Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 590678, 12 pages
http://dx.doi.org/10.1155/2015/590678

Research Article

A New Ensemble Method with Feature Space Partitioning for High-Dimensional Data Classification

Yongjun Piao,[1] Minghao Piao,[1] Cheng Hao Jin,[1] Ho Sun Shon,[2] Ji-Moon Chung,[3] Buhyun Hwang,[4] and Keun Ho Ryu[1]

[1]*Database and Bioinformatics Laboratory, College of Electrical and Computer Engineering, Chungbuk National University, Cheongju 362763, Republic of Korea*
[2]*Graduate School of Professional Science Master, Chungbuk National University, Cheongju 362763, Republic of Korea*
[3]*Department of Computer Science, Namseoul University, Cheonan 331707, Republic of Korea*
[4]*School of Electronics & Computer Engineering, Chonnam National University, Gwangju 500757, Republic of Korea*

Correspondence should be addressed to Keun Ho Ryu; khryu@dblab.chungbuk.ac.kr

Received 25 November 2014; Accepted 5 January 2015

Academic Editor: Sanghyuk Lee

Ensemble data mining methods, also known as classifier combination, are often used to improve the performance of classification. Various classifier combination methods such as bagging, boosting, and random forest have been devised and have received considerable attention in the past. However, data dimensionality increases rapidly day by day. Such a trend poses various challenges as these methods are not suitable to directly apply to high-dimensional datasets. In this paper, we propose an ensemble method for classification of high-dimensional data, with each classifier constructed from a different set of features determined by partitioning of redundant features. In our method, the redundancy of features is considered to divide the original feature space. Then, each generated feature subset is trained by a support vector machine, and the results of each classifier are combined by majority voting. The efficiency and effectiveness of our method are demonstrated through comparisons with other ensemble techniques, and the results show that our method outperforms other methods.

1. Introduction

The ultimate goal of supervised learning for classification is to mine previously unknown knowledge from existing data to predict a future event with best possible classification performance [1]. Classification algorithms typically deal with a set of records, each of which consists of a fixed number of features along with a class label that denotes its target. The algorithm then outputs a decision boundary that represents underlying patterns in the data. Many useful classification algorithms such as decision tree [2], neural network [3, 4], and support vector machine (SVM) [5] have been presented in the past. However, the increase in the data dimensionality may cause several issues with respect to scalability and learning performance in these classification algorithms. Moreover, the classification ability of a single classifier is limited.

In general, ensembles of classifiers provide better classification accuracy than a single predictor can do. To improve the classification accuracy, ensemble methods, also known as classifier combination, first generate a set of base classifiers from training data and then perform actual classification by combining the results of base classifiers. For achieving better accuracy of the combined set of multiple classifiers, each base classifier should be diverse and independent. When it comes to building each base classifier, ensemble classifier generation methods can be broadly categorized into four groups [6]: (i) by selecting different subsets of instances of training set to build each base classifier, (ii) by choosing different subsets of features of the input features to construct each base classifier, (iii) by being based on different categories of the class labels to build each base classifier, and (iv) by manipulating the learning algorithm. Among many methods, bagging [7] and

Input: training data D, Inducer I, number of bootstrap samples N
Output: Aggregated classifier C^*
Begin:
(1) for $i = 1$ to N {
(2) D' = bootstrap sample from D (sample with replacement)
(3) $C_i = I(D')$
(4) }
(5) $C^*(x) = \arg\max_{y \in Y} \sum_{i:C_i(x)=y} 1$
End

Algorithm 1: Bagging.

boosting [8] are two widely used ensemble methods. They resample the original data to create multiple training sets based on some sampling distribution and build the base classifier from each bootstrap sample. However, these methods are not guaranteed to generate fully independent individual base classifiers [9]. According to [10, 11], their theoretical and empirical results indicate that the most effective method of achieving independence is by training base classifiers on different feature subsets [12]. The basic idea of feature subset-based ensemble is simply to give each classifier a different projection of the training set [13]. In particular for high-dimensional data, adopting independent feature subsets for ensemble generation has shown to be more efficient [14] compared with manipulating the training samples. This may be due to the following: (i) a feature subset-based ensemble can perform faster due to the reduced size of input space; (ii) it can reduce the correlation among the classifiers. Among the feature subset-based ensemble methods, random forest [15] is a widely used approach that employs decision tree as a base classifier. It achieves diversity by randomly partitioning the original feature space instead of using whole features. However, random partition of the input space may increase the risk that irrelevant and redundant features can be included in the selected subset. Furthermore, decision tree methods have the so-called fragmentation problem as less and less training data are used to search for the root nodes of subtrees. If the training data do not have enough instances compared with dimensions, the performance of decision tree becomes typically very poor.

In this paper, we propose an ensemble framework for classifying high-dimensional data with each classifier constructed from a different set of features determined by redundant features partitioning. First, we suggest a multiple subset generation method based on feature relevance and redundancy to construct each classifier. Then, a number of classifiers are built from the generated subsets. Finally, the classification results of the classifiers are combined by majority voting. It is observed that the proposed ensemble method outperforms other ensemble methods by up to 6% in terms of classification accuracy.

2. Previous Work

2.1. Bagging. Bagging [7] is a method for generating multiple versions of classifiers and using these to get an aggregated classifier. Each base classifier is generated by different bootstrap samples. Algorithm 1 shows the bagging algorithm [14]. The algorithm takes training data D, inducer I, and the number of bootstrap samples N as input and then produces an ensemble classifier which is the combination of the classifiers trained from the multiple bootstrap samples. D' is obtained by repeatedly sampling instances from a dataset according to probability distribution (line 2). Since the sampling is done with replacement, some instances may appear several times in the same training set, while others may not. Consequently, N bootstrap samples, $D_1, D_2, \ldots, D_N$, are generated, from which a classifier C_i is trained by using each bootstrap sample D_i (line 3). Finally, a combined classifier C^* is built from $C_1, C_2, \ldots, C_i$, and C^* predicts the class label of a given instance x by counting votes (line 5).

