Pattern Recognition 43 (2010) 2106–2118

Contents lists available at ScienceDirect

Pattern Recognition

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

Bagging Constraint Score for feature selection with pairwise constraints

Dan Sun, Daoqiang Zhang *

Department of Computer Science and Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

ARTICLE INFO

Article history:
Received 2 December 2008
Received in revised form
20 November 2009
Accepted 14 December 2009

Keywords:
Feature selection
Constraint Score
Pairwise constraints
Bagging
Ensemble learning

ABSTRACT

Constraint Score is a recently proposed method for feature selection by using pairwise constraints which specify whether a pair of instances belongs to the same class or not. It has been shown that the Constraint Score, with only a small amount of pairwise constraints, achieves comparable performance to those fully supervised feature selection methods such as Fisher Score. However, one major disadvantage of the Constraint Score is that its performance is dependent on a good selection on the composition and cardinality of constraint set, which is very challenging in practice. In this work, we address the problem by importing Bagging into Constraint Score and a new method called Bagging Constraint Score (BCS) is proposed. Instead of seeking one appropriate constraint set for single Constraint Score, in BCS we perform multiple Constraint Score, each of which uses a bootstrapped subset of original given constraint set. Diversity analysis on individuals of ensemble shows that resampling pairwise constraints is helpful for simultaneously improving accuracy and diversity of individuals. We conduct extensive experiments on a series of high-dimensional datasets from UCI repository and gene databases, and the experimental results validate the effectiveness of the proposed method.

1. Introduction

Feature selection has been one of the key steps in mining high-dimensional data for decades. The idealized definition of feature selection is finding the minimally sized feature subset that is necessary and sufficient for a specific task [1]. Feature selection has several potential benefits, such as reducing measurement and storage demands, and defying the curse of dimensionality to enhance prediction performance, etc. [2,3].

The training data used in feature selection can be either labeled, unlabeled, or both, corresponding to supervised [4], unsupervised [5,6], or semi-supervised feature selection [7]. In supervised feature selection, feature relevance is evaluated by their correlation with the class labels, but in unsupervised ones, feature relevance is evaluated by their capability of keeping certain properties of the data, such as the variance or the locality preserving ability [8,9]. Supervised feature selection methods usually outperform unsupervised ones when labeled data are sufficient. However, obtaining a lot of labeled data is very expensive and inefficient due to the intervention of human experts. So in many real applications we are often confronted with the so-called 'small labeled-sample problem' [10,11]. To deal with that problem, semi-supervised feature selection methods which can use both labeled and unlabeled data to estimate feature relevance are proposed [7]. However, in semi-supervised feature selection, the supervision information used is still class labels.

In fact, besides the class labels there exist other forms of supervision information, e.g., pairwise constraints which specify whether a pair of instances belongs to the same class (*must-link* constraint) or different classes (*cannot-link* constraint) [12,13]. In our recent work [14], we have proposed a pairwise constraints-guided feature selection algorithm called Constraint Score, which has shown excellent performance on lots of datasets. However, one unresolved problem in Constraint Score is how to choose the appropriate constraint set because its performance is severely influenced by the composition and cardinality of constraint set, as in most constraint-based classification methods. For example, we run two versions of Constraint Score, denoted as Constraint Score-1 and Constraint Score-2, respectively in [14], on four datasets from UCI repository (see Table 1 for dataset description), i.e., *Credit Approval*, *Horse*, *Vehicle* and *Wine*, for 1000 times. For each run, we randomly generate 60 pairwise constraints from all of the constraints that can be generated from the total class labels, and use Constraint Score to select half features. For each dataset, half of the data are used for training and the rest for testing, and the nearest neighborhood classifier is adopted to evaluate the performance. Both the methods have large deviations in performances of the 1000 runs. Specifically, the differences between the maximum and the minimum accuracies of 1000 runs on four datasets are 21.5%, 32.1%, 20.9% and 34.1%, respectively for Constraint Score-1, and 23.6%, 27.8%, 19.4% and 31.8%, respectively for Constraint Score-2. It shows that constraint sets have a great effect on the performance of Constraint Score. However, to the best of our knowledge, it is still an 'open problem'

* Corresponding author. Tel.: +86 25 84896481x12217.
E-mail addresses: dansun@nuaa.edu.cn (D. Sun), dqzhang@nuaa.edu.cn (D. Zhang).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.12.011

Table 1
Statistics of the UCI datasets.

Datasets	Size	Dimension	# of classes
Credit Approval	690	14	2
Heart	270	13	2
Horse	368	27	2
Image	2310	19	7
Ionosphere	351	34	2
Labor	57	16	2
Sonar	208	60	2
Vehicle	846	18	4
Vowel	528	10	11
Wine	178	13	3

[15] to select good constraint set. Several active methods for constraint selection have also been proposed, all of which are based on querying the actual underlying class labels. They may collapse if they are restricted to select the constraints from a given constraint set.

In this work, we focus on the problem of feature selection with pairwise constraints from the ensemble perspective with the goal of improving classification accuracy. Bagging is a successful ensemble method based on *bootstrapping* and aggregating concepts, i.e., the training set is randomly sampled many times with replacement to construct several base classifiers which are then aggregated. Inspired by Bagging, we perform multiple Constraint Score on multiple bootstrapped constraints subsets instead of making efforts on finding single constraint set for single Constraint Score. Our algorithm, called Bagging Constraints Score (BCS), constructs individual components using different constraints subsets generated by resampling pairwise constraints in the given constraint set. Although significant work has covered ensemble feature selection [16–18], to the best of our knowledge, no previous feature selection research has tried to build ensemble by exploiting pairwise constraints.

On the other hand, it would be worth mentioning that we have proposed to build ensembles by exploiting pairwise constraints in one of our recent works [19]. A constraint preserving projection was learnt from the pairwise constraints and used to project original instances into a new data representation based on which a set of base classifiers is built. Whereas in this research we propose to use pairwise constraints for feature selection ensemble, which is apparently different from constraint-projection based ensemble method. Moreover, we have compared the performances of both the methods on several real datasets in the experiments.

The rest of this paper is organized as follows. Section 2 introduces some related work on feature selection, learning with pairwise constraints and ensemble learning. Section 3 briefly reviews Constraint Score algorithm as well as two other existing score functions used in supervised and unsupervised feature selections. Then we discuss the details of the proposed BCS algorithm in Section 4, and in Section 5 we report the experimental results on real datasets. Finally, we conclude this paper in Section 6.

2. Related work

2.1. Feature selection

Feature selection methods can be widely categorized into two groups, i.e., (1) filter methods [20] and (2) wrapper methods [21]. The filter methods evaluate the goodness of features by using the intrinsic characteristics of the training data and are independent of any learning algorithm. In contrast, the wrapper methods directly use predetermined learning algorithms to evaluate the features. The wrapper methods usually outperform the filter methods in terms of accuracy, but the former are computationally more expensive than the latter. When dealing with data with huge number of features, the filter methods are usually adopted due to their computational efficiency [22]. In this study, we are particularly interested in the filter methods and consider feature selection from an ensemble view.

Within the filter methods, different feature selection algorithms can further be categorized into two groups [22], i.e., (1) feature ranking methods, (2) subset search methods. The feature ranking methods evaluate the goodness of features individually and obtain a ranked list of selected features ordered by their goodness [23–26]. Laplacian Score [8] and Fisher Score [4,8] are two typical feature ranking methods widely used in feature selection methods. To be specific, Laplacian Score is unsupervised and does not use any class labels, but Fisher Score is supervised and uses all class labels. The former evaluates a feature by its power of locality preserving, but the latter seeks feature subsets which preserve the discriminative ability of a classifier. On the other hand, the subset search methods evaluate the goodness of each candidate feature subset and select the optimal one among them according to some evaluation measures. There are lots of evaluation measures, i.e., consistency, correlation, information measure, combined with many search strategies, i.e. complete, random and heuristic search, to give birth to different algorithms [22,27,28]. The common characteristics of those algorithms are time-consuming and cannot easily scale to very high-dimensional datasets. In contrast, feature ranking methods are very scalable to datasets with both a huge number of instances and a very high dimensionality [22]. The reason is that the feature ranking methods consider features individually while the subset selection methods consider feature selection as a combinatorial problem. Recently, Peng et al. [29] proposed a two-stage feature selection algorithm which combines the minimal-redundancy and maximal-relevance criterion (mRMR) with subset search methods.

2.2. Learning with pairwise constraints

Pairwise constraints arise naturally in many tasks such as image retrieval [12,30,31]. In those applications, considering the pairwise constraints are more practical than trying to obtain class labels, because the true labels may not be known *a priori*, but it could be easier for a user to specify whether some pairs of instances belong to the same class or not. Unlike the class labels, pairwise constraints can be derived from labeled data but not vice versa, and sometimes can be automatically obtained without human intervention. For those reasons, pairwise constraints have been widely used in distance metric learning [32], semi-supervised clustering [33], etc. In clustering tasks, Wagstaff [34] first shows that using randomly selected pairwise constraints can both increase clustering accuracy and decrease runtime.

Constraints can be easily generated with minimal effort or even automatically in some specific domains [12,35]. One example exists in the object recognition of the video sequences data. In the temporally successive frames of video, one can add must-link constraints between objects those occur in different frames but roughly the same location, as long as the scenes are the same. Similarly, the cannot-link constraints can be added between two objects which are in different locations but the same frame.

