Ensemble Classifiers for Steganalysis of Digital Media

Jan Kodovský, Jessica Fridrich, *Member, IEEE*, and Vojtěch Holub

***Abstract*—Today, the most accurate steganalysis methods for digital media are built as supervised classifiers on feature vectors extracted from the media. The tool of choice for the machine learning seems to be the support vector machine (SVM). In this paper, we propose an alternative and well-known machine learning tool – ensemble classifiers implemented as random forrests – and argue that they are ideally suited for steganalysis. Ensemble classifiers scale much more favorably w.r.t. the number of training examples and the feature dimensionality with performance comparable to the much more complex SVMs. The significantly lower training complexity opens up the possibility for the steganalyst to work with rich (high-dimensional) cover models and train on larger training sets – two key elements that appear necessary to reliably detect modern steganographic algorithms. Ensemble classification is portrayed here as a powerful developer tool that allows fast construction of steganography detectors with markedly improved detection accuracy across a wide range of embedding methods. The power of the proposed framework is demonstrated on three steganographic methods that hide messages in JPEG images.**

I. Introduction

The goal of steganalysis is to detect the presence of secretly hidden data in an object. Digital media files, such as images, video, and audio, are ideal cover objects for steganography as they typically consist of a large number of individual elements that can be slightly modified to embed a secret message. Moreover, such empirical covers are rather difficult to model accurately using statistical descriptors,[1] which substantially complicates detection of embedding changes. In particular, with the exception of a few pathological cases, the detection cannot be based on estimates of the underlying probability distributions of statistics extracted from cover and stego objects. Instead, detection is usually cast as a supervised classification problem implemented using machine learning.

Although there exists a large variety of various machine learning tools, support vector machines seem to be by far the most popular choice. This is due to the fact that SVMs are backed by a solid mathematical foundation cast within the statistical learning theory [51] and because they are resistant to overtraining and perform rather well even when the feature dimensionality is comparable or larger than the size of the training set. Moreover, robust and efficient open-source implementations are available for download and are easy to use [13], [10].

The complexity of SVM training, however, slows down the development cycle even for problems of a moderate size, as the complexity of calculating the Gram matrix representing the kernel is proportional to the square of the product of the feature dimensionality and the training set size. Moreover, the training itself is at least quadratic in the number of training samples. This imposes limits on the size of the problem one can handle in practice and forces the steganalyst to consciously design the features to fit within the complexity constraints defined by available computing resources. Ensemble classifiers give substantially more freedom to the analysts, who can now design the features virtually without constraints on feature dimensionality and the training set size to build detectors through a much faster development cycle.

Early feature-based steganalysis algorithms used only a few dozens of features, e.g., 72 higher-order moments of coefficients obtained by transforming an image using QMFs [14], 18 binary similarity metrics [3],

The work on this paper was supported by the Air Force Office of Scientific Research under the research grant FA9550-09-1-0147. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of AFOSR or the U.S. Government.

The authors are with the Department of Electrical and Computer Engineering, Binghamton University, NY, 13902, USA. Email: jan.kodovsky@binghamton.edu, fridrich@binghamton.edu, vholub1@binghamton.edu.

[1] In [5], arguments were made that empirical cover sources are fundamentally incognizable.

23 DCT features [17], and 27 higher-order moments of wavelet coefficients [21]. Increased sophistication of steganographic algorithms together with the desire to detect steganography more accurately prompted steganalysts to use feature vectors of increasingly higher dimension. The feature set designed for JPEG images described in [42] used 274 features and was later extended to twice its size [28] by Cartesian calibration, while 324- and 486-dimensional feature vectors were proposed in [48] and [11], respectively. The SPAM set for the second-order Markov model of pixel differences has a dimensionality of 686 [39]. Additionally, it proved beneficial to merge features computed from different domains to further increase the diversity. The 1234-dimensional Cross-Domain Feature (CDF) set [30] proved especially effective against YASS [50], [49], which makes embedding changes in a key-dependent domain. Because modern steganography [41], [15], [35] places embedding changes in those regions of images that are hard to model, increasingly more complex statistical descriptors of covers are required to capture a larger number of (weaker) dependencies among cover elements that might be disturbed by embedding [19], [18], [23], [29]. This historical overview clearly underlies a pressing need for scalable machine learning to facilitate further development of steganalysis.

To address the complexity issues arising in steganalysis today, in the next section we propose ensemble classifiers built as random forrests by fusing decisions of an ensemble of simple base learners that are inexpensive to train. By exploring several different possibilities for the base learners and fusion rules, we arrive at a rather simple, yet powerful design that appears to improve detection accuracy for all steganographic systems we analyzed so far. In Sections II-A–II-E, we discuss various implementation issues and describe the algorithms for determining the ensemble parameters. In the experimental Section III, we provide a tell-tale example of how an analyst might work with the new framework for JPEG-domain steganography. Comparison with SVMs in terms of complexity and performance appears in Section IV. Finally, the paper is concluded in Section V.

This paper is a journal version of our recent conference contribution [29]. The main difference is a complete description of the training process, including algorithms for determining the ensemble parameters, and a far more detailed comparison with SVMs. In our effort to provide an example of usage in practice, we introduce a compact general-purpose feature set for the DCT domain and use it to improve detection of nsF5 [20], Model-Based Steganography (MBS) [43], and YASS, three representatives of different embedding paradigms.

We use calligraphic font for sets and collections, while vectors or matrices are always in boldface. The symbol $\mathbb{N}_0$ is used for the set of positive integers, $\mathbf{I}$ is a unity matrix, and $\mathbf{X}^\mathsf{T}$ is the transpose of $\mathbf{X}$. The Iverson bracket $[P] = 1$ whenever the statement P is true and it is 0 otherwise.

II. ENSEMBLE CLASSIFICATION FOR STEGANALYSIS

When a new steganographic method is proposed, it is required that it not be detectable using known feature sets. Thus, as the first step in building a detector, the steganalyst needs to select a model for the cover source within which the steganography is to be detected. Another way of stating this is to say that the covers are represented in a lower-dimensional feature space before training a classifier. This is usually the hardest and most time-consuming part of building a detector and one that often requires a large number of experiments through which the analyst probes the steganographic method using various versions of features intuitively designed to detect the embedding changes. Thus, it is of utmost importance to be able to run through a large number of tests in a reasonable time. Moreover, one will likely desire to use features of high dimension, that is unless the steganographic method has a weakness and can be detected using a simple low-dimensional feature vector. Consequently, one will likely have to employ larger training sets to prevent overtraining and to build a more robust detector, which further increases the computational complexity. In an ideal world, the process of feature design should be fully automatized. Unfortunately, as the recent steganalysis competition BOSS showed [22], [19], [18], the current state of the art is not advanced enough to reach this goal and experience and insight still play an important role. The contribution of this paper can be viewed as a first step towards automatizing steganalysis. We provide a general framework together with a scalable machine learning tool that can substantially speed up the development cycle while allowing the steganalyst to work with very complex and potentially high-dimensional feature vectors as well as large training sets.

A. The ensemble

The proposed ensemble classifier consists of many base learners independently trained on a set of cover and stego images. Each base learner is a simple classifier built on a (uniformly) randomly selected subspace of the feature space. Given an example from

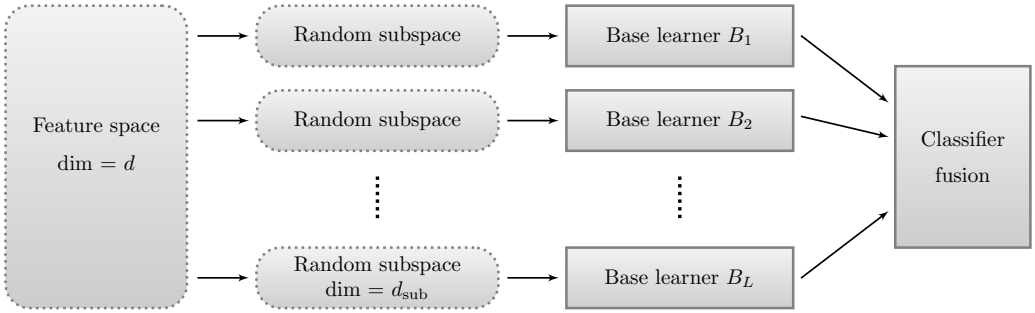


Figure 1. Diagram illustrating the proposed ensemble classifier. The random subspaces are constructed by selecting $d_{\rm sub} \ll d$ features randomly and uniformly from the entire feature space.

the testing set, the final decision is formed by aggregating the decisions of individual base learners. This supervised ensemble strategy will work only if the individual base learners are sufficiently diverse in the sense that they make different errors on unseen data. In order to further increase the mutual diversity of the base learners, each learner is trained on a bootstrap sample[2] drawn from the training set rather than on the whole training set. This strategy, known in the machine learning community as bootstrap aggregating or bagging [6], will also allow us to obtain an accurate estimate of the testing error, which will be important for determining optimal ensemble parameters. We note that the bootstrap samples are formed "by pairs," i.e., we make sure that the pairs of cover features and the corresponding stego features are preserved. This modification, which is specific for steganalysis, is rather important as it has been shown that breaking the cover-stego pairs into two sets, one of which is used for training and the other, testing, one for error estimation, may lead to a biased error estimate and, consequently, to a suboptimal performance [46], [27].