2.2. Boosting. Boosting [6] is also a widely used ensemble method developed to improve the performance of learning algorithms that generate multiple classifiers and vote on them. Unlike bagging, boosting assigns a weight to each training instance and may adaptively change the weight at the end of each boosting round. AdaBoost is an improved boosting algorithm whose pseudo code is shown in Algorithm 2 [14]. The algorithm takes as input training data D containing m instances, inducer I, and iteration parameter N and then outputs a combined classifier. Initially, all of the instances are equally assigned the same weight (line 1). Then, the algorithm gradually constructs classifiers by modifying the weights of training instances based on the previous classifier's performance (lines 2–9). This is accomplished by computing the new classifier while putting more emphasis on those objects previously found to be difficult to accurately classify. After generating each classifier, the proportion of incorrect classification rate is calculated (line 4). If the weighted error is larger than 0.5, the current D' will be set to a bootstrap sample with weight 1 for every instance. Otherwise, the weight of correctly classified instances will be updated by a factor inversely proportional to the error (lines 6–8). In other words, if the current classifier finds a certain object difficult to classify, then that object will be assigned a greater weight for the next iteration. Conversely, if an object is found to be easy to classify, then it will have smaller weight in the next iteration. Finally, the classifiers are combined using a weighted voting scheme (line 10).

Input: training data D size of m, Inducer I, number of iterations N
Output: Aggregated classifier C^*
Begin:
(1) $D' = D$ with instance weights assigned to be 1
(2) for $i = 1$ to N {
(3) $C_i = I(D')$
(4) $\varepsilon_i = \frac{1}{m} \sum_{x_j \in D' : C_i(x_j) \neq y_j} \text{weight}(x)$
(5) If $\varepsilon_i > 1/2$, set D' to a bootstrap sample from D with weight 1 for every instance and go to Step 3
(6) $\beta_i = \varepsilon_i / (1 - \varepsilon_i)$
(7) For each $x_j \in D'$, if $C_i(x_j) = y_j$ then $\text{weigh}(x_j) = \text{weight}(x_j) \cdot \beta_i$
(8) Normalize the weights of instances so the total weight of D' is m
(9) }
(10) $C^*(x) = \arg\max_{y \in Y} \sum_{i : C_i(x) = y} \log \frac{1}{\beta}$
End

Algorithm 2: AdaBoost.

Input: training data D, number of selected variables m, number of trees N
Output: Aggregated classifier C^*
Begin:
(1) for $i = 1$ to N {
(2) D' = bootstrap sample from D (sample with replacement)
(3) S' size of $m = S$ (S' will be randomly selected from original input space)
(4) $C_i = I(D', S')$ (I: Classification and regression tree)
(5) }
(6) $C^*(x) = \arg\max_{y \in Y} \sum_{i : C_i(x) = y} 1$
End

Algorithm 3: Random forest.

2.3. Random Forest. Random forest is an ensemble classification method consisting of multiple unpruned decision trees. Unlike bagging, random forest forms bootstrap samples by randomly partitioning the original feature space instead of using the whole input features. As shown in Algorithm 3, to construct individual decision trees, bootstrap samples are selected from the training instances with replacement (line 2). Then, classification and regression tree (CART) algorithm is applied to grow the decision tree. At the node selection stage, it decides the best splitting node from a randomly selected subspace of m features (lines 3-4).

2.4. Feature Subset-Based Ensembles. Bagging and boosting are the ensemble methods that manipulate the original instances. However, this kind of ensemble methods is difficult to accurately classify high-dimensional data like image or gene expression data. The reason is that image or gene expression data generally has very small number of samples compared with dimensions. Therefore, sampling the training instances will lead to lack of representative instances so that bagging and boosting will be susceptible to overfitting. In this case, feature subset-based ensemble method is more efficient [14] compared with manipulating the training samples. This may be due to the following: (i) a feature subset-based ensemble can perform faster due to the reduced size of input space; (ii) it can reduce the correlation among the classifiers. Besides random forest, various feature partitioning-based ensemble methods have been proposed. Ahn et al. [16] proposed an ensemble method that uses mutually exclusive subspaces to achieve diversity. The authors applied their method to bioinformatics and chemical domains and showed that their method can achieve better performance than that of random forest. Ming Ting et al. [17] also introduced a feature subset-based ensemble method that employs support vector machine as a base classifier. The feature space was divided into nonoverlapping local regions according to user-defined number of features. de Bock and Poel [18] proposed a rotation-based ensemble classifier that applied feature extraction methods such as principle component analysis and independent component analysis to generate subspace of features. However, these methods did not consider the correlation among features [19].

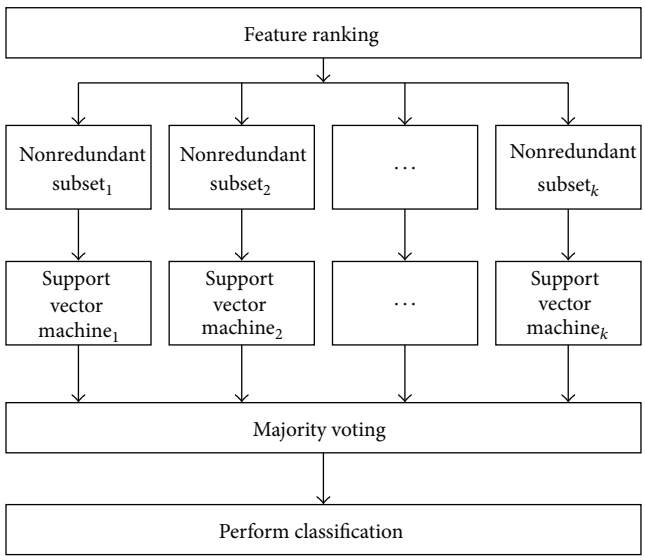

Figure 1: Framework of proposed method.

