Pattern Recognition 46 (2013) 769–787

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

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

Stratified sampling for feature subspace selection in random forests for high dimensional data

Yunming Ye [a,e,*], Qingyao Wu [a,e], Joshua Zhexue Huang [b,d], Michael K. Ng [c], Xutao Li [a,e]

[a] Department of Computer Science, Shenzhen Graduate School, Harbin Institute of Technology, China
[b] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
[c] Department of Mathematics, Hong Kong Baptist University, China
[d] Shenzhen Key Laboratory of High Performance Data Mining, China
[e] Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen, China

ARTICLE INFO

Article history:
Received 3 November 2011
Received in revised form
1 September 2012
Accepted 3 September 2012
Available online 12 September 2012

Keywords:
Stratified sampling
High-dimensional data
Classification
Ensemble classifier
Decision trees
Random forests

ABSTRACT

For high dimensional data a large portion of features are often not informative of the class of the objects. Random forest algorithms tend to use a simple random sampling of features in building their decision trees and consequently select many subspaces that contain few, if any, informative features. In this paper we propose a stratified sampling method to select the feature subspaces for random forests with high dimensional data. The key idea is to stratify features into two groups. One group will contain strong informative features and the other weak informative features. Then, for feature subspace selection, we randomly select features from each group proportionally. The advantage of stratified sampling is that we can ensure that each subspace contains enough informative features for classification in high dimensional data. Testing on both synthetic data and various real data sets in gene classification, image categorization and face recognition data sets consistently demonstrates the effectiveness of this new method. The performance is shown to better that of state-of-the-art algorithms including SVM, the four variants of random forests (RF, ERT, enrich-RF, and oblique-RF), and nearest neighbor (NN) algorithms.

1. Introduction

Random forest (RF) builds a classification ensemble with a set of decision trees that grow using randomly selected subspaces of data [1–5]. Experimental results have shown that random forest classifiers can achieve a high accuracy in data classification [6,4,2]. Interest in random forests continues to grow with recent theoretical analyses [7–9] and applications research in bioinformatics [10–14] and computer vision [15–19].

A key step in building a RF is generating different subspaces of features at each node of each unpruned decision tree that makes up the forest. Several subspace selection methods have been developed [5,1,4,2,3]. A simple random sampling of the available features is a common approach to selecting subspaces [2].

The performance of a RF depends on the performance of each decision tree, and the diversity of decision trees in the forest. Breiman [2] formulated the overall performance of a set of trees as the average strength and the average correlation between the trees. He showed that the generalization error of a RF classifier is bounded by the ratio of the average correlation between trees divided by the square of the average strength of the trees.

For high dimensional data a large proportion of the features may not be informative of the class of an object. The common random sampling method may select many subspaces that do not include informative features. As a consequence, the decision trees generated from these subspaces may suffer a reduction in their average strength, thus increasing the error bounds for the RF.

The main aim of this paper is to propose and develop a stratified sampling method for feature subspace selection for generating the decision trees of a RF. It is particularly relevant for high dimensional data. Our idea is to introduce a stratification variable to divide features into two groups: strong informative features and weak informative features. We then randomly select features from each group, ensuring we have representative features from each group. This approach ensures that each subspace contains enough useful features for classification purpose in high dimensional data.

In this paper we use both synthetic data sets and real world data sets from gene classification, image categorization and face recognition to demonstrate the proposed method's effectiveness. The performance of the proposed method is shown to better that of the current state-of-the-art algorithms including SVM, the four variants of random forests (RF, ERT, enrich-RF, and oblique-RF), and the kNN and naive Bayes algorithms.

* Corresponding author. Tel./fax: +86 755 2603 3008.
E-mail addresses: yeyunming@hit.edu.cn (Y. Ye), wuqingyao.china@gmail.com (Q. Wu), zx.huang@siat.ac.cn (J. Zhexue Huang), mng@math.hkbu.edu.hk (M.K. Ng), xutaolee08@gmail.com (X. Li).

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2012.09.005

The remainder of the paper is organized as follows. In Section 2, we review random forests. In Section 3, we present the stratified sampling method for feature subspace selection. In Section 4, we present and discuss the experimental results on various data sets. Conclusions and future work are presented in Section 5.

2. Random forests

A RF model consists of an ensemble of decision tree models. The concept of building multiple decision trees was introduced by Williams [20]. Ho [21] then developed the concept of taking a subspace of features when building each tree within the ensemble. Breiman [2] then developed the RF algorithm as it is commonly known today by introducing further randomness into the process.

The common algorithm can be described as follows. Given a training data set X taken from a space $\mathcal{S}$ of N features:

1. Use bagging [22] to generate K subsets $\{X_1, X_2, \ldots, X_K\}$ by randomly sampling X with replacement.
2. For each data set X_k, use the CART [23] to build a decision tree. At each node in building a decision tree, randomly sample a subspace of p features ($p \ll N$) from $\mathcal{S}$, and compute all possible splits based on the p features. The best split (e.g., the largest Gini measure) is used to continue with the divide and conquer process. Continue until a stopping criteria are met: i.e., all data are pure with respect to the class, have identical values for each attribute, or the number of instances remaining in the data subset is less than n_{min}.
3. Combine the K unpruned trees $h_1(X_1)$, $h_2(X_2)$, …, $h_K(X_K)$ into a RF ensemble, and use a vote among the trees as the ensemble classification decision.

An ensemble learner with excellent generalization accuracy has two properties: high accuracy of each component learner and high diversity in component learners. The theoretical and practical performance of ensemble classifiers is well documented [24,25].

Breiman [2] employs two randomization procedures to achieve diversity. Sufficiently diverse trees can be constructed using randomly selected training samples for each of the individual trees, and randomly selected subspaces of features for splitting at each node within a tree. Geurts et al. [3] proposed to extremely randomize trees by randomizing the attribute splitting threshold. Such randomization procedures were found to enhance the independence of individual classifiers in the ensemble and thus obtaining improved accuracy.

Random feature subspace sampling is the most common method to deliver diversity amongst decision trees due to its simplicity, efficiency and resulting classification performance. Despite its popularity, random subspace sampling may not be a good strategy to deal with high dimensional data. Recent methods build more discriminative subspaces by a weighting features individually [26]. The weights are computed with respect to the feature's correlation with the class feature. The resulting weights are treated as a probability by which a feature will be selected for inclusion in a feature subspace. This approach has been shown to deliver impressive improvements in the classification performance of individual trees, primarily because the subspaces will contain features more relevant to the class. However, the chances of introducing more correlated trees are also increased since the features with large weights are likely selected repeatedly. Our approach, introduced in this paper, uses a stratified sampling technique for feature subspace selection, in order to improve the independence of the resulting decision trees.

Stratified sampling is a well-known technique with many applications for large scale data management [27–29]. In stratified random sampling the population are divided into smaller subgroups called strata. Random sampling is then applied within each subgroup or stratum.

One of the advantages of stratified sampling is that it can capture key population characteristics. When we apply stratified sampling for feature subspace selection, a set of strong informative features forms a core stratum, and the weak informative features forms another stratum. As discussed above it is advantageous to guarantee the inclusion of informative features in a subspace when constructing a decision tree. Adding weakly informative features will offer considerable diversity amongst the resulting decision trees.

Based on these ideas, we introduce stratified random forests. The method builds random decision trees using stratified feature subspace sampling for each node within the tree building process. This method is described in the following section where we also show that it is robust in subspace diversity and is computationally efficient.

3. Stratified feature subspace selection

In this section, we first discuss the problem of a simple random sampling method for feature subspace selection. Then we introduce a stratified sampling method. Finally, we develop a new random forest algorithm, the stratified random forest (SRF).

3.1. High dimensional data issue

The classification performance of a decision tree depends on the correlation of the selected features to the class feature. To increase the strength of a tree we are required to select feature subspaces that contain features that are well correlated to the class feature. However, for high dimensional data, there are usually relatively few features that have high a correlation to the class feature.

Suppose we have only H features that are informative (i.e., highly correlated to the class feature) for classification purposes. The remaining $N-H$ features are then not informative. If a decision tree is grown in a subspace of p features, often with $p \ll N$, then the total number of possible subspaces is

$$\binom{N}{p} = \frac{N!}{(N-p)!p!}$$

The probability of selecting a subspace of $p > 1$ features without informative features is given by

$$\frac{\binom{N-H}{p}}{\binom{N}{p}} = \frac{\left(1-\frac{H}{N}\right)\cdots\left(1-\frac{H}{N}-\frac{p}{N}-\frac{1}{N}\right)}{\left(1-\frac{1}{N}\right)\cdots\left(1-\frac{p}{N}-\frac{1}{N}\right)} \approx \left(1-\frac{H}{N}\right)^p$$

In high dimensional data, $N \gg H$, and therefore the above probability is close to 1. That is, there is a very high chance that a subspace selected by a simple random sampling method will not contain any informative features. As a result, the trees generated will not be useful for classification purposes, and so the overall average strength of the trees is reduced. We propose the stratified sampling method to ensure all generated subspaces contain some informative features.

3.2. Stratified sampling method

The stratified sampling method introduced in this paper ensures the inclusion of strongly correlated features in all feature subspaces whilst also avoiding the same features being repeatedly selected. In this way we increase the overall average strength of the models and reduce the overall incidence of correlated trees.

Let $\boldsymbol{A}$ be a set of N features $\{A_1,A_2,\ldots,A_N\}$ describing a space $\mathcal{S}$. Let Y be a feature identifying the classes of objects. We consider a non-negative function φ as a measure of the informativeness of an input feature A_i with respect to the class feature Y. The resulting value of φ_i is normalized as follows:

$$\theta_i = \frac{\varphi_i}{\sum_{k=1}^{N} \varphi_k} \tag{1}$$

where θ_i is then in between 0 and 1 and measures the relative informativeness of feature A_i with respect to the set of features $\boldsymbol{A}$. We call a feature A_i strong (weak) if θ_i is large (small). We can then stratify the set $\boldsymbol{A}$ into two groups as $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$ by the following procedure:

(i) Sort the set of features based on $\{\theta_i\}$ in descending order.
(ii) Specify a threshold α and divide $\boldsymbol{A}$ into two disjoint groups so that $\boldsymbol{A}=\boldsymbol{A}_s \cup \boldsymbol{A}_w$, $\boldsymbol{A}_s \cap \boldsymbol{A}_w = \emptyset$, and $A_s = \{A_i \in A \,|\, \theta_i < \alpha\}$ and $A_w = \{A_i \in A \,|\, \theta_i \geq \alpha\}$.

The stratified sampling of a subspace with $p\ (>1)$ features can now be accomplished by selecting individual features at random from the two subgroups. The features are selected in proportion to the relative sizes of the two groups. That is, we randomly selected $p_s = p \times N_s/N$ features from the strong feature group, where N_s is the size of the strong feature group and N is the total number of features. We also randomly sample $p_w = p - p_s$ features from the weak feature group. These are then merged to form a subspace for tree construction. We specify p must contain at least one from each group. In this way, we can guarantee that the subspace at any node contains both strong and weak features.

For high dimensional data with a large portion of less informative features, the stratified sampling method will now provide more accurate results than conventional RF in classification. This is demonstrated in Section 4. We now analyze the subspace diversity that results from the proposed stratified sampling method.

3.2.1. Subspace diversity analysis

Let N_s and N_w be the numbers of features in $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$, respectively, $N_s+N_w=N$, where N is the number of features in $\boldsymbol{A}$. The possible number of the selections of p_s features from $\boldsymbol{A}_s$ is given by

$$C_s = \binom{N_s}{p_s} = \frac{N_s!}{(N_s-p_s)!p_s!}$$

The possible number of the selections of p_w features from $\boldsymbol{A}_w$ is given by

$$C_w = \binom{N-N_s}{p-p_s} = \frac{(N-N_s)!}{(N-N_s-p+p_s)!(p-p_s)!}$$

The subspace diversity C can be calculated by the total number of possible subspaces:

$$\begin{aligned} C = C_s \times C_w &= \frac{N_s!}{(N_s-p_s)!p_s!} \times \frac{(N-N_s)!}{(N-N_s-p+p_s)!(p-p_s)!} \\ &\approx \frac{(N_s)^{p_s}\left(1-\frac{p_s}{N_s}\right)^{(p_s-1)}}{p_s!} \times \frac{(N-N_s)^{(p-p_s)}\left(1-\frac{p-p_s}{N-N_s}\right)^{(p-p_s-1)}}{(p-p_s)!} \\ &= \frac{(N_s)^{p_s}(N-N_s)^{(p-p_s)}}{p_s!(p-p_s)!}\left(1-\frac{p_s}{N_s}\right)^{(p_s-1)}\left(1-\frac{p-p_s}{N-N_s}\right)^{(p-p_s-1)} \end{aligned}$$

If $N_s \ll N$, the diversity of subspaces can be represented as

$$C \approx \frac{(N_s)^{p_s}(N)^{(p-p_s)}}{p_s!(p-p_s)!}\left(1-\frac{p_s}{N_s}\right)^{(p_s-1)}$$

This formula shows that the diversity of subspaces increases as p increases as p_s increases. The stratified sampling method is sufficient in subspace diversity. For example, suppose the total number of features $N=100$, the number of strong informative features $N_s=50$, and we sample a subspace of 10 features containing five strong features. There are over 4 billions possible subspaces. If we set the subspace size $p=\text{int}(\log_2(N)+1)=7$ as suggested in [2], where int(x) is the first integer larger than x, we will also have over 300 millions of different subspaces.