To formally describe our ensemble classifier, we introduce the following notation. The symbol d stands for the feature space dimensionality, $d_{\rm sub}$ for the dimensionality of the feature subspace on which each base learner operates, $N^{\rm trn}$ and $N^{\rm tst}$ are the number of training and testing examples from each class,[3] and L is the number of base learners. Furthermore, we reserve $\mathbf{x}_m, \bar{\mathbf{x}}_m \in \mathbb{R}^d$, $m = 1, \ldots, N^{\rm trn}$, for the cover and stego feature vectors computed from the training set and $\mathbf{y}_k, \bar{\mathbf{y}}_k \in \mathbb{R}^d$, $k = 1, \ldots, N^{\rm tst}$, for the features obtained from the testing cover and stego examples, respectively. The set of all training and testing samples will be denoted $\mathcal{X}^{\rm trn} = \{\mathbf{x}_m, \bar{\mathbf{x}}_m\}_{m=1}^{N^{\rm trn}}$ and $\mathcal{Y}^{\rm tst} = \{\mathbf{y}_k, \bar{\mathbf{y}}_k\}_{k=1}^{N^{\rm tst}}$. For $\mathcal{D} \subset \{1, \ldots, d\}$, $\mathbf{x}^{(\mathcal{D})}$ is a $|\mathcal{D}|$-dimensional feature vector consisting only of those features from $\mathbf{x}$ whose indices are in $\mathcal{D}$, preserving their original order.

The individual base learners B_l, $l = 1, \ldots, L$, are mappings $\mathbb{R}^d \to \{0, 1\}$, where 0 stands for cover and 1

Algorithm 1 Ensemble classifier.

1: **for** l=1 to L **do**

2: Form a random subspace

$$\mathcal{D}_l \subset \{1, \ldots, d\}, |\mathcal{D}_l| = d_{\rm sub} < d$$

3: Form a bootstrap sample $\mathcal{N}_l^{\rm b}, \left|\mathcal{N}_l^{\rm b}\right| = N^{\rm trn}$ by uniform sampling with replacement from the set $\{1, \ldots, N^{\rm trn}\}$

4: Train a base learner B_l on features

$$\mathcal{X}_l = \left\{\mathbf{x}_m^{(\mathcal{D}_l)}, \bar{\mathbf{x}}_m^{(\mathcal{D}_l)}\right\}_{m \in \mathcal{N}_l^{\rm b}}$$

$\Rightarrow$ obtain eigenvector $\mathbf{v}_l$ and threshold T_l

5: For all test examples $\mathbf{y} \in \mathcal{Y}^{\rm tst}$ make lth decisions:

$$B_l(\mathbf{y}^{(\mathcal{D}_l)}) \triangleq \begin{cases} 1 & \text{when } \mathbf{v}_l^{\mathsf{T}} \mathbf{y}^{(\mathcal{D}_l)} > T_l \\ 0 & \text{otherwise.} \end{cases}$$

6: **end for**

7: Form the final decisions $B(\mathbf{y})$ by majority voting:

$$B(\mathbf{y}) = \begin{cases} 1 & \text{when } \sum_{l=1}^{L} B_l(\mathbf{y}^{(\mathcal{D}_l)}) > L/2 \\ 0 & \text{when } \sum_{l=1}^{L} B_l(\mathbf{y}^{(\mathcal{D}_l)}) < L/2 \\ \text{random} & \text{otherwise.} \end{cases}$$

8: **return** $B(\mathbf{y}), \mathbf{y} \in \mathcal{Y}^{\rm tst}$

[2]Bootstrap sample is a uniform sample with replacement.

[3]It is always assumed that the training set contains the same number of cover and stego images because the stego images are obtained by embedding a random message in each cover image from some finite sample of images from a given source.

for stego. Note that, even though defined on $\mathbb{R}^d$, all base learners are trained on feature spaces of a dimension d_{sub} that can be chosen to be much smaller than the full dimensionality d, which significantly lowers the computational complexity. Even though the performance of individual base learners can be weak, the accuracy quickly improves after fusion and eventually levels out for a sufficiently large L. The decision threshold of each base learner is adjusted to minimize the total detection error under equal priors on the training set:

$$P_{\text{E}} = \min_{P_{\text{FA}}} \frac{1}{2}\left(P_{\text{FA}} + P_{\text{MD}}(P_{\text{FA}})\right), \tag{1}$$

where $P_{\text{FA}}, P_{\text{MD}}$ are the probabilities of false alarms and missed detection, respectively.

We recommend to implement each base learner as the Fisher Linear Discriminant (FLD) [12] because of its low training complexity; the most time consuming part is forming the within-class covariance matrices and inverting their summation. Additionally, such weak and unstable classifiers desirably increase diversity.

Since the FLD is a standard classification tool, we only describe those parts of it that are relevant for the ensemble classifier. The lth base learner is trained on the set $\left\{\mathbf{x}_i^{(\mathcal{D}_l)}, \bar{\mathbf{x}}_i^{(\mathcal{D}_l)} | i \in \mathcal{N}_l^{\text{b}}\right\}$, where $\mathcal{D}_l \subset \{1, \dots, d\}$, $|\mathcal{D}_l| = d_{\text{sub}}$ is randomly selected subset and $\mathcal{N}_l^{\text{b}}$ is a bootstrap sample drawn from the set $\{1, \dots, N^{\text{trn}}\}$, $|\mathcal{N}_l^{\text{b}}| = N^{\text{trn}}$. Each base learner is fully described using the generalized eigenvector

$$\mathbf{v}_l = (\mathbf{S}_{\text{W}} + \lambda \mathbf{I})^{-1}(\boldsymbol{\mu} - \bar{\boldsymbol{\mu}}), \tag{2}$$

where $\boldsymbol{\mu}, \bar{\boldsymbol{\mu}} \in \mathbb{R}^{d_{\text{sub}}}$ are the means of each class

$$\boldsymbol{\mu} = \frac{1}{N^{\text{trn}}} \sum_{m \in \mathcal{N}_l^{\text{b}}} \mathbf{x}_m^{(\mathcal{D}_l)}, \quad \bar{\boldsymbol{\mu}} = \frac{1}{N^{\text{trn}}} \sum_{m \in \mathcal{N}_l^{\text{b}}} \bar{\mathbf{x}}_m^{(\mathcal{D}_l)}, \tag{3}$$

$$\mathbf{S}_{\text{W}} = \sum_m (\mathbf{x}_m^{(\mathcal{D}_l)} - \boldsymbol{\mu})(\mathbf{x}_m^{(\mathcal{D}_l)} - \boldsymbol{\mu})^{\mathsf{T}} + (\bar{\mathbf{x}}_m^{(\mathcal{D}_l)} - \bar{\boldsymbol{\mu}})(\bar{\mathbf{x}}_m^{(\mathcal{D}_l)} - \bar{\boldsymbol{\mu}})^{\mathsf{T}} \tag{4}$$

is the within-class scatter matrix, and λ is a stabilizing parameter to make the matrix $\mathbf{S}_{\text{W}} + \lambda \mathbf{I}$ positive definite and thus avoid problems with numerical instability in practice when S_{W} is singular or ill-conditioned.[4]

For a test feature $\mathbf{y} \in \mathcal{Y}^{\text{tst}}$, the lth base learner reaches its decision by computing the projection $\mathbf{v}_l^{\mathsf{T}} \mathbf{y}^{(\mathcal{D}_l)}$ and comparing it to a threshold (previously adjusted to meet a desired performance criterion). After collecting all L decisions, the final classifier output is formed by combining them using an unweighted (majority) voting strategy – the sum of the individual votes is compared to the decision threshold $L/2$. We note that this threshold may be adjusted within the interval $[0, L]$ in order to control the importance of the two different types of errors or to obtain the whole receiver operating characteristic (ROC curve). In all experiments in this paper, we adjust the threshold to $L/2$ as P_{E} is nowadays considered standard for evaluating the accuracy of steganalyzers in practice.

The pseudo-code for the entire ensemble classifier is described in Algorithm 1, while Figure 1 shows its high-level conceptual diagram. The classifier depends on two parameters, d_{sub} and L, which are determined using algorithms from Section II-C.

B. Illustrative example

Before finishing the description of the ensemble classifier with procedures for automatic determination of d_{sub} and L, we include a simple illustrative example to demonstrate the effect of the parameters on performance. We do so for the steganographic algorithm nsF5 (no-shrinkage F5) [20] as a modern representative of steganographic schemes for the JPEG domain, using a simulation of its embedding impact if optimal wet-paper codes were used.[5]

The image source is the CAMERA database containing 6,500 JPEG images originally acquired in their RAW format taken by 22 digital cameras, resized so that the smaller size is 512 pixels with aspect ratio preserved, converted to grayscale, and finally compressed with JPEG quality factor 75 using Matlab's command `imwrite`. The images were randomly divided into two halves for training and testing, respectively.

All ensemble classifiers were built to detect stego images embedded with relative payload 0.1 bpac (bits per non-zero AC DCT coefficient). Figure 2 (left) shows the classifier error P_{E} (1) on the testing set as a function of the number of fused base learners L, for three different feature sets and a fixed d_{sub}. The feature sets $\mathcal{F}_{\text{inter}}$ and $\mathcal{F}_{\text{intra}}$ capture inter- and intra-block dependencies among DCT coefficients and $\mathcal{F}^*$ is their union. The features are defined in Section III-B; here, we use them to merely illustrate that the classification accuracy quickly saturates with L.

The error P_{E} (after saturation) is shown as a function of the subspace dimensionality d_{sub} in Figure 2 (right). Observe an initial quick drop followed by a fairly flat minimum, after which P_{E} starts growing again, which is mainly because of the following two reasons. First, the individual base learners become more dependent and thus the ability of the ensemble classifier to form

[4]The parameter λ can be either fixed to a small constant value (e.g., $\lambda = 10^{-10}$) or dynamically increased once numerical instability is detected.

[5]A simulator of nsF5 embedding is provided at http://dde.binghamton.edu/download/nsf5simulator/.