3. Proposed Ensemble Method

We propose an ensemble method with multiple independent feature subsets to better classify high-dimensional data. The framework of the proposed ensemble is shown in Figure 1. The proposed method mainly consists of two phases: (i) generating multiple feature subsets based on the correlation among features and (ii) constructing the model from each feature subset using a machine learning algorithm as the base classifier and combining the results of all classifiers by majority voting. Next, we will illustrate each step in detail.

3.1. Feature Subset Generation. Generating feature subsets for ensemble can be viewed as multiple iterations of feature selection procedures. In the past, various feature selection techniques have been proposed such as chi-square test, mutual information, Pearson correlation coefficients, and Relief [20]. Although these methods are fast, they lack robustness when interactions among features exist. To select a relevant and nonredundant feature subset, a Fast Correlation-Based Filter (FCBF) [21] approach was proposed to remove the redundant as well as irrelevant features, and the Symmetrical Uncertainty (SU) was used to measure the correlation where

$$\begin{aligned} \mathrm{SU}(X,Y) &= 2\left[\frac{\mathrm{IG}(X \mid Y)}{H(X)+H(Y)}\right], \\ \mathrm{IG}(X \mid Y) &= H(X) - H(X \mid Y). \end{aligned} \tag{1}$$

Here $\mathrm{IG}(X \mid Y)$ is the information gain of X after observing variable Y. $H(X)$ and $H(Y)$ are the entropies of variables X and Y, respectively. FCBF removes irrelevant features by ranking correlation between features and classification classes. To remove redundant features, the authors introduced a concept of predominant feature. A feature is said to be predominant if it does not have any approximate Markov Blanket in the current set. For two relevant features F_i and F_j, F_j forms an approximate Markov Blanket for F_i if

$$\mathrm{SU}_{j,c} \geq \mathrm{SU}_{i,c}, \qquad \mathrm{SU}_{i,j} \geq \mathrm{SU}_{j,c}, \tag{2}$$

where $\mathrm{SU}_{i,c}$ is the correlation between feature i and class; $\mathrm{SU}_{j,c}$ is the correlation between feature j and class; $\mathrm{SU}_{i,j}$ is the correlation between feature and feature. Thus, FCBF is a process in which all predominant features are identified. They are searched as follows. First, the feature with the largest $\mathrm{SU}_{i,c}$ value is selected as a starting point. Next, all redundant features regarding this feature are removed. Then redundant features regarding the next feature with the largest SU in the remaining set are removed. The algorithm repeats this procedure until there are no redundant features existing.

Although FCBF has good performance on high-dimensional data, it is not suitable for ensemble learning because it was originally designed to select a single feature subset. Thus, we extend FCBF to generate multiple feature subsets. First, based on the correlation between features and classes (i.e., SU), all the features are sorted in a descending order. Then a relevant subset of features can be derived by a predefined threshold σ. If the SU value of a feature is larger than the threshold, the feature is considered to be relevant. Generally, we recommend setting the threshold to be 0 in order to consider all of the features in the redundancy analysis step except "waste-features" which have a 0 SU value with respect to the class. After that, redundancy analysis is conducted on the relevant subset. The main difference between our method and FCBF is that our method considers the removed features in FCBF. It is because we hypothesize that it may be interesting to pay attention to the removed features as FCBF removes less relevant ones between two redundant features, and in some cases, low ranked features can also play an important role when considering the combination of features. Thus, the features not selected in the previous iteration will be the input in the next iteration. For example, in the first iteration, the redundant features are removed from original space as is done in FCBF. In the second iteration, the same analysis is done as iteration 1 but for the removed subset in the first iteration not for the whole feature space. Then, the third subset is selected from the removed space in the second iteration. It is repeated until a user-defined number of subsets are selected.

3.2. Model Learning. Over the past few years, SVM has been widely used for classification because of its good performance on high-dimensional data [22]. SVM was developed by Vapnik to solve the problems occurring in applications such as handwritten digit recognition [23], object recognition [24], text classification [25], cancer diagnosis [23], and bioinformatics [26]. Hence, we use SVM as the base classifier in our ensemble method. The goal of SVM is to find a hyperplane with a maximal margin (distance between two groups of data points) as defined and illustrated in Figure 2. Given some data points that are assumed to be divided into two groups, circles and squares, the hyperplane can be written as

$$w \cdot x + b = 0, \tag{3}$$

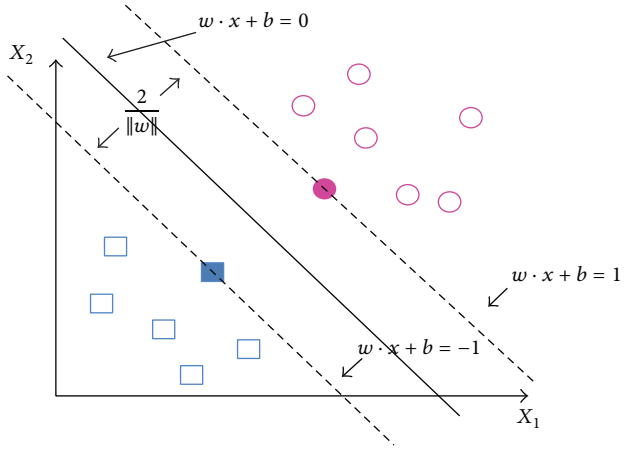

Figure 2: Example of support vector machine.

where w and b are parameters of the model: w denotes an orthogonal vector and b refers to a bias. SVM separates data points into two groups in such a way that they divide the data and there exist no data points in between them, and their distance, defined as margin, is maximized. Figure 2 shows two more hyperplanes placed at the boundary of two groups. These hyperplanes and the margin can be written as follows:

$$w \cdot x + b = 1, \qquad w \cdot x + b = -1,$$
$$\text{margin} = \frac{2}{\|w\|}. \tag{4}$$

Hence, the learning task in SVM can be formalized as the following constrained optimization problem:

$$\min_w \quad \frac{\|w\|^2}{2}, \tag{5}$$
$$\text{subject to} \quad y_i (w \cdot x_i + b) \geq 1, \quad i = 1, 2, \ldots, n.$$

This is also known as a convex optimization problem, which can be solved by using the standard Lagrange multiplier method:

$$L_p = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{N} \sigma_i \left(y_i (w \cdot x_i + b) - 1 \right), \tag{6}$$

where parameters σ_i are called the Lagrange multipliers. With the Lagrange multipliers, the decision function can be written as follows:

$$f(x) = \text{sgn} \left(\sum_{i=1}^{n} \sigma_i y_i K(x_i, x) + b \right). \tag{7}$$

Additionally, the results of each classifier are combined by majority voting, and classification of unknown data is performed based on the class label to obtain the most frequent votes. The mathematical function of our ensemble method with k classifiers can be written as

$$\text{class}(x) = \arg\max \left(\sum_{k} (f_k(x), c_i) \right). \tag{8}$$

Table 1: Datasets used in our experiments.