3.3. The SRF algorithm

The proposed algorithm (*stratified random forest* or just SRF) to build a random forest model from training data X with the stratified sampling method for feature subspace selection is summarized as follows:

1. For each feature A_i, compute its informativeness φ_i with an non-negative informative function φ, and normalize the resulting value to θ_i according to (1).
2. Specify a stratification threshold α to divide $\boldsymbol{A}$ into two groups $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$.
3. Use bagging [22] to generate K subsets $\{X_1,X_2,\ldots,X_K\}$.
4. Grow a decision tree $h_i(X_i)$ for each data set X_i. At each node, randomly sample a feature subspace with $p(>1)$ features from $\boldsymbol{A}_s$ and $\boldsymbol{A}_w$ proportionally. Use a Boolean test function τ on p features to divide the data into left and right children nodes. Continue this process until a stopping criteria are met: all data are pure or have identical value for each attribute, or the number of instances is less than n_{min}.
5. Combine the K unpruned trees $h_1(X_1)$, $h_2(X_2)$, …, $h_K(X_K)$ into a random forest ensemble and use voting to make the final classification decision.

3.3.1. Implementation

In the SRF algorithm, the informative function φ is used to evaluate the informativeness or predictive power of features.

We use the Fisher discriminant projection in the optimal direction of the features for calculating informativeness φ_i for the feature A_i. The projection computes $s=\mathbf{w}^T\mathbf{x}$ to relate to a class y in terms of input features $\mathbf{x}=(x_1,\ldots,x_N)$, where $\mathbf{w}=(w_1,\ldots,w_N)$ refers to the weights of the projection. When the feature is important (not important), the value of the weight is large (small). Therefore we use the absolute value of weight w_i as the informativeness φ_i of the feature A_i [30]. In the experiments of Section 4, we will use this informative measure to generate SRF.

According to the values of φ_i and its normalized value θ_i, the optimal solution of the stratification threshold α can be determined by minimizing the bound on generalization error of RF $PE^* \leq \overline{\rho}(1-s^2)/s^2$ [2]. For computational efficiency we employ the average of $\{\theta_i\}$ to be the stratification threshold.

Each tree is now built recursively in a top-down manner. We start building each tree from the training set. At each node, a feature subspace is randomly selected. To split the data we use a Boolean test on $\mathbf{w}^T\mathbf{x} \leq \tau$ or $\mathbf{w}^T\mathbf{x} > \tau$ where τ is the average value of means of the projected samples with respect to different class subsets. That is, $\tau=(1/C)\sum_{i=1}^{C}\tilde{m}_i$ where C is the number classes and $\tilde{m}_i$ is the projected mean of samples labeled by class c_i.

In summary, there are three parameters in the SRF algorithm: the subspace size p, the number of trees K and the minimum number of instances for splitting a node (n_{min}). In [2], K is usually set 100, and n_{min} is set to 1. For high dimensional data, more trees may be necessary. We thus use $K=500$ trees and $n_{min}=1$ as our default values. Unless otherwise stated, we use the default setting in all experiments in order to maximize the computational

advantage. The value of p for the size of subspace controls the strength and randomness of the generated trees. With $p=N$ we are essentially using a CART decision tree model builder, which produces trees with minimal diversity. With $p=1$ we produce completely random trees. In [2], it is suggested to use $p=\text{int}(\log_2 N+1)$, where int(x) refers to the first integer that is equal to or larger than x. In Section 4, we will consider the performance for different values of p in the proposed algorithm.

3.3.2. Computational complexity

We consider there are M observations (instances), N features (attributes), and p features in a subspace ($p \ll N$). In the worst case, the use of LDA multivariate method to build a decision tree requires $O(pM^2)$ operations. If we set the subspace size p to be about int($\log_2 N+1$), as suggested in [2], the required computational complexity is $O(\log(N)M^2)$. More precisely, we need to solve a least squares problem at each node with a training data matrix of size M'-by-p, computation is fast using a linear time algorithm [31] and requires $O(pM')$ operations, where M' is the number of instances in a node. Roughly speaking, the computational complexity is $O(\log(N)M)$ to generate the nodes at the same level of a tree. Suppose the tree height is h, the total computational complexity would be $O(\log(N)Mh)$. In the case when the tree is extremely unbalanced ($h \approx O(M)$), then the computational complexity is $O(\log(N)M^2)$. For a balanced tree ($h \approx O(\log n)$), the computational complexity is $O(\log(nN)M)$.

The stratification is done before starting to learn the forest. But we note that it is available to re-compute the weak/strong features after every split. For such procedure, it is likely that a node observes a new distribution over weak/strong features only based on local learning data at each node. Now the stratification task is to select features into the subspaces based on the observed distribution. We note that an additional cost of $O(M'N)$ at each node is required for re-computing strong/weak features, where M' is the size of local learning data in the node and N is the number of features.

3.3.3. Re-computation of the weak/strong features

We note that the stratification in SRF is pre-computed for each feature based on its univariate distribution before starting to learn the forest. It is computationally efficient for high dimensional data, but it may not be applicable for data sets with strong interactions among features. After a split of data, due to interactions, the informativeness of features will changed. The interactions among features are overlooked in our approach. We use LDA as multivariate splitting method. But it is linear and hence also does not capture interactions. The informative features may change during tree generation, whilst the pre-computed informative features are still used for every split. Thus, the performance of resulting forest may be degraded. An intuitive way to remedy this problem is to re-compute the informative features after every split. It is easy to come with data sets with strong interactions among the features, and the re-compute mode is useful than that of pre-compute mode in these data sets. We experiment to investigate how would these two different strategies in computing weak/strong features affect the performance of SRF. The experimental results illustrate the advantages and disadvantages of the two methods. Refer to Section 5 for detailed analysis and discussion.

4. Experimental results

A series of experiments were conducted on nine synthetic data sets, 12 gene data sets, two image classification data sets and two face recognition data sets. All data sets are high dimensional. Recent work [32] performed an empirical evaluation of different supervised learning algorithms (SVM, ANN, LR, NB, KNN, RF, Bagging, boosted trees, PRC) on various dimensions, and concluded that SVM and RF perform consistently better than the other algorithms. To evaluate the performance of our *stratified random forest* (SRF), we compare it with each of an SVM, two variants of the random forest algorithm (the conventional random forest (RF) by Breiman and extremely randomize trees (ERT) by Geurts), and the nearest neighbor (NN) and naive Bayes (NB) algorithms.

Below we describe the implementations and parameter settings for the different algorithms. We train the SVM with a linear kernel using LibSVM.[1] We vary the value of the regularization parameter by factors of 2 ranging from 2^{-5} to 2^{15}. The optimal regularization parameter with the highest validated accuracy on the training set is chosen. For both variants of random forest type methods all trees are generated using CART. Unless otherwise stated 500 trees are generated and a feature subspace size of $\log_2 N+1$ is used to build random forest classifiers. We also evaluate the performance of different parameter settings in the following tests (e.g., the size of the feature subspace, tree depth, and the number of trees). For the NN classifier, we use a cosine distance with the same weight for each feature. For naive Bayes, we use the implementation as used for image classification in the ICCV courses.[2]

To evaluate the performance we use two metrics: the test error (ERR) and the area under the ROC curve (AUC). The AUC can be used for binary classification problems. Hand and Till [33] introduced an extension to the standard two-class ROC for multi-class problems. They derived a formulation that measures the unweighted pairwise discriminability of classes as follows:

$$AUC_{total} = \frac{2}{|C|(|C|-1)} \sum_{\{c_i,c_j\}} AUC(c_i,c_j) \tag{2}$$

where $|C|$ is the number of classes and $AUC(c_i,c_j)$ is the area under the two-class ROC curve involving the classes c_i and c_j.

We used Breiman's method described in [2] to calculate the average *Strength*, the average *Correlation* and $c/s2$ as measures of the generalization error of a random forest. Following Breiman's notation in [2], we denote *Strength* as s and *Correlation* as ρ. Let X be a training data and Y be the class labels. Let $h_k(X_k)$ be the kth tree classifier grown from the kth training data X_k sampled from X with replacement. Assume the random forest contains K trees. Given $x_i \in X$, the out-of bag proportion of votes for x_i on class j is

$$Q(x_i,j) = \frac{\sum_{k=1}^{K} I(h_k(x_i)=j; x_i \notin X_k)}{\sum_{k=1}^{K} I(x_i \notin X_k)} \tag{3}$$

where $I(\cdot)$ is the indicator function. $Q(x_i,j)$ is the number of trees in the random forest, which were trained without x_i and classified x_i into class j, divided by the number of training data sets not containing x_i.

The out-of bag estimate for strength s is computed as follows:

$$s = \frac{1}{n} \sum_{i=1}^{n} \left(Q(x_i,y_i) - \max_{j \neq y_i} Q(x_i,j) \right) \tag{4}$$

where n is the number of objects in X and y_i indicates the true class of x_i.

The out-of bag estimate for correlation ρ is computed as follows:

$$\rho = \frac{\frac{1}{n}\sum_{i=1}^{n} (Q(x_i,y_i) - \max Q_{j \neq y}(x_i,j))^2 - s^2}{\left(\frac{1}{K} \sum_{k=1}^{K} \sqrt{p_k + \overline{p}_k + (p_k - \overline{p}_k)^2} \right)^2} \tag{5}$$

[1] Available at http://www.csie.ntu.edu.tw/cjlin/libsvm/.

[2] http://people.csail.mit.edu/fergus/iccv2005/bagwords.html.

Table 1
Description of nine benchmark data sets.

Name	Attributes	Instances	Classes
Credit-g	24	1000	2
Diabetes	8	768	2
Ecoli	7	336	8
Glass	9	214	6
Ionosphere	34	351	2
Liver-disorder	6	345	2
Sonar	60	208	2
Soybean	35	685	19
Vehicle	18	846	4

where

$$p_k = \frac{\sum_{i=1}^{n} I(h_k(x_i) = y_i; x_i \notin X_k)}{\sum_{i=1}^{n} I(x_i \notin X_k)} \tag{6}$$

and

$$\overline{p}_k = \frac{\sum_{i=1}^{n} I(h_k(x_i) = \widehat{j}(x_i, Y); x_i \notin X_k)}{\sum_{i=1}^{n} I(x_i \notin X_k)} \tag{7}$$

where

$$\widehat{j}(x_i, Y) = \arg\max_{j \neq y_i} Q(x, j) \tag{8}$$

is the class that obtains the maximal votes among all classes but the true class.

Given the strength and correlation, the out-of bag estimate of the $c/s2$ measure can be computed with ρ/s^2.

4.1. Synthetic data

In the first set of experiments, we generated high dimensional data sets with a small portion of strong features and a large portion of non-informative features. We used nine benchmark data sets (see Table 1) from the UCI data repository to generate high dimensional synthetic data sets.

Given a UCI benchmark data set with a set of features A, we generate a synthetic data set by adding the set B of noisy features by using uniform distributions. Here B is non-informative to the class feature. As a result, the synthetic data set contains two set of features A and B, where A is the set of original features and B is set of the non-informative features. In the experiments, we studied and considered different numbers of non-informative features, and the number $|B|$ of noisy features is controlled by k. More precisely, we set $|B| = k|A|$ and tested the total number of features by increasing k starting from 2 until total number ($|A| + |B|$) of features is greater than 1000.

For each synthetic data set, a random 10% of the data was set aside as testing data. The out-of-bag estimates were used to evaluate the strength, correlation and the ratio $c/s2$, while the separated 10% testing data set was used to evaluate the testing error. To avoid bias, we repeated the procedure 80 times and reported the average performance of these trials.

4.1.1. Results on synthetic data sets

Fig. 1 shows the strength of trees with respect to the number of non-informative features in the data set. In the figure, the x-axis refers to the number k multiplying the number of original features in each UCI data set. The solid lines with square records are the results for the stratified random forest (SRF), whilst the dashed lines are the results of random forest (RF) and extremely randomized trees (ERT).

We observe from the figure that the tree strengths of three methods are comparable among themselves when the number of non-informative features is small. As the number of non-informative features increases, the tree strengths of Breiman's method and extremely randomized trees drop significantly, while the tree strength of the proposed method is still stable. Fig. 2 shows the correlation of trees with respect to the number of non-informative features in the data set. We see from the figure that the correlations of trees of the three methods are not significantly different. According to Figs. 1 and 2, we find that the proposed stratified sampling method can increase the tree strength without much increase of the correlation of trees.

Fig. 4 shows the relation between the ratio $c/s2$ and the number of non-informative features in the data set.[3] According to Fig. 4, we can see that almost every SRF plot is below the plots for RF and ERT for $c/s2$ on all data sets. Also when the number of non-informative features increases, the ratio $c/s2$ of the proposed method still keeps no much degradation. These results imply that the proposed method can reduce the generalization error of random forests in high dimensional data classification, especially when the data contains many non-informative features.

According to the experimental results, obvious improvement in test error is observed on all data sets. Fig. 3 shows the results of the average testing error with respect to the number of non-informative features for the three random forest methods. The proposed method consistently outperforms Breiman's method and extremely randomized trees by the method of Geurts et al. The improvement becomes more and more significant as the number of non-informative features increases.

4.2. Gene data sets

Gene data is usually high-dimensional with many non-informative features and only a small number of training observations available. For this experiment we compare performance across a wide range of gene data sets as described in Table 2. The table is divided into two parts. The upper part includes four binary classification data sets. The bottom part includes eight multi-class data sets.[4] In summary, these 12 data sets have between 2 and 26 distinct classes, between 60 and 308 instances (patients) and from 2000 to 16,063 features (genes). The number of features is significantly larger than the number of instances in all data sets (the instance per feature rate of these data sets is listed in Table 2).

4.2.1. Results on gene data sets

In these experiments the subspace size is controlled by a multiplier k. Specifically, we set $p = k(\log_2 N + 1)$, and increase the value of k from 5 to 50 by a factor of 5. We test the performance of the three random forest algorithms at each subspace size p using fivefold cross-validation. We report the average ERR and AUC results over the different trials. The detailed results are shown in Figs. 5 and 6. The solid lines with square records are the results for SRF, whilst the dashed lines are the results of RF and ERT. We can see that almost every SRF plot is below the plots for RF and ERT for ERR and above for AUC. These show a clear advantage offered by SRF in reducing the test error and increasing the AUC for high dimensional data. The improvement weakly depends on a good choice of the *optimal* subspace size in the SRF. The performance of the SRF is clearly better than those of RF and ERT for different subspace sizes.