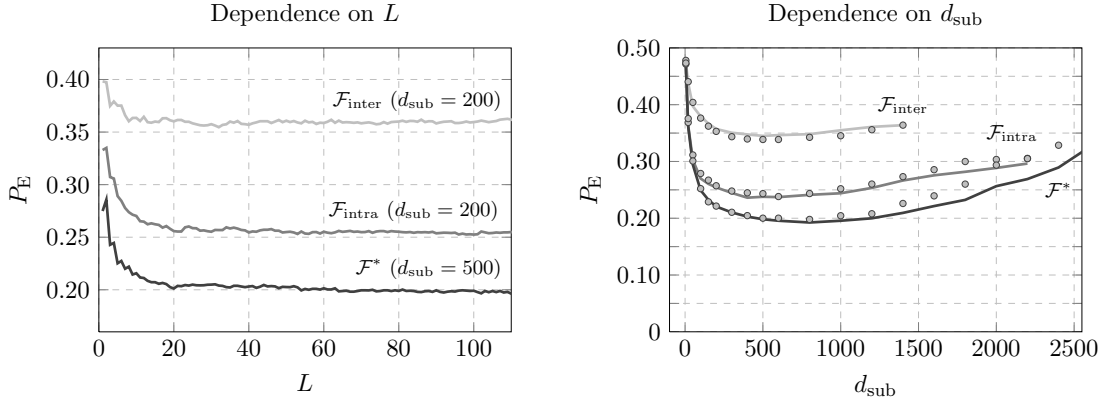


Figure 2. Left: Detection error P_E quickly saturates with the number of fused learners L. Right: The detection error after saturation as a function of d_{sub}. The dots represent out-of-bag error estimates E_{OOB} (see Section II-C). Feature sets considered: $\mathcal{F}_{\text{inter}}$, $\mathcal{F}_{\text{intra}}$, and $\mathcal{F}^*$ with dimensionalities 1550, 2375, and 3925 (see Section III). Target algorithm: nsF5 with payload 0.1 bpac.

non-linear boundaries decreases. Second, the individual FLDs start to suffer from overtraining as the subspace dimensionality increases while the training size remains the same.

C. Parameter determination

To complete the description of the classifier, we now supply a procedure for determining d_{sub} and L. Since each base learner B_l is trained on $\mathcal{X}_l = \{\mathbf{x}_m^{(\mathcal{D}_l)}, \bar{\mathbf{x}}_m^{(\mathcal{D}_l)}\}_{m \in \mathcal{N}_l^{\text{b}}}$, where $\mathcal{N}_l^{\text{b}}$ is a bootstrap sample of $\{1, \ldots, N^{\text{trn}}\}$ with roughly 63% of unique samples, the remaining 37% can be used for validation as B_l provides a single vote for each $\mathbf{x} \notin \mathcal{X}_l$. This procedure bears similarity to cross-validation in SVM training.

After n base learners are trained, each training sample $\mathbf{x} \in \mathcal{X}^{\text{trn}}$ will collect on average $0.37 \cdot n$ predictions that can be fused using the majority voting strategy into a prediction $B^{(n)}(\mathbf{x}) \in \{0, 1\}$. The following is an unbiased estimate of the testing error known as the "out-of-bag" (OOB) estimate:

$$E_{\text{OOB}}^{(n)} = \frac{1}{2N^{\text{trn}}} \sum_{m=1}^{N^{\text{trn}}} \left(B^{(n)}(\mathbf{x}_m) + 1 - B^{(n)}(\bar{\mathbf{x}}_m) \right). \tag{5}$$

The term comes from bagging (bootstrap aggregating) which is a well-established technique for reducing the variance of classifiers [6]. In contrast to bagging, we use a different random subset of features for training each base learner. Figure 2 (right) illustrates that $E_{\text{OOB}}^{(n)}$ is indeed an accurate estimate of the testing error.

1) Stopping criterion for L: As is already apparent from Figure 2 (left), the classification accuracy saturates rather quickly with L. The speed of saturation, however, depends on the accuracy of individual base learners, on the relationship between d and d_{sub}, and is also data dependent. Therefore, we determine L dynamically by observing the progress of the OOB estimate (5) and stop the training once the last K moving averages calculated from μ consecutive E_{OOB} values lie in an ϵ-tube:

$$L = \arg\min_n \left\{ n; \left| \min_{i \in P_K(n)} M_\mu(i) - \max_{i \in P_K(n)} M_\mu(i) \right| < \epsilon \right\}, \tag{6}$$

where

$$M_\mu(i) = \frac{1}{\mu} \sum_{j=i-\mu+1}^{i} E_{\text{OOB}}^{(j)} \tag{7}$$

and $P_K(n) = \{n - K, \ldots, n\}$. The parameters K, μ, and ϵ are user-defined and control the trade-off between computational complexity and optimality. In all our experiments in this paper, we fixed $K = 50$, $\mu = 5$, and $\epsilon = 0.005$. According to our experiments, this choice of the parameters works universally for different steganalysis tasks, i.e., for both small and large values of d and d_{sub}, for a wide range of payloads (low and high E_{OOB} values), and different steganographic algorithms.

Note that while $E_{\text{OOB}}^{(L)}$ is formed by fusing $0.37 \cdot L$ decisions (on average), the predictions on testing samples will be formed by fusing all L decisions. The final out-of-bag estimate meeting the criterion (6) will be denoted $E_{\text{OOB}} \equiv E_{\text{OOB}}^{(L)}$.

2) Subspace dimension d_{sub}: Since the classification accuracy is fairly insensitive to d_{sub} around its

minimum (Figure 2 (right)), most simple "educated guesses" of d_{sub} give already near-optimal performance, which is important for obtaining quick insights for the analyst. Having said this, we now supply a formal procedure for automatic determination of d_{sub}. Because $P_{\text{E}}(d_{\text{sub}})$ is unimodal in d_{sub}, the minimum can be found through a one-dimensional search over d_{sub} using $E_{\text{OOB}}(d_{\text{sub}})$ as an estimate of $P_{\text{E}}(d_{\text{sub}})$. Since the matrix inversion in (2) requires $O(d_{\text{sub}}^3)$ operations, to avoid evaluating $E_{\text{OOB}}(d_{\text{sub}})$ for large values of d_{sub}, we approach the minimum "from the left" using a simple direct-search derivation-free technique inspired by the compass search [31]. The pseudo-code, shown in Algorithm 2, can be interpreted as follows. Starting with a small value of d_{sub}, we keep increasing it by a pre-defined step Δ_d as long as $E_{\text{OOB}}(d_{\text{sub}})$ decreases. Once the error passes its minimum and starts increasing again, we go back to the lowest point and the step size is refined, $\Delta_d \leftarrow \Delta_d/2$, until the solution is found within the desired tolerance τ (Stage 2).

The parameters τ, Δ_d, and ϵ_d control the trade-off between the training time and optimality of the solution. In particular, τ specifies the desired relative tolerance within which the lowest value of E_{OOB} is to be found, Δ_d determines the initial step size, and ϵ_d specifies the robustness w.r.t. statistical fluctuations of $E_{\text{OOB}}(d_{\text{sub}})$ – it identifies the moment when to stop increasing d_{sub} and proceed to Stage 2.

Similarly as in Section II-C1, the choice of the parameters seems to be rather universal – in all experiments conducted in this paper we used $\tau = 0.02$, $\Delta_d = 200$, and $\epsilon_d = 0.005$ as a good compromise between the training time and the classification accuracy.

Algorithm 2 One-dimensional search for d_{sub}. To simplify the boundary issues, we define $E_{\text{OOB}}(d_{\text{sub}}) = 1$ for all $d_{\text{sub}} \notin [0, d]$.

Set parameters τ, Δ_d, and ϵ_d
//Stage 1: first pass with Δ_d
Initialize $k \leftarrow 0$, $E^*_{\text{OOB}} \leftarrow 1$, $d^*_{\text{sub}} \leftarrow 0$
repeat
  $k \leftarrow k + 1$
  Train ensemble classifier and obtain out-of-bag error estimate $E_{\text{OOB}}(k\Delta_d)$
  if $E_{\text{OOB}}(k\Delta_d) < E^*_{\text{OOB}}$ **then**
    $E^*_{\text{OOB}} \leftarrow E_{\text{OOB}}(k\Delta_d)$
    $d^*_{\text{sub}} \leftarrow k\Delta_d$
  end if
until $E_{\text{OOB}}(k\Delta_d) > E^*_{\text{OOB}} + \epsilon_d$
//Stage 2: localize the minimum by refining Δ_d
repeat
  Obtain $E_1 \equiv E_{\text{OOB}}(d^*_{\text{sub}} - \Delta_d)$
  Obtain $E_2 \equiv E_{\text{OOB}}(d^*_{\text{sub}})$
  Obtain $E_3 \equiv E_{\text{OOB}}(d^*_{\text{sub}} + \Delta_d)$
  if $1 \geq \frac{2E_2}{E_1+E_3} > 1 - \tau$ or Δ_d too small **then**
    return d^*_{sub}
  else
    $E^*_{\text{OOB}} \leftarrow \min\{E_1, E_2, E_3\}$
    $d^*_{\text{sub}} \leftarrow d_{\text{sub}}$ yielding E^*_{OOB}
    $\Delta_d \leftarrow \Delta_d/2$
  end if
until 1

D. Relationship to prior art

Boosting [45] is a general method of creating an accurate predictor by combining many weaker learners through a properly chosen aggregation strategy. Since the first successful ensemble systems were proposed, boosting has evolved into a well developed discipline whose popularity keeps on growing due to the simplicity of the approach, its straightforward parallel implementation, and high accuracy.

One of the earliest boosting frameworks is AdaBoost proposed by Freund and Schapire [16]. AdaBoost trains individual weak learners (base learners) sequentially and every base learner focuses on those samples that were more difficult to classify by previous base learners. This is achieved by a continuous adjustment of the training sample weights throughout the learning – the weights of training samples that were misclassified are increased while the weights of those that were classified correctly are decreased. The final decision is formed as a weighted combination of individual predictions with weights corresponding to the standalone accuracy of each base learner. AdaBoost is a deterministic meta-algorithm applicable to any classifier (base learner) capable of handling weighted training samples.

A different way of boosting the performance through an ensemble of weaker learners is bagging (or bootstrap aggregating) due Breiman [6], a concept already mentioned in Section II-A as a part of our proposed steganalysis framework. In bagging, every base learner is trained on a different bootstrap sample drawn from the original training set and their individual predictions are then combined through a simple majority voting scheme (averaging). The success of bagging relies on the *instability* of base learners w.r.t. small changes in

the training set. An important by-product of bagging is the ability to continuously monitor the testing error estimate (OOB).