Dataset	Instances	Features	Classes
AR10P	130	2,400	10
ORL10P	100	10,304	10
PIE10P	210	2,420	10
PIX10P	100	10,000	10
Leukemia	72	12,582	3
Prostate	102	12,600	2

4. Experimental Results

4.1. Dataset. To evaluate the effectiveness of our method, we used six publicly available datasets from two different domains, four from face recognition and two from DNA microarray data classification. The purpose of using first four datasets, namely, AR10P [27], ORL10P [28], PIE10P [29], and PIX10P [30], is to show how well our method can classify the image data. Each dataset has a large number of features compared with the number of instances. The last two datasets are Leukemia and Prostate datasets from DNA microarray experiments and are used to show how well our method can distinguish different types of cancers. The Leukemia dataset [31] contains a total of 72 samples in three classes, acute lymphoblastic leukemia (ALL), acute myeloid leukemia (AML), and mixed-lineage leukemia gene (MLL), which have 24, 28, and 20 samples, respectively. The number of features is 12,582. The Prostate dataset, first published in [32], embeds a two-class classification problem and contains 102 samples and 12,600 genes. One of the tasks addressed by the authors was to build a model that can distinguish between normal and tumorous prostate tissues. The summary of the datasets is shown in Table 1.

4.2. Performance Evaluation. We compared our methods with widely used ensemble methods: bagging, AdaBoost, and random forest. For bagging, AdaBoost, and our method, SVM was used as the base classifier for fair comparison. The number of classifiers for each ensemble method was set to 20. To obtain a statistically reliable predictive measurement, we performed 10 runs of 10-fold cross validation on all the datasets. In 10-fold cross validation, each dataset was randomly partitioned into ten parts. Nine parts were used as the training set, and the remaining one was used as the testing dataset. Selecting the kernel and appropriate parameters plays an important role in SVM classification performance. The RBF kernel is a commonly used kernel for three reasons [33]. First, the RBF kernel can handle nonlinear relationship between class labels and attributes. Second, it has fewer hyperparameters that influence the complexity of the model selection than that of the polynomial kernel. Third, the RBF kernel has fewer numerical difficulties. In our experiments, we chose the RBF kernel function, and the parameters c and r of RBF kernel must be optimized for each dataset. To determine the best values of c and r, we conducted a grid-search approach using 10-fold cross validation. A number of pairs of (c, r) values were attempted, and the pair with the best

Table 2: Performance of proposed method on AR10P dataset. Each row indicates the performance on each class and the last row shows the weighted average.

Class	Instances	TP rate	FP rate	Precision	Recall	F-measure	Roc area
1	13	1	0.017	0.867	1	0.929	1
2	13	0.923	0	1	0.923	0.96	0.995
3	13	0.923	0	1	0.923	0.96	0.998
4	13	1	0.009	0.929	1	0.963	1
5	13	0.923	0.009	0.923	0.923	0.923	0.983
6	13	1	0	1	1	1	1
7	13	0.923	0	1	0.923	0.96	0.992
8	13	1	0	1	1	1	1
9	13	1	0	1	1	1	1
10	13	1	0	1	1	1	1
Average		0.969	0.003	0.972	0.969	0.969	0.977

Table 3: Performance of proposed method on ORL10P dataset. Each row indicates the performance of each run and the last row shows the weighted average of 10 runs.

Class	Instances	TP rate	FP rate	Precision	Recall	F-measure	Roc area
1	10	1	0	1	1	1	1
2	10	1	0	1	1	1	1
3	10	1	0	1	1	1	1
4	10	1	0	1	1	1	1
5	10	1	0	1	1	1	1
6	10	1	0	1	1	1	1
7	10	1	0	1	1	1	1
8	10	1	0	1	1	1	1
9	10	1	0	1	1	1	1
10	10	1	0	1	1	1	1
Average		1	0	1	1	1	1

Table 4: Performance of proposed method on PIE10P dataset. Each row indicates the performance of each run and the last row shows the weighted average of 10 runs.

Class	Instances	TP rate	FP rate	Precision	Recall	F-measure	Roc area
1	21	1	0	1	1	1	1
2	21	1	0	1	1	1	1
3	21	1	0.005	0.955	1	0.977	1
4	21	1	0.005	0.955	1	0.977	0.998
5	21	1	0	1	1	1	1
6	21	0.952	0	1	0.952	0.976	0.996
7	21	1	0	1	1	1	1
8	21	0.952	0	1	0.952	0.976	0.979
9	21	1	0	1	1	1	1
10	21	1	0	1	1	1	1
Average		0.99	0.001	0.991	0.99	0.99	0.997

accuracy was picked in the range of $c \in \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$ and $r \in \{2^{-15}, 2^{-13}, \ldots, 2^{3}\}$.

Tables 2 through 7 show the performance of our proposed method in terms of TP rate, FP rate, precision, recall, F-measure, and ROC area, respectively. TP rate and FP rate refer to the proportion of actual positive instances correctly predicted as positive and the proportion of actual negative instances wrongly predicted as positive [34–36], respectively. Precision is computed as the number of true positive instances divided by the total number of instances labelled as belonging to the positive class. Recall is defined as the number of true positive instances divided by the total number of instances actually belonging to the positive class. F-measure is an evaluation metric that combines precision and recall

Table 5: Performance of proposed method on PIX10P dataset. Each row indicates the performance of each run and the last row shows the weighted average of 10 runs.

