An Effective Ensemble-based Classification Algorithm for High-Dimensional Steganalysis

Fengying He
College of Mathematics and Computer Science Fuzhou University, Fuzhou, China
Email: hfy@fzu.edu.cn

Shangping Zhong and Kaizhi Chen
College of Mathematics and Computer Science Fuzhou University, Fuzhou, China
Email: {spzhong, ckz}@fzu.edu.cn

Abstract—Recently, ensemble learning algorithms are proposed to address the challenges of high dimensional classification for steganalysis caused by the curse of dimensionality and obtain superior performance. In this paper, we extend the state-of-the-art steganalysis tool developed by Kodovsky and Fridrich: the Kodovsky’s ensemble classifier and propose a novel method, called CS-RS for high-dimensional steganalysis. Different from the Kodovsky’s ensemble classifier which selects features in a completely random way, the proposed CS-RS modifies the generation method of feature subspaces. Firstly, our method employs the chi-square statistic (CS) to measure the weight of each feature in the original feature space and sorts features according to weights. Then the sorted original feature space is partitioned into two parts according to a given dividing point: high correlation part and low correlation part. Finally, the feature subset is formed by selecting features randomly in each part according to the given sampling rate. Experiments with the steganographic algorithms HUGO demonstrate that the proposed CS-RS using the FLD classifier offers training complexity comparable to the Kodovsky’s classifier and significantly increases the performance of the Kodovsky’s classifier in less than 1000-dimensional feature subspaces, gaining 1.2% on the optimal result. In addition, the proposed algorithm outperforms Bagging and AdaBoost and can offer accuracy comparable to L-SVM.

Index Terms—high-dimensional feature; steganalysis; ensemble classifier; chi-square statistic; fld

I. INTRODUCTION

With the increased sophistication of steganographic algorithms, steganalysis has already begun using feature spaces of increased dimensionality [1]. The most accurate spatial domain steganalysis of embedding (LSB matching) uses the 686-dimensional SPAM features [2] while a 1,234-dimensional Cross-Domain Feature (CDF) set was employed in [3] to attack YASS [4]. Moreover, the recent results of the steganalysis competition BOSS [5] indicate that there is little hope that a human-designed low-dimensional feature space effective against HUGO [6] exists and constructed a 24993-dimensional HOLMES feature proved especially effective against HUGO.

Classifying high dimensional features poses a serious challenge to steganalysis due to several reasons. The required number of labeled samples for supervised learning methods increases exponentially with the dimensionality to achieve high generalization accuracy [7]. Increase in the dimensionality makes machine learning methods computationally intractable [8]. The data points become sparse in the higher dimensions, which makes the learning task very difficult if not impossible.

Even though the support vector machine (SVM) seems to be the most popular machine learning tool used in steganalysis, SVMs are quite restrictive due to the complexity of SVM will be increased rapidly with the dimensionality of feature space growing. Traditional approach that tackles classification problems with a high-dimensional feature space employs dimensionality reduction or projection techniques like Principal Component Analysis (PCA) or KL-transformation. These project the original attribute or feature space into lower dimensions by creating new dimensions that are combinations of the original features. While such techniques help in reducing dimensionality, one drawback is that the new dimensions can be difficult to interpret, thus making it hard to relate the instances to the original dimensions [9]. A second drawback for the dimension reduction techniques like PCA is the huge computational complexity when calculate the covariance matrix of feature vectors with thousands of dimensions [10].

To address the challenges associated with the curse of dimensionality arising in steganalysis, Kodovsky [11] proposed ensemble classifier built as random forest with the FLD [12] as a base learner. In Kodovsky’s ensemble classifier, each base learner is trained on randomly selected subspaces of the original feature space, the dimensionality of the random subspaces can be chosen to be much smaller than the full dimensionality, which significantly decreases the training complexity and each learner is trained on a bootstrap sample drawn from the training set rather than on the whole training set, which further increases the mutual diversity of the base learners. Experiment results demonstrated the ensemble classifier in [11] can provides performances equivalent to that of a
Support Vector Machine (SVM) \(^{[13]}\) for steganalysis of large databases with large feature vectors.

