Steganalysis in high dimensions: Fusing classifiers built on random subspaces

Jan Kodovský and Jessica Fridrich

Department of Electrical and Computer Engineering
Binghamton University, State University of New York

ABSTRACT

By working with high-dimensional representations of covers, modern steganographic methods are capable of preserving a large number of complex dependencies among individual cover elements and thus avoid detection using current best steganalyzers. Inevitably, steganalysis needs to start using high-dimensional feature sets as well. This brings two key problems – construction of good high-dimensional features and machine learning that scales well with respect to dimensionality. Depending on the classifier, high dimensionality may lead to problems with the lack of training data, infeasibly high complexity of training, degradation of generalization abilities, lack of robustness to cover source, and saturation of performance below its potential. To address these problems collectively known as the curse of dimensionality, we propose ensemble classifiers as an alternative to the much more complex support vector machines. Based on the character of the media being analyzed, the steganalyst first puts together a high-dimensional set of diverse "prefeatures" selected to capture dependencies among individual cover elements. Then, a family of weak classifiers is built on random subspaces of the prefeature space. The final classifier is constructed by fusing the decisions of individual classifiers. The advantage of this approach is its universality, low complexity, simplicity, and improved performance when compared to classifiers trained on the entire prefeature set. Experiments with the steganographic algorithms nsF5 and HUGO demonstrate the usefulness of this approach over current state of the art.

1. MOTIVATION

Today, security of steganographic algorithms designed for empirical cover sources is evaluated using steganalyzers built as binary classifiers trained on cover and stego features. Originally, the features were designed by hand to capture the impact of known steganographic schemes.[12] A cleaner strategy is to obtain the features by adopting a model for individual cover elements and use its sampled form as the feature.[7, 22, 27] Close attention has usually been paid to keep the dimensionality of the feature space low due to potential problems with the Curse of Dimensionality (CoD). However, modern steganography methods, such as HUGO[23] are designed to approximately preserve a high-dimensional representation of covers* and thus many complex dependencies among pixels, as well.

With the increased sophistication of steganographic algorithms, steganalysis has already begun using feature spaces of increased dimensionality. The most accurate spatial domain steganalysis of ±1 embedding (LSB matching) uses the 686-dimensional SPAM features[22] while a 1,234-dimensional Cross-Domain Feature (CDF) set was was employed in[18] to attack YASS.[25] Moreover, the recent results of the steganalysis competition BOSS[11] indicate that there is little hope that a human-designed low-dimensional feature space effective against HUGO[23] exists.

In this paper, we address both issues – design of useful high-dimensional feature spaces and scalable machine learning approach. We form a high-dimensional feature space of "prefeatures" whose role is to capture as many statistical dependencies among individual cover elements as possible. This is achieved either by merging existing feature sets or by forming the prefeatures from joint statistics of groups of cover elements that exhibit the strongest relationship. The emphasis here is on diversity while one should not be as concerned with dimensionality. Having formed the prefeature space, the steganalyst builds a scalable ensemble classifier by fusing the decisions of a set

E-mail: {jan.kodovsky, fridrich}@binghamton.edu; J.F.: http://www.ws.binghamton.edu/fridrich

*HUGO works in a feature space of dimensionality 10^7.