Class	Instances	TP rate	FP rate	Precision	Recall	F-measure	Roc area
1	10	1	0	1	1	1	1
2	10	1	0	1	1	1	1
3	10	1	0	1	1	1	1
4	10	1	0	1	1	1	1
5	10	1	0	1	1	1	1
6	10	1	0	1	1	1	1
7	10	0.9	0	1	0.9	0.947	0.947
8	10	1	0.011	0.909	1	0.952	0.996
9	10	1	0	1	1	1	1
10	10	1	0	1	1	1	1
Average		0.99	0.001	0.991	0.99	0.99	0.994

Table 6: Performance of proposed method on Leukemia dataset. Each row indicates the performance of each run and the last row shows the weighted average of 10 runs.

Class	Instances	TP rate	FP rate	Precision	Recall	F-measure	Roc area
ALL	24	1	0	1	1	1	1
AML	20	0.95	0	1	0.95	0.974	0.975
MLL	28	1	0.023	0.966	1	0.982	0.989
Average		0.986	0.009	0.987	0.986	0.986	0.989

Table 7: Performance of proposed method on Prostate dataset. Each row indicates the performance of each run and the last row shows the weighted average of 10 runs.

Class	Instances	TP rate	FP rate	Precision	Recall	F-measure	Roc area
N	50	0.98	0.058	0.942	0.98	0.961	0.96
T	52	0.942	0.02	0.98	0.942	0.961	0.96
Average		0.961	0.038	0.962	0.961	0.961	0.96

as follows: F-measure = 2 $*$ (precision $*$ recall)/(precision + recall). ROC area is defined as the area under the Receiver Operating Characteristic (ROC) curve. Each row of the tables indicates the performance on each class and the last row shows the averaged performance. From the tables, the average TP rate is found to be 0.969, 1, 0.99, 0.99, 0.986, and 0.961, and the average FP rate is found to be 0.003, 0, 0.001, 0.001, 0.009, and 0.038 on six datasets. Hence, we can easily observe that our method makes good prediction. Moreover, the ROC area on ORL10, PIE10P, and PIX10P is almost 100%. From the tables, it is clear that our method shows good performance on many different evaluation measures.

Figures 3 and 4 exhibit the box plot of classification accuracies of our method, bagging, AdaBoost, and random forest. On AR10P dataset, the proposed method shows best average prediction accuracy which is 96.152%. On Leukemia dataset, it is clear that the proposed method is found to result in best average prediction accuracy, which is 98.75% (standard deviation = 0.44), while the other methods are found to be 94.30% (standard deviation = 1.38), 96.80% (standard deviation = 0.67), and 82.08% (standard deviation = 2.81) for bagging, AdaBoost, and random forest, respectively. Similar results can also be found in other figures. One interesting observation is that random forest has relatively poor performance. It may be because random forest uses decision tree as base classifier, while other methods use SVM. It is well known that SVM has better performance on high-dimensional data than decision trees. The classification accuracies of each run can be seen in Appendix if the reader is interested.

To test the statistical significance of differences among classifiers, a paired-samples t-test is performed regarding bagging and the proposed method. We selected bagging because it showed the best average classification accuracy among the existing methods in most cases. On ORL10P dataset, we selected random forest instead of bagging, because the performance of our method and bagging is exactly the same. From Table 8, the hypothesis that the mean accuracy of proposed method is equal to the mean accuracy of bagging has been significantly rejected (t = 3.407, P value = 0.007 on AR10P, t = 9.391, P value = 0.000 on ORL10P, t = 4.303, P value = 0.002 on PIE10P, t = 11.599, P value = 0.000 on PIX10P, t = 9.816, P value = 0.000 on Leukemia, and t = 9.000, P value = 0.000 on Prostate dataset) with 5% significance level. It means that the differences among classifiers are statistically significant.