But the features in subspaces used to train the base learners in \([11]\) were selected randomly from the original feature space, this approach is not suitable for high dimensional feature space consisting of thousands of features, because the random sampling may result in the selected subspace contains many features which are uninformative to classification, thus affecting the classification performance of the base learners.

In this paper, we propose a new method, called CS-RS which modifies the generation method of feature subspaces in \([11]\) and therefore enhances classification performance over high-dimensional feature space. Firstly the weight of each feature in original feature space is computed with respect to the correlation between the feature and the class label using the chi-square statistic, the weights are treated as the classification ability of the features, the higher the weight, the more informative the feature to the classification. Then the features in original feature space are sorted from high to low according to weights and the sorted original feature space is partitioned into two parts according to a given dividing point. Finally, the feature subset is formed by selecting features randomly in each part according to the given sampling rate. Such a method increases the probability for informative features to be included in each subspace, thus increases the performance of the base learners and without sacrificing diversity between them due to the random selection of features in each part. Experiments with steganographic algorithm HUGO show that, the proposed algorithm is an effective method for solving the problems caused by high-dimensional feature vectors, the detection accuracy is much better than the Kodovsky’s classifier in low dimensional feature space without increasing training complexity obviously. The proposed algorithm gets the best result of 85.97% which increases the optimal value of the Kodovsky’s classifier of 1.2%. In addition, the proposed algorithm outperforms Bagging and AdaBoost and can offer accuracy comparable and often even better to L-SVM.

The rest of this paper is organized as follows. In Section II, we recall the important concepts of the Kodovsky’s classifier. In Section III, we describe the CS-RS algorithm in detail. In Section IV, we implement experiments and present an empirical analysis of the proposed algorithm. Finally, we give the conclusion and some suggestions for future work in the last Section.

II. THE KODOVSKY’S ENSEMBLE CLASSIFIER

The ensemble classifier proposed in \([11]\) consists of \(L\) independently trained base learners, designed to keep low complexity and overall simplicity. To facilitate the description, we use the same notation as in \([11]\). Denote \(d\) as the dimensionality of the high dimensional feature space, \(N^{trn}\) and \(N^{sub}\) the number of training and testing samples from each class, \(x_m \in \mathbb{R}^d, m=1,\ldots, N^{trn}\) as the feature vector of dimension \(d\) computed from the training set, \(y_k \in \mathbb{R}^d, k=1,\ldots, N^{sub}\) as the feature vector of dimension \(d\) computed from the testing set and \(L\) the number of base learners. The set of all training and testing samples will be denote \(X^{trn} = \{x_m\}_{m=1}^{N^{trn}}\) and \(Y^{sub} = \{y_k\}_{k=1}^{N^{sub}}\).

For \(D \subseteq \{1,\ldots,d\}\), \(x^{(D)}\) is a \(|D|\)-dimensional feature vector consisting only of those features from \(x\) whose indices are in \(D\), preserving their original order.

The Kodovsky’s classifier is to learn separately individual base learners. Each base learner, denoted \(B_0\) with \(l \in \{1,\ldots,L\}\) is a mapping \(R^{d_{sub}} \rightarrow \{0,1\}\), ‘0’ standing for cover and ‘1’ for stego, trained on \(X_l^{(D)} = \{x_m^{(D)}\}_{m=1}^{N^{sub}}\), where \(D_l \subseteq \{1,\ldots,d\}\|D_l| = d_{sub} < d\) is the \(l\)th random subspace and \(N^{sub} = |N^{sub}|\) is the \(l\)th bootstrap sample of the set of indices \(\{1,\ldots, N^{sub}\}\). For all test examples \(y \in Y^{sub}\), the final prediction \(B(y)\) is obtained by a majority vote:

\[
B(y) = \begin{cases} 
1 & \text{if } \frac{1}{L} \sum_{j=1}^{L} B_j(y) > \frac{L}{2} \\
0 & \text{otherwise}
\end{cases}
\]

The Kodovsky’s ensemble classifier is of low computational complexity and achieves performance as good as or even better than the SVM. But the features in \(D_l\) are selected randomly from the whole high-dimensional feature space, which may result in \(D_l\) contains many features uninformative to classification, thus affecting the classification performance of the base learners. We should improve it to achieve higher performance.