The random forest [7] is an extended version of bagging in the sense that it also trains individual base learners on bootstrap samples of the training set. The base learners are, however, additionally randomized by making them dependent on a random vector that is drawn independently and from one fixed distribution. In [7], each base learner is a decision tree whose splitting variables are chosen randomly as a small subset of the original variables. The final prediction is again formed as a majority vote. This additional randomization introduces instability (and thus diversity) to the individual base learners and substantially speeds-up the training. On the other hand, the accuracy of individual base learners decreases, which is to be expected. However, it turns out that the combined prediction generally yields comparable or even better results than bagging or AdaBoost. We would like to stress that unlike in AdaBoost, the random forest treats individual base learners equally in forming the final decision – this is because all the base learners were generated using the same random procedure.

The steganalysis system proposed in Section II-A could be categorized as a random forest with the FLD as a base learner. The random component is in the feature subspace generation and is a crucial part of the system as using the full feature space would be computationally intractable due to high feature dimensionality.

The idea of forming random subspaces from the original feature space is not new and is known under different names. Decision forests [24], attribute bagging [9], CERP (Classification by Ensembles from Random Partitions) [1], or the recently proposed RSE (Random Subsample Ensemble) [47] are all ensemble-based classifiers sampling the feature space prior base learner training to either increase the diversity among classifiers or reduce the original high dimension to manageable values.

Most ensemble systems described in the literature use base learners implemented as classification trees even though other classifiers may be used. For example, SVMs are used as base learners in [33], while in [2] a set of different base learners are compared, including logistic regression, L-SVM, and FLD. Our decision to select the FLD was based on numerous experiments we performed and will be discussed in more detail in the next section. Briefly, FLDs are very fast and provided overall good performance when combined together into a final vote.

Besides ensemble classification, there exist numerous other well-developed strategies for reducing the training complexity. One popular choice are dimensionality reduction techniques that can be either unsupervised (PCA) or supervised (e.g., feature selection [34]) applied prior to classification as a part of the feature preprocessing. However, such methods are rarely suitable for applications in steganalysis when no small subset of features can deliver performance similar to the full-dimensional case. The dimensionality reduction and classification can also be performed simultaneously either by minimizing an appropriately constructed single objective function directly (SVDM [38]) or by constructing an iterative algorithm for dimensionality reduction with a classification feedback after every iteration. In machine learning, these methods are known as embedded and wrapper methods [34].

Finally, the idea of using a committee of detectors *for steganalysis* appeared in [25]. However, the focus of the work was elsewhere – several classifiers were trained individually to detect different steganographic methods and their fusion was shown to outperform a single classifier trained on a mixture of stego images created by those methods.

E. Discussion

To the best of our knowledge, a fully automatized framework combining random feature subspaces and bagging into a random forest classifier, together with an efficient utilization of out-of-bag error estimates for stopping criterion and the search for the optimal value of the subspace dimension is a novel contribution not only in the field of steganalysis, but also in the ensemble classification literature. The ensemble classifier as described in Section II provided the best overall performance and complexity among many different versions we have investigated. In particular, we studied whether it is possible to improve the performance by selecting the features randomly but non-uniformly and we tested other base learners and aggregation rules for the decision fusion. We also tried to replace bagging with cross-validation and to incorporate the ideas of AdaBoost [16] into the framework. Even though none of these modifications brought an improvement, we believe they deserve to be commented on and we discuss them in this section.

Depending on the feature set and the steganographic algorithm, certain features react more sensitively to embedding than others. Thus, it seemingly makes sense to try improve the performance by selecting the more influential features more frequently instead of uniformly at random. We tested biasing the random

selection to features with a higher individual Fisher ratio. However, any deviation from the uniform distribution lead to a drop in the performance of the entire ensemble. This is likely because biased selection of features decreases the diversity of the individual base learners. The problem of optimum trade-off between diversity and accuracy of the base learners is not completely resolved in the ensemble literature [36], [8] and we refrain from further analyzing this important issue in this paper.

Next, we investigated whether base learners other than FLDs can improve the performance. In particular, we tested linear SVMs (L-SVMs), kernelized FLDs [37], decision trees, naive Bayesian classifiers, and logistic regression. In summary, none of these choices proved a viable alternative to the FLD. Decision trees were unsuitable due to the fact that in steganalysis it is unlikely to find a small set of influential features (unless the steganography has some basic weakness). All features are rather weak and only provide detection when considered as a whole or in large subsets. Interestingly, the ensemble with kernelized FLD, L-SVM, and logistic regression had performance comparable to FLDs, even though the individual accuracies of base learners were higher. Additionally, the training complexity of these alternative base learners was much higher. Also, unlike FLD, both L-SVM and kernelized FLD require pre-scaling of features and a parameter search, which further increases the training time.

The last element we tested was the aggregation rule. The voting as described in Algorithm 1 could be replaced by more sophisticated rules [32]. For example, when the decision boundary is a hyperplane, one can compute the projections of the test feature vector on the normal vector of each base learner and threshold their sum over all base learners. Alternatively, one could take the sum of log-likelihoods of each projection after fitting models to the projections of cover and stego training features (the projections are well-modeled by a Guassian distribution). We observed, however, that all three fusion strategies gave essentially identical results. Thus, we selected the simplest rule – the majority voting as our final choice.

Apart from optimizing individual components of the system, we also tried two alternative designs of the framework as a whole. First, we replaced the bootstrapping and out-of-bag error estimation with k-fold cross-validation. This modification yielded similar results as the original system based on OOB estimates.

The second direction of our efforts was to incorporate the ideas of AdaBoost. There are several ways of doing so. We can boost individual FLDs using AdaBoost and use them as base learners for the ensemble framework. Alternatively, we could use the whole ensemble as described in Section II as a base learner for AdaBoost. Both options, however, dramatically increase the complexity of the system. Additionally, we would lose the convenience of estimating the testing error simply as OOB estimates.

Another option is to apply AdaBoost into the framework *directly* by adjusting the training sample weights as the training progresses and replacing the final majority voting with a weighted sum of the individual predictions. There are, however, two complications: every base learner is trained in a different feature space and on different training samples. An appealing (and simple) way of resolving these problems is to use the cross-validation variant of the ensemble mentioned above. Using k-fold cross-validation, the entire process could be viewed as training k parallel ensemble systems, each of them trained on a different (but fixed) training set consisting of $k-1$ folds. AdaBoost could then be used to boost each of these k sub-machines, while the testing error estimation (and thus the automatic ensemble parameter search procedure) could be carried out through the folds left out.

We implemented this modified system and subjected it to numerous comparative experiments under different steganalysis scenarios. However, no performance gain has been achieved. Therefore, we conclude that the implementation as a random forest built from equally weighted FLDs trained on different random subspaces is the overall best approach among those we tested.

More details about our experiments with k-fold cross-validation and AdaBoost appear in the technical report [26], where we summarize all the comparative results mentioned in the previous paragraphs.

III. EXAMPLE: STEGANALYSIS IN JPEG DOMAIN

We now demonstrate the power and advantages of the proposed approach on a specific tell-tale example of how a steganalyst might use ensemble classifiers to assemble a feature set in practice and build a classifier. While the detector built here markedly outperforms existing state of the art, we stress that the purpose of this illustrative exposition is not to optimize the feature set design w.r.t. a specific algorithm and cover source. We intend to study this important topic as part of our future work.

A. Co-occurrences

Capturing dependencies among pairs of individual DCT coefficients through co-occurrence matrices is a

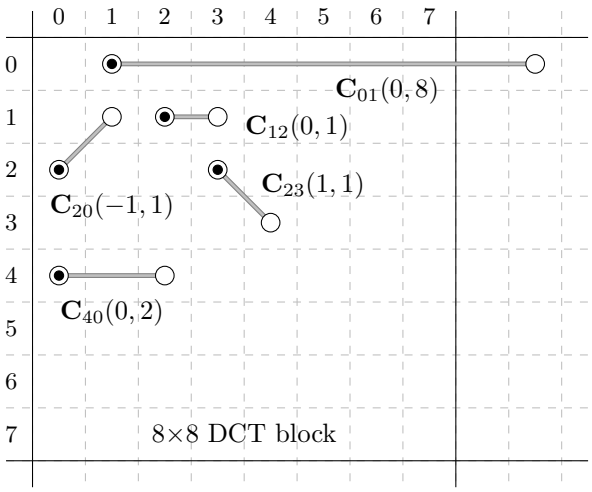


Figure 3. Graphical explanation of features $\mathbf{C}_{xy}(\Delta x, \Delta y)$. The symbols ◉— and —○ denote the first and the corresponding second DCT mode.

common practice in modern steganalysis [48], [11], [42]. However, the matrices are usually constructed from *all* coefficients in the DCT plane, which implies that coefficients from different DCT modes are, in spite of their different statistical nature, treated equally. Since ensemble classifiers are not limited by high dimensionality, we explore the possibility of improving detection by modeling the distribution of pairs of DCT coefficients on a mode-by-mode basis.