Fig. 7 further shows the strength and correlation of SRF and RF with respect to different feature subspace sizes. Here, we present

[3] The lines for other synthetic data sets are similar.

[4] Available at http://www.gems-system.org and http://www.upo.es/eps/bigs/datasets.html.

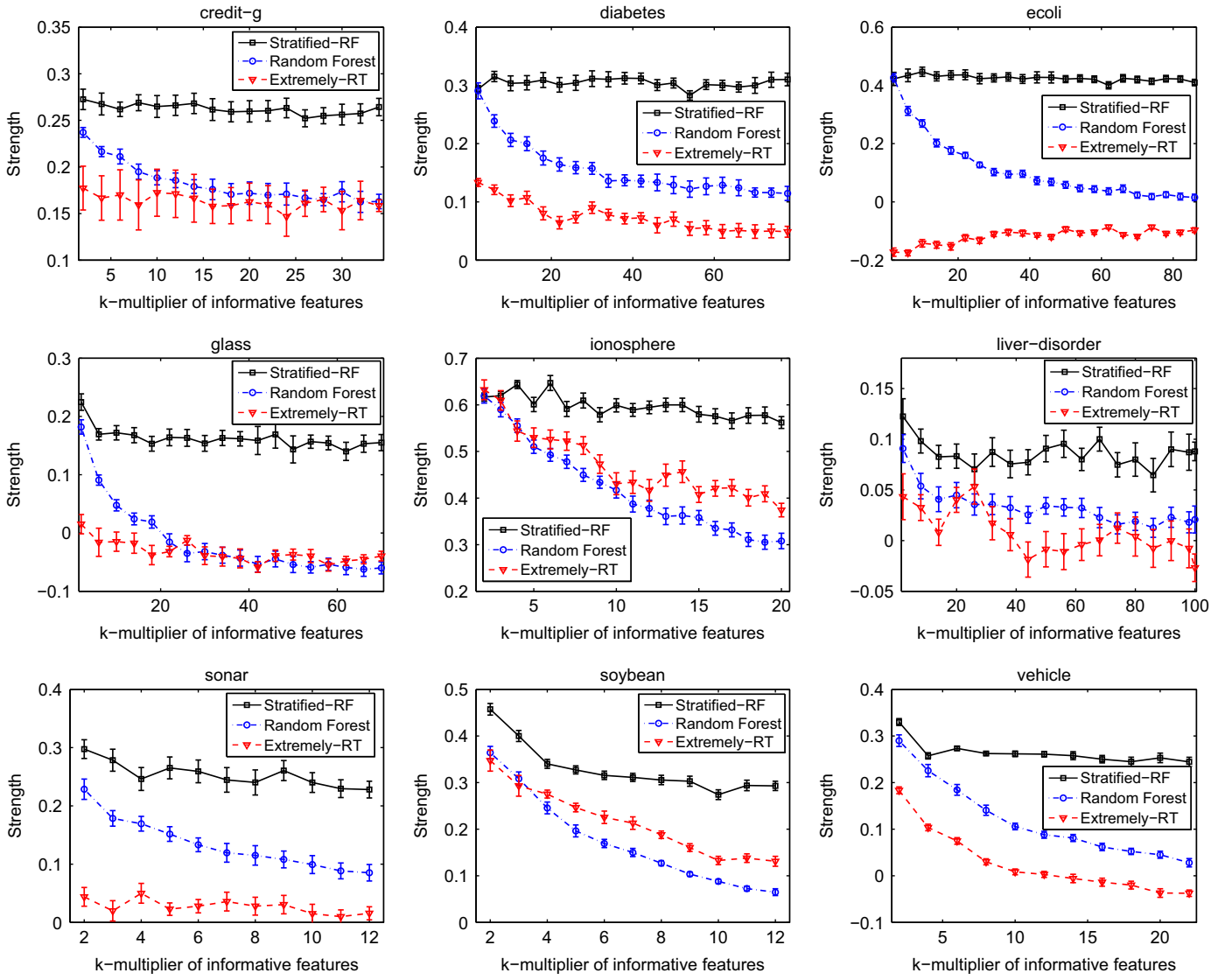

Fig. 1. Strength (mean and std-dev) changes against the number of non-informative features on the nine synthetic data sets.

results on four gene data sets in the upper part of Table 2.[5] From Fig. 7, we can observe the improvement in tree strength and note that the improvement becomes more significant as the size of the feature subspace increases. The tree strength of RF remains about the same. We can also observe that the correlation amongst the trees for the two methods is not significantly different. Experimentally, we find that the SRF can increase the tree strength without changing the correlation significantly. According to Breiman's [2] analysis we will obtain a better error bound for SRF.

We also evaluated the robustness of SRF against two input parameters (the number of trees K, and the minimum number of n_{min} instances for splitting a node). We test the values of K from 20 to 200 while the other parameters are fixed ($p=50\times(\log_2 N+1)$ and $n_{min}=1$). The results can be seen in Table 3. When the number of trees increases, the error rates and variances decrease since the ensemble containing more trees will fit the data much better. We still find that the performance of SRF is better than those of RF and ERT. We also test the values of n_{min} from 1 to 10 while holding other parameters fixed ($p=50\times(\log_2 N+1)$ and $K=200$), with the results shown in Table 4. We find that the performance of SRF, RF and ERF is not significantly affected, and SRF is still best among the three.

Fig. 8 shows the performance of each learning method (only using the strong features and using all features) on the 12 gene data sets. For SVM method, we try linear kernel (SVM-L) and polynomial kernel (SVM-P) with parameter selection following cross-validation. For the k NN classifier, k is selected from range 5 to 50. Then we perform fivefolds with selected *optimal* parameters and report the mean test error results for using strong features and all features. For the three random forest type methods we consider the average results of each across different feature subspaces trials (corresponding to the results shown in Figs. 5 and 6). We report the results on the average mean of the test error for different trials. We can see from Fig. 8 that the performance of RF and ERT is not significant change while training only on the strong features, this mainly due to the decision trees generated only on strong features may suffer an increase in correlation. Whilst SRF method is a good alternative for tradeoff between strength and correlation and have better performance. We can clearly see that SRF achieves the best performance across all comparison methods and all gene data sets. In additional, the performance of SRF using all features is better than those of only using strong features. This result implies

[5] Results on other data sets are similar.

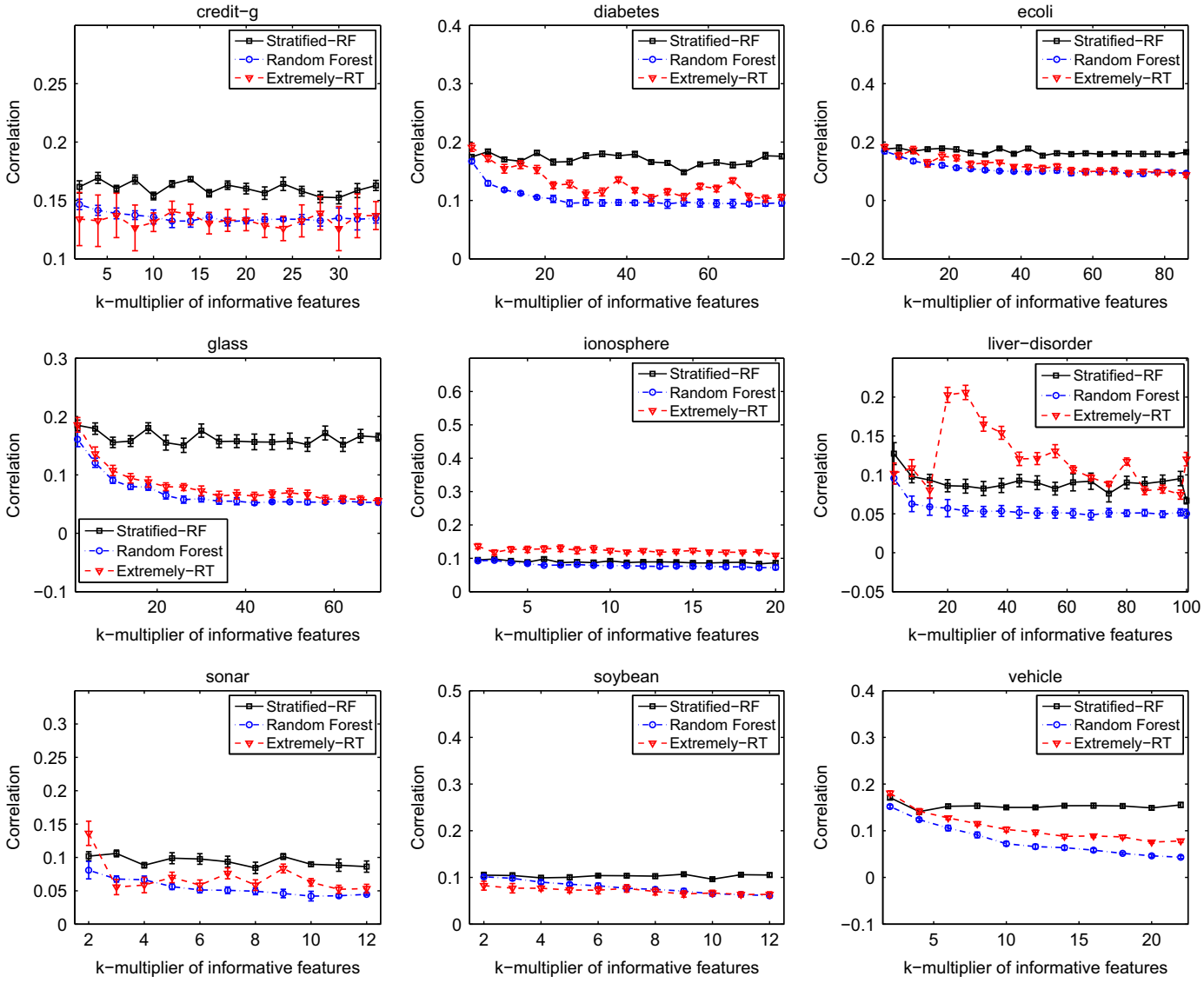

Fig. 2. Correlation changes against the number of non-informative features on the nine synthetic data sets.

that it is good to using all feature to add randomness in SRF, so that the ensemble learning with stratified feature subspace selection is better.

4.2.2. *On the effect of subspace size*

Experimental improvements are demonstrated on the gene data sets in comparison to Breiman's random forest (RF) and Geurts' extremely randomized trees (extremely-RT or ERT) with subspace size chosen of order log$N$. Instead of extremely-RT, there are also a lot of interesting related works of improving random forest. For instance, the enriched random forests (enriched-RF or ERF) algorithm proposed in [26] uses probability sampling to select subspaces, which can also be viewed as a stratified sampling method, but based on one distribution over all features. Concretely, enriched-RF uses a specific informative measure to create one distribution over all features, reflecting their strength and probability to be selected and let a node sample feature subspaces from this distribution. The larger the probability, the higher the chance to be included in the subspaces. Here, the informativeness of the features can be evaluated from univariate measures or multivariate measures. Univariate measures, such as chi-square (CHI), calculate the discriminative power of each feature individually. On the other hand, multivariate measures, such as Fisher's linear discriminant function (LDA), output the informativeness of all features simultaneously. Both univariate measures and multivariate measures can be used in enriched-RF and SRF to measure the strength of each feature. Univariate measures are more computationally efficient in practice, while multivariate measures are more effective for finding informative features with interactions. In SRF, we use LDA as a specific multivariate measure methodology to measure the strength of the features. Specially, the computational cost required is $O(MN)$ with a training data matrix of size M-by-N, where M is the number of samples and N is the number of features. But using an efficient algorithm implementation in [31], LDA can be computed with $O(Mn)$ time, where $n(\leq N)$. Therefore, it is also computational feasible to handle high dimensional data.

Another related work is regarding random forest generated with strong splits. A random forest can be generated from univariate splits or multivariate splits. Univariate split (e.g., Gini measure or information gain) is to identify one best feature at each node for data splitting. While multivariate split relies on combining multiple features together in the splitting process.