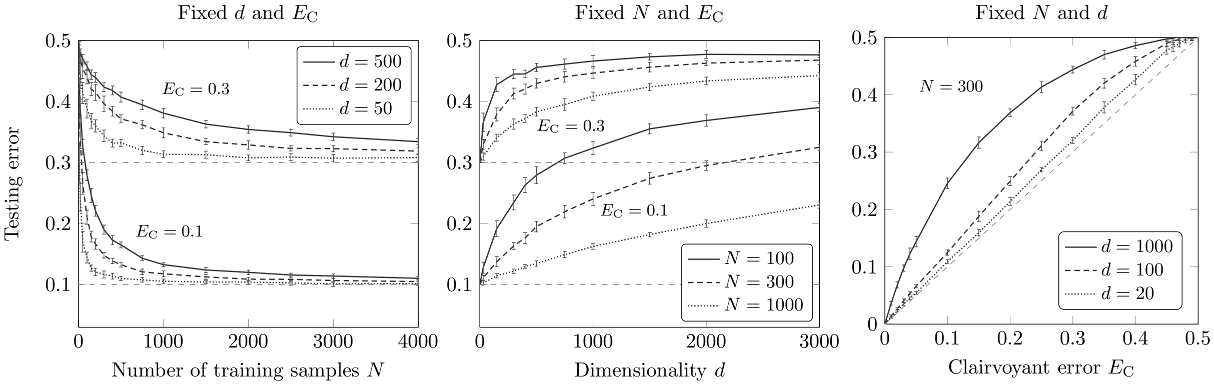


Figure 1. Effect of N,d, and P_{C} on the classification performance when fixing P_{C} (D_{C}) and d (left), N and P_{C} (middle), and N and d (right). All experiments were repeated 50 times and the median values are plotted, together with their MAD.

of simple base learners built on random subspaces of the prefeature space. This machine-learning strategy is introduced as a low-cost and scalable alternative to Support Vector Machines (SVMs) and it can achieve performance as good as or even better than the SVMs.

We explain our approach in six sections. In the next section, on a carefully designed artificial scenario we point out the complications that manifest when training a classifier in high dimensions. The ensemble classifier is described in Section 3. Several strategies for forming prefeatures in both JPEG and spatial domain are introduced in Section 4 and 5, where we include all experiments. The goal is to demonstrate the merit of using high-dimensional prefeatures in combination with ensemble classifiers by comparing to selected existing steganalyzers. Section 6 summarizes our contribution.

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. MAD stands for the Median Absolute Deviation.

2. CURSE OF DIMENSIONALITY

The purpose of this section is to shed more light on the difficulties one may encounter when classifying in high dimensions. We do so on an artificially created supervised classification problem.

Let us assume that we have N training examples of dimensionality d from two classes $\mathcal{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N/2)}\}$, $\mathcal{Y} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N/2)}\}$, $\mathbf{x}^{(m)}, \mathbf{y}^{(m)} \in \mathbb{R}^d$. Each example is formed by d i.i.d. realizations of a Gaussian random variable with mean 0 (for class $\mathcal{X}$) and $s > 0$ (for class $\mathcal{Y}$). Assuming both classes are equiprobable, for a given test sample $\mathbf{z} \in \mathbb{R}^d$, the optimal test statistic is the sample mean $\overline{\mathbf{z}}$ thresholded with $s/2$. We call this classifier *clairvoyant*. The total testing error of this classifier is $E_{\mathrm{C}} = 1 - \Phi(s\sqrt{d}/2)$ where $\Phi(x)$ is the c.d.f. of a standard normal variable. Furthermore, we define the class distinguishability as $D_{\mathrm{C}} = 1 - 2E_{\mathrm{C}}$.

The performance of detectors built using machine learning tools may be quite different from the clairvoyant classifier. In this section, we will study the performance of linear SVMs (L-SVMs) for different number of training examples, N, feature dimensionality, d, and class distinguishability, E_{C}. Since we know that the optimal separating boundary is linear, kernelized SVMs cannot give better results.

Figure 1 shows the results of experiments when two of the three parameters, N, d, and E_{C} (or D_{C}) are fixed while the remaining one varies. For fixed dimensionality d and class distinguishability D_{C}, with increasing number of training examples, N, the testing error[†] of the L-SVM approaches that of the clairvoyant detector (Figure 1 (left)). Furthermore, higher d or lower D_{C} require more samples for the L-SVM to perform well.

In Figure 1 (middle), we fix the number of the training samples N and the class distinguishability D_{C}, and increase the dimensionality d. Note that since we fixed D_{C}, the Gaussian shift s decreases as d grows, and can

[†] Our testing set consists of 4000 samples generated from the underlying cover/stego distributions.

be analytically expressed as $s = 2/\sqrt{d}\Phi^{-1}(1 - E_{\mathrm{C}})$. We show the testing errors obtained for two different values of $E_{\mathrm{C}} = 0.1$ and 0.3. We can clearly see the negative effect of the increased dimension.