However, many studies have shown that the performances of most constraint-based methods are severely influenced by the composition and cardinality of constraint set [15,36–39]. Davidson et al. [36] firstly attempted to measure constraint set utility

for partitional clustering algorithms and explain the reason for uselessness or even adverse effects of some constraint sets. Several constraint selection methods have also been proposed. Basu et al. [37] proposed to actively select informative pairwise constraints utilizing the farthest-first traversal scheme. Melville et al. [38] and Greene et al. [39] identified informative constraints in an ensemble way. However, all those algorithms are based on querying the actual underlying class labels. If they are restricted to select the constraints from an initial given constraint set, those active methods for constraint selection may fail. In fact, in many applications it is not always possible for an algorithm to actively select constraints because it will need human intervention.

2.3. Ensemble learning

Ensemble learning improves generalization performance of individual learners by combining the outputs of a set of diverse base classifiers. Previous theoretical and empirical researches have shown that an ensemble is always more accurate than individual components in the ensemble, if and only if individual members are both accurate and diverse [40,41].

Lots of methods have been developed for constructing classification ensembles. The most popular techniques are *Bagging* [42], *Boosting* [43], and the *Random Subspace* methods [44]. Both Bagging and Boosting train base classifiers by resampling training sets, while the Random Subspace method utilizes the random selection of feature subspaces to construct individual classifiers. These classifiers are usually combined by simple majority voting in the final decision rule. One difference between Bagging and Boosting lies in that the former obtains a bootstrap sample by uniformly sampling with replacement from original training set, while the latter resamples or reweighs the training data by emphasizing more on instances that are misclassified by previous classifiers [19]. Recently, besides classification ensemble, there also appears clustering ensemble which combines a diverse set of individual clustering for better consensus solutions [45,46].

3. Constraint Score

In this section, we briefly review the Constraint Score algorithm as well as two other score functions widely used in feature selection methods, namely, Laplacian Score [8] and Fisher Score [4,8]. The Laplacian Score is unsupervised and does not use any class labels, while the Fisher Score is supervised and uses all class labels. In contrast, Constraint Score uses partial supervision in the form of pairwise constraints.

The Laplacian Score is an effective unsupervised feature selection method, which evaluates a feature by its power of locality preserving. Let f_{ri} denote the r-th feature of the i-th sample $\mathbf{x}_i$, $i=1,\ldots,m$; $r=1,\ldots,n$. Define $\mu_r=\frac{1}{m}\sum_i f_{ri}$, then the Laplacian Score of the r-th feature is defined as follows [8]:

$$L_r=\frac{\sum_{i,j}(f_{ri}-f_{rj})^2 S_{ij}}{\sum_i (f_{ri}-\mu_r)^2 D_{ii}} \tag{1}$$

where D is a diagonal matrix with $D_{ii}=\sum_j S_{ij}$, and S_{ij} is defined as follows:

$$S_{ij}=\begin{cases} e^{-\|\mathbf{x}_i-\mathbf{x}_j\|^2/t} & \text{If } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are neighbors,} \\ 0 & \text{Otherwise.} \end{cases} \tag{2}$$

where t is a constant to be set and '$\mathbf{x}_i$ and $\mathbf{x}_j$ are neighbors' means either $\mathbf{x}_i$ is among k nearest neighbors of $\mathbf{x}_j$, or $\mathbf{x}_j$ is among k nearest neighbors of $\mathbf{x}_i$.

The Fisher Score is a supervised feature selection method using all class labels and seeks feature subsets which preserve the discriminative ability of a classifier. Let μ_r^i and $(\sigma_r^i)^2$ be the mean and variance of class i, $i=1,\ldots,c$, corresponding to the r-th feature and n_i is the number of samples in class i. Then the Fisher Score of the r-th feature is defined as follows [8]:

$$F_r=\frac{\sum_{i=1}^{c} n_i(\mu_r^i-\mu_r)^2}{\sum_{i=1}^{c} n_i(\sigma_r^i)^2}. \tag{3}$$

Inspired by both Laplacian Score and Fisher Score, we proposed Constraint Score [14], which used only partial supervision information in the form of pairwise constraints to seek the most representative feature subsets. The basic idea of Constraint Score is to select features with the best constraint preserving ability.

Given a set of data samples $\mathbf{X}=[\mathbf{x}_1, \mathbf{x}_2,\ldots,\mathbf{x}_m]$, and some supervision information in the form of pairwise must-link constraints $M=\{(\mathbf{x}_i, \mathbf{x}_j)|\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same class$\}$ and pairwise cannot-link constraints $C=\{(\mathbf{x}_i, \mathbf{x}_j)|\ \mathbf{x}_i$ and $\mathbf{x}_j$ belong to the different classes$\}$. Let f_{ri} denotes the r-th feature of the i-th sample $\mathbf{x}_i$, $i=1,\ldots,m$; $r=1,\ldots,n$. For evaluating the score of the r-th feature using the constraints in M and C, the two different Constraint Scores of the r-th feature, C_r^1 and C_r^2, which should be minimized, are defined as follows [14]:

$$C_r^1=\frac{\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in M}(f_{ri}-f_{rj})^2}{\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in C}(f_{ri}-f_{rj})^2}, \tag{4}$$

$$C_r^2=\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in M}(f_{ri}-f_{rj})^2-\lambda\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in C}(f_{ri}-f_{rj})^2. \tag{5}$$

The intuition of Eqs. (4) and (5) is simple and natural. That is, we want to select features with the best constraint preserving ability. Specifically, if there is a must-link constraint between two data samples, a 'good' feature should be the one on which those two data samples are close to each other; on the other hand, if there is a cannot-link constraint between two data samples, a 'good' feature should be the one on which those two samples are far away from each other. Since the distance between instances in the same class is typically smaller than that in different classes, a regularization coefficient λ is set in Eq. (5) to balance the two terms of it.

In [14], algorithm using Eq. (4) is denoted as Constraint Score-1, and algorithm using Eq. (5) is denoted as Constraint Score-2. The details of Constraint Score algorithms are shown in Fig. 1.

4. Bagging Constraint Score

Constraint Score is a new filter method for feature selection based on pairwise constraints. It has been shown that, with only a

Algorithm: Constraint Score

Input: Dataset $\mathbf{X}=[\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_m]$;

must-link constraint set $M=\{(\mathbf{x}_i, \mathbf{x}_j)\}$,

cannot-link constraint set $C=\{(\mathbf{x}_i, \mathbf{x}_j)\}$,

λ (for **Constraint Score-2** only),

Output: The ranked feature list

Step 1: For each of the n features, compute its constraint score using Eq. (4) (for **Constraint**

Score-1) or Eq. (5) (for **Constraint Score-2**);

Step 2: Rank the features according to their constraint scores in ascending order.

Fig. 1. The Constraint Score algorithm [14].

small amount of constraints, Constraint Score can achieve comparable performance to Fisher Score using full class labels, and significantly outperforms unsupervised feature selection methods such as Laplacian Score. However, as we have mentioned above, the performance of Constraint Score is notably influenced by the composition and cardinality of constraint set as in most constraint-based learning algorithms. That is, it often results in highly unstable results when different sets of pairwise constraints are used. However, to the best of our knowledge, choosing the most appropriate constraint sets for different algorithms and tasks is still very challenging. In addition, in many applications it is not always possible for an algorithm to actively select constraints because it will need the human intervention. So a more practical question is can we make the best use of the available pairwise constraints for learning when the constraint set is given in advance.

For that purpose, in this research we use pairwise constraints in an ensemble approach. Firstly, pairwise constraints in the given constraint set are divided into several constraints subsets by bootstrapping, i.e., random sampling with replacement. Then multiple processes of Constraint Score are carried out on those individual constraints subsets to select corresponding feature subset. Finally, the individual components are constructed by utilizing those different feature subsets. Our method is motivated by the following facts: (1) pairwise constraints guided feature selection method, namely, Constraint Score, usually achieves better performance than corresponding unsupervised feature selection and is comparable to supervised feature selection; (2) classification accuracy after Constraint Score is very unstable when different constraint sets are used, i.e., they have a large diversity along different constraint sets. It is well known that the effectiveness of an ensemble of learners depends on both the accuracy and the diversity of the individual components. Because our method has the potential to simultaneously improve both the accuracy and the diversity of individuals, we expect it should achieve better ensemble solution. The experimental results presented in the rest of this paper validate the motivation behind our method. The details of the proposed Bagging Constraint Score (BCS) algorithm are described as follows.

We are given a set of labeled data $\mathbf{X}=\{\mathbf{x}_i\}_{i=1}^m$, corresponding class labels $\mathbf{Y}=\{\mathbf{y}_i\}_{i=1}^m$, a set of must-link constraints M and cannot-link constraints C. We assume the ensemble size is L, the number of selected features is n_f. At first, we take L bootstrapped samples $\{M^b\}_{b=1}^L$ and $\{C^b\}_{b=1}^L$ (the random selection with replacement) of the must-link constraint set M and cannot-link constraint set C, respectively. Consequently, we construct L individual components using these L constraint sets. To be specific, for each of the L constraint sets (including both must-link and cannot-link constraint sets), the Constraint Score algorithm is performed to compute the scores of all features, and then the features are ranked according to their scores in ascending order. After that, the dataset is projected to subspace by selecting the former n_f features. Then, a base classifier is built on the projected dataset $\mathbf{Z}=\{\mathbf{z}_i\}_{i=1}^m$. Finally, by combining the outputs of L base classifiers (here we adopt majority vote), we can expect better performance on stability and accuracy. Following Constraint Score in [14], we denote Bagging Constraint Score (BCS) algorithm using Eqs. (4) and (5) as Bagging Constraint Score-1 (BCS-1) and Bagging Constraint Score-2 (BCS-2), respectively. Fig. 2 summarizes the whole procedure of the proposed BCS algorithm.