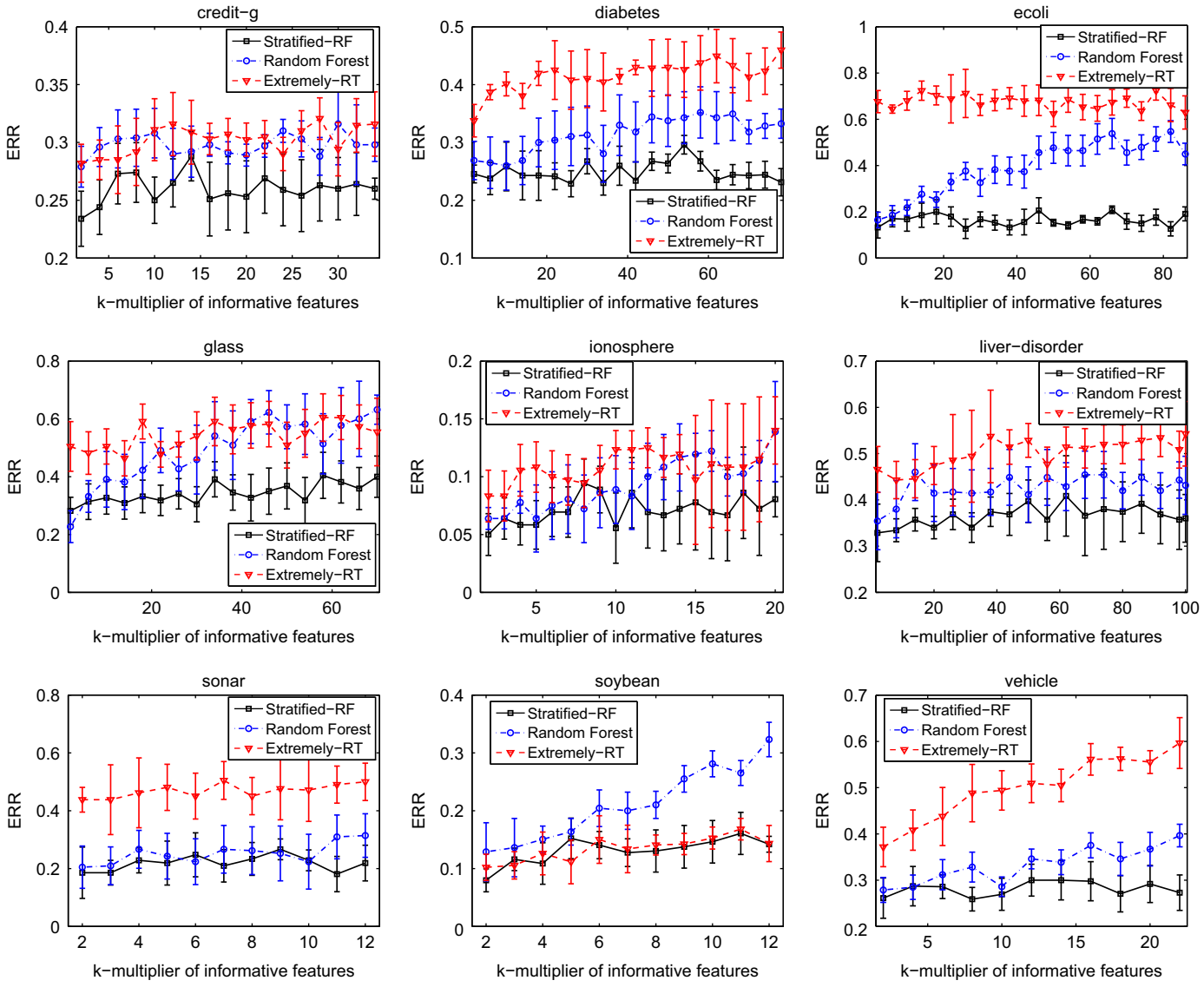

Fig. 3. Testing error (mean and std-dev) changes against the number of non-informative features on the nine synthetic data sets.

Finding strong splits based on multivariate splits is a recent research topic in decision trees. One popular method is the oblique random forests (oblique-RF or ORF) proposed in [34], which applies linear discriminative models to learn the splitting rule, resulting decision boundaries that are arbitrary and oblique to the axes. This approach has been shown to deliver impressive improvements in classification performance of data with many correlated numerical features and few samples [34].

We experiment to compare our SRF method with random forests learning by these two closely related approaches (enriched-RF and oblique-RF) on the gene data sets. It would be interesting to see how results change with different subspace size settings. Thus we use different specifical setting of subspace sizes: $\log_2 N+1$, $\sqrt{N}$, $N/10$, $N/4$ and $N/2$. Table 5 shows the test error results of SRF, RF, enriched-RF and oblique-RF with varying subspaces sizes. Examining the results of enrich-RF shows that enrich-RF performs better than RF when small subspaces are chosen, but it fall behind RF with large subspace, i.e., $p=N/10$. Enrich-RF do better in small subspaces because of the subspaces will contain more relevant features using probability sampling based on the effective distribution over features. However, the chances of introducing more correlated trees are also increased since the features with large weights are likely selected repeatedly. Examining the results of oblique-RF shows that oblique-RF outperforms RF, regardless of the subspace sizes. Because gene data sets consist of many correlated numerical features. For simple univariate splits currently used in RF, one optimal feature is chosen to the learning data at each node for separating classes, leading to deeply nested decision rules and complex decision boundaries. Whilst oblique-RF relies on stronger splits with multiple features, combining the effect of multiple features in the splitting results in decision boundaries that appears to be smoother and better adapted to correlated numerical data [34].

Our SRF method takes into account both stratified sampling for feature subspace selection and strong splits for data splitting in RF. As can be seen, the performance of SRF is clearly in favor of all other RF methods (RF, enriched-RF, oblique-RF) across all data sets and subspace sizes. This result indicates that it is advantageous to combine stratified sampling and strong splits to learn RF. Here, we give a further analysis of the characteristics of the SRF method in terms of comparing with enriched-RF and oblique-RF. Enriched-RF uses one distribution over all features to select subspaces, features are selected based on probability on the distribution. Whilst SRF is based on weak/strong feature stratification, this is another distribution and features

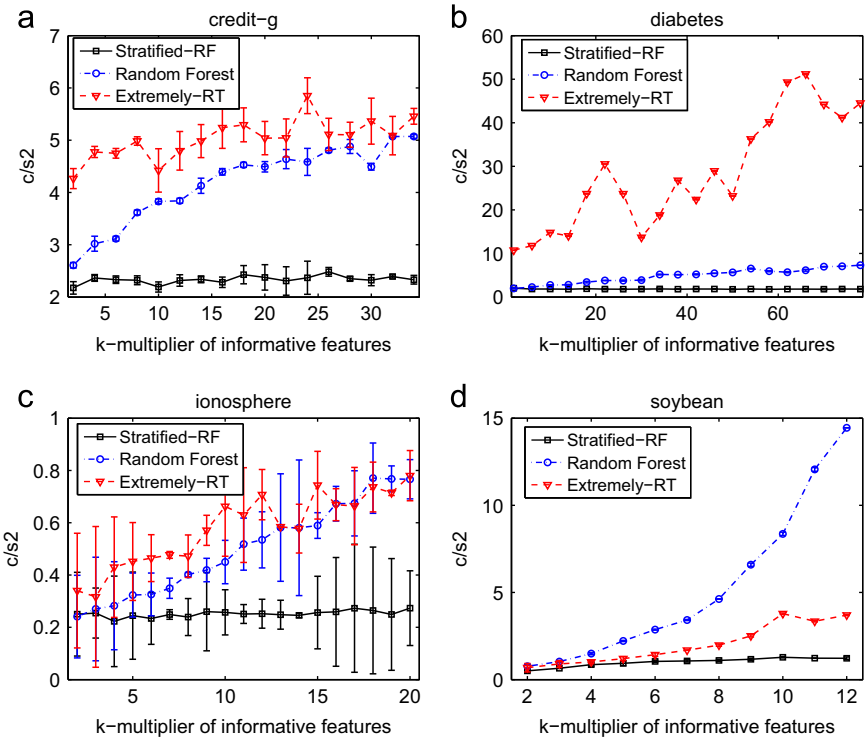

Fig. 4. $c/s2$ (mean and std-dev) changes against the number of non-informative features on the: (a–d) credit-g, diabetes, ionosphere and soybean synthetic data sets.

Table 2
Description of 12 gene data sets.

Name	Att.	Ins.	Class	Instance per feature
Colon	2000	62	2	0.0310
DLBCL	5469	77	2	0.0141
Embryonal-T	7129	60	2	0.0084
Prostate-tumor	10,509	102	2	0.0097
Leukemia1	5327	72	3	0.0135
9-tumors	5727	60	9	0.0105
Brain-tumor1	5920	90	5	0.0152
Leukemia2	11,225	72	3	0.0064
11-tumors	12,533	174	11	0.0139
Lung-cancer	12,600	203	5	0.0161
14-tumors	15,009	308	26	0.0205
GCM	16,063	190	14	0.0118

are randomly selected from weak/strong subsets. It is shown in Table 5 that RF can be improved when larger subspaces are chosen. But using probability sampling on one distribution may impose a bias to features with large weights, especially for large subspaces. The decision trees generated from these subspaces may suffer a reduction in diversity. On the other hand, the SRF method that select subspaces based on stratified sampling on distributions over weak/strong features is relatively insensitive to the choice of subspace sizes. The main different between these two sampling methods is that we select features based on the proportion in the SRF method. Therefore, we have less strong features compared with weak features. Meanwhile, we select weak/strong features randomly, again offer considerable diversity in subspaces. Such randomization procedure enhances the independence of individual trees in RF and obtaining better results.

Oblique-RF directly applies LDA-like split at each node with a set of randomly selected features, while SRF performs the stratification before starting to learn the forest and makes use of more discriminative subspaces for splitting. Experimentally, we find that a RF model learn from subspaces with stratified sampling is better than one without. Interestingly, we observe that SRF(LDA) with feature distribution in stratification induced by LDA informative measure has the highest overall performance. The SRF(LDA) method employs LDA-like model for both informative measure and splitting measure. Both of them have desired properties for handling data with many correlated numerical features, mixing them would help to obtain better RF for high dimensional gene data.

We also compare the best test error achieved by different methods when the optimal subspace size is chosen for each method. The result is given in Fig. 9. We can see that our proposed SRF approach performs better than the oblique-RF method which improves over both RF and enrich-RF. SRF consistently gives significantly lower test error performance on all data sets.

In summary, SRF method is very effective for the classification of high dimensional gene data. Gene data has three special characteristics: (i) many weak-informative features are included (e.g., see Fig. 10(a) and (b)); (ii) many features are correlated; (iii) the number of training samples is very small with respect to the high dimensionality. These characteristics present considerable challenges for constructing decision trees with good strength. RF generated with a simple random sampling strategy for selecting feature subspaces could achieve a good correlation but will have a high probability of missing informative features. Thus the average strength of trees will be decreased. Our SRF method overcomes this problem by utilizing more discriminative subspaces and stronger splits in tree generation. We have shown that SRF is effective for high dimensional gene data classification and can obtain better results against other RF methods.

4.3. *Image classification data sets*

Image classification has emerged as an important area in computer vision. Many of the most popular current methods for image classification are to sample patches from the image by using a bag-of-features representation. In our experiments we

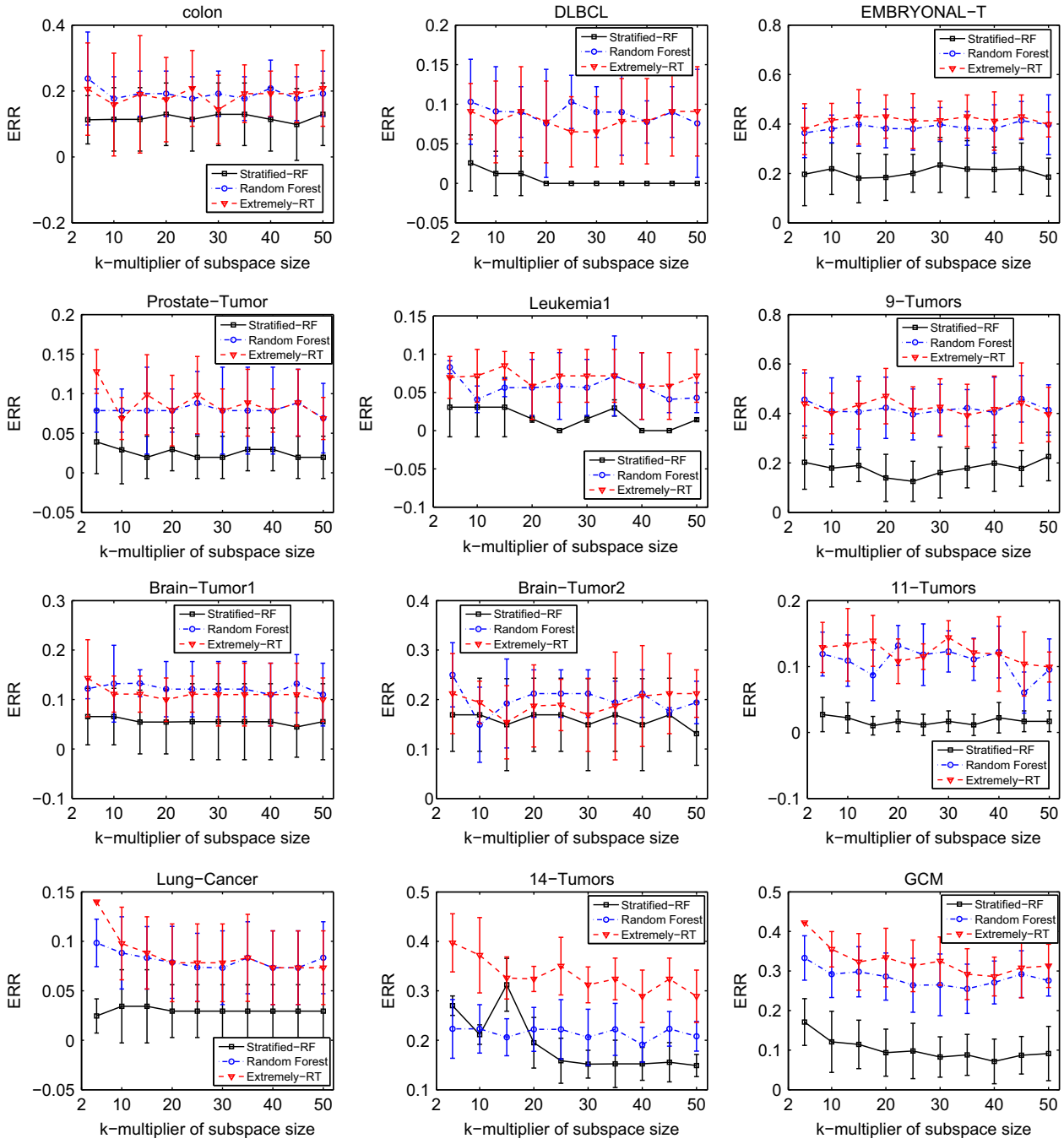

Fig. 5. Testing error (mean and std-dev) changes against the size of feature subspace on the 12 gene data sets.

present a comparison of results for the image classification task on two benchmark data sets: Caltech categories[6] and horses.[7] For the Caltech problem, we use a subset of 100 images from the Caltech face data set and 100 images from the Caltech background data set following the demonstration of image classification in ICCV.[8] The horse data contains a positive class of 170 horse images and a negative class of 170 images which do not include horses. We randomly chose half of the data as the training data set and the other half as the testing data set.