In the last experiment, we switch the roles of E_{C} and d – we fix d and vary E_{C} by properly adjusting the shift s. Figure 1 (right) shows the testing errors for three different values of d and for a fixed number of training samples $N = 300$. The curse of dimensionality manifests the most for moderate distinguishability. For perfectly separable classes with $E_{\mathrm{C}} \approx 0$, L-SVM handles high dimensionality well. When $E_{\mathrm{C}} \approx 1$ (classes become indistinguishable), both the clairvoyant and L-SVM classifiers start making random guesses. As before, higher dimensionality leads to a larger difference between the L-SVM and the clairvoyant classifier.

We conclude that there are three main factors that can negatively influence the performance of machine learning tools: (1) small number of training samples, (2) low class distinguishability, (3) high dimensionality. Weak steganographic methods are easily detectable because they disturb some elementary cover properties that can be captured by a low-dimensional feature vector with high distinguishability. A fairly small training dataset is then usually sufficient to train a classifier with an excellent performance. On the other hand, more advanced steganographic methods (and these are of our interest) require high-dimensional feature spaces capable of capturing more complex dependencies among individual cover elements, which in turn necessitates more training samples.

A seemingly straightforward strategy to improve the performance of existing steganalyzers may be to increase the size of the training set. This way we allow the machine learning tool to better utilize the given feature space and we may use feature spaces of higher dimensions without degradation of performance. However, sooner or later one will likely encounter technical problems with data or memory management, or the training would be unacceptably long. Furthermore, in many practical scenarios, the steganalyst lacks information about the cover source (only a limited number of cover examples are available). Here, training the classifier on a different cover source may result in a serious drop in testing performance.[5, 15]

3. PROPOSED FRAMEWORK

This section contains the description of our proposed framework for building steganalyzers. Instead of handcrafting low-dimensional features with good class distinguishability and directly applying a machine learning tool, we divide the problem into two separate stages:

1. Form a set of *diverse* prefeatures that capture as many dependencies among individual cover elements as possible. The emphasis is on diversity while making sure all prefeatures are well populated when computed on a database of covers. At this stage, we do not attempt to limit the dimensionality in any way.
2. In the second, discriminative stage, we take into account the steganographic method under investigation. Our goal is maximizing the classification accuracy on the prefeature space using machine learning techniques that scale well with feature dimensionality.

The construction of prefeatures is discussed at the beginning of each experimental section. Here, we focus on high-dimensional classification and introduce our ensemble classifier.

3.1 High-dimensional classification – options

We start with the following rhetoric question: "What is the best way of utilizing the distinguishing power of high-dimensional prefeatures for steganalysis, given a limited amount of training samples?" One option is a direct application of a classification tool. It is known that SVMs are quite robust to the CoD, so provided it is computationally feasible, this should always be tried. And we may indeed obtain a satisfying performance, depending on the security of the steganographic method (and the secret message length). In this paper, we are more interested in the scenario where the steganographic method is difficult to detect and when the classification cannot be performed directly on the prefeature space due to computational issues. Furthermore, in Section 2 we saw that direct application of machine learning when dimensionality is much higher than the number of training examples may lead to poor performance.

There exist several well developed strategies that can be grouped into the following three broad categories:

1. **Reduce dimensionality and then classif**y. The high dimensionality of the prefeature space is first reduced using a dimensionality-reduction technique that can be either unsupervised (PCA) or supervised (e.g., feature selection[20]). The goal is to reduce the impact of the CoD on the subsequent classification problem. In a “traditional” approach to steganalysis, this reduction is usually achieved using human insight and heuristics.

2. **Reduce dimensionality and simultaneously classify**. Here, the dimensionality reduction and classification are combined into a single task. One can minimize an appropriately constructed single objective function directly (SVDM[21]) or, in general, construct an iterative algorithm for dimensionality reduction with a classification feedback after every iteration. In machine learning, these methods are known as embedded methods and wrapper methods.[20]

3. **Ensemble methods** follow a simple recipe – reduce dimensionality randomly, construct a classifier (base learner) on the reduced space, and set it aside. Repeating this procedure many times, each base learner is built on different subspace of the original space. The final decision is formed by aggregating the decisions of individual classifiers.[19] This is the direction pursued here.