Now we analyze the time complexity of Bagging Constraint Score. First, we assume that the number of pairwise constraints used in BCS is l, which is bounded by $0<l<O(m^2)$. The main procedures of BCS have two parts: (1) feature selection part consisting of steps 1(a)–(c); (2) learning part. Here we are

Algorithm: Bagging Constraint Score (BCS)

Input:

training data $\mathbf{X}=\{\mathbf{x}_i\}_{i=1}^m$,

base learning algorithm Learner,

must-link constraint set $M=\{(\mathbf{x}_i,\mathbf{x}_j)\}$, cannot-link constraint set $C=\{(\mathbf{x}_i,\mathbf{x}_j)\}$,

λ (for **Constraint Score-2** only), number of selected features n_f, ensemble size L,

Class labels $\mathbf{Y}=\{\mathbf{y}_i\}_{i=1}^m$ corresponding to X

Method:

For $b=1, 2, \ldots, L$;

a) Take a bootstrapped sample M^b of the must-link constraints set M and a bootstrapped sample C^b of the cannot-link constraints set C;

b) For each of the n features, compute its constraint score using Eq. (4) (for **Bagging Constraint Score-1**) or Eq. (5) (for **Bagging Constraint Score-2**);

c) Rank the features according to their constraint scores in ascending order and get the training data set $\mathbf{Z}=\{\mathbf{z}_i\}_{i=1}^m$ by selecting the former n_f features;

d) Call Learner, providing it with the training data set;

e) Get a hypothesis $h_b: \mathbf{Z}\rightarrow\mathbf{Y}$;

Endfor

Output:

the final hypothesis $h_f(\mathrm{x})=\arg\max_{y\in Y,\, z\in Z}\sum_{b:h_b(z)=y}1$

Fig. 2. The proposed BCS algorithm.

interested in time complexity of the feature selection part of BCS, which contains: (1) sampling pairwise constraints L times needs $O(Ll)$ operations; (2) on L constraints subsets, Constraint Score is performed to evaluate the n features, which needs totally $O(nLl)$ operations; (3) ranking n features needs totally $O(nL\log n)$ operations. Hence, the overall time complexity of the feature selection part of BCS is $O(Ll+nL\max(l,\log n))$. In contrast, the time complexities of Constraint Score and Fisher Score are respectively $O(n\max(l,\log n))$ and $O(n\max(c,\log n))$.

It's noteworthy that our proposed BCS algorithm is different from the trivial combination of Constraint Score and Bagging. For comparison, we also implement trivial Constraint Score+Bagging algorithm (CSB), which creates an ensemble based on different bootstrapped instance sets with the same features obtained from one Constraint Score. The detailed procedure of CSB is as follows. At first, we rank the features in ascending order by performing Constraint Score using all available pairwise constraints in given constraint set to select the former n_f features. Then we sample the training set with only the above selected n_f features, to generate the bootstrapped instance sets. At last, we construct classifiers on each of these bootstrapped instance sets and aggregate them by a simple majority vote. Similarly, we denote CSB algorithms based on Constraint Score-1 and Constraint Score-2 as CSB-1 and CSB-2, respectively. The differences between BCS and CSB algorithms are apparent. Firstly, in BCS, Bagging is used in sampling pairwise constraints instead of sampling data instances, which is used in CSB. Secondly, multiple Constraint Score corresponding to multiple bootstrapped constraints subsets are carried out instead of single Constraint Score with single given constraint set which is used in CSB. The experimental results in the later section validate the great superiority of our algorithm over the trivial CSB method.

5. Experimental results

In order to investigate the performance of our proposed Bagging Constraint Score algorithm (BCS), an empirical study is

conducted. Experiments are conducted on several high-dimensional datasets taken from the UCI machine learning repository [47] and some gene expression databases [48]. We assume that pairwise constraints are given in advance and are obtained by considering the underlying class labels of pairs of instances. Besides, we use equal numbers of must-link and cannot-link constraints just following Constraint Score proposed in our previous paper [14], where it has been shown that in most cases using equal numbers of must-link and cannot-link constraints are more helpful than using unbalanced must-link and cannot-constraints. The results of algorithms using constraints are averaged over 100 independent runs with different sets of constraints. It's worth noting that in each round, the randomly selected constraints in all algorithms are the same. The parameters in Eq. (5) are always set to λ=0.1, if without extra explanations.

The performances of all algorithms are measured by the classification accuracy using selected features on testing data. For each dataset, we choose the first half of samples from each class as the training data, and the remaining data for testing except for those having a predefined partition of the objects into training and testing subsets. Moreover for UCI datasets, besides using the fixed training and test sets, we also test performances of different algorithms using 10 times 10-fold cross validation (CV). The classifiers are the nearest neighborhood (NN) and support vector machine (SVM) [49] classifiers. The ensemble size, i.e., the number of base classifiers, is set as 10 if without extra explanations. Furthermore, if there is a tie, we assign the test

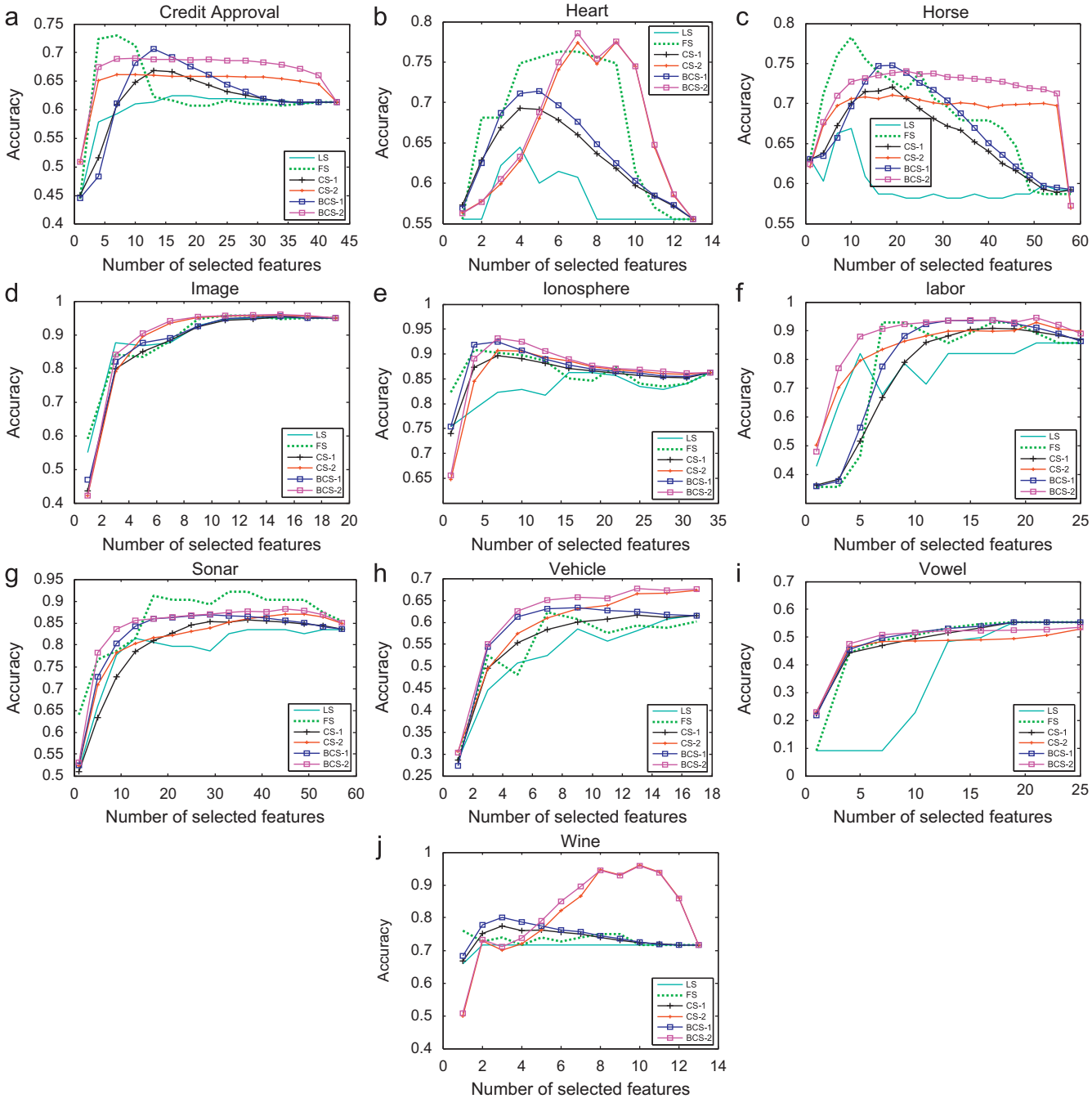

Fig. 3. Accuracy vs. different number of selected features on 10 UCI datasets. The ensemble size is 10 and 60 pairwise constraints including 30 must-link and 30 cannot-link constraints are used.

instance to the randomly selected class. Each of the base classifiers uses the same number of selected features.