A popular and powerful approach to classifying images is to treat them as a collection of visual words. Several quantization (coding) techniques have been explored to learn the *visual dictionary* (textons) for this image representation process [35–37]. The most common visual coding method is based on *k*-means vector quantization. A visual descriptor function (e.g., SIFT [38]) is used to extract local features by sampling subwindows at random from the training images, either at sparse points of interest or within dense regions [39]. The collection of descriptors are then clustered using a *k*-means approach. The set of estimated cluster centers are treated as universal visual words to produce a visual codebook. For classification an image is represented by a histogram of visual words. Any classifier can be used for the purpose of image classification. Advanced methods for learning visual words can be found in [17,40,15].

[6] http://www.vision.caltech.edu/html-files/archive.html.
[7] http://pascal.inrialpes.fr/data/horses.
[8] http://people.csail.mit.edu/torralba/shortCourseRLOC/.

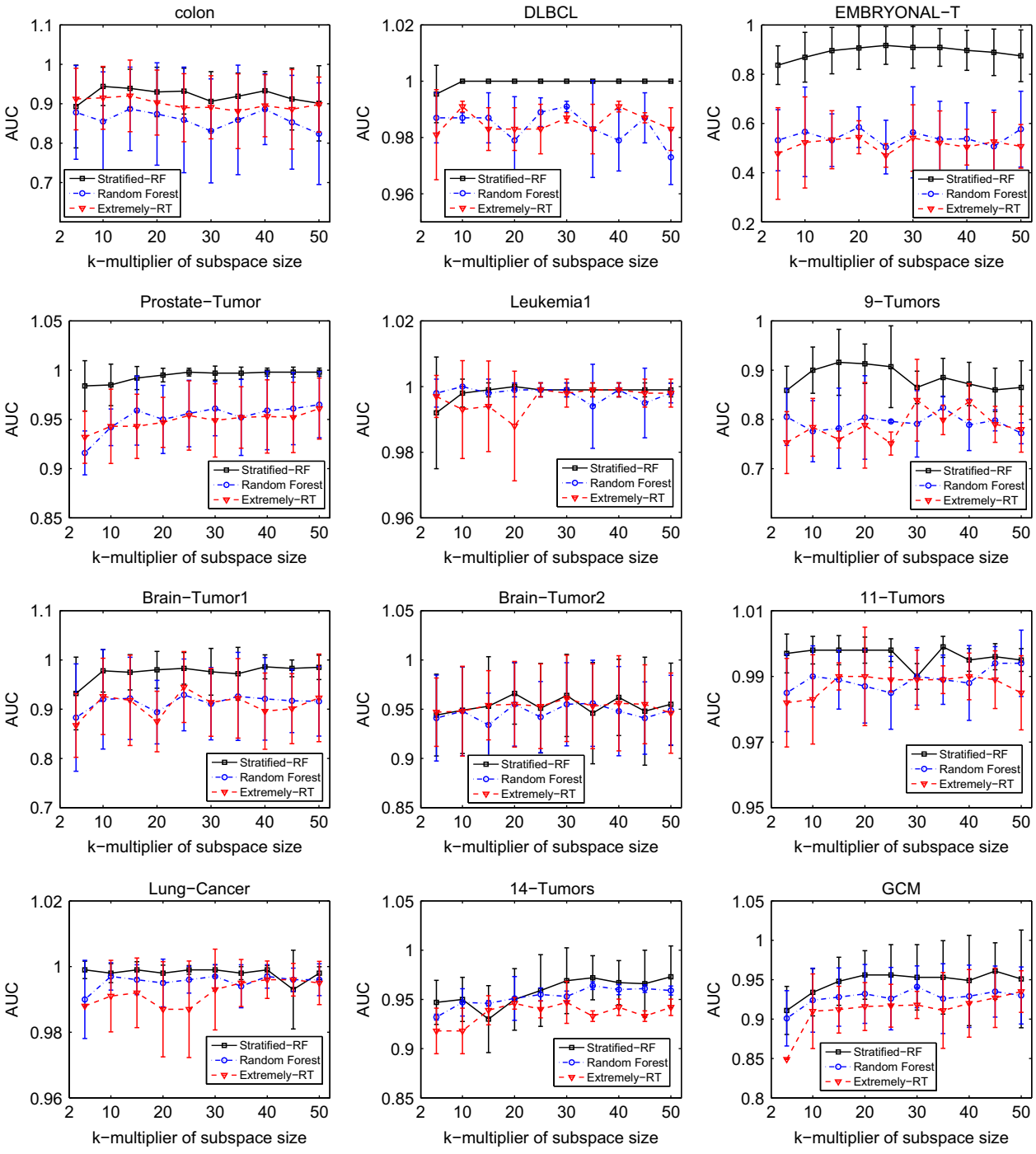

Fig. 6. AUC (mean and std-dev) changes against the size of feature subspace on the 12 gene data sets.

In our experiments we explore the effectiveness of the proposed SRF method as a classifier. We use traditional *k*-means quantization as a visual coding method to generate the visual codebook and control the number of centers to produce the different vocabularies (dimensions). In Fig. 11, we show the informativeness of each feature of the Caltech and Horse data sets of codebook sizes of 5000 and 15,000. We see from the figure that the group of strong features and the group of weak features are quite distinguished for both data sets with various codebook sizes. We expect quite good performance from SRF.

Next we compare the performance of SRF with SVM, naive Bayes [41], and again the two other variants of random forests (RF and ERT) over different codebook sizes. We measure the performance in terms of the classification error rate (ERR) and area under ROC curve (AUC).

We choose a feature subspace $p = 10 \times (\log_2 N + 1)$ as a good trade-off between speed and accuracy. The classification performance and computational time run on a 3.0 GHz Linux server are shown in Fig. 12 and Tables 6 and 7, respectively. As illustrated in Fig. 12, SRF consistently outperforms the other four methods across various codebook sizes. Fig. 12(a) and (b) also shows the clear robustness of SRF. The SRF curves are relatively stable with respect to different codebook sizes. However, the curves of the other four methods change significantly. In Tables 6 and 7 we find

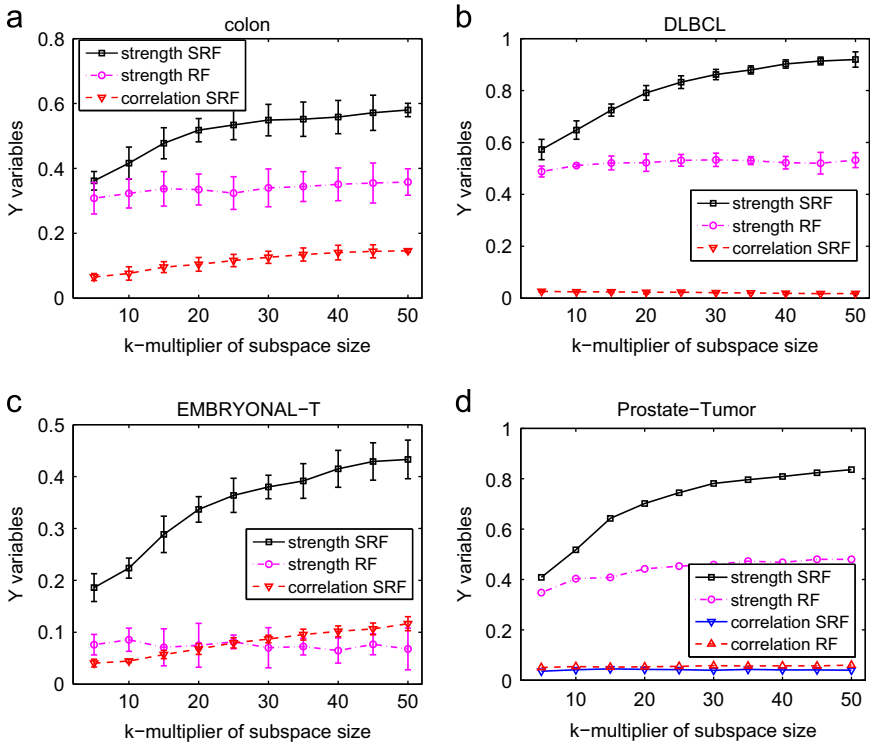

Fig. 7. Strength and correlation (mean and std-dev) changes against the size of feature subspace on four gene data sets.

Table 3
Test error (mean ± std-dev%) against different K on four gene data sets.

Data set	Method	20 trees	50 trees	80 trees	100 trees	200 trees
colon	RF	22.6 ± 3.8	19.9 ± 2.5	19.1 ± 2.4	18.8 ± 2.1	19.2 ± 1.9
	ERT	24.4 ± 2.6	23.2 ± 3.9	21.2 ± 2.6	22.6 ± 2.2	18.6 ± 2.1
	SRF	**13.1 ± 1.4**	**12.8 ± 1.5**	**12.5 ± 0.7**	**12.8 ± 1.4**	**11.8 ± 1.0**
DLBCL	RF	8.9 ± 3.2	8.1 ± 1.1	8.0 ± 2.3	8.3 ± 1.9	7.9 ± 1.0
	ERT	11.8 ± 2.5	12.3 ± 2.9	12.0 ± 2.1	11.8 ± 2.4	8.1 ± 1.0
	SRF	**0.6 ± 0.5**	**0.5 ± 0.4**	**0.5 ± 0.4**	**0.5 ± 0.4**	**0.5 ± 0.9**
Embryonal-T	RF	42.0 ± 3.8	40.1 ± 2.9	38.9 ± 3.1	38.8 ± 3.9	40.8 ± 1.7
	ERT	42.4 ± 4.8	40.9 ± 3.3	39.9 ± 2.5	40.8 ± 2.8	41.5 ± 1.7
	SRF	**21.9 ± 3.6**	**22.4 ± 2.2**	**21.8 ± 1.9**	**21.0 ± 2.6**	**20.5 ± 1.8**
Prostate-T	RF	10.4 ± 4.0	8.8 ± 1.5	8.1 ± 1.9	8.2 ± 1.5	8.0 ± 0.6
	ERT	14.2 ± 2.6	10.6 ± 2.8	9.5 ± 1.8	9.6 ± 2.0	8.7 ± 1.8
	SRF	**3.5 ± 2.6**	**2.8 ± 1.6**	**2.9 ± 0.9**	**2.4 ± 0.7**	**2.5 ± 0.7**

Table 4
Test error (mean ± std-dev%) against different n_{min} on four gene data sets.

Data set	Method	1 instance	2 instances	4 instances	6 instances	10 instances
colon	RF	19.2 ± 1.9	19.2 ± 2.1	18.0 ± 1.6	18.8 ± 1.5	18.2 ± 1.3
	ERT	18.6 ± 2.1	22.0 ± 2.0	22.8 ± 1.9	21.2 ± 2.9	22.0 ± 2.5
	SRF	**11.8 ± 1.0**	**12.5 ± 1.3**	**12.2 ± 1.1**	**12.6 ± 1.4**	**12.6 ± 1.4**
DLBCL	RF	6.9 ± 1.0	6.5 ± 1.9	6.9 ± 1.4	6.4 ± 1.6	6.3 ± 1.9
	ERT	8.1 ± 1.0	10.8 ± 1.2	10.5 ± 1.4	10.9 ± 1.7	11.0 ± 1.6
	SRF	**0.5 ± 0.9**	**0.5 ± 0.4**	**0.5 ± 0.4**	**0.5 ± 0.4**	**0.5 ± 0.4**
Embryonal-T	RF	40.8 ± 1.7	39.6 ± 2.3	39.9 ± 2.3	39.7 ± 2.2	39.8 ± 1.8
	ERT	41.5 ± 1.7	38.5 ± 1.8	38.7 ± 2.0	39.1 ± 0.9	38.9 ± 1.5
	SRF	**20.5 ± 1.8**	**21.3 ± 2.1**	**20.6 ± 1.5**	**20.9 ± 2.1**	**20.5 ± 1.8**
Prostate-T	RF	8.0 ± 0.6	7.6 ± 1.4	7.6 ± 1.2	7.8 ± 1.6	7.7 ± 1.0
	ERT	8.7 ± 1.8	8.2 ± 1.6	9.3 ± 1.3	8.3 ± 1.4	8.7 ± 1.6
	SRF	**2.5 ± 0.7**	**2.2 ± 0.5**	**2.6 ± 0.5**	**2.6 ± 0.7**	**2.7 ± 0.4**