III. THE CS-RS ALGORITHM

Theory and experiments show that \([14-17]\), increasing the performance of the base learners and the diversity between them helps to improve the generalization performance of the ensemble classifier. Therefore, how to improve the performance of base classifiers, while enhancing the diversity between them is the main goal of this study. The main idea of the proposed CS-RS is: 1) firstly, the chi-square statistic is adopted to compute the weight of each feature in original feature space. The weights are treated as the classification ability of the features, the higher the weight, the more informative the feature to the classification; 2) the features in original feature space are sorted from high to low according to weights and the sorted original feature space is partitioned into two parts according to a given dividing point: high correlation part and low correlation part. Then each part is assigned a sampling rate, the feature subset is formed by selecting features randomly in each part according to the given sampling rate, by this way, we can guarantee good performance of base learners, while ensuring diversity; 3) train base classifiers on feature subsets using bootstrap sample and combine their individual decisions to form the final class predictor.
A. Computation of Feature Weight using Chi-Square Statistic

The chi-square test [18] is a kind of hypothesis test method used widely. It is used to analysis the correlation of two variables. The basic idea of the chi-square test is to calculate the difference between the expected frequency and the actual frequency, denoted as the chi-square statistic. The larger the value of the chi-square statistic, the stronger the correlation of the two variables is.

Let Y be the class with 2 distinct class labels \( y_j \) (j=1,2). For the purposes of our discussion we consider the \( i \)-th dimensional feature \( F \) with \( P \) distinct values in training set \( D \) of dimension \( d \), \( l \in \{1,...,d\} \). We denote the distinct features by \( f_i \), for \( i=1,...,p \). Considering all combinations of the distinct features of \( F \) and the labels of \( Y \), we can obtain a contingency table[19] of \( F \) against \( Y \) as shown in Table I. In Table I, the symbol \( \text{val}_{ij} \) stands for the number of the subset of \( D \) satisfying the condition that \( F=f_j \) and \( Y=y_j \).

<table>
<thead>
<tr>
<th>( F=f_1 )</th>
<th>( \text{val}_{11} )</th>
<th>( \text{val}_{12} )</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F=f_2 )</td>
<td>( \text{val}_{21} )</td>
<td>( \text{val}_{22} )</td>
<td>( \text{val} )</td>
</tr>
<tr>
<td>( \text{Total} )</td>
<td>( \text{val}_{1} )</td>
<td>( \text{val}_{2} )</td>
<td>( \text{val} )</td>
</tr>
</tbody>
</table>

As shown in Table I, the far right column contains the marginal totals for feature \( F \):

\[
\text{val}_i = \sum_{j=1}^{2} \text{val}_{ij} \quad i = 1,..., p
\]  

(1)

The bottom row is the marginal totals for class \( Y \):

\[
\text{val}_j = \sum_{i=1}^{2} \text{val}_{ij} \quad j = 1,2
\]  

(2)

The grand total (the total number of samples) is in the bottom right corner:

\[
\text{val} = \sum_{i=1}^{2} \sum_{j=1}^{2} \text{val}_{ij}
\]  

(3)

The weight of the feature \( F \) is computed as:

\[
\text{corr}_{es}(F,Y) = \sum_{i=1}^{2} \sum_{j=1}^{2} \frac{(\text{val}_{ij} - t_{ij})^2}{t_{ij}}
\]  

(4)

Where \( \text{val}_{ij} \) is the actual frequency from the contingency table and \( t_{ij} \) is the expected frequency computed as:

\[
t_{ij} = \frac{\text{val}_i \times \text{val}_j}{\text{val}}
\]  

(5)

Supposed training set \( D \) has \( d \)-dimensional features \( F_1,F_2,...,F_d \), to ensure the sum of the feature weights is equal to 1, we compute the normalized weight of each feature as:

\[
w_i = \frac{\sqrt{\text{corr}_{es}(F_i,Y)}}{\sum_{i=1}^{d} \sqrt{\text{corr}_{es}(F_i,Y)}} \quad (1 \leq i \leq d)
\]  

(6)

The feature with larger \( w_i \) has a stronger power in predicting the classes of samples.