In order to make the supervised ensemble strategy work, the individual base learners have to be sufficiently diverse in the sense that they should make different errors on unseen data. The diversity is often more important than the accuracy of the individual classifiers, provided their performance is better than random guessing. From this point of view, overtrained base learners are not a big issue. In fact, ensemble classification is often applied to relatively weak and unstable classifiers since these yield higher diversity. It was shown that even fully overtrained base learners, when combined through a classification ensemble, may produce accuracy comparable to state-of-the-art techniques.[8]

The idea of injecting an element of randomness into classification, has been previously used in different forms. In bagging,[3] individual classifiers are generated by training a classifier on different bootstrap samples from the training set. In,[4] the randomness is in the construction of individual classifiers (classification trees) rather than in the training set. In both cases, a set of “weak” classifiers is created, and combined into an ensemble of classifiers. Different combining methods may then be applied to form the final, accurate classification tool,[19] however, a simple majority vote is often sufficient. The process of improving the accuracy of a set of base learners by a proper aggregation strategy is known as boosting.[26]

There exist methods based on random projections that leverage the Johnson–Lindenstrauss Theorem.[1,2] In a nutshell, the JL Theorem states that with high probability the distances and angles between features are approximately preserved when the feature space is projected onto a random subspace of a certain smaller dimension. When an SVM is applied in the projected space, the margin (distinguishability) between classes stays approximately the same and one thus alleviates the CoD.

After numerous experiments with ensemble classifiers on the nsF5 algorithm, we arrived at a construction that worked the best for us not only for JPEG images but also for the spatial domain. We make no claim that the classifier described below is the best possible approach. This work should rather be viewed as the first and quite promising step in this direction while additional effort is certainly required to study and optimize its performance across various embedding algorithms.

3.2 The proposed ensemble classifier

Let d be the dimensionality of the prefeatures, N^{trn} and N^{tst} the number of training and testing examples from each class, and L the number of base learners whose decisions will be fused. Furthermore, let $\mathbf{x}^{(m)}, \mathbf{y}^{(m)} \in \mathbb{R}^d$, $m = 1, \ldots, N^{\mathrm{trn}}$, and $\mathbf{b}^{(k)} \in \mathbb{R}^d$, $k = 1, \ldots, N^{\mathrm{tst}}$, be the cover and stego prefeature vectors for the training set and prefeatures for the testing examples, respectively. The ensemble classifier is described using Algorithm 1. For $\mathcal{D} \subset \{1, \ldots, d\}$, $\mathbf{x}^{(\mathcal{D})}$ is the subset of features $\{\mathbf{x}^{(k)}\}_{k\in\mathcal{D}}$.

The individual classifiers F_l map to $\{0, 1\}$, where 0 stands for cover and 1 for stego. The base learners are Fisher Linear Discriminants (FLDs) because such low complexity, weak, and unstable classifiers desirably increase diversity. After collecting L base learners, the final class predictor is formed by combining their individual decisions using an unweighted (majority) voting strategy.

Algorithm 1 Ensemble classifier.

```
for $l$=1 to $L$ do
    Randomly select $\mathcal{D}_l \subset \{1,\ldots,d\}, |\mathcal{D}_l| = d_{\text{red}} < d$
    Train a classifier $F_l$ on cover features $\mathbf{x}_m^{(\mathcal{D}_l)}$ and stego features $\mathbf{y}_m^{(\mathcal{D}_l)}$, $m = 1,\ldots,N^{\text{trn}}$. Each classifier is a mapping $F_l : \mathbb{R}^{d_{\text{red}}} \to \{0,1\}$.
    Make decisions using $F_l$:
        $$F_l(\mathbf{b}) \triangleq [F_l(\mathbf{b}^{(1)})),\ldots,F_l(\mathbf{b}^{(N^{\text{tst}})})] \in \{0,1\}^{N^{\text{tst}}}. \quad (1)$$
end for
Fuse all decisions by voting for each test example $k \in \{1,\ldots,N^{\text{tst}}\}$:
    $$F(k) = \begin{cases} 1 & \text{when } \sum_{l=1}^{L} F_l(\mathbf{b}^{(k)}) > L/2 \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$
return $F(k), k = 1,\ldots,N^{\text{tst}}$
```

Note that all base learners in the algorithm are trained on feature spaces of a fixed dimension d_{red} that can be chosen to be significantly smaller than the full dimensionality d. Even though the performance of individual base learners can be weak, the accuracy quickly improves after fusion and eventually levels out. The voting could be replaced by other aggregation rules. For example, when the decision boundary is a hyperplane, one can use the sum of projections on the normal vector of each classifier or the sum of likelihoods of each projection after fitting models to the projections of cover and stego features. Because in our experiments all three fusion strategies gave essentially identical results, we recommend using voting due to its simplicity. Finally, the individual classifiers should be adjusted to meet a desired performance criterion.