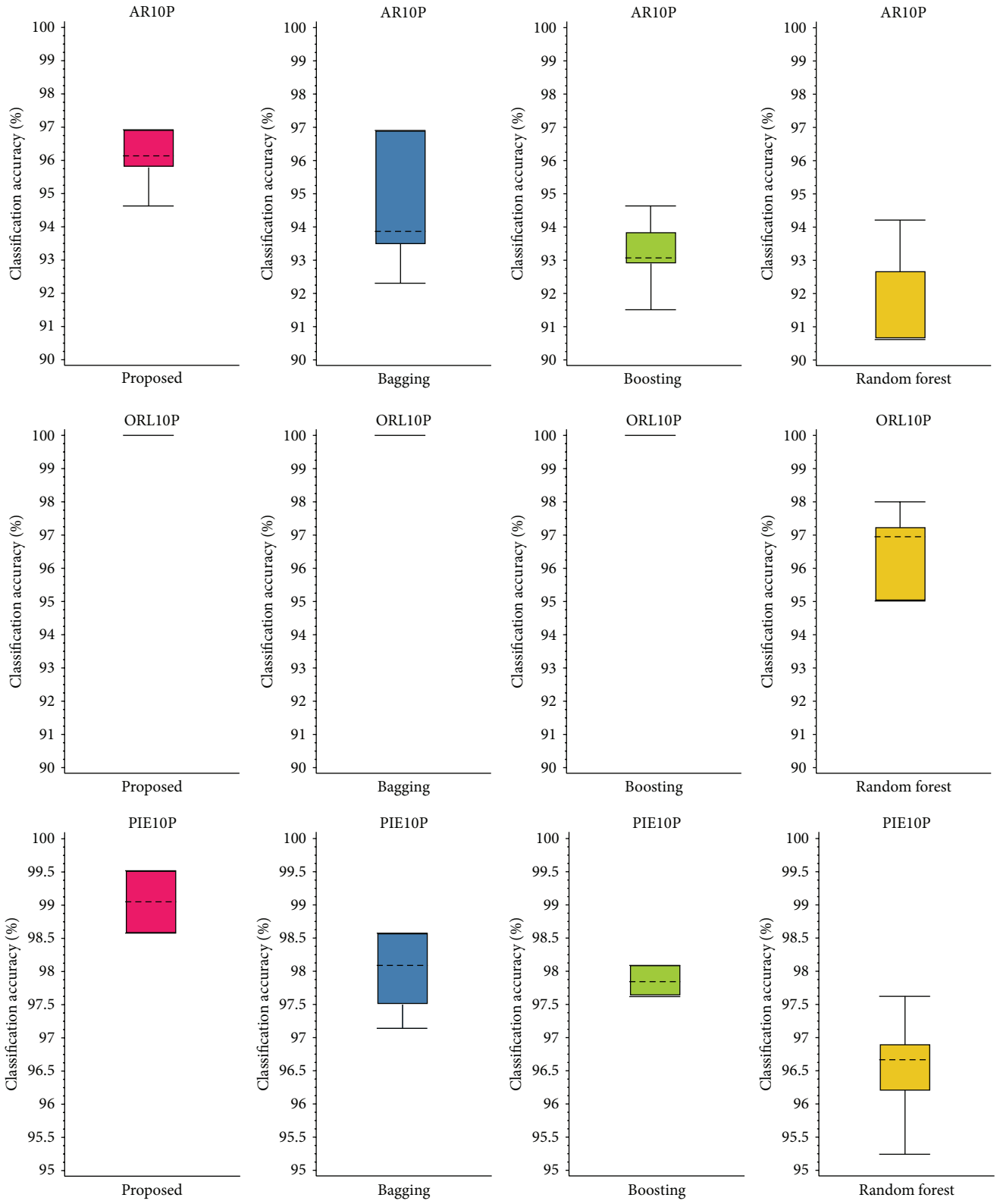

Figure 3: Box plot of classification accuracy on AR10P, ORL10P, and PIE10P datasets.

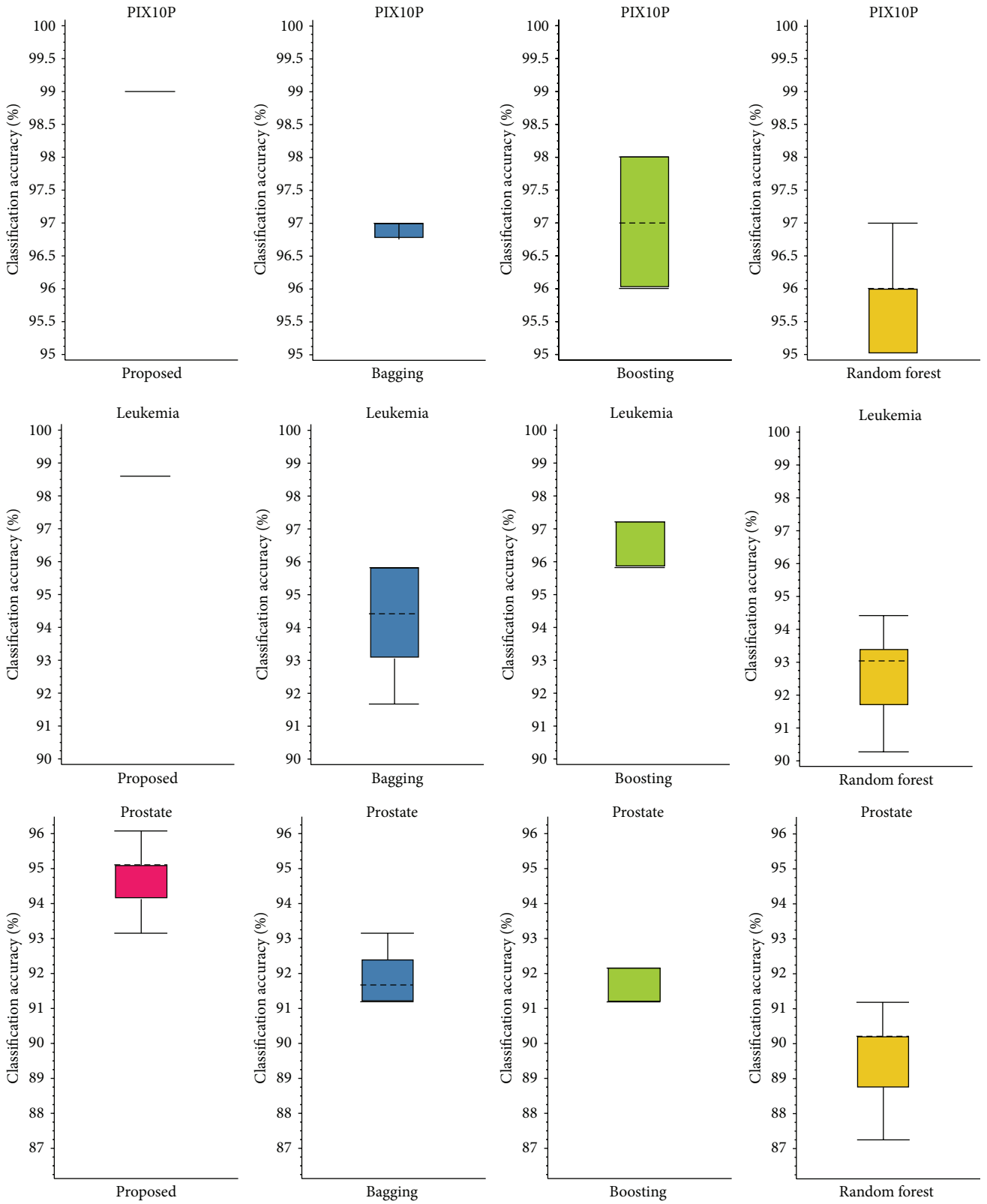

Figure 4: Box plot of classification accuracy on PIX, Leukemia, and Prostate datasets.