B. Improved Algorithm

We modify Kodovsky’s algorithm [11] using the new sampling selection method to form feature subspaces.

Denote \( d \) as the dimensionality of the high-dimensional feature space, \( d_{sub} \) for the dimensionality of the feature subspace on which each base learner operate, \( L \) the number of base learners and \( N_{trn}^w \) the number of training samples from each class. The improved algorithm is called as CS-RS algorithm, described as follows:

Algorithm 1: CS-RS Algorithm

1. Input:
   - \( F \): the feature space \{\( F_1,...,F_d \)\},
   - \( d_{sub} \): the size of subspaces,
   - \( L \): the number of base learners,
   - \( dp \): the dividing point,
   - \( p \): the sampling rate,
   - \( x \): the test sample.
2. For \( i=1 \) to \( d \) do
   Calculate the weight of the \( i \)-th dimensional feature \( w_i \) using formula (6).
3. End for
4. Sort the original feature space according to the weight of each feature.
5. Divide the sorted feature space \( F \) into two parts according to the dividing point \( dp \): \[ F_{\leq dp}, F_{> dp} \] and \[ F_{dp+1}, F_d \].
6. For \( i=1 \) to \( L \) do
   (1) Draw a bootstrap sample \( N_{b}^{i} ; |N_{b}^{i}| = N_{trn}^w \) by uniform sampling with replacement from the training set \{\( 1,...,N_{trn}^w \); \}
   (2) Select features randomly in each part according to the given sampling rate \( p \) to form subset \( D_{b}^{i} ; |D_{b}^{i}| = d_{sub} < d \); 
   (3) Train a base learner \( B_i \) on features \[ X_{b}^{i} = \{X_{m_{b}^{i}}^{(L)} \} \] \( m_{b}^{i} \) \; ;
   (4) Project \( x \) onto the feature subset \( X_{b}^{i} \) and obtain the testing sample subset \( x_{b}^{i} \);
   (5) Predict \( x_{b}^{i} \) using \( B_i \) and obtain the prediction \( B_{b}^{i}(x_{b}^{i}) \), \( B_{b}^{i}(x_{b}^{i}) \in \{0,1\} \).
7. End for
8. Obtain the final prediction of \( x \) by majority voting:

\[
B(x) = \begin{cases} 
1 & \text{when } \sum_{i=1}^{L} B_{b}^{i}(x_{b}^{i}) > \frac{L}{2} \\
0 & \text{when } \sum_{i=1}^{L} B_{b}^{i}(x_{b}^{i}) < \frac{L}{2} \\
\text{random} & \text{otherwise}
\end{cases}
\]
The above algorithm has the following characteristics:

(1) The diversity between base learners is obtained by training them on bootstrap samples and the randomization in the feature subspace generation.

(2) The classification accuracy of base classifiers is improved, because the probability for informative features to be included in each subspace is increases. It is suggested that the feature subset can contain more features with stronger classification ability to improve performance.

(3) The flexibility of forming the feature subspace can be effectively enhanced by setting different sampling rate in corresponding parts to adjust dynamically the distribution of features of different parts in the random subspace for each base learner.

IV. EXPERIMENT RESULTS AND ANALYSIS

A. Experimental Setting

We demonstrate the power of the proposed CS-RS algorithm by applying it to the recently proposed adaptive spatial-domain steganographic algorithm called HUGO. All experiments were carried out on the database BossBase v1.00 (http://www.agents.cz/boss/BOSSFinal/). This training database is made of 10000 512×512 greyscale cover images in the pgm format, and the same 10000 images embedding a message at 0.4 bpp with HUGO algorithm with default parameters. 9000 covers and the associated 9000 stegos are selected randomly as testing set and another 2000 images as a training set. On each image, we extract 12753-dimensional SRM feature set [20] for the experiments.

The program code are implemented using C/C++ language, the test platform is WIN7 operating system, Intel Xeon E5300 2.60GHz, 8GB Memory.