Let $\mathbf{D}_{xy}^{(i,j)}$ denote the (x,y)th DCT coefficient in the (i,j)th 8×8 block, $[x,y] \in \{0,\dots,7\} \times \{0,\dots,7\}$, $i = 1,\dots,8\lceil M/8 \rceil$, $j = 1,\dots,8\lceil N/8 \rceil$, where $M \times N$ are image dimensions. Our features will be formed as two-dimensional co-occurrence matrices $\mathbf{C}_{xy}(\Delta x, \Delta y)$ for coefficient pair $[x,y]$ and $[x+\Delta x, y+\Delta y]$. Formally, $\mathbf{C}_{xy}(\Delta x, \Delta y) = \{c_{kl}\}_{k,l=-T}^{T}$ is a $(2T+1)^2$-dimensional matrix with

$$c_{kl} = \frac{1}{Z} \sum_{i,j} \left[\left(\left\langle \mathbf{D}_{xy}^{(i,j)} \right\rangle_T = k \right) \wedge \left(\left\langle \mathbf{D}_{x+\Delta x, y+\Delta y}^{(i,j)} \right\rangle_T = l \right) \right], \tag{8}$$

where the normalization constant Z ensures that $\sum_{k,l} c_{kl} = 1$. The symbol $[\cdot]$ in (8) is the Iverson bracket, $\wedge$ stands for the logical "and" operator, and the truncation operator $\langle \cdot \rangle_T$ is defined as

$$\langle x \rangle_T = \begin{cases} x & \text{if } x \in [-T, T] \\ T \cdot \operatorname{sign}(x) & \text{otherwise.} \end{cases} \tag{9}$$

Note that in the definition of $\mathbf{C}_{xy}(\Delta x, \Delta y)$, we do not constrain Δx and Δy and allow $[x + \Delta x, y + \Delta y]$ to be out of the range $\{0,\dots,7\} \times \{0,\dots,7\}$ to more easily describe co-occurrences for inter-block coefficient pairs (e.g., $\mathbf{D}_{x+8,y}^{(i,j)} = \mathbf{D}_{xy}^{(i+1,j)}$). Figure 3 illustrates the notation on selected features, covering examples of both intra- and inter-block co-occurrences.

Assuming the statistics of natural images do not change after mirroring about the main diagonal, the symmetry of DCT basis functions w.r.t. the main block diagonal allows us to replace $\mathbf{C}_{xy}$ with the more robust

$$\tilde{\mathbf{C}}_{xy}(\Delta x, \Delta y) = \mathbf{C}_{xy}(\Delta x, \Delta y) + \mathbf{C}_{yx}(\Delta y, \Delta x). \tag{10}$$

Moreover, since for natural images $\{\tilde{c}_{kl}\}_{k,l=-T}^{T}$ will be sign-symmetrical, $\tilde{c}_{kl} \approx \tilde{c}_{-k,-l}$, we form

$$\bar{\mathbf{C}}_{xy}(\Delta x, \Delta y) = \{\bar{c}_{kl}\}_{k,l=-T}^{T}, \tag{11}$$

where $\bar{c}_{kl} = \tilde{c}_{kl} + \tilde{c}_{-k,-l}$. The redundant portion of $\bar{\mathbf{C}}_{xy}(\Delta x, \Delta y)$ can now be removed obtaining thus the final form of the co-occurrence, which we denote again $\bar{\mathbf{C}}_{xy}(\Delta x, \Delta y)$, with dimensionality $[(2T+1)^2 - 1]/2 + 1$.

B. Building the ensemble

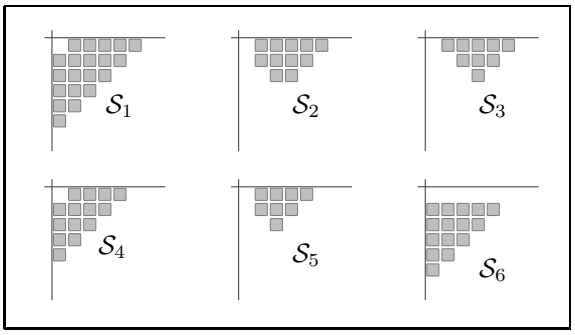
First, we assemble the feature space. By fixing $T = 3$ for all co-occurrences, each $\bar{\mathbf{C}}_{xy}(\Delta x, \Delta y)$ will have exactly 25 elements. Next, we form ten different sets of co-occurrence-based features to capture various dependencies among DCT coefficients while focusing on lower frequencies as they contain the vast majority of nonzero coefficients. The first six feature sets listed in Table I capture intra-block relationships among DCT coefficients: $\mathcal{F}_{\text{h}}$ - horizontally and vertically neighboring pairs; $\mathcal{F}_{\text{d}}$ - diagonally and semi-diagonally neighboring pairs; $\mathcal{F}_{\text{oh}}$ - "skip one" horizontally neighboring pairs; $\mathcal{F}_{\text{x}}$ - pairs symmetrically positioned w.r.t. the 8×8 block diagonal; $\mathcal{F}_{\text{od}}$ - "skip one" diagonally and semi-diagonally neighboring pairs; and $\mathcal{F}_{\text{m}}$ - "horse-move" positioned pairs. The remaining four sets capture inter-block relationships between coefficients from neighboring blocks: $\mathcal{F}_{\text{ih}}$ - horizontal neighbors in the same DCT mode; $\mathcal{F}_{\text{id}}$ - diagonal neighbors in the same DCT mode; $\mathcal{F}_{\text{is}}$ - semi-diagonal neighbors in the same DCT mode; $\mathcal{F}_{\text{ix}}$ - horizontal neighbors in DCT modes symmetrically positioned w.r.t. the 8×8 block diagonal. The union of all six intra-block feature sets is denoted $\mathcal{F}_{\text{intra}}$, and the union of the four inter-block feature sets is denoted $\mathcal{F}_{\text{inter}}$. Finally, we define $\mathcal{F}^* = \mathcal{F}_{\text{intra}} \cup \mathcal{F}_{\text{inter}}$.

The diagrams in Table I (right) specify the sets of DCT modes used for the construction of individual feature sets. The different shapes of these regions are due to different relative positions of DCT pairs in the corresponding features and due to symmetrization defined by (10).

Continuing with the construction of the ensemble classifier, we will assume that the analyst's goal is to attack the nsF5 algorithm for a fixed payload of

Table I
DESCRIPTION OF FEATURE SETS DEFINED IN THE TEXT. THE DIAGRAMS ON THE RIGHT DEFINE THE SETS OF DCT MODES, $\mathcal{S}_i$, $i = 1, \ldots, 6$, USED IN THE DEFINITIONS ON THE LEFT.

		Definition	Dim
$\mathcal{F}_{\rm h}$	=	$\{\bar{\mathbf{C}}_{xy}(0,1); [x,y] \in \mathcal{S}_1\}$	500
$\mathcal{F}_{\rm d}$	=	$\{\bar{\mathbf{C}}_{xy}(1,1); [x,y] \in \mathcal{S}_2\}$	500
		$\cup\{\bar{\mathbf{C}}_{xy}(1,-1); [x,y] \in \mathcal{S}_3\}$	
$\mathcal{F}_{\rm oh}$	=	$\{\bar{\mathbf{C}}_{xy}(0,2); [x,y] \in \mathcal{S}_4\}$	350
$\mathcal{F}_{\rm x}$	=	$\{\bar{\mathbf{C}}_{xy}(y-x, x-y); [x,y] \in \mathcal{S}_3\}$	225
$\mathcal{F}_{\rm od}$	=	$\{\bar{\mathbf{C}}_{xy}(2,2); [x,y] \in \mathcal{S}_5\}$	425
		$\cup\{\bar{\mathbf{C}}_{xy}(2,-2); [x,y] \in \mathcal{S}_3\}$	
$\mathcal{F}_{\rm m}$	=	$\{\bar{\mathbf{C}}_{xy}(-1,2); [x,y] \in \mathcal{S}_6\}$	375
$\mathcal{F}_{\rm ih}$	=	$\{\bar{\mathbf{C}}_{xy}(0,8); [x,y] \in \mathcal{S}_1\}$	500
$\mathcal{F}_{\rm id}$	=	$\{\bar{\mathbf{C}}_{xy}(8,8); [x,y] \in \mathcal{S}_2\}$	275
$\mathcal{F}_{\rm is}$	=	$\{\bar{\mathbf{C}}_{xy}(-8,8); [x,y] \in \mathcal{S}_2\}$	275
$\mathcal{F}_{\rm ix}$	=	$\{\bar{\mathbf{C}}_{xy}(y-x, x-y+8); [x,y] \in \mathcal{S}_1\}$	500



0.10 bpac.[6] The CAMERA database of cover and stego images is randomly split into a training and testing set of equal sizes. The analyst first evaluates the performance of each feature set listed in Table I individually and then chooses to merge those sets that give good detection to build the final ensemble classifier. This process is carried out on images from the training set only. The goodness of each feature set is evaluated by training the ensemble classifier and computing its OOB error estimate $E_{\rm OOB}$ as explained in Section II-C.

The evolution of the error after merging the individual feature sets is schematically shown in Figure 4. Interestingly, with every new added collection of co-occurrence-matrix-based features, the value of $E_{\rm OOB}$ decreases. Even though the inter-block features $\mathcal{F}_{\rm inter}$ by themselves perform poorly in comparison with $\mathcal{F}_{\rm intra}$, after merging them to form $\mathcal{F}^* = \mathcal{F}_{\rm inter} \cup \mathcal{F}_{\rm intra}$, the error further decreases to 0.1974. The performance of $\mathcal{F}^*$ is finally improved by an additional 2.5% using Cartesian calibration [28]. We denote the final $7,850$-dimensional Cartesian-calibrated feature set as $\mathcal{CF}^*$.

C. Testing phase

Once the features are optimized in terms of the lowest $E_{\rm OOB}$ (obtained from the training set), we proceed to the actual testing phase – we create class predictions for all test images and compare the predicted labels with the ground truth. The obtained testing error is $P_{\rm E} = 0.1702$ and is consistent with the OOB estimate $E_{\rm OOB} = 0.1728$. The detection results for other payloads are shown in Figure 5.

Since the feature construction was essentially reduced to adding more features capturing different dependencies among DCT coefficients, it is natural to ask whether the feature set $\mathcal{CF}^*$ is effective for other steganographic methods. To this end, we steganalyzed two additional algorithms: YASS [50] with five different settings (3, 8, 10, 11, and 12)[7] as reported in [30] and MBS [43] with payloads from 0.01 to 0.05 bpac. We chose these algorithms in order to cover three very different embedding paradigms – nsF5 minimizes the embedding impact, YASS masks embedding changes by additional JPEG compression, and MBS represents a model-preserving embedding paradigm.