3.3 Implementation issues

An important advantage of the proposed algorithm is its low computation complexity – if implemented correctly, the training does not depend on the prefeature space dimensionality d, which can be achieved by storing the prefeatures as individual files. The complexity is therefore driven by d_{red} rather then d, as only selected d_{red} prefeatures need to be accessed at a time. The ensemble classifier depends on two parameters – d_{red} and L. As will become apparent in the next experimental section, the classification accuracy saturates rather quickly with L. For the fastest performance one should choose the smallest L that gives satisfactory performance. The optimal value of d_{red} is more critical as values that are too small or too large may give sub-optimal results. Obviously, one could implement a one-dimensional grid search similar to cross-validation in SVMs to find the optimal value of d_{red}. Fortunately, we observed that the optimum is quite flat, which means that the grid could be sparse. In all our tests, we simply selected d_{red} by hand after a few initial trials.

4. TESTING THE FRAMEWORK IN JPEG DOMAIN

We analyze the proposed steganalysis framework and demonstrate the effects of various design parameters on the nsF5,[14] which is currently the most secure algorithm that directly manipulates quantized DCT coefficients.

All experiments were carried out on a database of 6,500 JPEG images coming from more than 20 different cameras. All images were converted to grayscale, resized so that the smaller side had 512 pixels with aspect ratio preserved, and JPEG compressed with quality factor 75. A randomly selected half of the images was used for training and the other half for testing. We used LIBSVM,[6] the publicly available implementation of SVMs. The decision threshold of all classifiers (including the FLD base learners) was always adjusted to produce the minimum overall average classification error

$$P_{\text{E}} = \min_{P_{\text{FA}}} \frac{1}{2}\left(P_{\text{FA}} + P_{\text{MD}}(P_{\text{FA}})\right) \quad (3)$$

on the training data (P_{FA} and P_{MD} are the false-alarm and missed-detection probabilities.). This error is also used to report the accuracy of detection in the entire paper.

4.1 Forming the prefeatures

Our prefeature sets for the JPEG domain will be built by various combinations of three Cartesian-calibrated[17] feature sets: the 2×274-dimensional CC-PEV set,[24] the CC-SHI[27] set with 2×324 features, and the $2 \times 24,000$-dimensional CC-C set. The first two are described in their corresponding publications, while the third set is formed by co-occurrence matrices of selected coefficient pairs. It was built in a scalable manner to capture as many dependencies among DCT coefficients as possible. We describe it next.

We denote by $\mathbf{D}_{kl}^{(i,j)}$ the (k,l)th coefficient in the (i,j)th 8×8 block, $k,l = 0,\dots,7$, $i = 1,\dots,8\lceil M/8\rceil$ and $j = 1,\dots,8\lceil N/8\rceil$, where $M \times N$ are image dimensions. First, just as in CC-PEV and CC-SHI, the coefficients are truncated: $\mathbf{D}_{kl}^{(i,j)} \leftarrow \mathrm{trunc}_T(\mathbf{D}_{kl}^{(i,j)})$, where $\mathrm{trunc}_T(x) = x$ when $x \in [-T,T]$, and $\mathrm{trunc}_T(x) = T\mathrm{sign}(x)$ otherwise ($T \in \mathbb{N}_0$). The prefeatures are co-occurrence matrices for pairs of DCT coefficients. A pair is defined as

$$\mathbf{P}\left(\Delta i, \Delta j, k_1, l_1, k_2, l_2\right) = \left\{ \left[\mathbf{D}_{k_1 l_1}^{(i,j)}, \mathbf{D}_{k_2 l_2}^{(i+\Delta i, j+\Delta j)}\right] \,\middle|\, i = 1,\dots,N^{(i)}, j = 1,\dots,N^{(j)} \right\}, \tag{4}$$