B. Analysis of Experimental Results

In order to show the advantages of the new generation method of feature subset, we compare our proposed method and the Kodovsky’s method. The dimensionality of the subspace can significantly influence the performance of the ensemble classifier. To evaluate the performance of our proposed method in a fair and reasonable way, we run both algorithms against different sizes of feature subspaces. The dimensionality of the subspace is 100, 150, 200, 250, 300, 350, 400, 450, 500, 600, 700, 800, 900, 1000, 2000, 5000, respectively. Due to the computational resource limitation, for a given subspace size we set the number of base learners $L=71$.

For our CS-RS algorithm, we set the dividing point $dp=5000$, the sampling rate $p=[0.5,0.5]$, $[0.6,0.4]$, $[0.7,0.3]$, $[0.8,0.2]$, respectively. To facilitate the description, our method is described as CS-RS $x_y$, Where $x$ represents the sampling rate in the first part and $y$ represents the sampling rate in the second part. For example, the CS-RS $6.4$ represents the sampling rate in the first part is $60\%$ and the sampling rate in the second part is $40\%$.

For each algorithm, there are totally 16 sets of experiments carried out in this paper. To make the results statistically reliable, each set of experiment has been repeated for 10 times independently to take the average value of detection accuracy, true positive rate and true negative rate as the final results, as shown in Table II. The best results are marked in bold. $T_p$ represents the true positive rate, $T_n$ represents the true negative rate and $T$ represents the detection accuracy. From the results shown in Table II, we observe that:

(1) In the feature subspaces with less than 1000 dimensionality, the proposed CS-RS algorithm obtains an average detection accuracy of 83.92% which increases the average detection accuracy of the Kodovsky’s ensemble classifier of 2.6%. Compared with the Kodovsky’s algorithm, the CS-RS algorithm can get the optimal value in lower dimensional feature subspace, the detection accuracy of the CS-RS is more than 85% with $d_{sub}=300$ while the Kodovsky’s algorithm gets the optimal value of 84.98% with $d_{sub}=500$. We also can see, the CS-RS algorithm gets the highest detection accuracy of 85.97% which increases the optimal value of the Kodovsky’s ensemble classifier of 1.2% and gets the average detection accuracy of all subspaces of 82.84% which increases the average detection accuracy of the Kodovsky’s ensemble classifier of all subspaces of 2.2%. But when $d_{sub}>1000$, with the increase of dimensionality, the advantages of our algorithm is weakened, even though the performance still gets a small improvement with an average detection accuracy=77.26%. When $d_{sub}=5000$, the performance of two algorithms is very similar. The reason for that is our algorithm greatly increases the possibility of informative features to be selected, which significantly improves the classification ability of base learners in low dimensionality feature subspace and the diversity between them is increased due to the randomization in the feature subspace generation, thus, the performance of our algorithm is greatly improved. Then with the increase of dimensionality, the classification ability of base learners is still improved, but the diversity of base learners is declined, therefore, the performance of our ensemble is weakened.

(2) For the proposed algorithm, increase appropriately the proportion of the features in the first part can extend the subspace’s classification ability to improve the fusion decision. But if select too much features with higher classification ability, the accuracy of ensemble classifier will decline. This further indicates that we should make a trade-off between diversity and accuracy of the base classifier. From Table II, it can be seen when $p$ is $\{0.7, 0.3\}$, the detection accuracy is the best on the whole, which provides the basis for feature selection in the practical application.

(3) As is already apparent from Table II, the detection accuracy of both algorithms first increases and then decreases with the increase of dimensionality of feature subset, which is mainly because of the following two reasons. First, the redundancy between features will increase with the growing dimension sizes of subsets, the individual base learners become more dependent and thus...
the ability of the ensemble classifier to form non-linear boundaries decreases. Second, the base classifiers start to suffer from overtraining as the subspace dimensionality increases while the training size remains the same.

In order to visualize the performance of the algorithms, the Receiver Operating Characteristic (ROC) curve is used when comparing both methods. Fig. 1 plotted the curves in different dimensions of the subspaces, which are 100, 500, 1000 and 5000 respectively. The great advantages of our proposed algorithm in detection accuracy can be obviously seen from Fig.1.