5.1. Results on UCI datasets

First, we evaluate our proposed BCS (BCS-1 and BCS-2) algorithms on 10 UCI datasets which have relatively high dimensions from UCI repository. The statistics are presented in Table 1.

5.1.1. Comparison with feature selection methods

First we compare our proposed BCS (including BCS-1 and BCS-2) with the following four feature selection algorithms: (1) Laplacian Score (LS) [8]; (2) Fisher Score (FS) [8]; (3) Constraint Score-1 (CS-1) [14]; (4) Constraint Score-2 (CS-2) [14]. For each dataset, a total of 60 pairwise constraints including 30 must-link and 30 cannot-link constraints are used in CS-1, CS-2, BCS-1 and BCS-2 methods. The number of constraints is still very small compared with the total number of possible constraints that can be generated from training data.

Fig. 3 plots the curves of the six algorithms for accuracy vs. different numbers of selected features. And in Table 2, we record the highest accuracy of all algorithms after optimal dimensions are selected. The numbers of optimal dimensions are also presented in Table 2, and we can see that most methods tend to best (or nearly best) when less than half the number of features are selected, except for the CS-2 and BCS-2 on *Wine* where more features are needed to achieve the best performances. It can be seen from both Fig. 3 and Table 2 that, almost in all cases the ensemble methods, i.e. BCS-1 and BCS-2, achieve better performances than those of single methods including LS, CS-1 and CS-2, respectively and even outperform the fully supervised Fisher Score method in several cases. These results show that the proposed BCS method can utilize constraints in a more effective

Table 2
Highest accuracy of different algorithms on UCI datasets (the numbers in the brackets represent the optimal features).

Datasets	LS	FS	CS-1	CS-2	BCS-1	BCS-2
Credit Approval	62.5(16)	**73.8(5)**	66.9(13)	66.2(9)	70.6(12)	69.4(11)
Heart	64.4(4)	76.3(6)	69.3(4)	77.5(9)	71.4(5)	**78.6(7)**
Horse	69.6(12)	**78.8(21)**	72.2(12)	71.1(17)	74.8(17)	74.2(20)
Image	95.0(12)	95.7(11)	95.4(15)	95.5(16)	95.8(13)	**96.0(13)**
Ionosphere	86.3(16)	91.4(6)	89.7(8)	90.5(8)	92.9(6)	**93.6(8)**
Labor	85.7(20)	92.9(7)	90.9(18)	92.5(21)	94.1(16)	**94.6(15)**
Sonar	84.5(14)	**93.2(42)**	85.8(37)	87.5(44)	87.3(26)	89.0(45)
Vehicle	61.9(18)	64.7(8)	61.9(18)	67.4(17)	63.3(9)	**68.5(13)**
Vowel	55.3(19)	**57.2(12)**	55.3(18)	55.3(27)	55.3(18)	55.3(27)
Wine	71.6(2)	76.1(1)	77.4(3)	96.5(10)	80.0(3)	**96.6(10)**

Table 3
Highest accuracy of different algorithms on UCI datasets (CV, NN).

Datasets	LS	FS	CS-1	CS-2	BCS-1	BCS-2
Credit Approval	67.2 ± 5.1	$\mathbf{79.7 \pm 5.0}$	70.2 ± 2.2	70.2 ± 3.5	73.1 ± 2.4	72.8 ± 3.5
Heart	65.5 ± 7.8	79.5 ± 8.7	70.2 ± 5.9	80.2 ± 7.9	72.4 ± 6.2	$\mathbf{81.7 \pm 7.7}$
Horse	71.8 ± 6.9	$\mathbf{82.6 \pm 4.9}$	74.8 ± 3.7	78.4 ± 4.3	77.5 ± 3.8	80.5 ± 4.0
Image	96.5 ± 1.5	97.1 ± 2.0	96.7 ± 1.8	97.0 ± 1.6	97.3 ± 1.9	$\mathbf{98.0 \pm 1.8}$
Ionosphere	92.7 ± 4.2	$\mathbf{94.2 \pm 2.7}$	90.7 ± 2.6	91.0 ± 2.3	92.2 ± 2.8	93.3 ± 2.7
Labor	93.9 ± 13.3	98.1 ± 6.3	95.5 ± 7.2	97.2 ± 6.3	96.3 ± 7.3	$\mathbf{99.0 \pm 6.4}$
Sonar	87.2 ± 8.5	$\mathbf{93.4 \pm 7.9}$	85.8 ± 8.4	86.6 ± 7.8	87.2 ± 7.7	89.5 ± 7.6
Vehicle	68.6 ± 4.2	67.7 ± 5.0	65.5 ± 5.8	69.0 ± 5.4	66.4 ± 5.3	$\mathbf{71.5 \pm 5.5}$
Wine	77.5 ± 9.5	83.6 ± 7.6	82.5 ± 7.1	95.9 ± 2.8	83.7 ± 5.8	$\mathbf{97.0 \pm 2.4}$
BCS-1 vs. others	5-0-4	1-2-6	2-0-7	4-0-5	–	0-4-5
BCS-2 vs. others	5-0-4	2-2-5	4-0-5	3-0-6	4-0-5	–

Bottom rows of the table present Win–Loss–Tie (W–L–T) comparisons between BCS (BCS-1 and BCS-2) against other approaches after t-tests at 95% significance level.

Table 4
Highest accuracy of different algorithms on UCI datasets (CV, SVM).

Datasets	LS	FS	CS-1	CS-2	BCS-1	BCS-2
Credit Approval	68.6 ± 3.7	85.9 ± 3.8	77.7 ± 2.6	83.5 ± 3.6	79.1 ± 2.4	$\mathbf{87.6 \pm 3.5}$
Heart	63.4 ± 9.8	83.9 ± 6.3	73.4 ± 4.6	84.5 ± 6.2	75.9 ± 5.7	$\mathbf{85.8 \pm 6.0}$
Horse	73.5 ± 7.1	$\mathbf{88.0 \pm 4.2}$	81.2 ± 4.3	84.8 ± 3.1	81.2 ± 5.0	86.6 ± 2.9
Image	87.9 ± 2.2	90.3 ± 2.0	88.2 ± 1.8	89.0 ± 1.6	88.7 ± 1.9	$\mathbf{91.5 \pm 1.8}$
Ionosphere	94.3 ± 5.2	94.5 ± 4.9	94.0 ± 4.4	94.3 ± 4.1	94.6 ± 4.4	$\mathbf{95.1 \pm 4.2}$
Labor	95.7 ± 8.4	99.5 ± 0	98.0 ± 2.2	99.8 ± 0	98.3 ± 1.4	$\mathbf{99.8 \pm 0}$
Sonar	74.0 ± 6.1	$\mathbf{83.1 \pm 9.4}$	68.0 ± 6.4	68.3 ± 6.4	69.1 ± 6.7	70.0 ± 7.3
Vehicle	$\mathbf{67.0 \pm 4.5}$	60.2 ± 3.9	51.9 ± 3.2	57.9 ± 3.8	56.2 ± 3.9	60.5 ± 3.8
Wine	52.6 ± 10.9	78.8 ± 10.6	70.6 ± 6.3	96.5 ± 2.7	73.9 ± 5.9	$\mathbf{97.6 \pm 2.7}$
BCS-1 vs. others	5-1-3	0-4-5	0-0-9	0-4-5	–	0-5-4
BCS-2 vs. others	6-1-2	2-2-5	5-0-4	2-0-7	5-0-4	–

Bottom rows of the table present Win–Loss–Tie (W–L–T) comparisons between BCS (BCS-1 and BCS-2) against other approaches after t-tests at 95% significance level.

way than Constraint Score. Fig. 3 and Table 2 also indicate that BCS-2 often performs better than BCS-1 just like CS-2 which often performs better than CS-1 due to the use of regularization. It is noteworthy that FS achieves performance superior to both BCS-1 and BCS-2 in 4 of 10 cases which proves the importance of supervised information.

Tables 3 and 4 give the highest accuracies plus the standard deviations of different algorithms under 10 times 10-fold cross validation using NN and SVM classifiers, respectively. Here we only perform experiments on nine UCI datasets because the dataset *Vowel* has a predefined partition of training and test datasets. We also give the significance test results at the bottom

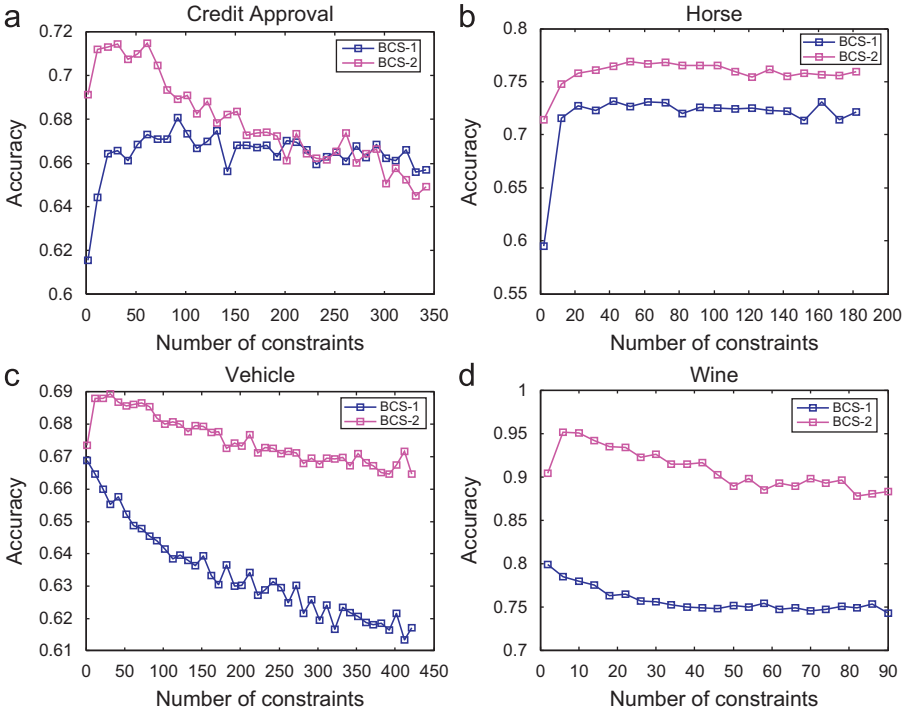

Fig. 4. Accuracy vs. different number of pairwise constraints used in each individual component on four UCI datasets.