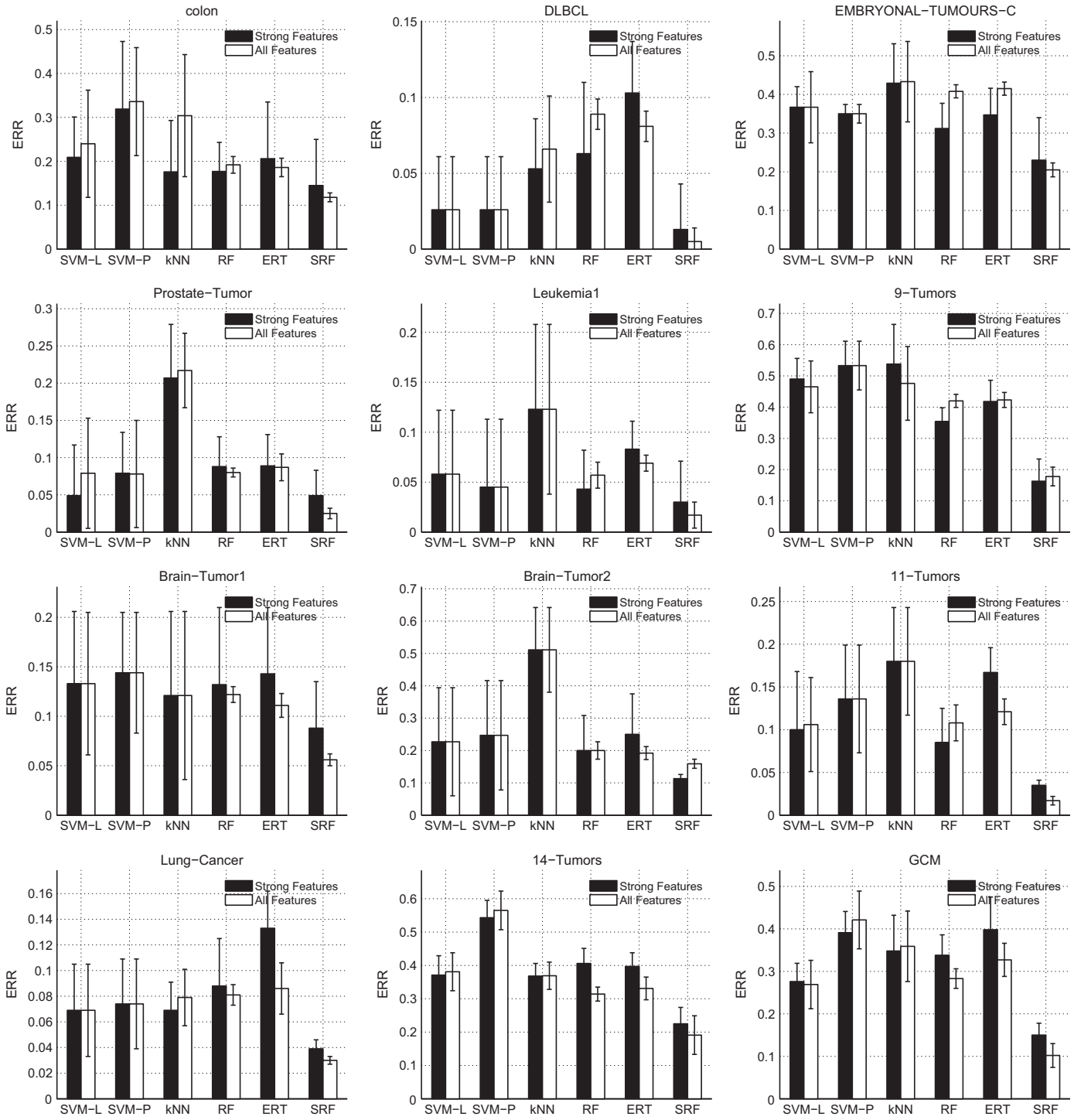

Fig. 8. Testing error (mean and std-dev) changes against the number of features on the 12 gene data sets.

that the running time of each method increases with increases in codebook size.

We also consider a more up-to-date vision data, namely PASCAL VOC 2007. The PASCAL VOC 2007 data[9] contains about 10,000 images split into train, validation and test sets, and labeled with 20 object classes. In our experiment, we use the MATLAB code of a recent work [42] which investigated several popular encoding methods on this data set. The baseline encoding method in [42] with hard quantization histogram and an additive χ^2 kernel map to all codes is used to build the bag of visual words data model for the learning classifiers, as it is shown that the baseline encoding method can achieve good performance for the PASCAL VOC 2007 data set [42]. We compare SRF with standard RF and linear kernel SVM implemented with LibSVM. The performance is evaluated as Area Under ROC curves (AUC) across all classes. Table 8 shows the AUC of SRF and SVM on the PASCAL VOC 2007 data set. Experimentally, we find that the proposed SRF

[9] Available at http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/.

Table 5
Test error (mean ± std-dev%) of different random forest algorithms against different subspace sizes on four gene data sets.

Data set	Method	$\log_2 N+1$	$\sqrt{N}$	$N/10$	$N/4$	$N/2$
colon	Random forest	20.8 ± 11.5	20.6 ± 15.8	17.7 ± 6.6	17.7 ± 6.6	17.7 ± 6.6
	Enrich-RF(CHI)	19.2 ± 6.9	17.7 ± 6.6	21.2 ± 5.1	21.2 ± 5.1	21.2 ± 5.1
	Enrich-RF(LDA)	17.7 ± 6.6	17.7 ± 6.6	17.7 ± 6.6	16.2 ± 5.9	17.7 ± 6.6
	Oblique-RF	17.7 ± 6.6	11.5 ± 7.3	12.9 ± 9.5	14.5 ± 10.5	16.0 ± 12.5
	SRF(CHI)	17.6 ± 8.6	11.9 ± 9.5	12.9 ± 9.5	**12.9 ± 9.5**	**16.0 ± 12.5**
	SRF(LDA)	**16.8 ± 7.5**	**11.3 ± 7.3**	**11.3 ± 7.3**	**12.9 ± 9.5**	**13.6 ± 6.6**
DLBCL	Random forest	12.8 ± 8.6	12.8 ± 8.6	5.3 ± 5.6	7.8 ± 5.4	7.9 ± 8.6
	Enrich-RF(CHI)	7.8 ± 5.4	7.8 ± 2.7	10.4 ± 7.4	10.4 ± 7.4	10.4 ± 7.4
	Enrich-RF(LDA)	6.0 ± 3.7	6.0 ± 3.7	6.7 ± 8.2	6.7 ± 8.2	6.7 ± 8.2
	Oblique-RF	6.9 ± 4.4	2.9 ± 3.5	2.9 ± 3.5	6.7 ± 8.2	7.9 ± 8.6
	SRF(CHI)	6.5 ± 4.4	**1.3 ± 3.0**	2.6 ± 3.5	**2.6 ± 3.5**	**1.3 ± 3.0**
	SRF(LDA)	**5.3 ± 3.0**	2.6 ± 3.5	**1.3 ± 2.8**	**2.6 ± 3.5**	**1.3 ± 3.0**
Embryonal-T	Random forest	44.4 ± 13.9	39.7 ± 8.6	36.2 ± 10.2	39.8 ± 5.0	44.7 ± 7.1
	Enrich-RF(CHI)	39.7 ± 12.0	34.8 ± 9.0	34.7 ± 8.0	38.7 ± 8.0	38.7 ± 8.0
	Enrich-RF(LDA)	39.7 ± 9.3	34.8 ± 14.8	34.8 ± 8.7	38.2 ± 5.6	43.0 ± 8.7
	Oblique-RF	37.4 ± 13.5	32.5 ± 11.2	34.8 ± 7.6	36.5 ± 12.8	38.2 ± 11.4
	SRF(CHI)	36.7 ± 8.6	31.8 ± 7.6	31.7 ± 11.8	**36.5 ± 14.1**	**38.2 ± 11.4**
	SRF(LDA)	**30.0 ± 11.7**	**26.8 ± 11.5**	**20.6 ± 8.1**	38.2 ± 11.4	**38.2 ± 11.4**
Prostate-T	Random forest	12.7 ± 2.5	7.9 ± 4.5	6.9 ± 4.4	5.9 ± 4.1	6.9 ± 5.6
	Enrich-RF(CHI)	7.9 ± 2.7	6.9 ± 2.7	8.9 ± 9.6	7.9 ± 6.9	7.9 ± 6.9
	Enrich-RF(LDA)	5.9 ± 4.1	6.9 ± 2.7	8.9 ± 9.6	7.9 ± 7.5	7.9 ± 7.5
	Oblique- RF	9.9 ± 7.1	4.9 ± 4.0	3.9 ± 4.3	3.9 ± 4.0	**5.9 ± 6.5**
	SRF(CHI)	7.4 ± 2.7	4.4 ± 4.9	2.7 ± 4.3	**3.9 ± 4.0**	**5.9 ± 6.5**
	SRF(LDA)	**6.2 ± 2.7**	**2.5 ± 4.4**	**2.5 ± 2.7**	**3.9 ± 4.0**	**5.9 ± 6.5**

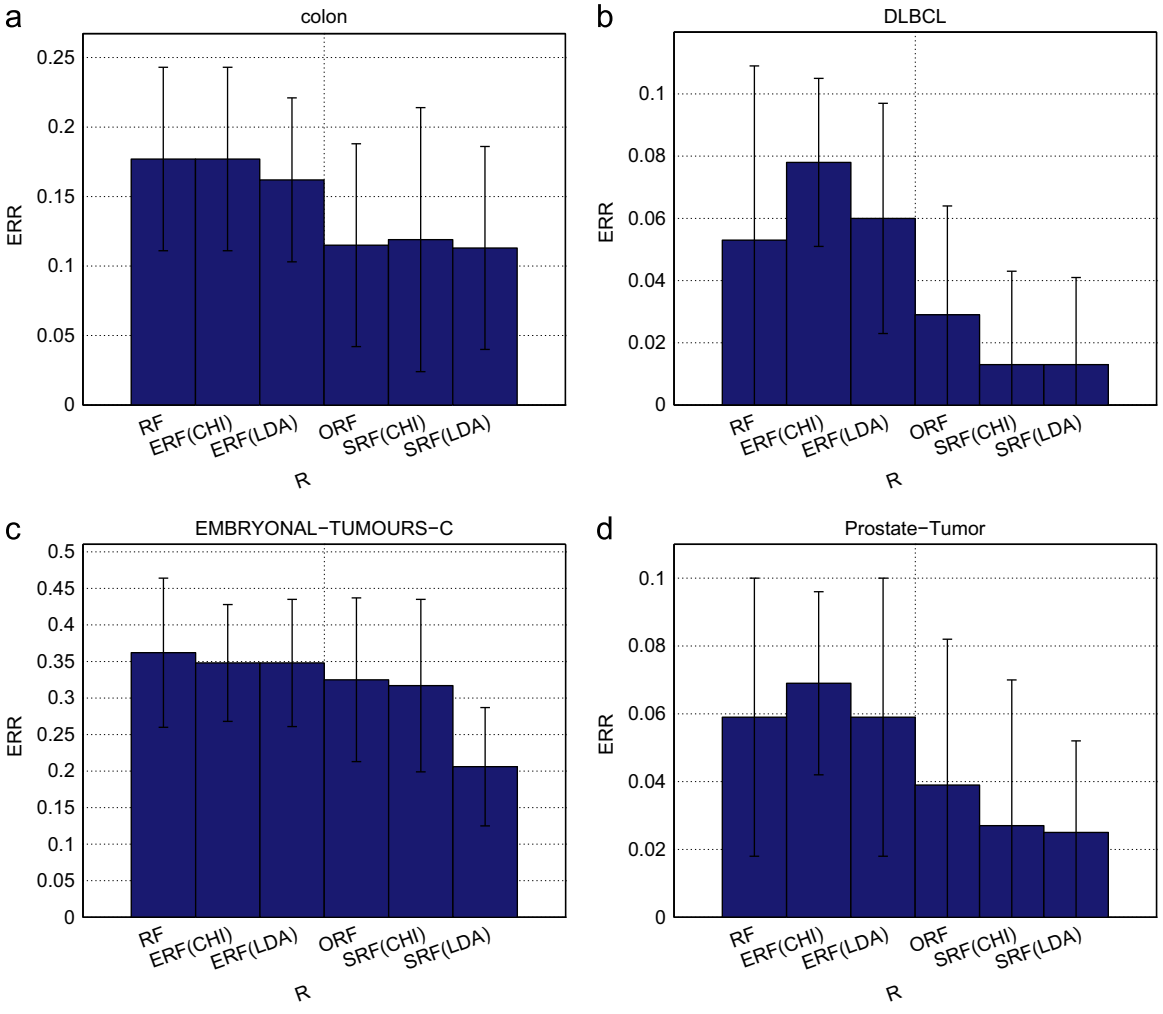

Fig. 9. The best test error results of different compared methods when the optimal subspace size is chosen for each method.

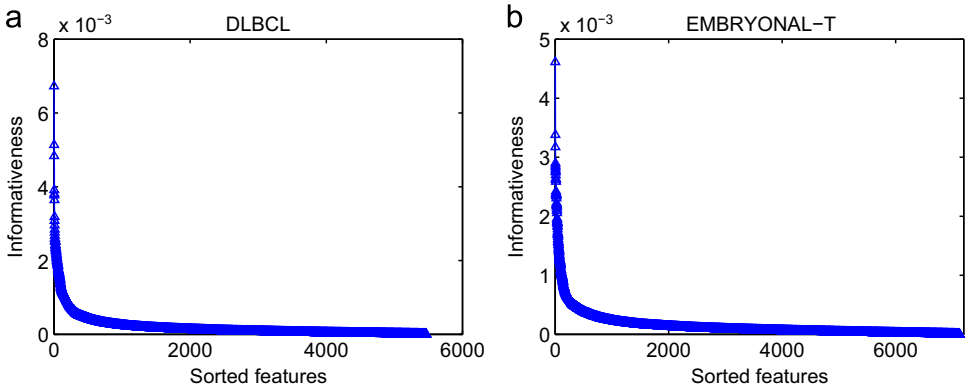

Fig. 10. Distributions of feature informativeness: (a) (b) DLBCL and Embryonal-T Gene data.