<table>
<thead>
<tr>
<th>$d_{sub}$</th>
<th>CS-RS_5.5</th>
<th>CS-RS_6.4</th>
<th>CS-RS_7.3</th>
<th>CS-RS_8.2</th>
<th>Ensemble[11]</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>80.54</td>
<td>81.46</td>
<td>81.00</td>
<td>81.31</td>
<td>80.69</td>
</tr>
<tr>
<td>150</td>
<td>84.45</td>
<td>80.05</td>
<td>82.25</td>
<td>82.02</td>
<td>80.88</td>
</tr>
<tr>
<td>200</td>
<td>85.12</td>
<td>80.76</td>
<td>82.94</td>
<td>87.69</td>
<td>79.93</td>
</tr>
<tr>
<td>250</td>
<td>83.97</td>
<td>82.85</td>
<td>83.41</td>
<td>86.78</td>
<td>81.46</td>
</tr>
<tr>
<td>300</td>
<td>86.73</td>
<td>84.05</td>
<td>85.39</td>
<td>86.45</td>
<td>84.71</td>
</tr>
<tr>
<td>350</td>
<td>85.93</td>
<td>85.05</td>
<td>85.49</td>
<td>86.91</td>
<td>84.77</td>
</tr>
<tr>
<td>400</td>
<td>84.46</td>
<td>86.12</td>
<td>85.29</td>
<td>87.15</td>
<td>83.55</td>
</tr>
<tr>
<td>450</td>
<td>83.61</td>
<td>80.71</td>
<td>82.16</td>
<td>82.33</td>
<td>81.57</td>
</tr>
<tr>
<td>500</td>
<td>84.22</td>
<td>86.96</td>
<td>85.59</td>
<td>88.14</td>
<td>83.18</td>
</tr>
<tr>
<td>550</td>
<td>82.51</td>
<td>87.11</td>
<td>84.81</td>
<td>86.28</td>
<td>83.98</td>
</tr>
<tr>
<td>600</td>
<td>85.57</td>
<td>85.09</td>
<td>85.33</td>
<td>86.52</td>
<td>84.60</td>
</tr>
<tr>
<td>650</td>
<td>84.25</td>
<td>81.91</td>
<td>83.08</td>
<td>83.34</td>
<td>83.18</td>
</tr>
<tr>
<td>700</td>
<td>85.57</td>
<td>85.09</td>
<td>85.33</td>
<td>86.52</td>
<td>84.60</td>
</tr>
<tr>
<td>750</td>
<td>84.25</td>
<td>81.91</td>
<td>83.08</td>
<td>83.34</td>
<td>83.18</td>
</tr>
<tr>
<td>800</td>
<td>85.57</td>
<td>85.09</td>
<td>85.33</td>
<td>86.52</td>
<td>84.60</td>
</tr>
<tr>
<td>850</td>
<td>84.25</td>
<td>81.91</td>
<td>83.08</td>
<td>83.34</td>
<td>83.18</td>
</tr>
<tr>
<td>900</td>
<td>85.57</td>
<td>85.09</td>
<td>85.33</td>
<td>86.52</td>
<td>84.60</td>
</tr>
<tr>
<td>950</td>
<td>84.25</td>
<td>81.91</td>
<td>83.08</td>
<td>83.34</td>
<td>83.18</td>
</tr>
<tr>
<td>1000</td>
<td>85.57</td>
<td>85.09</td>
<td>85.33</td>
<td>86.52</td>
<td>84.60</td>
</tr>
<tr>
<td>1050</td>
<td>84.25</td>
<td>81.91</td>
<td>83.08</td>
<td>83.34</td>
<td>83.18</td>
</tr>
<tr>
<td>1100</td>
<td>85.57</td>
<td>85.09</td>
<td>85.33</td>
<td>86.52</td>
<td>84.60</td>
</tr>
<tr>
<td>1150</td>
<td>84.25</td>
<td>81.91</td>
<td>83.08</td>
<td>83.34</td>
<td>83.18</td>
</tr>
<tr>
<td>1200</td>
<td>85.57</td>
<td>85.09</td>
<td>85.33</td>
<td>86.52</td>
<td>84.60</td>
</tr>
</tbody>
</table>