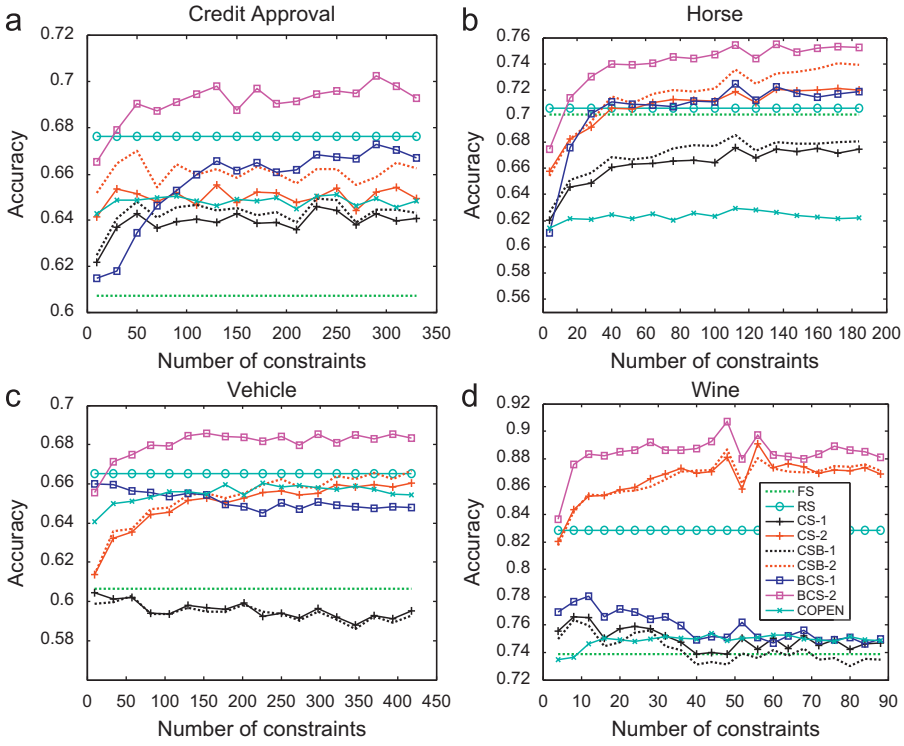

Fig. 5. Accuracy vs. different number of pairwise constraints (for CS-1, CS-2, CSB-1, CSB-2, BCS-1 and BCS-2) on four UCI datasets. The ensemble size is 10 and the desired number of selected features is half of the number of original features for all methods.

rows of both tables. From Tables 3 and 4, the analogous trend between BCS and other feature selection methods can be observed as in Table 2, i.e., BCS-2 and FS usually have better performances than all other methods. It can also be seen from Tables 2 and 3 that nearly all algorithms obtain better performances using 10-fold CV than using equally-divided training and test sets. Finally, the results in Tables 3 and 4 show that the improvements of our BCS methods over other approaches are not only applied to NN classifier, but also are applicable to other classifiers.

5.1.2. Comparison with ensemble learning methods

In this subsection, we investigate comparisons between our BCS and other ensemble learning methods, including Random Subspace (RS) [44] and the trivial Constraint Score+Bagging (denoted as CSB, including CSB-1 and CSB-2 following CS-1 and CS-2, respectively) methods introduced in the end of Section 4. As Ho [44] shows that in RS, using half of the feature components yield the best accuracy, in the experiments we fix the desired number of selected features as half of the number of original features for all methods.

On the other hand, besides constructing ensemble based on feature selection, in our recent work, we have proposed a method called COPEN which uses pairwise constraints for classifier ensemble [19]. In COPEN, we first use the constraint projection (CP) method to transfer information in pairwise constraints into new data representation, and then construct a set of base classifiers under that new representation. For completeness, we also compare BCS with COPEN.

For each dataset, we set the number of total given constraints as the size of the training dataset (denoted as m), and vary the number of constraints used in each individual component from two to m. Intuitively, it is disadvantageous to large datasets when the individual components select the same number of constraints as the size of training data. Especially as the ensemble size is not increased corresponding to the increment of the number of constraints. To verify our intuition, we observe the accuracy when varying the number of constraints used in each individual component and the results are given in Fig. 4.

From Fig. 4, we can clearly see that, when the number of constraints is large, the accuracy declines as the number of each individual component used increases, which is especially apparent on the large datasets, e.g. *Credit Approval* and *Vehicle*. Thus on the two datasets, we experimentally set the number of constraints used in each individual component as 20% of the total given constraints. Fig. 5 shows the accuracy under fixed number of selected features with different numbers of pairwise constraints on four UCI datasets (from two to the number of training data).

Fig. 5 demonstrates that BCS-2 significantly outperforms the other algorithms even if a few constraints are used. Generally, the performance of BCS-2 increases fast in the beginning (with few constraints) and is less insensitive to the number of constraints with relatively enough constraints. BCS-1 offers better performance than those of CS-1 and CSB-1 in most cases, but worse than that of BCS-2. On *Credit Approval* and *Horse*, CSB-1 and CSB-2 are a bit superior to CS-1 and CS-2, respectively. However, the situation on other two datasets is almost on the contrary, which is especially apparent between CSB-1 and CS-1. In almost all cases both CSB-1 and CSB-2 are inferior to BCS-1 and BCS-2, respectively. Although Fisher Score utilizes all labeled data, in most cases its performance is inferior to BCS and CSB, which proves again the effectiveness of ensemble methods. The Random Subspace method, which selects features randomly without any form of supervision information, is naturally inferior to BCS-2. Finally, from Fig. 5, we can see that BCS is distinctly superior to COPEN, which suggests that selecting feature subsets for ensemble is better than projecting features into new subspaces on those four datasets.

It's noteworthy that when the numbers of constraints are too small, both CS-1 and BCS-1 perform poorly on *Credit Approval* and *Horse*. It implies that under those extreme cases, BCS-1 can help little in improving the performances of CS. A similar but not so obvious phenomenon can also be found between CS-2 and BCS-2. In other words, the 'cardinality' problem also exists in our BCS method. Fortunately, both BCS and CS are less insensitive to the number of constraints when relatively enough constraints are used. Finally, it's interesting to note that on *Vehicle* and *Wine*, the performances of CS-1, CSB-1 and BCS-1 decrease when more constraints are used.

5.1.3. Comparison with constraint selection methods

In this subsection, we introduce the constraint selection method proposed by Greene et al. [39] to our CS and BCS methods. The goal of that work is identifying informative constraints for semi-supervised clustering. In this study, we just take advantage of their method to select constraints and then employ our CS and BCS methods to do feature selection. It is noteworthy that the settings of the constraint selection method are the same as those in [39], except that we generate an ensemble consisting of 10 members for each dataset other than 2000. Following CS and BCS, we name these new methods ICS (ICS-1 and ICS-2) and BICS (BICS-1 and BICS-2), respectively since the informative constraints are introduced. For consistency, the experimental setup of ICS and BICS is the same as those of CS and BCS, respectively except that the given constraints are selected actively rather than randomly.

Fig. 6 shows the plots of accuracy under fixed number of selected features vs. different numbers of pairwise constraints on two UCI datasets. It can be seen from Fig. 6 that, for *Credit*

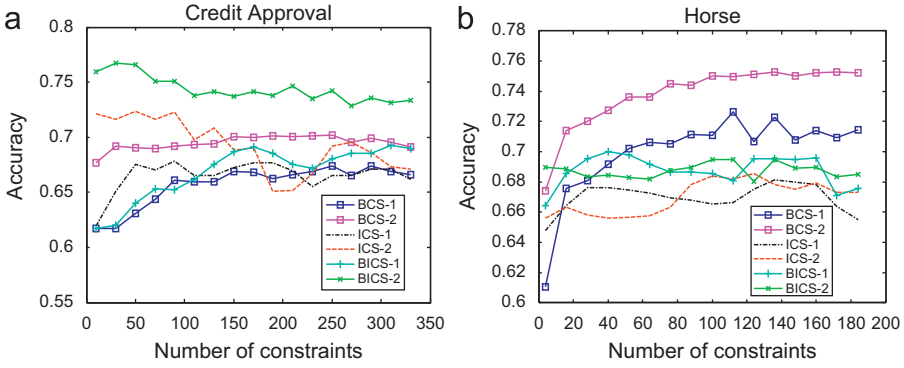

Fig. 6. Accuracy vs. different number of pairwise constraints on *Credit Approval* and *Horse*. The ensemble size is 10 and the desired number of selected features is half of the number of original features for all methods.