where $\Delta i, \Delta j \in \mathbb{N}_0$, $k_1, l_1, k_2, l_2 \in \{0,\dots,7\}$ are the parameters of the pair, and $N^{(i)} = 8\lceil M/8\rceil - \Delta i$ and $N^{(j)} = 8\lceil N/8\rceil - \Delta j$. To be properly defined, the following condition must hold: $|\Delta i| + |\Delta j| + |k_1 - k_2| + |l_1 - l_2| > 0$. For example, the six-tuple $(0,0,0,1,0,2)$ describes pairs of horizontally neighboring DCT modes $[0,1]$ and $[0,2]$ in the same 8×8 block (intra-block), while $(1,0,1,1,1,1)$ are pairs of modes $[1,1]$ located in the two vertically neighboring 8×8 blocks (inter-block). Note that unlike CC-SHI or CC-PEV, the CC-C set can capture inter-block dependencies between two different DCT modes.

The co-occurrence matrices are formed from pairs in the usual manner:

$$\mathbf{C}_{st} = \frac{1}{N^{(i)} N^{(j)}} \sum_{i=1}^{N^{(i)}} \sum_{j=1}^{N^{(j)}} \left| \left\{ [a,b] \in \mathbf{P}\left(\Delta i, \Delta j, k_1, l_1, k_2, l_2\right) \middle| a = s, b = t \right\} \right|. \tag{5}$$

In this paper, we fix $T = 4$ yielding the co-occurrence matrix of dimension 81 for every pair of DCT modes characterized by the 6-tuple $(\Delta i, \Delta j, k_1, l_1, k_2, l_2)$. In order to make the construction of the individual co-occurrence matrices more systematic, we first sort all possible DCT mode pairs based on their importance, and then start forming the co-occurences from the most important to the least important ones. As a measure of importance, we used the mutual information (MI) computed over a large enough set of randomly selected DCT coefficient pairs from those two modes. We denote the feature set created as the top N concatenated co-occurrence matrices as CN, yielding the dimension $N \times 81$. After Cartesian calibration, the dimensionality of what we denote as CC-CN doubles to $2 \times N \times 81$. For example, the set CC-C300 has 48,600 features.

Note that the procedure for generating the prefeatures is an unsupervised technique and does not depend on the analyzed steganographic method. If we were to capture only those dependencies that are disturbed by embedding, one should use other importance measures than MI, such as the FLD ratio[9] or MMD.[16] But we refrain from doing so for two reasons. First, we want the prefeatures to be universal and applicable to a wider range of steganographic methods. Second, embedding invariants may be very useful for steganalysis, provided they are correlated with some other cover statistic that is disturbed by embedding. Think of the number of ones in a JPEG image embedded with Jsteg.[28]

4.2 Effect of different parameters on steganalysis of nsF5

We now analyze the influence of various design parameters on detection accuracy and we do so for the nsF5 algorithm. All graphs and tables report the median error P_{E} of the steganalyzer. The error bars are MADs obtained over ten random 50/50 splits of the database into the training and testing sets.

In our first experiment, we used existing state-of-the-art feature sets as prefeatures – CC-PEV and CC-SHI. Table 1 shows the results of a soft-margin Gaussian SVM (G-SVM) and our ensemble classifier both using the same prefeatures. The G-SVM was trained using five-fold cross-validation on the following multiplicative grid

$$C = 10^a,\ \gamma = \frac{1}{N_{\mathrm{F}}} \cdot 2^b,\ a \in \{-3,\dots,4\},\ b \in \{-5,\dots,2\}, \tag{6}$$

Prefeatures : CC-SHI

Payload	G-SVM	Ensemble class.
0.05	0.3662 ± 0.0022	0.3640 ± 0.0039
0.10	0.2339 ± 0.0028	0.2302 ± 0.0042
0.15	0.1194 ± 0.0035	0.1159 ± 0.0018
0.20	0.0595 ± 0.0014	0.0507 ± 0.0016

Prefeatures: CC-PEV

Payload	G-SVM	Ensemble class.
0.05	0.3316 ± 0.0028	0.3235 ± 0.0019
0.10	0.1798 ± 0.0018	0.1712 ± 0.0017
0.15	0.0818 ± 0.0008	0.0745 ± 0.0008
0.20	0.0349 ± 0.0011	0.0286 ± 0.0010

Table 1. Steganalyzer error P_{E} for nsF5 using G-SVM and the proposed ensemble classifier with $d_{\mathrm{red}} = 400$ and $L = 31$.