Figure 1. ROC curve of different algorithms in different dimensions of the subspace.
We also compare the training time of the CS-RS algorithm and the Kodovsky’s algorithm for different $d_{\text{sub}}$. The experiment is repeated 20 times, and we calculate the average value of the training time as the final result. The training time of the CS-RS algorithm is the average value of the CS-RS algorithm with different sampling rate. The comparison is reported in Table III. This experiment reveals that although our algorithm needs to compute the feature weights, there is little difference between the training times of both algorithms. The CS-RS algorithm can offer training complexity comparable to the Kodovsky’s classifier.

**TABLE III**

<table>
<thead>
<tr>
<th>$d_{\text{sub}}$</th>
<th>Ensemble [11]</th>
<th>CS-RS</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.380 min</td>
<td>0.457 min</td>
</tr>
<tr>
<td>150</td>
<td>0.397 min</td>
<td>0.532 min</td>
</tr>
<tr>
<td>200</td>
<td>0.428 min</td>
<td>0.587 min</td>
</tr>
<tr>
<td>250</td>
<td>0.465 min</td>
<td>0.617 min</td>
</tr>
<tr>
<td>300</td>
<td>0.490 min</td>
<td>0.648 min</td>
</tr>
<tr>
<td>350</td>
<td>0.515 min</td>
<td>0.670 min</td>
</tr>
<tr>
<td>400</td>
<td>0.545 min</td>
<td>0.695 min</td>
</tr>
<tr>
<td>450</td>
<td>0.588 min</td>
<td>0.732 min</td>
</tr>
<tr>
<td>500</td>
<td>0.622 min</td>
<td>0.753 min</td>
</tr>
<tr>
<td>600</td>
<td>0.648 min</td>
<td>0.802 min</td>
</tr>
<tr>
<td>700</td>
<td>0.682 min</td>
<td>0.855 min</td>
</tr>
<tr>
<td>800</td>
<td>0.723 min</td>
<td>0.873 min</td>
</tr>
<tr>
<td>900</td>
<td>0.802 min</td>
<td>0.930 min</td>
</tr>
<tr>
<td>1000</td>
<td>0.893 min</td>
<td>1.003 min</td>
</tr>
<tr>
<td>2000</td>
<td>1.730 min</td>
<td>1.942 min</td>
</tr>
<tr>
<td>5000</td>
<td>5.247 min</td>
<td>5.590 min</td>
</tr>
</tbody>
</table>

Next we investigated the impact of the number of base learners $L$ on our algorithm and compared the results with those of the Kodovsky’s algorithm. Fig.2 shows the detection accuracy on the testing set of our algorithm as a function of the number of fused base learners $L$, for 5000 as the dividing point and \{0.8,0.2\} as the sampling rate. Fig.3 shows the detection accuracy on the testing set of the Kodovsky’s algorithm as a function of the number of fused base learners $L$. The performance of the two algorithms for different subspace ($d_{\text{sub}}=200, 500$ and 1000, respectively) is shown in Fig.2 and Fig.3. Obviously, the classification accuracy of both algorithms quickly saturates with $L$. The classification accuracy is tending towards stability and does not change seriously when $L>71$. For the fastest performance, one should choose the smallest $L$ that gives satisfactory performance. We suggest the number of base learners $L=71$.