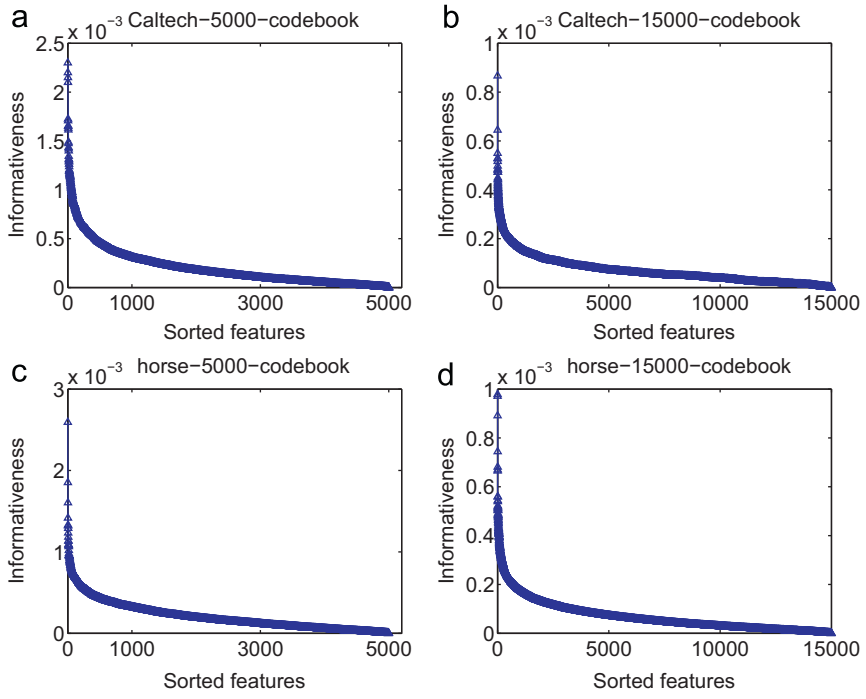

Fig. 11. Distributions of feature informativeness on Caltech and Horse image data.

method compares favorably against RF and SVM in AUC (0.8192 versus 0.7893 and 0.8838).

4.4. Face recognition data sets

In this section we consider the problem of human face recognition and present experiments on two publicly available databases. We conduct dimension reduction on the face recognition problem using the Eigenface [43] and use features selected randomly[10] to compare the performance across various feature dimensions.

The extended Yale B database consists of 2414 face images of 38 individuals [44]. The AT &T ORL data set consists of 400 face images of 40 individuals [45].[11] We randomly choose half of the images of each individual for training and the remaining images for testing. We perform dimension reduction and compute the recognition rate with various features. The feature space dimensions used are 30, 56, 120, and 504. As the data is reduced to lower dimensions, the subspace size for the SRF algorithm is set to $d/2$, where d is the reduced dimension.

Fig. 13 shows the recognition performance against various feature dimensions for five different classifiers: SVM, NN, RF, ERT, and SRF. We can see that SRF consistently outperforms the other four different classifiers across different feature dimensions. More specifically, the best recognition rate for SRF on the YaleB database using Eigenface features (Fig. 13(a)) is 95.86%. This compares well with the maximum recognition rates using RF, ERT, SVM and NN, using Eigenfaces, which were 91.70%, 92.30%, 92.61% and 92.12%, respectively. Similarly, the best recognition rate of SRF is 94.86% by using random features (Fig. 13(b)), where the recognition rates of RF, ERT, SVM and NN are 93.40%, 93.60%, 92.12% and 90.95%.

For the ORL database, the performance of SRF is quite consistent with respect to various Eigenfaces, being between 94.0% and 94.5% (see Fig. 13(c)). The performances of the other methods (RF, ERT, SVM and NN) are between 91.5% and 93.5%, 92.0% and

[10] We randomly select pixels from the face image as features.

[11] The cropped and aligned faces for these two data sets are available at http://www.zjucadcg.cn/dengcai/Data/FaceData.html.

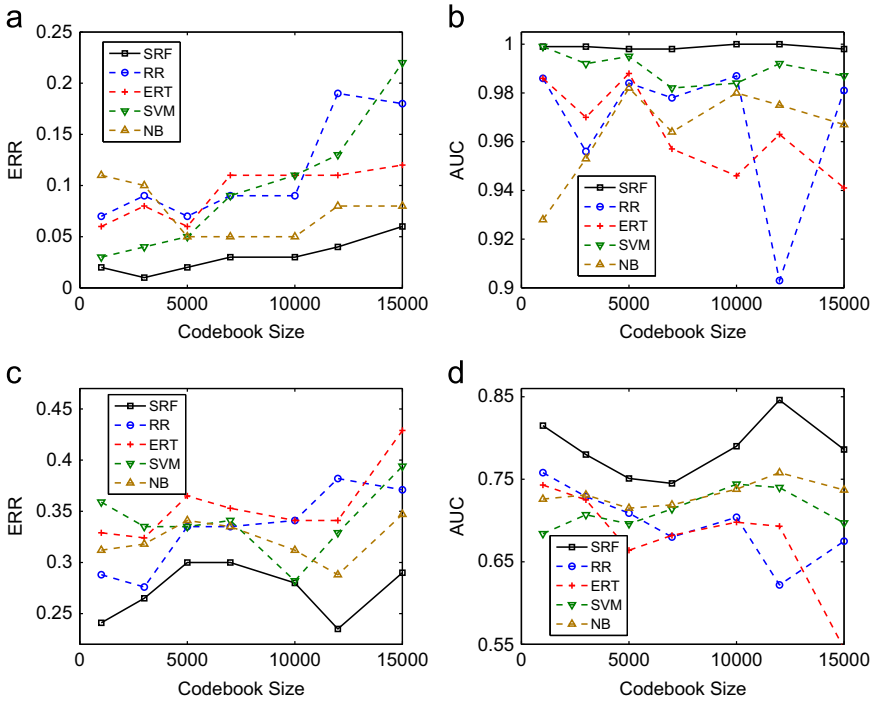

Fig. 12. Test error and AUC results on image classification data sets; (a) and (b) the Caltech data set (face versus background), (c) and (d) the horse data set (horse versus none).

Table 6
Computational time (seconds) for Caltech data set over various codebook sizes.

Visual words size	Caltech data set						
	SRF			RF		ERT	
	Stratified sampling	Trees growing	Avg. depth	Trees growing	Avg. depth	Trees growing	Avg. depth
1000	0.14	23.17	2.01	130.32	4.24	164.35	4.81
3000	0.05	27.79	2.01	209.36	5.59	270.62	6.56
5000	0.08	32.01	2.15	284.08	6.63	309.72	7.67
7000	0.11	40.82	2.34	310.74	7.64	379.77	8.97
10,000	0.18	63.71	2.72	338.10	9.25	364.34	11.08
12,000	0.23	92.73	2.78	352.20	9.94	411.20	11.81
15,000	0.53	124.65	3.06	427.94	11.87	495.07	14.19

Table 7
Computational time (seconds) for Horse data set over various codebook sizes.

Visual words size	Horse data set						
	SRF			RF		ERT	
	Stratified sampling	Trees growing	Avg. depth	Trees growing	Avg. depth	Trees growing	Avg. depth
1000	0.06	30.63	2.02	290.00	5.53	342.44	6.62
3000	0.13	36.62	2.15	383.42	7.58	457.38	9.03
5000	0.21	43.20	2.26	413.85	9.41	495.41	11.26
7000	0.21	51.33	2.40	455.09	10.59	556.77	12.69
10,000	0.31	68.05	2.76	480.74	11.92	607.73	14.34
12,000	0.61	104.01	2.84	527.02	12.52	672.28	14.95
15,000	0.66	148.34	3.19	563.08	13.69	692.03	16.60

93.0%, 93.0% and 93.5% and 89.5% and 90.0% respectively. Also, SRF using random features (Fig. 13(d)) outperforms the other four methods on this data set, achieving a maximum recognition rate of 93.50%.

5. Discussions

In the experiment, we compare the proposed SRF method with variants of random forest methods: conventional random forest (RF), extremely randomized trees (ERT), enrich random forest (enrich-RF), and oblique random forest (oblique-RF). Different randomization techniques are used for tree growing in these compared methods. In general, the performance of a random forest depends on the performance of each tree, and the diversity of trees in the forest. Actually, analysis of these methods leads to better understanding of the ensemble methods with trees.

The accuracy of random forest methods is influenced by the subspace size p. Table 5 shows how the test error of different random forest methods varies with different p values. We observe that there are two types of trends: monotonically decreasing and then asymptotically stable, decreasing followed by increasing. In hindsight, the results make sense: with small subspaces $p=\log_2 N+1$, most of the features are not included at each node to develop the tree. As a result, important features are missed, leading to poor performance. On the other extreme, with a high value of $p=N$, all features are considered for the search of the optimal split during tree generation. It has impact on deteriorating the diversity of the induction algorithms that seek for the best split locally at each tree node. In this case, the ensemble algorithms degenerate, and equivalent to single decision tree algorithms. For a medium number of p, the random forest methods can capture the randomization that they should have. From the results in Table 5, we can observe that RF requires larger subspace setting, e.g., $p=N/10$ in order to achieve excellent performance. In contract, enrich-RF, oblique-RF and our proposed SRF perform well under the setting of $p=\sqrt{N}$.

It is also well known that for random forest ensemble methods, the behavior of prediction error is a monotonically decreasing function of number of trees K. We can see from Table 3 that the higher the value of K, the better the accuracy. The choice of an appropriate value of K will thus essentially depend on the resulting compromise between computational

Table 8
The results of area under ROC curves (AUC) for SVM and SRF across all classes.

	Plane	Bike	Bird	Boat	Bottle	
SVM	0.9072	0.7559	0.6556	0.8774	0.7617	
RF	0.8675	0.7444	0.6415	0.8025	0.7474	
SRF	**0.9537**	**0.8839**	**0.8679**	**0.9238**	**0.8134**	
	Bus	Car	Cat	Chair	Cow	
SVM	0.9003	0.8393	0.8260	0.8133	0.8436	
RF	0.8866	0.8578	0.8279	0.8092	0.7935	
SRF	**0.9396**	**0.9148**	**0.8947**	**0.8553**	**0.8869**	
	Table	Dog	Horse	Motorbike	Person	
SVM	0.8581	0.6583	0.8627	0.8483	0.7367	
RF	0.8480	0.6825	0.8558	0.8301	0.7266	
SRF	**0.8705**	**0.8497**	**0.9340**	**0.9125**	**0.8325**	
	Plant	Sheep	Sofa	Train	TV	Mean
SVM	0.7728	0.8636	0.8096	0.9220	0.8707	0.8192
RF	0.6737	0.7241	0.7797	0.8820	0.8060	0.7893
SRF	**0.8284**	**0.8710**	**0.8364**	**0.9257**	**0.8817**	**0.8838**

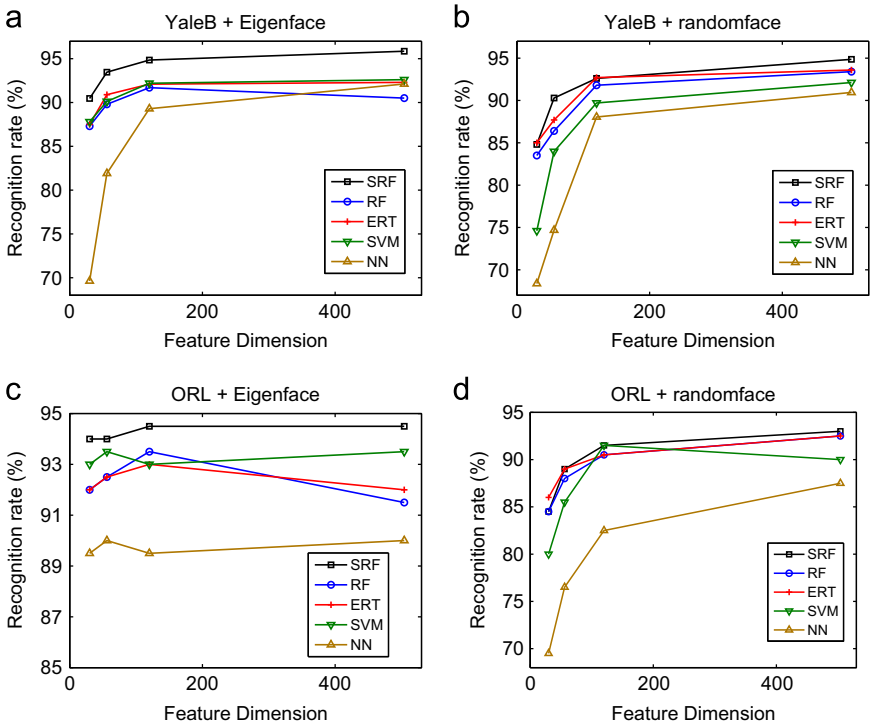

Fig. 13. Recognition rates on YaleB and ORL data sets using various feature dimensions: (a), (c) using Eigenface features. (b), (d) using random features.

requirements and accuracy. Different randomization methods may have different convergence profiles on different problems. In general, the extremely randomized trees method converges more slowly compared to other tree ensemble methods (RF and SRF) due to the totally randomized trees involved.

An interesting observation is that the performance of extremely randomized trees by the method of Geurts et al. appears considerably worse than or equal to those of standard RF on the synthetic data sets and gene data sets, which seems to contradict the original paper on extremely randomized trees. In [3], Geurts et al. have evaluated the performance of extremely randomized trees in 24 different data set with relevant features. The attribute size is not large (between 2 and 617), and the number of instances is much more than the number of attributes in these data sets (the number of instances per attribute, i.e., instance size divided by attribute size M/N, is between 10 and 1000). It is shown that the classification performance of extremely randomized trees is as accurate as, or even more accurate than RF as well as other ensemble methods on these data sets [3]. Thus we expect that extremely randomized trees can achieve good performance on data sets with relevant features and sufficient training instances. But, in our experiment, the synthetic data sets that we compiled include many noisy features. Whilst, for the gene data, the number of learning instances is insufficient compared to the large number of attributes (M/N is only between 0.003 and 0.031). In the case of handling a large number of noisy features and using insufficient learning instances, there is a high chance to miss the informative features in decision trees by using extremely random

Table 9
Test error (mean ± std-dev%) of pre-computed SRF and re-computed SRF on gene data sets.