Table 8: Paired t-test between proposed method and bagging.

	Mean	Standard deviation	Standard error mean	95% confidence interval of the difference		t	df	Sig.
				Lower	Upper			
Proposed method – bagging on AR10P dataset	1.536	1.495	0.451	0.531	2.541	3.407	10	0.007
Proposed method – forest on ORL10P dataset	3.500	1.179	0.373	2.657	4.343	9.391	9	0.000
Proposed method – bagging on PIE10P dataset	1.048	0.770	0.244	0.497	1.599	4.303	9	0.002
Proposed method – bagging on PIX10P dataset	2.100	0.568	0.180	1.694	2.506	11.699	9	0.000
Proposed method – bagging on Leukemia dataset	4.445	1.432	0.453	3.421	5.469	9.816	9	0.000
Proposed method – bagging on Prostate dataset	2.940	1.033	0.327	2.201	3.679	9.000	9	0.000

Table 9: Classification accuracy of 10 runs of 10-fold cross validation for proposed method, bagging, AdaBoost, and random forest on AR10P.

	Proposed	Bagging	AdaBoost	Random forest
1	96.92	93.85	93.08	92.69
2	96.92	96.92	93.08	91.15
3	94.62	92.31	91.54	90.62
4	94.62	95.38	94.62	92.69
5	96.15	93.85	93.08	94.23
6	96.15	92.31	93.85	92.69
7	96.15	93.85	93.85	92.62
8	96.92	96.92	92.31	90.62
9	96.15	93.85	93.85	92.69
10	96.92	96.92	93.08	90.62

Table 10: Classification accuracy of 10 runs of 10-fold cross validation for proposed method, bagging, AdaBoost, and random forest on ORL10P.

	Proposed	Bagging	AdaBoost	Random forest
1	100	100	100	95
2	100	100	100	98
3	100	100	100	97
4	100	100	100	95
5	100	100	100	97
6	100	100	100	95
7	100	100	100	97
8	100	100	100	97
9	100	100	100	96
10	100	100	100	98

Table 11: Classification accuracy of 10 runs of 10-fold cross validation for proposed method, bagging, AdaBoost, and random forest on PIE10P.

	Proposed	Bagging	AdaBoost	Random forest
1	99.05	98.57	98.09	97.62
2	98.57	97.62	98.09	95.24
3	98.57	98.09	98.09	96.67
4	98.57	98.09	97.62	96.67
5	99.52	97.62	97.62	96.19
6	99.05	97.14	97.62	96.19
7	99.52	98.57	97.62	98.10
8	99.52	97.14	97.62	96.67
9	99.52	98.57	98.09	96.19
10	98.57	98.57	98.09	96.67

Table 12: Classification accuracy of 10 runs of 10-fold cross validation for proposed method, bagging, AdaBoost, and random forest on PIX10P.

	Proposed	Bagging	AdaBoost	Random forest
1	99	97	97	96
2	99	96	96	96
3	99	96	98	95
4	99	97	96	96
5	99	97	98	97
6	99	97	97	95
7	99	98	97	96
8	99	97	98	95
9	99	97	97	95
10	99	97	96	96

5. Conclusion

In this paper, we presented a feature partitioning-based ensemble method to better classify high-dimensional data. In our method, each base classifier was trained from different feature space by dividing redundant features into different subsets. SVM was used as the base classifier and the results of each SVM were merged by a majority voting method. For the experiments, we used six publicly available datasets in two different domains. Through the experiments, we demonstrated that dividing the redundant features into several parts for ensemble construction can achieve better performance

Table 13: Classification accuracy of 10 runs of 10-fold cross validation for proposed method, bagging, AdaBoost, and random forest on Leukemia.

	Proposed	Bagging	AdaBoost	Random forest
1	98.61	95.83	97.22	91.67
2	100	94.44	97.22	93.06
3	98.61	95.83	97.22	91.67
4	98.61	95.83	97.22	90.28
5	98.61	91.67	95.83	93.06
6	98.61	94.44	97.22	94.44
7	98.61	94.44	97.22	93.06
8	98.61	93.06	95.83	91.67
9	98.61	93.06	97.22	93.06
10	98.61	94.44	95.83	97.22

Table 14: Classification accuracy of 10 runs of 10-fold cross validation for proposed method, bagging, AdaBoost, and random forest on Prostate.

	Proposed	Bagging	AdaBoost	Random Forest
1	95.1	91.18	91.18	89.22
2	95.1	92.16	92.16	90.20
3	95.1	93.14	92.16	90.20
4	94.12	91.18	89.22	87.25
5	94.12	91.18	91.18	85.29
6	93.14	92.16	92.16	90.20
7	95.1	91.18	91.18	89.22
8	96.08	92.16	91.18	90.20
9	95.1	93.14	92.16	90.20
10	95.1	91.18	91.18	91.18

for classification on high-dimensional data and that our proposed algorithm has higher prediction accuracies than other ensemble classification algorithms.

Appendix

See Tables 9, 10, 11, 12, 13, and 14.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (No. 2013R1A2A2A01068923) and (No. 2013R1A1A206518), Export Promotion Technology Development Program, Ministry of Agriculture, Food and Rural Affairs (No. 114083-3), and Special Research Program of Chonnam National University, 2009 (2009-0413).

References

[1] H. Zhao, A. P. Sinha, and W. Ge, "Effects of feature construction on classification performance: an empirical study in bank failure prediction," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2633–2644, 2009.

[2] B. Pradhan, "A comparative study on the predictive ability of the decision tree, support vector machine and neuro-fuzzy models in landslide susceptibility mapping using GIS," *Computers & Geosciences*, vol. 51, pp. 350–365, 2013.

[3] D. C. Ciresan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural Networks*, vol. 30, pp. 333–338, 2012.

[4] C. H. Jin, G. Pok, H.-W. Park, and K. H. Ryu, "Improved pattern sequence-based forecasting method for electricity load," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 9, no. 6, pp. 670–674, 2014.