The features in first part (high correlation part) have stronger classification ability. The dividing point $dp$ adjust the proportion of the features in first part selected in random subspace. Obviously, increasing appropriately the proportion of the features in first part selected in the random subspace can improve the classification ability of the ensemble. But when the value of $dp$ increases to a certain extent, the accuracy of ensemble classifier will decline because of the weaken diversity. Fig.4 gives the curves of the detection accuracy with respect to the values of parameter $dp$ in the case of CS-RS with the number of base learners $L=71$, subspace dimensionality $d_{\text{sub}}=300$ and sampling rate $p=\{0.5,0.5\}, \{0.7,0.3\}$ respectively. It can be seen that the detection accuracy varies with different $dp$. The detection accuracy first increases and then tends to be stable, and finally decreases with the value of $dp$ increasing from small to large. When $dp=1000$, the performance of the CS-RS algorithm and the Kodovsky’s algorithm is vary similar. It can be seen that the highest detection accuracy always occurs when $dp\in\{3000,5000\}$. For gaining good steganalysis performance in various application scenarios, we suggest $dp=5000$ as the default value.
According to the theory of ensemble classifier \cite{14}, the individual base learners have to be sufficiently diverse in the sense that they should make different errors on unseen data. The proportion mechanism to select features at each part according the sampling rate is to increase the diversity between them. To verify the effectiveness of the sampling rate used in our algorithm, we compare our algorithm, the Kodovsky’s algorithm and the CS-RS (part1) algorithm which only selects features from the first part to form feature subspace for different \(d_{sub}\). The comparison is shown in Fig.5. From it we can see: (1) The performance of the CS-RS algorithm with different sampling rate is better than the CS-RS (part1) algorithm in all subspaces except \(d_{sub}=450\) and 800. (2) The CS-RS (part1) algorithm gets better performance than the Kodovsky’s algorithm in 10 out of 16 subspaces with different dimensionality and provides performances equivalent to the Kodovsky’s algorithm in 3 subspaces. In two subspaces(\(d_{sub}=900\) and 5000) the CS-RS (part1) algorithm gets the worst results while our CS-RS algorithm gets optimal results in all subspaces with different dimensionality. The results indicate using the sampling rate in our algorithm can improve the diversity between base learners.

To assess whether our algorithm can be competitive in terms of accuracy and cpu time with L-SVM and other popular ensemble algorithms, namely Bagging and AdaBoost when a high-dimensional feature space is used for steganalysis, the performance of our algorithm with \(d_{sub}=5000\), \(dp=5000\) and the number of base learners \(L=71\) is contrasted with Bagging, AdaBoost and L-SVM with 5000-dimensional feature set extracted from SRM in Table IV. We use FLD as base learner, the parameters of Bagging and AdaBoost are kept at their default values in WEKA, the training parameters of L-SVM are obtained using five-fold cross-validation to search over the grid of the cost parameter \(C \in \{10^\alpha\} , \alpha \in \{-4,...,3\}\), which is recommended in \cite{11}. The results in Table IV indicate, the proposed CS-RS outperforms Bagging and AdaBoost, increasing the performance of Bagging and AdaBoost more than 10% with much lower training complexity. The performance of our algorithm is similar to or even better than L-SVM, but the time required for training of our algorithm is a litter higher than L-SVM due to computing the weights of features.

![Figure 4. Detection accuracy varies with different dividing point dp](image)

![Figure 5. Comparison of the detection accuracy of different algorithms for different feature subspace](image)
new CS-RS algorithm that incorporates the new subspace to obtain more accurate and robust detectors. We presented a method on feature space with increasing higher dimensionality to learners, without sacrificing diversity between them. To ensure the randomization of features by the randomly generation method. In the proposed algorithm, the feature subset can contain more informative features, while can effectively defeat the HUGO steganographic methods rate. Experimental results show that the proposed scheme for creating an ensemble model from large data.

ACKNOWLEDGMENT
This study was supported by the Technology Innovation Platform Project of Fujian Province under Grant No.2009J1007, the Natural Science Foundation of Fujian Province under Grant No. 2010J01331. The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

REFERENCES


Fengying He She is currently the Lecturer in the College of Mathematics and Computer Science, Fuzhou University of China. She was born in1979. Her current research interests include machine learning, pattern recognition and multimedia security.

Shangping Zhong He received his PhD in Computer Science and Technology from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2005. He is currently a professor in the College of Mathematics and Computer Science, Fuzhou University, China. He was born in 1969. His current research interests include machine learning, pattern recognition and multimedia security.

Kaizhi Chen He is currently a Lecturer in the College of Mathematics and Computer Science, Fuzhou University, China. He was born in 1983.His current research interests include machine learning, pattern recognition and multimedia security.