Approval the performance of BCS is comparable to that of ICS. For *Horse*, BCS is superior to ICS. Fig. 6 also indicates that in most cases our BCS method which randomly selects constraints is superior to ICS method which actively selects constraints. On the other hand, comparing BICS-1 with BCS-1 and BICS-2 with BCS-2, respectively, we can see that ensemble feature selection methods with actively selected constraints are not always superior to those with randomly selected constraints. Besides, BICS-1 significantly improves ICS-1 on *Horse*, and BICS-2 significantly improves ICS-2 on *Credit Approval*. It implies that our notion of ensemble is propitious to feature selection with not only randomly selected constraints but also actively selected constraints. Finally, it's interesting to note that on *Horse*, both ICS and BICS with active constraints selection are much inferior to BCS with random constraints selection.

Table 5
Statistics of the gene expression databases.

Datasets	Size	Dimension	# of classes
Colon Cancer	62	2000	2
Leukemia	72	7129	2

5.2. Results on gene expression databases

In this subsection, several experiments are carried out on two gene expression databases with huge number of features, i.e., *Colon Cancer* [48] and *Leukemia* [50], whose statistics are presented in Table 5. As before, a total of 60 pairwise constraints including 30 must-link and 30 cannot-link constraints are used in both CS and BCS. Because *Leukemia* has a predefined partition of the objects into training and testing

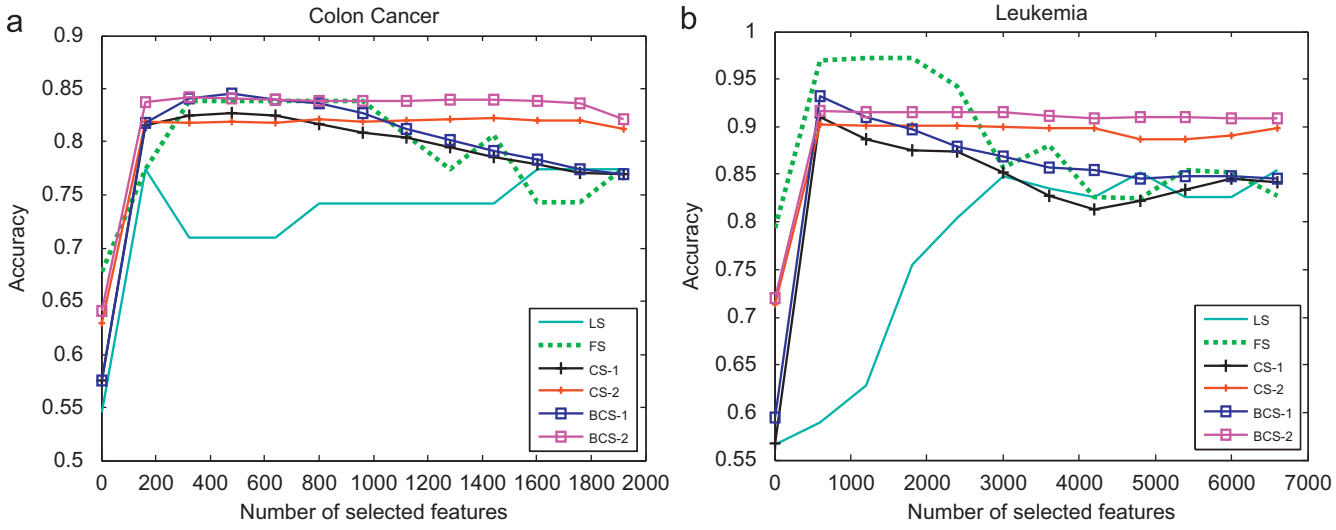

Fig. 7. Accuracy vs. different number of selected features on gene expression databases. The ensemble size is 10 and 60 pairwise constraints including 30 must-link and 30 cannot-link constraints are used.

Table 6
Highest accuracy of different algorithms on gene expression databases, bold texts represent the corresponding numbers of selected features.

Datasets	LS	FS	CS-1	CS-2	BCS-1	BCS-2
Colon Cancer	78.0(165)	83.9(665)	82.9(533)	82.3(533)	84.5(425)	**85.7(220)**
Leukemia	85.3(2796)	**97.0(516)**	92.4(375)	90.5(469)	93.1(443)	92.3(608)

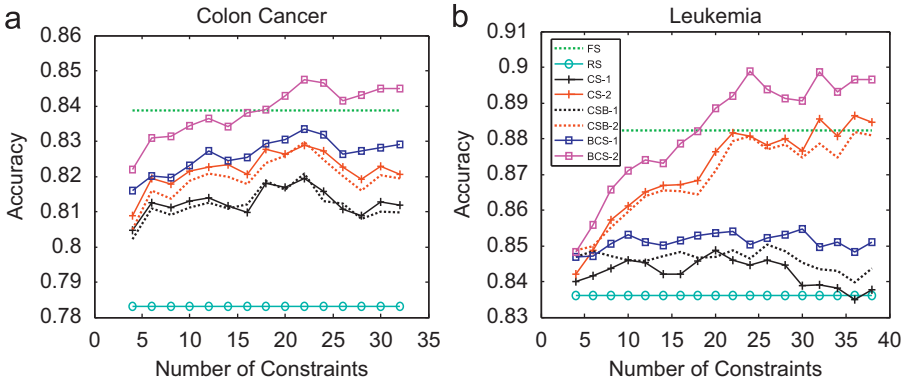

Fig. 8. Accuracy vs. different number of pairwise constraints (for CS-1, CS-2, CSB-1, CSB-2, BCS-1 and BCS-2) on gene expression databases. The ensemble size is 10 and the desired number of selected features is half of the number of original features for all methods.

subsets, the ensembles on this dataset are performed on the predefined training and testing sets.

Fig. 7 shows the plots for accuracy vs. different numbers of selected features on *Colon Cancer* and *Leukemia* databases. As shown in Fig. 7, on both the databases, BCS-1 and BCS-2 outperform CS-1 and CS-2, respectively, and both achieve significantly better performances than that of LS again. Moreover, on *Colon Cancer*, the accuracy of BCS-2 is higher than FS in most cases. However, it is noteworthy that on *Leukemia*, FS is superior to other methods when few features are selected.

Furthermore, Table 6 compares the highest accuracy of all algorithms after optimal dimensions are selected. On *Colon Cancer*, it can be observed that BCS achieves significantly better performance than that of CS, and even outperforms the fully supervised Fisher Score method. Though the situation is not so good on *Leukemia*, we can see that BCS-1 and BCS-2 are still superior to CS-1 and CS-2, respectively. The numbers of optimal dimensions are also presented in Table 6. We can see that in most cases selecting too many features is not propitious to the improvement of performance.

Fig. 8 shows the plots for accuracy under fixed number of selected features vs. different numbers of pairwise constraints on two gene expression databases. Like in Fig. 5, the performances of BCS-2 on both gene expression databases are remarkable. No matter how many constraints are used, BCS-2 achieves higher accuracy than CS-1, CS-2, CSB-1, CSB-2 and RS. Besides, the accuracy of BCS-2 is lower than FS in the beginning (with few constraints), but increases fast at the end (with relatively more constraints), which is more apparent on *Leukemia* than on *Colon Cancer*.

5.3. Further discussions

5.3.1. Effect of ensemble size

We compare three types of ensemble methods, i.e., Bagging Constraint Score (including both BCS-1 and BCS-2), Constraint Score Bagging (including both CSB-1 and CSB-2) and the Random Subspace method under a range of ensemble sizes. For BCS and CSB, we set the total number of pairwise constraints as the number of instances. Besides, on *Credit Approval* and *Vehicle*, we set the number of constraints used in each individual component as 20% of the total number of given constraints again. The numbers of selected features are fixed as half of the number of original features. The results on four UCI datasets and two gene expression datasets are shown in Fig. 9.

As can be seen from the diagrams, nearly all ensemble methods have the same trend. That is, they generally obtain