Figure 5 shows the steganalysis results in terms of the median (MED) error values and median absolute deviations (MAD) over ten independent splits of the CAMERA database into training and testing sets. The detection improved markedly for all three algorithms when compared with the results obtained using a Gaussian SVM (G-SVM) with CDF or CC-PEV features.

Note that the additional randomization in the proposed framework due to random subspaces and bootstrapping does not increase the statistical variations in the test error much as the obtained results of MAD values are comparable to those obtained using the G-SVM. In terms of the worst case analysis of the proposed framework with $\mathcal{CF}^*$ features, we measured that the worst value of $P_{\rm E}$ is, on average over all three algorithms and all the considered payloads, only by 0.39% higher then the median value.

IV. COMPARISON WITH SVMS

The ensemble classifier is proposed here as an alternative tool to SVMs for feature development in steganalysis and for building steganalyzers. In this section, we compare it with both Gaussian and linear SVMs in terms of training complexity and performance.

[6] As in experiments in Figure 2, the stego images were obtained using the nsF5 simulator http://dde.binghamton.edu/download/nsf5simulator/

[7] According to [30], these were the five most secure settings that incorporated the improvements introduced in [44].

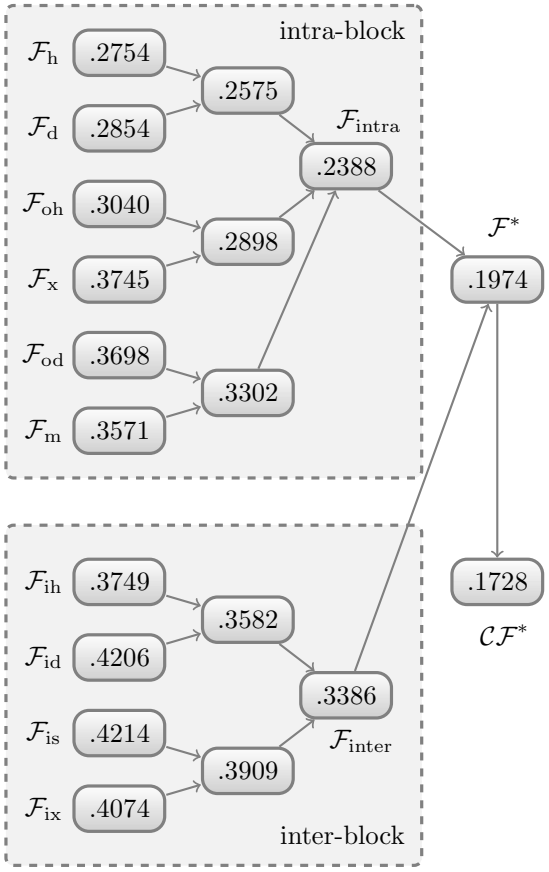


Figure 4. Evolution of E_{OOB} as individual feature sets are merged together.

A. Complexity

As the feature-space dimensionality and the number of training samples increase, the complexity and memory requirements of SVMs increase quite rapidly. The complexity of training scales at least quadratically with N^{trn} and grows as the cost parameter[8] C increases or as the classes become less distinguishable. Performing a proper k-fold cross-validation over the pre-defined grid of the penalty parameter $C \in \mathcal{G}_C$ requires repeating the training $k \cdot |\mathcal{G}_C|$ times. Furthermore, in case of the non-linear G-SVM, the grid of hyper-parameters is two-dimensional as we also need to search for the optimal value of the kernel width $\gamma \in \mathcal{G}_\gamma$, which makes the training even more expensive. Additionally, a G-SVM needs the kernel matrix of size $(N^{\mathrm{trn}})^2$ to be stored in the memory, which becomes prohibitive even for moderately large training sets and requires a more advanced caching solution.

In contrast, training the ensemble classifier requires much more moderate computer resources. To compute the scatter matrix $\mathbf{S}_{\mathrm{W}}$ (4) for one base learner, $O(N^{\mathrm{trn}} d_{\mathrm{sub}}^2)$ operations are needed while the matrix inversion in (2) can be carried out in $O(d_{\mathrm{sub}}^3)$ operations. Thus, the total training complexity for a fixed value of d_{sub} is $O(LN^{\mathrm{trn}} d_{\mathrm{sub}}^2) + O(Ld_{\mathrm{sub}}^3)$, which is linear w.r.t. N^{trn} and does not directly depend on d. As the search for the optimal value of d_{sub} (described in Section II-C2) is performed essentially by increasing the value of d_{sub} until the optimal value is found, the complexity is dominated by the largest value of the random subspace dimensionality tried during the search, which is always close to the optimal d_{sub}. We also note that, unlike SVM, FLD is scale-invariant and the features do not need to be normalized.

With respect to the memory requirements, each base learner needs access to only $2N^{\mathrm{trn}} d_{\mathrm{sub}}$ features at a time. Therefore, the classifier could be implemented so that one never needs to load all d-dimensional features into the memory during training, which is a favorable property especially for very large d. The ensemble classifier is represented using the set of L generalized eigenvectors (2) and the corresponding thresholds, which requires the total storage space of $O(Ld_{\mathrm{sub}})$.

B. Experimental comparisons

In the first experiment, we attack the steganographic algorithm nsF5 using the 548-dimensional CC-PEV features. We use this fairly low-dimensional feature set[9] so that we can easily train the following three classifiers without running into complexity issues:

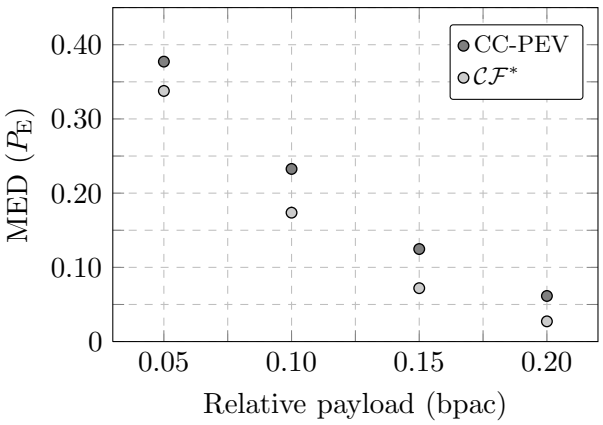
- Gaussian SVM implemented using the publicly available package LIBSVM [10]. The training includes a five-fold cross-validation search for the optimal hyper-parameters – the cost parameter C and the kernel width γ. It was carried out on the multiplicative grid $\mathcal{G}_C \times \mathcal{G}_\gamma$, $\mathcal{G}_C = \{10^a\}$, $a \in \{0, \dots, 4\}$, $\mathcal{G}_\gamma = \left\{\frac{1}{d} \cdot 2^b\right\}$, $b \in \{-4, \dots, 3\}$.
- Linear SVM implemented using the package LIBLINEAR [13]. The five-fold cross-validation is used to search over the grid of the cost parameter $C \in \{10^a\}$, $a \in \{-4, \dots, 3\}$.
- Ensemble classifier implemented in Matlab as described in Section II-A, including the search for the optimal value of d_{sub} and the automatic determination of L. Our implementation is available for download at http://dde.binghamton.edu/download/ensemble/.

[8]The parameter C is a cost for misclassification in the objective function of SVM training.

[9]Note the shift in the notion of what constitutes a *low*-dimensional feature set.

Steganalysis of nsF5

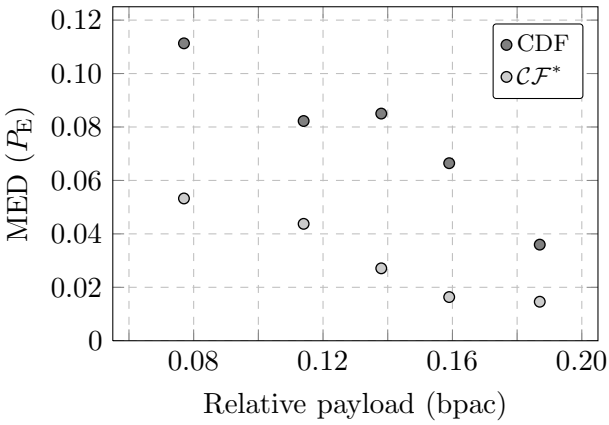
payload	CC-PEV		$\mathcal{CF}^*$	
(bpac)	MED	MAD	MED	MAD
0.05	0.3772	0.00246	0.3377	0.00345
0.10	0.2326	0.00231	0.1737	0.00130
0.15	0.1247	0.00385	0.0720	0.00165
0.20	0.0615	0.00200	0.0273	0.00140



Steganalysis of YASS

payload	CDF		$\mathcal{CF}^*$	
(bpac)$^+$	MED	MAD	MED	MAD
0.077 (12)	0.1113	0.00363	0.0532	0.00165
0.114 (11)	0.0822	0.00149	0.0437	0.00095
0.138 (8)	0.0850	0.00366	0.0271	0.00100
0.159 (10)	0.0664	0.00078	0.0164	0.00125
0.187 (3)	0.0360	0.00217	0.0146	0.00045

$^+$the number in parentheses denotes the YASS setting



Steganalysis of MBS

payload	CC-PEV		$\mathcal{CF}^*$	
(bpac)	MED	MAD	MED	MAD
0.01	0.3924	0.00292	0.3710	0.00180
0.02	0.2875	0.00208	0.2560	0.00110
0.03	0.2072	0.00169	0.1684	0.00140
0.04	0.1365	0.00215	0.1087	0.00185
0.05	0.0924	0.00208	0.0684	0.00160

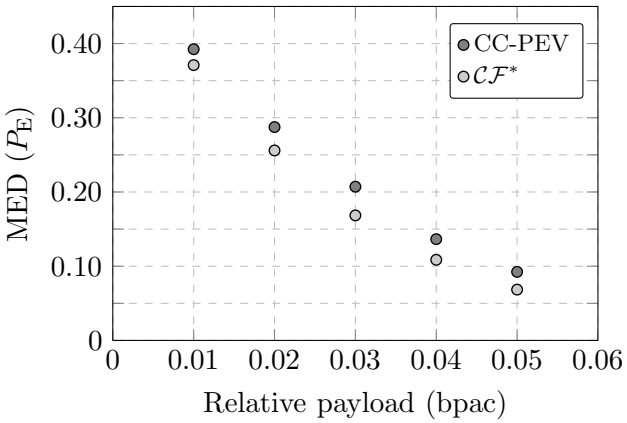


Figure 5. Steganalysis of nsF5, YASS, and MBS. The performance of the ensemble classifier using $\mathcal{CF}^*$ features is compared to the state-of-the-art G-SVM steganalysis with CC-PEV (or CDF) features. We report median (MED) testing error over ten different splits of the CAMERA database into a training and testing set, as well as the median absolute deviation (MAD) values.