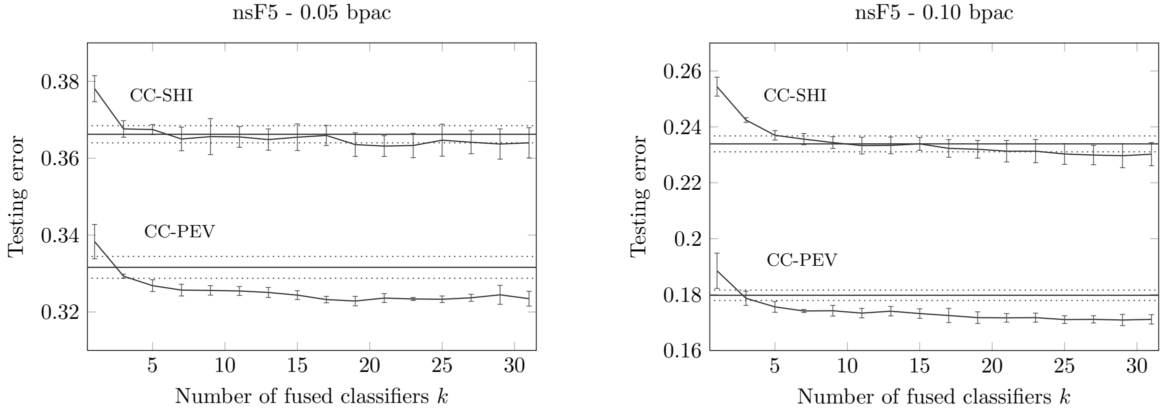


Figure 2. Steganalyzer error P_{E} for nsF5 at payload 0.05 bpac (left) and 0.10 bpac (right). The ensemble classifier, which used $d_{\mathrm{red}} = 400$, is compared to a G-SVM (horizontal reference lines) on CC-PEV and CC-SHI. The dotted horizontal lines show the MAD error bars for the G-SVM.

where N_{F} is the number of prefeatures. The ensemble classifier parameters were fixed at $d_{\mathrm{red}} = 400$ and $L = 31$. In Figure 2, we plot the progress of the testing error P_{E} for selected payloads in bpac (bits per non-zero AC DCT coefficient) as a function of the number of fused base learners L. The results indicate that the proposed method quickly levels up with increasing number of base learners and the performance is similar to G-SVM.

To give an idea about the time savings, the ensemble classifier was trained with CC-PEV features with $L = 31$ and $d_{\mathrm{red}} = 400$ in about 70 seconds,[‡] while training a G-SVM on the same features (with optimal values of the hyperparameters C and γ already found) took approximately 3.5 times longer. The full two-dimensional grid-search for C and γ with five-fold cross-validation took about 8 hours! The difference in complexity becomes even more striking with increased prefeature dimensionality. Training a $50,000$-dimensional prefeature takes the ensemble classifier with $L = 99$ and $d_{\mathrm{red}} = 2000$ approximately 20 minutes, while training a G-SVM with the optimal hyperparameters already found takes about 7.5 hours. The full grid-search is practically impossible to execute in a reasonable time without a high-performance cluster.

The next experiment involved the CC-CN prefeatures introduced in Section 4.1. The number of co-occurrence matrices was fixed to $N = 300$. The C300 and CC-C300 sets correspond to the non-calibrated and Cartesian-calibrated versions of dimensionalities $24,300$ and $48,600$, respectively. The steganalyzer error is shown in Table 2. Despite the very high dimensionality, the calibrated prefeatures perform better. When randomly selecting the prefeatures, the second half of the prefeatures (the reference features) were treated in the same way as the first half. We also tried applying the Cartesian calibration individually, i.e., when randomly selecting d_{red} prefeatures, $d_{\mathrm{red}}/2$ of them were randomly selected from the original features and then supplied with their calibrated counterparts, yielding the desired dimensionality d_{red}. The performance was very similar. The former approach is favorable since it is more general and allows us to incorporate the whole concept of calibration, regardless of its type, into the problem of generating prefeatures.