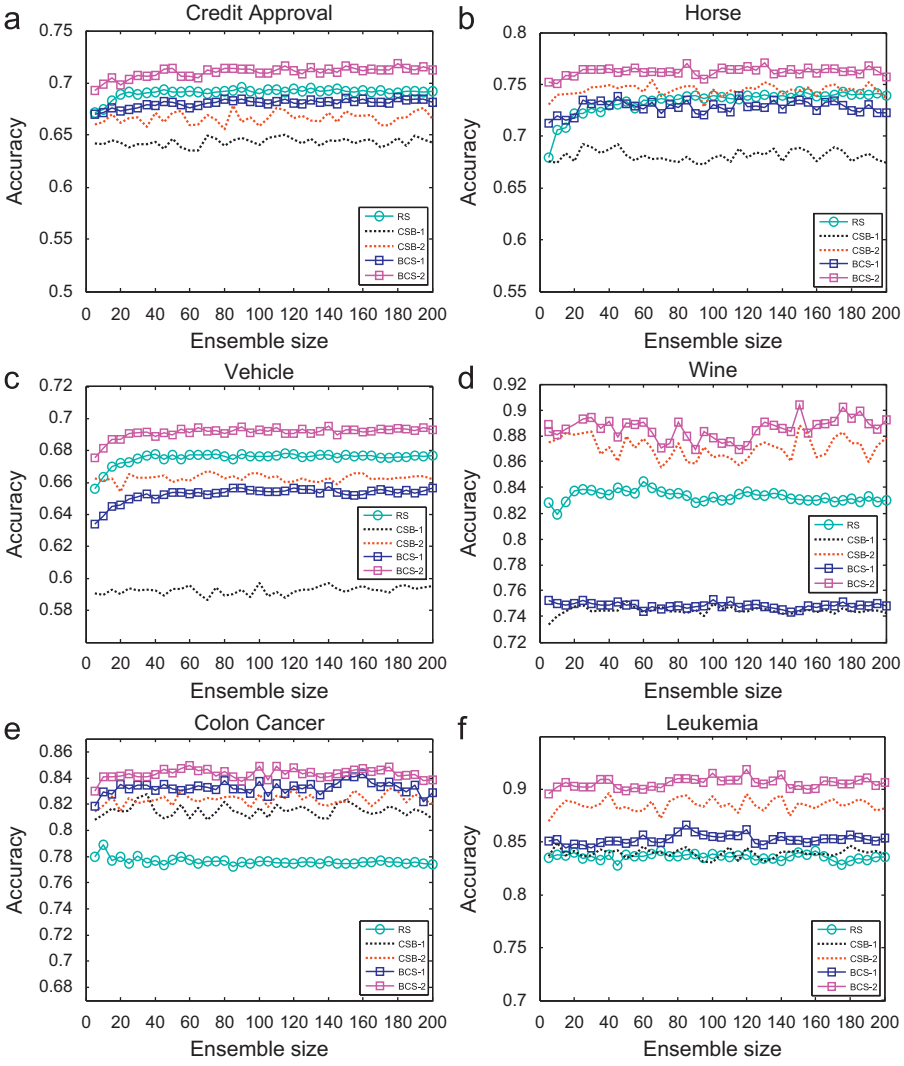

Fig. 9. Accuracy vs. ensemble size on four UCI and two gene expression datasets. The total number of pairwise constraints is the number of instances and the desired number of selected features is half of the number of original features.

improved performances as the ensemble size increases. But after some demarcation point, e.g. the value around 20, the accuracies are less insensitive to the increase of the ensemble size. In all cases, BCS-2 is superior to all other algorithms again.

5.3.2. Analysis of diversity

In order to understand how our ensemble approach works, we use kappa measure to plot diversity–error diagram [51]. Because the CSB methods are apparently inferior to our proposed BCS methods, we only show diversity–error diagrams of BCS-1, BCS-2 and RS on four UCI datasets and two gene expression datasets in Fig. 10. All ensembles consist of 50 individual classifiers. We fix the desired number of selected features as half of the number of original features for all the methods and the total number of pairwise constraints as the number of instances again. On the x-axis of a kappa–error diagram is k for each pair of classifiers in the ensemble and on the y-axis is the averaged individual error E. As small values of k indicate better diversity and small values of E indicate better accuracy, the most desirable pairs of classifiers will close to the bottom left corner of the graph. It is worth noting that on *Wine*, there are only few points corresponding to BCS-1 and BCS-2, which is because there are many classifier pairs having the same diversity and accuracy. For a visual evaluation of the relative positions of the clouds of kappa–error points, we also plot the centroids of the clouds for the respective ensemble methods on each dataset in Fig. 10. Specifically, diamond denotes centroid of RS, dot denotes that of BCS-1, and triangle denotes that of BCS-2.

As can be seen from Fig. 10(a)–(d) that, except on *Wine*, BCS-2 achieves slightly less diversity but much lower error than that of RS. On each UCI dataset, BCS-2 is more accurate and more diverse than BCS-1. BCS-1 gets higher accuracies than those of RS on *Credit Approval* and *Horse*, but lower accuracies on the remaining two datasets. In general, BCS-2 is not as diverse as RS but apparently has the more accurate classifiers than both RS and BCS-1. However, in the former experiments, we have shown that BCS-2 has better performance than RS. It seems that BCS-2 can achieve a better trade-off between accuracy and diversity than other methods. That is, it builds ensemble based on reasonably diverse but markedly accurate individual components [51].

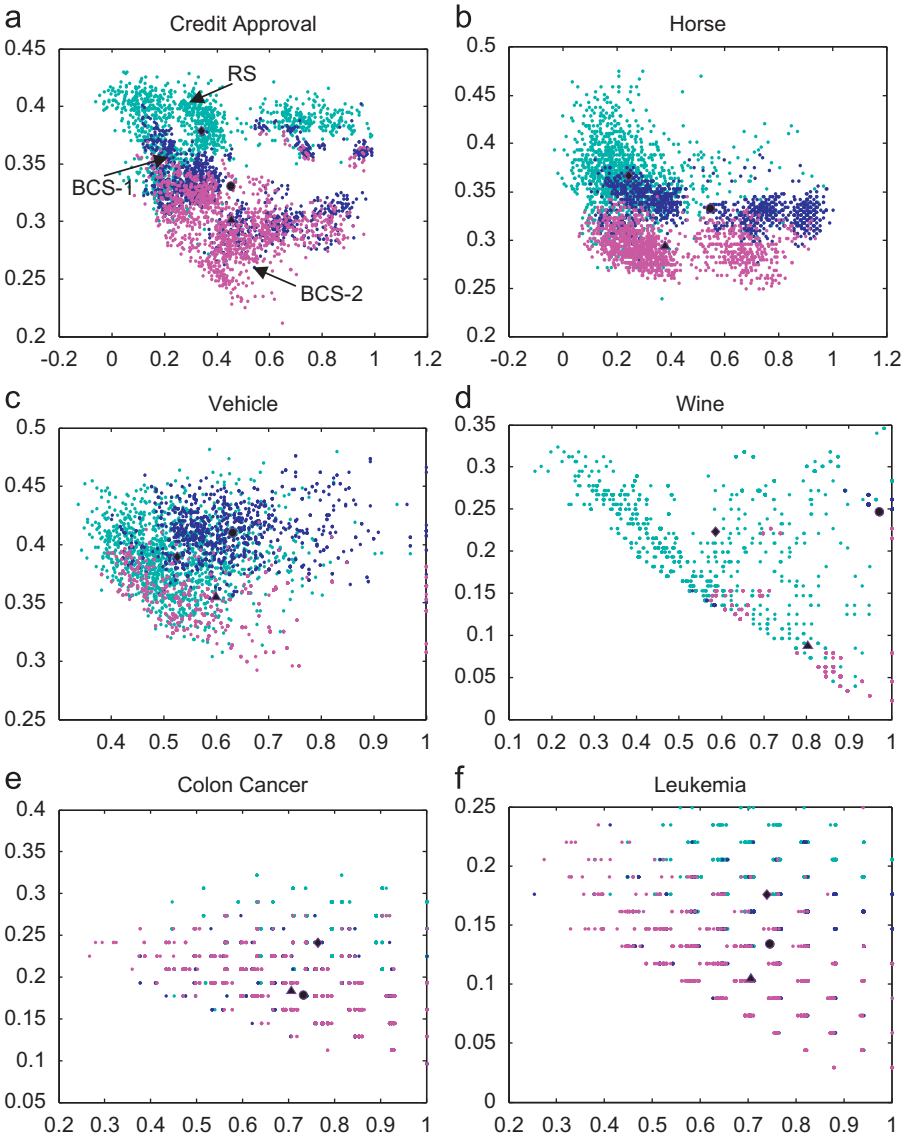

Fig. 10. The diversity–error diagrams on four UCI and two gene expression datasets. The ensemble size is 50. The total number of pairwise constraints is equal to the number of instances and the desired number of selected features is half of the number of original features.

In Fig. 10(e) and (f), the situation is clear. BCS-2 is remarkably more accurate and more diverse than RS. BCS-1 is comparable to BCS-2 on *Colon Cancer* and inferior to it on *Leukemia*. Thus from the aspect of diversity–accuracy of the ensemble, it is not surprising that BCS-2 can achieve the best ensemble performance among the three methods.

6. Conclusion

Constraint Score which uses pairwise constraints for feature selection has shown good performance in our previous work [14]. However, one important yet unresolved problem is how to choose the most appropriate constraint set. In this study, we address this important issue from a different view. Instead of making efforts on choosing constraints for single feature selection, we directly investigate the problem of how to best use the available constraints. We propose an ensemble method called Bagging Constraint Score (BCS), which constructs individual components utilizing different feature subsets obtained by performing multiple Constraint Score on multiple bootstrapped constraints subsets. Extensive experiments on a series of UCI and gene expression databases have verified the effectiveness of our approach.

We have introduced constraint selection method to our BCS method. However, preliminary experimental results show that the ensemble feature selection methods with actively selected constraints are not always superior to those with randomly selected constraints, which deserve further investigation. Besides, the proposed BCS is a general framework which is not restricted to classification task, but can be used for more general tasks such as clustering. Furthermore, the proposed BCS can naturally be extended for semi-supervised cases if we replace the classification and clustering steps with semi-supervised classification [33] and semi-supervised clustering [34], respectively, which deserves further research. Moreover, in the current work we are more concerned on the classification accuracy of BCS and lose sight of its property on selecting features. In the future work, we will include an analysis of the features selected by each individual component of BCS and find effective ways for obtaining a single feature subset by aggregating all feature subsets generated from multiple Constraint Score.

Acknowledgments

We want to thank the anonymous reviewers for their valuable comments, which have improved this paper significantly. This work is supported by National Science Foundation of China under Grant no. 60875030.