Data set	Pre-computed SRF (strong)	Pre-computed SRF	Re-computed SRF
Colon	12.9 ± 9.5	11.3 ± 9.5	11.0 ± 9.5
DLBCL	2.6 ± 3.0	1.3 ± 3.0	1.3 ± 3.0
Embryonal-T	34.9 ± 10.6	31.7 ± 11.8	31.0 ± 11.2
Prostate-T	3.9 ± 4.0	2.7 ± 4.3	2.5 ± 4.3

Table 10
Computing times (mean ± std-dev seconds) of pre-computed SRF and re-computed SRF on gene data sets.

Data set	Pre-computed SRF (strong)	Pre-computed SRF	Re-computed SRF
Colon	6080 ± 136	4850 ± 273	12,600 ± 249
DLBCL	7490 ± 445	5230 ± 308	17,300 ± 1420
Embryonal-T	6050 ± 170	4780 ± 273	16,600 ± 1190
Prostate-T	9530 ± 212	7290 ± 233	28,100 ± 2170

sampling strategy, and the number of instance in the data set is not large enough to generate numerous nodes to make the resulting forest produce promising performance.

We also experiment to compare the performances between three implementations of our proposed SRF method: pre-computed SRF (strong), pre-computed SRF and re-computed SRF. In the pre-computed SRF (strong) method, we only select strong features into the subspaces. On the other hand, in the pre-computed SRF method and the re-computed SRF method, we include both strong and weak features. In this comparison, we generate 200 trees for each compared method and report their best results with optimal subspace setting. Tables 9 and 10 provide, respectively, classification accuracy and CPU times (in seconds) for three compared methods, averaged over fivefold cross-validation on gene data sets. All the comparisons are performed in a computer with 2.66 GHz CPU and 3.5 GB RAM.

Regarding classification accuracy, Table 9 shows that re-computed SRF is able to deliver better performance than pre-computed SRF with only strong features. This result indicates that interactions among features in a tree generation will have an impact on strong features. Given the pre-computed weak/strong features, we do not advocate using only strong features, but instead, ensuring the use of weak features together with strong features in tree generation. We believe that it is beneficial to include weak features into the subspaces because a weak feature may become a high informative feature after the split. We note that the pre-computed SRF method selecting both strong and weak features can deliver similar performance in comparison to re-computed SRF. Indeed, pre-computed SRF with both weak and strong features consistently produces promising results across all the experiments. Regarding computational time, Table 10 shows that pre-computed SRF is significantly faster than that of the re-computed SRF. The pre-computed SRF is on the average 3 times faster than the re-computed SRF.

In summary, pre-computed SRF by performing stratification before generating the trees is more computational efficient in comparison to re-computed SRF. But pre-computed SRF does not consider feature interactions. This shortcoming can be overcome to certain content by including the weak features into the subspaces. The implementation of re-computed SRF, on the other hand, has an advantage in handling feature interactions. Re-computed SRF delivers excellent performance on high dimensional data sets. However, re-computing requires additional operations, and the computational complexity is linearly on the number of features. When there is large number of features, the computations would become rather expensive. The choice of an appropriate implementation of SRF will essentially depend on the compromise between computational requirement and accuracy.

6. Conclusion

In this paper we have presented a stratified sampling method to select subspaces of features for building decision trees with random forests. The motivation for this new subspace feature selection method is to increase the strength and maintain the diversity of the trees in random forest for high dimensional data. This should result in a reduction of the generalization error and an increase in classification accuracy.

We introduce a stratified sampling method in contracts to the common random sampling method for high dimensional data. The approach identifies two groups of features: strong and weak features. The tree building algorithm then randomly chooses features from each group in building a tree.

Experimental results on several high dimensional data sets in gene classification, machine learning, computer vision and face recognition problems are presented. They show that for a very minimal additional computational cost, random forests with high classification accuracy can be built. We demonstrated that the performance of the proposed SRF method is better than current state-of-the-art algorithms including SVM, four common variants of random forest ensemble methods (RF, ERT, enrich-RF, and oblique-RF), nearest neighbor (NN), and naive Bayes (NB).

Acknowledgments

This research is supported in part by NSFC under Grant no. 61073195, and Shenzhen Science and Technology Program under Grant nos. CXB201005250024A and CXB201005250021 A.M. Ng's research supported in part by Centre for Mathematical Imaging and Vision, HKRGC Grant no. 201812 and HKBU FRG Grant no. FRG2/11-12/127.

References

[1] T.K. Ho, The random subspace method for constructing decision forests, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8) (1998) 832–844.
[2] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.
[3] P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Machine Learning 63 (1) (2006) 3–42.
[4] T. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization, Machine Learning 40 (2) (2000) 139–157.
[5] Y. Amit, D. Geman, Shape quantization and recognition with randomized trees, Neural Computation 9 (7) (1997) 1545–1588.
[6] R.E. Banfield, L.O. Hall, K.W. Bowyer, W.P. Kegelmeyer, A comparison of decision tree ensemble creation techniques, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (1) (2007) 173–180.
[7] G. Biau, L. Devroye, G. Lugosi, Consistency of random forests and other averaging classifiers, The Journal of Machine Learning Research 9 (2008) 2015–2033.
[8] Y. Lin, Y. Jeon, Random forests and adaptive nearest neighbors, Journal of the American Statistical Association 101 (474) (2006) 578–590.
[9] N. Meinshausen, Quantile regression forests, The Journal of Machine Learning Research 7 (2006) 999.
[10] R. Díaz-Uriarte, A. de Andrés, Gene selection and classification of microarray data using random forest, BMC Bioinformatics 7 (1) (2006) 3.
[11] A. Bureau, J. Dupuis, K. Falls, K. Lunetta, B. Hayward, T. Keith, P. Van Eerdewegh, Identifying SNPs predictive of phenotype using random forests, Genetic Epidemiology 28 (2) (2005) 171–182.
[12] B. Goldstein, A. Hubbard, A. Cutler, L. Barcellos, An application of random forests to a genome-wide association dataset: methodological considerations and new findings, BMC Genetics 11 (1) (2010) 49.

[13] Y. Meng, Y. Yu, L. Cupples, L. Farrer, K. Lunetta, Performance of random forest when SNPs are in linkage disequilibrium, BMC Bioinformatics 10 (1) (2009) 78.
[14] D. Schwarz, I. König, A. Ziegler, On safari to random jungle: a fast implementation of random forests for high-dimensional data, Bioinformatics 26 (14) (2010) 1752.
[15] J. Shotton, M. Johnson, R. Cipolla, Semantic texton forests for image categorization and segmentation, in: Proceedings of the IEEE CVPR Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[16] F. Moosmann, E. Nowak, F. Jurie, Randomized clustering forests for image classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (9) (2008) 1632–1646.
[17] V. Lepetit, P. Lagger, P. Fua, Randomized trees for real-time keypoint recognition, in: Proceedings of the IEEE CVPR Conference on Computer Vision and Pattern Recognition, 2005, pp. 775–781.
[18] A. Bosch, A. Zisserman, X. Muoz, Image classification using random forests and ferns, in: Proceedings of the IEEE ICCV International Conference on Computer Vision, 2007, pp. 1–8.
[19] G. Rogez, J. Rihan, S. Ramalingam, C. Orrite, P. Torr, Randomized trees for human pose detection, in: Proceedings of the IEEE CVPR Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[20] G.J. Williams, Combining decision trees: initial results from the MIL algorithm, in: J.S. Gero, R.B. Stanton (Eds.), Artificial Intelligence Developments and Applications: Selected Papers from the First Australian Joint Artificial Intelligence Conference, Sydney, Australia, 2–4 November, 1987, Elsevier Science Publishers, North-Holland, 1988, pp. 273–289.
[21] T.K. Ho, Random decision forests, in: Proceedings of the 3rd International Conference on Document Analysis and Recognition, IEEE Computer Society, 1995, pp. 278–282.
[22] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.
[23] L. Breiman, J. Friedman, R. Olsen, C. Stone, Classification and Regression Trees, 1984.
[24] T. Dietterich, Machine learning research: four current directions, AIM Magazine 18 (1997) 97–136.
[25] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: bagging, boosting, and variants, Machine Learning 36 (1–2) (1999) 105–139.
[26] D. Amaratunga, J. Cabrera, Y.S. Lee, Enriched random forests, Bioinformatics 24 (18) 2010–2014.
[27] T. Liu, F. Wang, G. Agrawal, Stratified sampling for data mining on the deep web, in: Proceedings of the IEEE ICDM International Conference on Data Mining, 2010, pp. 324–333.
[28] S. Chaudhuri, G. Das, V. Narasayya, Optimized stratified sampling for approximate query processing, ACM Transactions on Database Systems (TODS) 32 (2) (2007) 9.
[29] S. Fernandes, C. Kamienski, J. Kelner, D. Mariz, D. Sadok, A stratified traffic sampling methodology for seeing the big picture, Computer Networks 52 (2008) 2677–2689.
[30] J. Weston, A. Elisseeff, B. Schölkopf, M. Tipping, Use of the zero norm with linear models and kernel methods, The Journal of Machine Learning Research 3 (2003) 1439–1461.
[31] D. Cai, X. He, J. Han, Srda: an efficient algorithm for large-scale discriminant analysis, IEEE Transactions on Knowledge and Data Engineering 20 (1) (2007) 1–12.
[32] R. Caruana, N. Karampatziakis, A. Yessenalina, An empirical evaluation of supervised learning in high dimensions, in: Proceedings of the IEEE ICML International Conference on Machine Learning, 2008, pp. 96–103.
[33] D. Hand, R. Till, A simple generalisation of the area under the roc curve for multiple class classification problems, Machine Learning 45 (2) (2001) 171–186.
[34] B. Menze, B. Kelm, D. Splitthoff, U. Koethe, F. Hamprecht, On oblique random forests, Machine Learning and Knowledge Discovery in Databases (2011) 453–469.
[35] M. Varma, A. Zisserman, A statistical approach to texture classification from single images, International Journal of Computer Vision 62 (1–2) (2005) 61–81.
[36] J. Zhang, M. Marszalek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: a comprehensive study, in: Proceedings of the IEEE CVPR Conference on Computer Vision and Pattern Recognition, 2007, pp. 13–13.
[37] S. Zhu, Statistical modeling and conceptualization of visual patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (6) (2003) 691–712.
[38] D. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.
[39] E. Nowak, F. Jurie, B. Triggs, Sampling strategies for bag-of-features image classification, in: Proceedings of the ECCV European Conference on Computer Vision, 2006, pp. 490–503.
[40] V. Lepetit, P. Fua, Keypoint recognition using randomized trees, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (9) (2006) 1465–1479.
[41] G. Csurka, C. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: Proceedings of the ECCV European Conference on Computer Vision, vol. 1, 2004, p. 22.
[42] K. Chatfield, V. Lempitsky, A. Vedaldi, A. Zisserman, The devil is in the details: an evaluation of recent feature encoding methods, URL: http://www.robots.ox.ac.uk/~vgg/research/encoding_eval/.
[43] M. Turk, A. Pentland, Eigenfaces for recognition, Journal of Cognitive Neuroscience 3 (1) (1991) 71–86.
[44] A. Georghiades, P. Belhumeur, D. Kriegman, From few to many: illumination cone models for face recognition under variable lighting and pose, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (6) (2001) 643–660.
[45] F. Samaria, A. Harter, Parameterisation of a stochastic model for human face identification, in: Proceedings of the IEEE Conference on Applications of Computer Vision, 1994, pp. 138–142.

Yunming Ye received the Ph.D. degree in Computer Science from Shanghai Jiao Tong University. He is now an Associate Professor in the Department of Computer Science, Harbin Institute of Technology. His research interests include data mining, text mining, and clustering algorithms.

Qingyao Wu received his B.Sc. degree at the South China University of Technology in 2007, and the M.Sc. degree at Shenzhen Graduate School, Harbin Institute of Technology in 2009. He is currently pursuing Ph.D. degree in the Department of Computer Science, Shenzhen Graduate School, Harbin Institute of Technology. His research interests include data mining, machine learning and bioinformatics, particularly in multi-label learning and ensemble learning.

Joshua Zhexue Huang is a Professor and Chief Scientist at Shenzhen Institutes of Advanced Technology Chinese Academy of Sciences, and Honorary Professor at Department of Mathematics, The University of Hong Kong. He is known for his contribution to a series of k-means type clustering algorithms in data mining that is widely cited and used, and some have been included in commercial software. He has led the development to the open source data mining system AlphaMiner (http://www.alphaminer.org) that is widely used in education, research and industry. He has extensive industry expertise in business intelligence and data mining and has been involved in numerous consulting projects in Australia, Hong Kong, Taiwan and mainland China. Dr. Huang received his Ph.D. degree from the Royal Institute of Technology in Sweden. He has published over 100 research papers in conferences and journals.

Michael K. Ng is a Professor in the Department of Mathematics at the Hong Kong Baptist University. He obtained his B.Sc. degree in 1990 and M.Phil. degree in 1992 at the University of Hong Kong, and Ph.D. degree in 1995 at Chinese University of Hong Kong. He was a Research Fellow of Computer Sciences Laboratory at Australian National University (1995–1997), and an Assistant/Associate Professor (1997–2005) of the University of Hong Kong before joining Hong Kong Baptist University. His research interests include bioinformatics, data mining, image processing, scientific computing and data mining, and he serves on the editorial boards of international journals, see http://www.math.hkbu.edu.hk/~mng/.

Xutao Li received the B.S. and M.S. degrees in Computer Science from Lanzhou University and Harbin Institute of Technology in China in 2007 and 2009, respectively. He is currently working towards the Ph.D. degree in the Department of Computer Science, Harbin Institute of Technology. His research interests include data mining, clustering algorithms, and machine learning.