[5] Z. Qi, Y. Tian, and Y. Shi, "Robust twin support vector machine for pattern classification," *Pattern Recognition*, vol. 46, no. 1, pp. 305–316, 2013.

[6] A. Rahman and B. Verma, "Ensemble classifier generation using non-uniform layered clustering and Genetic Algorithm," *Knowledge-Based Systems*, vol. 43, pp. 30–42, 2013.

[7] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[8] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *International Conference on Machine Learning*, pp. 148–156, 1996.

[9] S. B. Cho and H.-H. Won, "Cancer classification using ensemble of neural networks with multiple significant gene subsets," *Applied Intelligence*, vol. 26, no. 3, pp. 243–250, 2007.

[10] K. Tumer and J. Ghosh, "Classier combining: analytical results and implications," in *Proceedings of the National Conference on Artificial Intelligence*, pp. 126–132, Portland, Ore, USA, 1996.

[11] K. Tumer and N. C. Oza, "Decimated input ensembles for improved generalization," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '99)*, pp. 3069–3074, July 1999.

[12] R. Bryll, R. Gutierrez-Osuna, and F. Quek, "Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets," *Pattern Recognition*, vol. 36, no. 6, pp. 1291–1302, 2003.

[13] L. Rokach, "Genetic algorithm-based feature set partitioning for classification problems," *Pattern Recognition*, vol. 41, no. 5, pp. 1676–1700, 2008.

[14] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[15] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[16] H. Ahn, H. Moon, M. J. Fazzari, N. Lim, J. J. Chen, and R. L. Kodell, "Classification by ensembles from random partitions of high-dimensional data," *Computational Statistics and Data Analysis*, vol. 51, no. 12, pp. 6166–6179, 2007.

[17] K. Ming Ting, J. R. Wells, S. Chuan Tan, S. Wei Teng, and G. I. Webb, "Feature-subspace aggregating: ensembles for stable and unstable learners," *Machine Learning*, vol. 82, no. 3, pp. 375–397, 2011.

[18] K. W. de Bock and D. V. D. Poel, "An empirical evaluation of rotation-based ensemble classifiers for customer churn prediction," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12293–12301, 2011.

[19] Y. Yang, Z. Ma, A. G. Hauptmann, and N. Sebe, "Feature selection for multimedia analysis by sharing information among multiple tasks," *IEEE Transactions on Multimedia*, vol. 15, no. 3, pp. 661–669, 2013.

[20] I. A. Gheyas and L. S. Smith, "Feature subset selection in large dimensionality domains," *Pattern Recognition*, vol. 43, no. 1, pp. 5–13, 2010.

[21] L. Yu and H. Liu, "Feature selection for high-dimensional data: a fast correlation-based filter solution," in *Proceedings of the 12th International Conference on Machine Learning*, Washington, DC, USA, 2003.

[22] Y. Piao, M. Piao, K. Park, and K. H. Ryu, "An ensemble correlation-based gene selection algorithm for cancer classification with gene expression data," *Bioinformatics*, vol. 28, no. 24, pp. 3306–3315, 2012.

[23] M. F. Akay, "Support vector machines combined with feature selection for breast cancer diagnosis," *Expert Systems with Applications*, vol. 36, no. 2, pp. 3240–3247, 2009.

[24] S. J. Hanson and Y. O. Halchenko, "Brain reading using full brain support vector machines for object recognition: there is no 'face' identification area," *Neural Computation*, vol. 20, no. 2, pp. 486–503, 2008.

[25] W. Zaghloul, S. M. Lee, and S. Trimi, "Text classification: neural networks vs support vector machines," *Industrial Management & Data Systems*, vol. 109, no. 5, pp. 708–717, 2009.

[26] L. Zhang, B. Liao, D. Li, and W. Zhu, "A novel representation for apoptosis protein subcellular localization prediction using support vector machine," *Journal of Theoretical Biology*, vol. 259, no. 2, pp. 361–365, 2009.

[27] A. Martinez and R. Benavente, "The AR face database," CVC Technical Report 24, 1998.

[28] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proceedings of the 2nd IEEE Workshop on Applications of Computer Vision*, pp. 138–142, December 1994.

[29] T. Sim, S. Baker, and M. Bsat, "The cmu pose, illumination, andexpression (pie) database of human faces," Tech. Rep. CMU-RI-TR-01-02, Robotics Institute, Carnegie Mellon University, 2001.

[30] D. Hond and L. Spacek, "Distinctive descriptions for face processing," in *Proceedings of the 8th British Machine Vision Conference (BMVC '97)*, pp. 320–329, Essex, UK, 1997.

[31] S. A. Armstrong, J. E. Staunton, L. B. Silverman et al., "MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia," *Nature Genetics*, vol. 30, no. 1, pp. 41–47, 2002.

[32] D. Singh, P. G. Febbo, K. Ross et al., "Gene expression correlates of clinical prostate cancer behavior," *Cancer Cell*, vol. 1, no. 2, pp. 203–209, 2002.

[33] H.-H. Hsu, C.-W. Hsieh, and M.-D. Lu, "Hybrid feature selection by combining filters and wrappers," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8144–8150, 2011.

[34] M. Piao, H. S. Shon, J. Y. Lee, and K. H. Ryu, "Subspace projection method based clustering analysis in load profiling," *IEEE Transactions on Power Systems*, vol. 29, no. 6, pp. 2628–2635, 2014.

[35] M. E. A. Bashir, D. G. Lee, M. Li et al., "Trigger learning and ECG parameter customization for remote cardiac clinical care information system," *IEEE Transactions on Information Technology in Biomedicine*, vol. 16, no. 4, pp. 561–571, 2012.

[36] Y. Lee, Y. J. Jung, K. W. Nam, S. Nittel, K. Beard, and K. H. Ryu, "Geosensor data representation using layered slope grids," *Sensors*, vol. 12, no. 12, pp. 17074–17093, 2012.