References

[1] K. Kira, L. Rendell, The feature selection problem: traditional methods and a new algorithm, in: Proceedings of Ninth National Conference on Artificial Intelligence, 1992, pp. 129–134.
[2] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.
[3] J. Hua, W. Tembe, E. Dougherty, Performance of feature-selection methods in the classification of high-dimension data, Pattern Recognition 42 (3) (2009) 409–424.
[4] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, Journal of Machine Learning Research 5 (2004) 1205–1224.
[5] J. Dy, C. Brodley, A. Kak, L. Broderick, A. Aisen, Unsupervised feature selection applied to content-based retrieval of lung images, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (2003) 373–378.
[6] J. Dy, C.E. Brodley, Feature selection for unsupervised learning, Journal of Machine Learning Research 5 (2004) 845–889.
[7] Z. Zhao, H. Liu, Semi-supervised feature selection via spectral analysis, in: Proceedings of the Seventh SIAM International Conference on Data Mining, Minneapolis, MN, 2007.
[8] X. He, D. Cai, P. Niyogi, Laplacian score for feature selection, in: Advances in Neural Information Processing Systems, vol. 17, MIT Press, Cambridge, MA, 2005, pp. 507–514.
[9] X. He, P. Niyogi, Locality preserving projections, in: Advances in Neural Information Processing Systems, vol. 16, MIT Press, Cambridge, MA, 2004, pp. 153–160.
[10] A. Jain, D. Zongker, Feature selection: evaluation, application, and small sample performance, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (2) (1997) 153–158.
[11] J. Wang, K. Plataniotis, J. Lu, A. Venetsanopoulos, Kernel quadratic discriminant analysis for small sample size problem, Pattern Recognition 41 (5) (2008) 1528–1538.
[12] A. Bar-Hillel, T. Hertz, N. Shental, D. Weinshall, Learning a mahalanobis metric from equivalence constraints, Journal of Machine Learning Research 6 (2005) 937–965.
[13] D. Zhang, Z. Zhou, S. Chen, Semi-supervised dimensionality reduction, in: Proceedings of the Seventh SIAM International Conference on Data Mining, Minneapolis, MN, 2007, pp. 629–634.
[14] D. Zhang, S. Chen, Z. Zhou, Constraint score: a new filter method for feature selection with pairwise constraints, Pattern Recognition 41 (5) (2008) 1440–1451.
[15] K. Wagstaff, Value, cost, and sharing: open issues in constrained clustering, in: International Workshop on Knowledge Discovery in Inductive Databases, 2007, pp. 1–10.
[16] D. Opitz, Feature selection for ensembles, in: Proceedings of the 16th National Conference on Artificial Intelligence, AAAI, 1999, pp. 379–384.
[17] A. Tsymbal, S. Puuronen, I. Skrypnyk, Ensemble feature selection with dynamic integration of classifiers, in: Proceedings of the International ICSC Congress on Computational Intelligence Methods and Applications CIMA'2001, Bangor, Wales, UK, 2001, pp. 558–564.
[18] A. Tsymbal, S. Puuronen, D.W. Patterson, Ensemble feature selection with simple Bayesian classification, Information Fusion 4 (2003) 87–100.
[19] D. Zhang, S. Chen, Z. Zhou, Q. Yang, Constraint projections for ensemble learning, in: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, Chicago, IL, 2008, pp. 758–763.
[20] H. Almuallim, T. Dietterich, Learning with many irrelevant features, in: Proceedings of the Ninth National Conference on Artificial Intelligence, San Jose, 1991, pp. 547–552.
[21] R. Kohavi, G. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1–2) (1997) 273–324.
[22] L. Yu, H. Liu, Feature selection for high-dimensional data: a fast correlation-based filter solution, in: Proceedings of the 20th International Conferences on Machine Learning, Washington DC, 2003, pp. 601–608.
[23] H. Liu, J. Sun, L. Liu, H. Zhang, Feature selection with dynamic mutual information, Pattern Recognition 42 (7) (2009) 1330–1339.
[24] D. Ziou, T. Hamri, S. Boutemedjet, A hybrid probabilistic framework for content-based image retrieval with feature weighting, Pattern Recognition 42 (7) (2009) 1511–1519.
[25] A. Anand, G. Pugalenthi, P. Suganthan, Predicting protein structural class by SVM with class-wise optimized features and decision probabilities, Journal of Theoretical Biology 253 (2) (2008) 375–380.
[26] Y. Li, B. Lu, Feature selection based on loss-margin of nearest neighbor classification, Pattern Recognition 42 (9) (2009) 1914–1921.
[27] S. Nakariyakul, D. Casasent, An improvement on floating search algorithms for feature subset selection, Pattern Recognition 42 (9) (2009) 1932–1940.
[28] H. Yoon, K. Yang, C. Shahabi, Feature subset selection and feature ranking for multivariate time series, IEEE Transactions on Knowledge and Data Engineering 17 (2005) 1186–1198.
[29] H. Peng, F. Long, C. Ding, Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (8) (2005) 1226–1238.
[30] K. Yap, A soft relevance framework in content-based image retrieval systems, IEEE Transactions on Circuits and Systems for Video Technology 15 (2005) 1557–1568.
[31] K. Wu, Fuzzy SVM for content-based image retrieval—A pseudo-label support vector machine framework, IEEE Computational Intelligence Magazine 1 (2006) 10–16.
[32] E. Xing, A.Y. Ng, M. Jordan, S. Russell, Distance metric learning, with application to clustering with side-information, in: Advances in Neural Information Processing Systems, vol. 15, MIT Press, Cambridge, MA, 2003, pp. 505–512.
[33] X. Zhu, Semi-supervised learning literature survey, Technique Report 1530, Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI, 2006. 〈http://www.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf〉.
[34] K. Wagstaff, C. Cardie, Constrained k-means clustering with background knowledge, in: Proceedings of the 17th International Conference on Machine Learning, 2000, pp. 1103–1110.
[35] R. Yan, J. Zhang, J. Yang, A. Hauptmann, A discriminative learning framework with pairwise constraints for video object classification, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, 2004, pp. 284–291.
[36] I. Davidson, K. Wagstaff, S. Basu, Measuring constraint-set utility for partitional clustering algorithms, in: Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases, 2006, pp. 115–126.

[37] S. Basu, A. Banerjee, R. Mooney, Active semi-supervision for pairwise constrained clustering, in: Proceedings of the Fourth SIAM International Conference on Data Mining, 2004, pp. 333–344.
[38] P. Melville, R. Mooney, Diverse ensembles for active learning, in: Proceedings of the 21st International Conferences on Machine Learning, 2004, pp. 584–591.
[39] D. Greene, P. Cunningham, Constraint selection by committee: an ensemble approach to identifying informative constraints for semi-supervised clustering, in: Proceedings of the 18th European Conference on Machine Learning, Warsaw, Poland, 2007, pp. 17–21.
[40] T. Dietterich, Ensemble methods in machine learning, in: Proceedings of the First International Workshop on Multiple Classifier Systems, Springer, Berlin, 2000, pp. 1–15.
[41] K. Liu, Ensemble component selection for improving ICA based microarray data prediction models, Pattern Recognition 42 (7) (2009) 1274–1283.
[42] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.
[43] Y. Freund, R. Schapire, Experiments with a new boosting algorithm, in: Proceedings of the 13th International Conference on Machine learning, 1996, pp. 148–156.
[44] T. Ho, The random subspace method for construction decision forests, IEEE Transactions on Pattern Analysis and Machine Intelligence (1998) 832–844.
[45] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, Journal of Machine Learning Research 3 (2002) 583–618.
[46] A. Topchy, A. Jain, W. Punch, Clustering ensembles: models of consensus and weak partitions, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (12) (2005) 1866–1881.
[47] C. Blake, E. Keogh, C. Merz, UCI repository of machine learning databases, 〈http://www.ics.uci.edu/~mlearn/MLRepository.html〉, Department of Information and Computer Science, University of California, Irvine, 1998.
[48] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, A. Levine, Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays, Proceedings of the National Academy of Sciences 96 (12) (1999) 6745–6750.
[49] C. Chang, C. Lin, LIBSVM: a library for support vector machines, 2001, software available at 〈http://www.csie.ntu.edu.tw/~cjlin/libsvm〉.
[50] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, E. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (1999) 531–537.
[51] J. Rodríguez, L. Kuncheva, C. Alonso, Rotation forest: a new classifier ensemble method, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (10) (2006) 1619–1630.

About the Author—DAN SUN received her bachelor's degree in computer science from Nanjing University of Aeronautics and Astronautics, China, in 2007. She is currently working on Master Degree in Department of Computer Science and Engineering at Nanjing University of Aeronautics and Astronautics. Her research interests include machine learning, pattern recognition and data mining.

About the Author—DAOQIANG ZHANG received B.Sc. and Ph.D. degrees in Computer Science from Nanjing University of Aeronautics and Astronautics, China, in 1999 and 2004, respectively. From 2004 to 2006, he was a postdoctoral fellow in the Department of Computer Science & Technology at Nanjing University. He joined the Department of Computer Science and Engineering of Nanjing University of Aeronautics and Astronautics as a Lecturer in 2004, and is a professor at present. His research interests include machine learning, pattern recognition, data mining, and image processing. In these areas he has published over 40 technical papers in refereed international journals or conference proceedings. He was nominated for the National Excellent Doctoral Dissertation Award of China in 2006, and won the best paper award at the Ninth Pacific Rim International Conference on Artificial Intelligence (PRICAI'06). He has served as a program committee member for several international and native conferences. He is a member of Chinese Association of Artificial Intelligence (CAAI) Machine Learning Society.