Splitting the CAMERA database in two halves – one for training and the other half for testing, a set of stego images was created for each relative payload $\alpha \in \{0.05, 0.1, 0.15, 0.2\}$ bpac. Table II shows the detection accuracy and the training time of all three classifiers. The time was measured on a computer with the AMD Opteron 275 processor running at 2.2 GHz.

The performance of all three classifiers is very similar, suggesting that the optimal decision boundary between cover and stego images in the CC-PEV feature space is linear or close to linear. Note that while the testing errors are comparable, the time required for training differs substantially. While the G-SVM took several hours to train, the training of L-SVMs was accomplished in 14–23 minutes. The ensemble classifier is clearly the fastest, taking approximately 2 minutes to train across all payloads.

With the increasing complexity and diversity of cover models, future steganalysis must inevitably start using larger training sets. Our second experiment demonstrates that the computational complexity of the ensemble classifier scales much more favorably w.r.t. the training set size N^{trn}.[10] To this end, we fixed the payload to 0.10 bpac and extended the CAMERA database by 10,000 images from the BOWS2 competition [4] and 9,074 BOSSbase images [40]. Both databases were JPEG compressed using the quality factor 75. The resulting collection of images allowed us to increase the training size to $N^{\text{trn}} = 25,000$.

Table III shows the training times for different N^{trn}. The values for the L-SVM and the ensemble classifier are averages over five independent random selections of training images. As the purpose of this experiment is rather illustrative and the G-SVM classifier is apparently computationally infeasible even for relatively low values of N^{trn}, we report its training times measured only for a single randomly selected training set and drop it from experiments with $N^{\text{trn}} > 5,000$ entirely. It is apparent that although the L-SVM training is computationally feasible even for the largest values of N^{trn}, the ensemble is still substantially faster and the difference quickly grows with the training set size.

In our last experiment, we compare the performance of the L-SVM with the ensemble classifier when a high-dimensional feature space is used for steganalysis. The G-SVM was not included in this test due to its high complexity. We consider the 7,850-dimensional feature vector $\mathcal{CF}^*$ constructed in Section III and use it to attack nsF5, YASS, and MBS. This experiment reveals how well both classifiers handle the scenario when the feature space dimensionality is larger than the number of training samples. The comparison is reported in Table IV. The ensemble classifier is substantially faster and delivers a higher accuracy for all three algorithms (the decision boundary seems to be farther from linear than in the case of CC-PEV features).

[10]The actual number of training samples is $2N^{\text{trn}}$ as we take both cover and stego features.

Table II
Steganalysis of nsF5 using CC-PEV features. The running times and P_{E} values are medians (MED) over ten independent splits of the CAMERA database into training and testing sets. We also report median absolute deviation (MAD) values for P_{E}.

bpac	Classifier	P_{E} MED	P_{E} MAD	Training time
0.05	G-SVM	0.3772	0.00246	7 hr 29 min
	L-SVM	0.3802	0.00269	23 min
	Ensemble	0.3695	0.00145	2 min
0.10	G-SVM	0.2326	0.00231	6 hr 52 min
	L-SVM	0.2421	0.00246	27 min
	Ensemble	0.2226	0.00265	3 min
0.15	G-SVM	0.1247	0.00385	5 hr 39 min
	L-SVM	0.1342	0.00185	26 min
	Ensemble	0.1160	0.00170	3 min
0.20	G-SVM	0.0615	0.00200	4 hr 51 min
	L-SVM	0.0638	0.00123	27 min
	Ensemble	0.0547	0.00150	3 min

Table III
Dependence of the training time on N^{trn}. Target algorithm: nsF5 with 0.10 bpac.

N^{trn}	G-SVM	L-SVM	Ensemble
1,000	33 min	5 min	< 1 min
2,000	2 hr 27 min	10 min	1 min
3,000	5 hr 24 min	14 min	1.5 min
4,000	9 hr 31 min	20 min	2 min
5,000	13 hr 47 min	27 min	2 min
10,000	×	54 min	4 min
15,000	×	1 hr 23 min	5 min
20,000	×	1 hr 52 min	6 min
25,000	×	2 hr 21 min	8 min

V. SUMMARY

The current trend in steganalysis is to train classifiers with increasingly more complex cover models and large data sets to obtain more accurate and robust detectors. The complexity of the tool-of-choice, the support vector machine, however, does not scale favorably w.r.t. feature dimensionality and the training set size. To facilitate further development of steganalysis, we propose an alternative – ensemble classifiers built by fusing decisions of weak and unstable base learners imple-

Table IV
STEGANALYSIS USING $\mathcal{CF}^*$ FEATURES. THE L-SVM CLASSIFIER IS COMPARED WITH THE PROPOSED ENSEMBLE CLASSIFIER.

	L-SVM		Ensemble	
	P_{E}	time	P_{E}	time
nsF5 0.05	0.3518	9 hr 27 min	0.3377	31 min
nsF5 0.10	0.1851	8 hr 04 min	0.1737	37 min
nsF5 0.15	0.0809	6 hr 36 min	0.0720	24 min
nsF5 0.20	0.0292	5 hr 37 min	0.0273	15 min
YASS 3	0.0222	5 hr 47 min	0.0146	48 min
YASS 8	0.0377	6 hr 12 min	0.0271	45 min
YASS 10	0.0241	5 hr 35 min	0.0164	59 min
YASS 11	0.0616	5 hr 46 min	0.0437	58 min
YASS 12	0.0711	6 hr 12 min	0.0532	1 min
MBS 0.01	0.3806	13 hr 39 min	0.3710	43 min
MBS 0.02	0.2654	14 hr 58 min	0.2560	1 hr 9 min
MBS 0.03	0.1820	13 hr 52 min	0.1684	57 min
MBS 0.04	0.1094	11 hr 23 min	0.1087	59 min
MBS 0.05	0.0715	10 hr 24 min	0.0684	49 min

mented as the Fisher Linear Discriminant. The training complexity of the ensemble scales much more favorably allowing the steganalyst to work with high-dimensional feature spaces and large training sets, removing thus the limitations imposed by the available computing resources that have often curbed the detector design in the past. The ensemble is especially useful for fast feature development when attacking a new scheme. Performance-wise, ensemble classifiers offer accuracy comparable and often even better to the much more complex SVMs at a fraction of the computational cost. We show a specific example how one can build a diverse high-dimensional feature space for analysis of JPEG images and readily use it to markedly improve the detection of nsF5, YASS, and MBS. A Matlab implementation of the ensemble is available at http://dde.binghamton.edu/download/ensemble/.

Our future work will be directed towards optimizing the feature design for specific cover sources and embedding algorithms to establish new benchmarks for steganalysis in JPEG as well as the spatial domain.

REFERENCES

[1] 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. *Comput. Stat. Data Anal.*, 51:6166–6179, August 2007.

[2] A. Assareh, M.H. Moradi, and L.G. Volkert. A hybrid random subspace classifier fusion approach for protein mass spectra classification. In *Proceedings of the 6th European conference on Evolutionary computation, machine learning and data mining in bioinformatics*, EvoBIO'08, pages 1–11, Berlin, Heidelberg, 2008. Springer-Verlag.

[3] İ. Avcıbaş, M. Kharrazi, N. D. Memon, and B. Sankur. Image steganalysis with binary similarity measures. *EURASIP Journal on Applied Signal Processing*, 17:2749–2757, 2005.

[4] P. Bas and T. Furon. BOWS-2. http://bows2.gipsa-lab.inpg.fr, July 2007.

[5] R. Böhme. *Improved Statistical Steganalysis Using Models of Heterogeneous Cover Signals*. PhD thesis, Faculty of Computer Science, Technische Universität Dresden, Germany, 2008.

[6] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, August 1996.

[7] L. Breiman. Random forests. *Machine Learning*, 45:5–32, October 2001.

[8] G. Brown. An information theoretic perspective on multiple classifier systems. In J. Benediktsson, J. Kittler, and F. Roli, editors, *Multiple Classifier Systems*, volume 5519 of *Lecture Notes in Computer Science*, pages 344–353. Springer Berlin, Heidelberg, 2009.

[9] R. Bryll, R. Gutierrez-Osuna, and F. Quek. Attribute bagging: Improving accuracy of classifier ensembles by using random feature subsets. *Pattern Recognition*, 36(6):1291–1302, 2003.

[10] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[11] C. Chen and Y.Q. Shi. JPEG image steganalysis utilizing both intrablock and interblock correlations. In *Circuits and Systems, ISCAS 2008. IEEE International Symposium on*, pages 3029–3032, May 2008.

[12] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. New York: John Wiley & Sons, Inc., 2nd edition, 2001.

[13] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. Software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear.

[14] H. Farid and L. Siwei. Detecting hidden messages using higher-order statistics and support vector machines. In F. A. P. Petitcolas, editor, *Information Hiding, 5th International Workshop*, volume 2578 of Lecture Notes in Computer Science, pages 340–354, Noordwijkerhout, The Netherlands, October 7–9, 2002. Springer-Verlag, New York.

