actually proposals in various economic and political fields. Different proposals are likely to have overlapped part focused by the public.

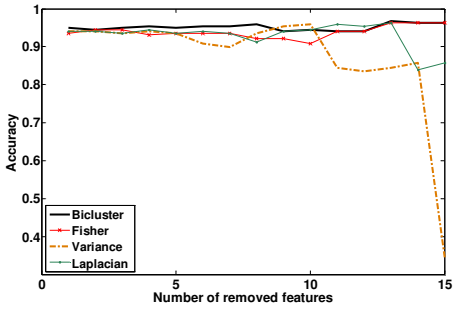


(a)

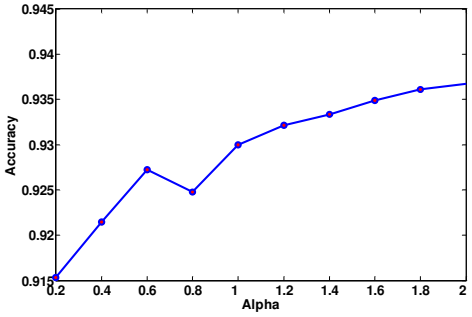


(b)

Figure. 6. Feature selection performance on the wdbc data sets. (a) The classification accuracy, and (b) accuracies against α for Bicluster score.



(a)



(b)

Fig.7. Feature selection performance on the house-votes-84 data sets. (a) The classification accuracy, and (b) accuracies against α for Bicluster score.

In addition, a group of congressmen from a specific party may have positive views on a subgroup of proposals and negative views on another subgroup. Hence, there exist many subsets of features which internally influence each other. Our algorithm is able to discover these feature subsets, meanwhile, the interactions among these proposals to be voted can accordingly help in differentiating the congressmen's political stands. The influence of different values of α is shown in Fig. 7(b), and provides further support to this point. It can be seen that the total classification accuracy keeps increasing as α is increased.

IV. Discussions and Conclustions

In this paper, a novel unsupervised feature selection algorithm is proposed. This algorithm is based on an unsupervised biclustering algorithm which can discover local coherent patterns in a data matrix. The discovered local patterns including a subset of instances and a subset of features simultaneously reveal both the separability of instances and inter-dependencies among features. Thus, we propose a new scoring scheme which is called Bicluster score. Like a wrapper method, Bicluster score firstly discover biclusters in a data matrix, then calculate two subsidiary scores by considering the clustered instances and features for each bicluster, and finally compute the Bicluster score by summing the two subsidiary scores.

The experimental results using three UCI data sets demonstrate that Bicluster score can outperform the two often used unsupervised feature ranking algorithm and produce comparable or even better results than Fisher score which is a supervised method. In particular, our algorithm significantly outperforms the other three algorithms using the wine data set, indicating that the features (chemical components) have relatively strong correlations. In contrast, for the wdbc data, Fisher score demonstrates the best performance on feature selection, showing that the features extracted from breast tumor images are relatively independent to each other. As a result, our algorithm is unable to improve the classification results by considering the correlations among the features. For the house-votes-84 data set, our algorithm generates the best results, illustrating the features (i.e. the proposals to be voted) are inter-related to each other and some of them have similar influences on a subgroup of congressmen from a specific political organization.

In summary, the results demonstrate that Bicluster score is able to conduct feature selection on several UCI and real microarray data sets with good performance. By discovering biclusters and ranking the features according to these biclusters, it makes use of both the filter and wrapper methods' characteristics for selecting features and can be expected to be suitable for various data sets, especially the ones with strong interdependencies among the features.

References

[1] A.K. Jain, R.P.W. Duin, and J.C. Mao, "Statistical pattern recognition: A review," IEEE Trans. Pattern Anal. Mach. Intell., vol.22, no.1, pp. 4-37, 2000.

[2] H. Liu, and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," IEEE Trans. Knowl. Data Eng., vol. 17, pp. 491-502, 2005.

[3] H. Liu, and H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic, Boston, 2001.

[4] R. Kohavi, and G.H. John, "Wrappers for feature subset selection," Artif. Intell., vol. 97, no.1-2, pp. 273-324, 1997.

[5] J.G. Dy, C.E. Brodley, A. Kak, L.S. Broderick, and A.M. Aisen, "Unsupervised feature selection applied to content-based retrieval of lung images," IEEE Trans. Pattern Anal. Mach. Intell., vol.25, no.3, pp. 373-378, 2003.

[6] P. Pudil, J. Novovicova, and J. Kittler, "Floating search methods in feature selection," Pattern Recogn. Letters, vol.15, no.11, pp. 1119-1125, 1994.

[7] Z.X. Zhu, Y.S. Ong, K.W. Wong, and K.T. Seow, "Experimental condition selection in whole-genome functional classification," in: Proceedings of the 2004 IEEE Conference on Cybernetics and Intelligent Systems, Singapore, 2004.

[8] H.C. Peng, F.H. Long, and C. Ding, "Feature selection based on mutual information: Criteria of Max-dependancy, Max-relevance and Min-redundancy," IEEE Trans. Pattern Anal. Mach. Intell., vol.27, no.8, 1226-1237, 2005.

[9] Y. Cheng, and G.M. Church, "Biclustering of Expression Data," in: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB), 2000, pp. 93-103.

[10] S.C. Madeira, and A.L.Oliveira, "Biclustering Algorithms for Biological Data Analysis: A Survey," IEEE Trans. Comput. Biol. Bioinform., vol.1 no.1, pp. 24-45, 2004.

[11] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995.

[12] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in: Advances in Neural Information Processing Systems, 17, MIT Press, Cambridge, MA, 2005.

[13] P. Mitra, C.A. Murthy, S.K. Pal, "Unsupervised feature selection using feature similarity," IEEE Trans. Pattern Anal. Mach. Intell., vol.24, no.3, pp. 301-312, 2002.

[14] H.L. Wei, and S.A. Billings, "Feature subset selection and ranking for data dimensionality reduction," IEEE Trans. Pattern Anal. Mach. Intell., vol.29, no.1, pp. 162-166, 2007.

[15] M. Law, M. Figueiredo, and A.K. Jain, "Simultaneous feature selection and clustering using mixture models," IEEE Trans. Pattern Anal. Mach. Intell., vol.26, no.9, pp. 1154-1166, 2004.

[16] http://www.ics.uci.edu/~mlearn/MLSummary.html

[17] D.Q. Zhang, S.C. Chen, and Z.H. Zhou, "Constraint Score: A new filter method for feature selection with pairwise constraints," Pattern Recognition, vol.41, pp.1440-1451, 2008.