[15] T. Filler and J. Fridrich. Gibbs construction in steganography. *IEEE Transactions on Information Forensics and Security*, 5(4):705–720, 2010.

[16] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, London, UK, 1995. Springer-Verlag.

[17] J. Fridrich. Feature-based steganalysis for JPEG images and its implications for future design of steganographic schemes. In J. Fridrich, editor, *Information Hiding, 6th International Workshop*, volume 3200 of Lecture Notes in Computer Science, pages 67–81, Toronto, Canada, May 23–25, 2004. Springer-Verlag, New York.

[18] J. Fridrich, J. Kodovský, M. Goljan, and V. Holub. Breaking HUGO – the process discovery. In T. Filler, T. Pevný, A. Ker, and S. Craver, editors, *Information Hiding, 13th International Workshop*, volume 6958 of Lecture Notes in Computer Science, Prague, Czech Republic, May 18–20, 2011.

[19] J. Fridrich, J. Kodovský, M. Goljan, and V. Holub. Steganalysis of content-adaptive steganography in spatial domain. In T. Filler, T. Pevný, A. Ker, and S. Craver, editors, *Information Hiding, 13th International Workshop*, volume 6958 of Lecture Notes in Computer Science, Prague, Czech Republic, May 18–20, 2011.

[20] J. Fridrich, T. Pevný, and J. Kodovský. Statistically undetectable JPEG steganography: Dead ends, challenges, and opportunities. In J. Dittmann and J. Fridrich, editors, *Proceedings of the 9th ACM Multimedia & Security Workshop*, pages 3–14, Dallas, TX, September 20–21, 2007.

[21] M. Goljan, J. Fridrich, and T. Holotyak. New blind steganalysis and its implications. In E. J. Delp and P. W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII*, volume 6072, pages 1–13, San Jose, CA, January 16–19, 2006.

[22] G. Gül, A.E. Dirik, and İ. Avcıbaş. Steganalytic features for JPEG compression-based perturbed quantization. *IEEE Signal Processing Letters*, 14(3):205–208, March 2000.

[23] G. Gül and F. Kurugollu. A new methodology in steganalysis : Breaking highly undetactable steganograpy (HUGO). In T. Filler, T. Pevný, A. Ker, and S. Craver, editors, *Information Hiding, 13th International Workshop*, volume 6958 of Lecture Notes in Computer Science, Prague, Czech Republic, May 18–20, 2011.

[24] T.K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:832–844, 1998.

[25] M. Kharrazi, H.T. Sencar, and N. Memon. Improving steganalysis by fusion techniques: A case study with image steganography. *LNCS Transactions on Data Hiding and Multimedia Security I*, 4300:123–137, 2006.

[26] J. Kodovský. Ensemble classification in steganalysis – cross-validation and AdaBoost. Technical report, Binghamton University, August 2011.

[27] J. Kodovský. On dangers of cross-validation in steganalysis. Technical report, Binghamton University, August 2011.

[28] J. Kodovský and J. Fridrich. Calibration revisited. In J. Dittmann, S. Craver, and J. Fridrich, editors, *Proceedings of the 11th ACM Multimedia & Security Workshop*, pages 63–74, Princeton, NJ, September 7–8, 2009.

[29] J. Kodovský and J. Fridrich. Steganalysis in high dimensions: Fusing classifiers built on random subspaces. In N. D. Memon, E. J. Delp, P. W. Wong, and J. Dittmann, editors, *Proceedings SPIE, Electronic Imaging, Security and Forensics of Multimedia XIII*, volume 7880, pages OL 1–13, San Francisco, CA, January 23–26, 2011.

[30] J. Kodovský, T. Pevný, and J. Fridrich. Modern steganalysis can detect YASS. In N. D. Memon, E. J. Delp, P. W. Wong, and J. Dittmann, editors, *Proceedings SPIE, Electronic Imaging, Security and Forensics of Multimedia XII*, volume 7541, pages 02–01–02–11, San Jose, CA, January 17–21, 2010.

[31] T.G. Kolda, R.M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45:385–482, 2003.

[32] L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.

[33] L.I. Kuncheva, J.J. Rodriguez Diez, C.O. Plumpton, D.E.J. Linden, and S.J. Johnston. Random subspace ensembles for fMRI classification. *IEEE Transactions on Medical Imaging*, 29:531–542, 2010.

[34] T.N. Lal, O. Chapelle, J. Weston, and A. Elisseeff. Embedded methods. In I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, editors, *Feature Extraction: Foundations and Applications, Studies in Fuzziness and Soft Computing*, pages 137–165. Physica-Verlag, Springer, 2006.

[35] W. Luo, F. Huang, and J. Huang. Edge adaptive image steganography based on LSB matching revisited. *IEEE Transactions on Information Forensics and Security*, 5(2):201–214, June 2010.

[36] J. Meynet and J.-P. Thiran. Information theoretic combination of classifiers with application to AdaBoost. In M. Haindl, J. Kittler, and F. Roli, editors, *Multiple Classifier Systems*, volume 4472 of *Lecture Notes in Computer Science*, pages 171–179. Springer Berlin / Heidelberg, 2007.

[37] S. Mika, G. Ratsch, J. Weston, B. Schölkopf, and K.R. Mullers. Fisher discriminant analysis with kernels. In *Proceedings of the 1999 IEEE Signal Processing Society Workshop Neural Networks for Signal Processing IX*, pages 41–48, Madison, WI, August 1999.

[38] F. Pereira and G. Gordon. The support vector decomposition machine. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 689–696, Pittsburgh, PA, 2006.

[39] T. Pevný, P. Bas, and J. Fridrich. Steganalysis by subtractive pixel adjacency matrix. *IEEE Transactions on Information Forensics and Security*, 5(2):215–224, June 2010.

[40] T. Pevný, T. Filler, and P. Bas. Break Our Steganographic System, http://boss.gipsa-lab.grenoble-inp.fr.

[41] T. Pevný, T. Filler, and P. Bas. Using high-dimensional image models to perform highly undetectable steganography. In R. Böhme and R. Safavi-Naini, editors, *Information Hiding, 12th International Workshop*, volume 6387 of Lecture Notes in Computer Science, pages 161–177, Calgary, Canada, June 28–30, 2010. Springer-Verlag, New York.

[42] T. Pevný and J. Fridrich. Merging Markov and DCT features for multi-class JPEG steganalysis. In E. J. Delp and P. W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX*, volume 6505, pages 3 1–3 14, San Jose, CA, January 29–February 1, 2007.

[43] P. Sallee. Model-based steganography. In T. Kalker, I. J. Cox, and Y. Man Ro, editors, *Digital Watermarking, 2nd International Workshop*, volume 2939 of Lecture Notes in Computer Science, pages 154–167, Seoul, Korea, October 20–22, 2003. Springer-Verlag, New York.

[44] A. Sarkar, K. Solanki, and B.S. Manjunath. Further study on YASS: Steganography based on randomized embedding to resist blind steganalysis. In E. J. Delp and P. W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, volume 6819, pages 16–31, San Jose, CA, January 27–31, 2008.

[45] R.E. Schapire. The boosting approach to machine learning: An overview. In D. D. Denison, M. H. Hansen, C. Holmes, B. Mallick, and B. Yu, editors, *Nonlinear Estimation and Classification*. Springer, 2003.

[46] V. Schwamberger and M.O. Franz. Simple algorithmic modifications for improving blind steganalysis performance. In J. Dittmann and S. Craver, editors, *Proceedings of the 12th ACM Multimedia & Security Workshop*, pages 225–230, Rome, Italy, September 9–10, 2010.

[47] G. Serpen and S. Pathical. Classification in high-dimensional feature spaces: Random subsample ensemble. In *Machine Learning and Applications, 2009. ICMLA '09. International Conference on*, pages 740 –745, December 2009.

[48] Y.Q. Shi, C. Chen, and W. Chen. A Markov process based approach to effective attacking JPEG steganography. In J. L. Camenisch, C. S. Collberg, N. F. Johnson, and P. Sallee, editors, *Information Hiding, 8th International Workshop*, volume 4437 of Lecture Notes in Computer Science, pages 249–264, Alexandria, VA, July 10–12, 2006. Springer-Verlag, New York.

[49] K. Solanki, N. Jacobsen, U. Madhow, B.S. Manjunath, and S. Chandrasekaran. Robust image-adaptive data hiding based on erasure and error correction. *IEEE Transactions on Image Processing*, 13(12):1627–1639, Dec 2004.

[50] K. Solanki, A. Sarkar, and B.S. Manjunath. YASS: Yet another steganographic scheme that resists blind steganalysis. In T. Furon, F. Cayre, G. Doërr, and P. Bas, editors, *Information Hiding, 9th International Workshop*, volume 4567 of Lecture Notes in Computer Science, pages 16–31, Saint Malo, France, June 11–13, 2007. Springer-Verlag, New York.

[51] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

Jan Kodovský is currently pursuing the Ph.D. degree at the Electrical and Computer Engineering Department of Binghamton University, New York. He received his M.S. degree in Mathematical Modeling from the Czech Technical University in Prague in 2006. His current research interests include steganalysis, steganography, and applied machine learning.

Vojtěch Holub is currently pursuing the Ph.D. degree at the department of Electrical and Computer Engineering at Binghamton University, New York. His main focus is on steganalysis and steganography. He received his M.S. degree in Software Engineering from the Czech Technical University in Prague in 2010.

Jessica Fridrich holds the position of Professor of Electrical and Computer Engineering at Binghamton University (SUNY). She has received her PhD in Systems Science from Binghamton University in 1995 and MS in Applied Mathematics from Czech Technical University in Prague in 1987. Her main interests are in steganography, steganalysis, digital watermarking, and digital image forensic. Dr. Fridrich's research work has been generously supported by the US Air Force and AFOSR. Since 1995, she received 19 research grants totaling over $7.5mil for projects on data embedding and steganalysis that lead to more than 110 papers and 7 US patents. Dr. Fridrich is a member of IEEE and ACM.