[‡] This was achieved on a DELL XPS M1530 machine with 4GB RAM and Intel Core 2 Duo processor at 2.50 GHz.

	Ensemble classifier				G-SVM
Payload	C300	CC-C300	CC-C100	CC-C50	CC-PEV
0.05	0.3273	0.3195	0.3249	0.3422	0.3316
0.10	0.1775	0.1628	0.1703	0.1932	0.1798
0.15	0.0756	0.0631	0.0639	0.0799	0.0818
0.20	0.0326	0.0226	0.0218	0.0290	0.0349

Table 2. Steganalyzer error P_{E} using the ensemble classifier with different CC-C prefeatures for $d_{\mathrm{red}} = 2000$ and $L = 99$ when applied to nsF5. As a reference, we supply the G-SVM performance using CC-PEV features.

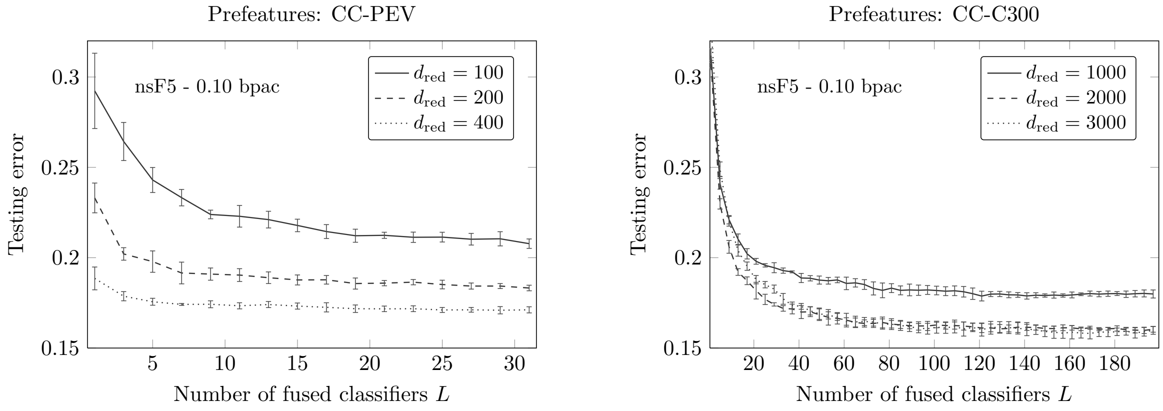


Figure 3. Error P_{E} of the ensemble classifier for different choices of d_{red} for nsF5 and payload of 0.10 bpac.

We also varied the number of co-occurrence matrices $N \in \{50, 100, 300\}$ to find out the effect of the prefeature space dimensionality. As Table 2 shows, more co-occurrence matrices generally improve the performance. This is more apparent for smaller payloads with smaller class distinguishability. For a large payload of 0.20 bpac, however, increasing N from 100 to 300 does not bring any improvement.

Figure 3 shows that the performance of the ensemble classifier highly depends on the reduced dimensionality d_{red}. The figure reports the detection error for two prefeature sets, CC-PEV and CC-C300, for the payload of 0.10 bpac. As already mentioned in Section 3.3, the best value of d_{red} could be found using a one-dimensional search.

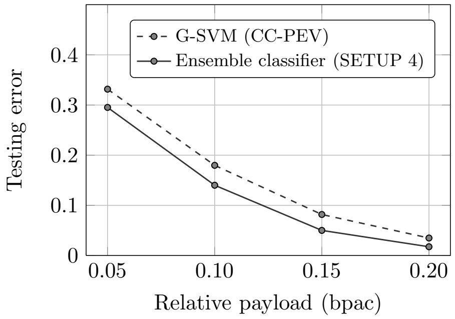
So far, we have shown that the proposed method is capable of working with different prefeatures and that its performance can achieve results similar to a G-SVM. Unlike a G-SVM, though, our approach is scalable w.r.t. the prefeature-space dimensionality. An interesting question is whether the state-of-the-art feature sets can be improved by merging them into a single set and applying the framework. Our next experiment demonstrates that it is indeed possible. The results for the following four setups are shown in Table 3.

- SETUP 1 – Prefeatures: CC-PEV and CC-SHI of dimensionality 1,196, $d_{\mathrm{red}} = 600$.
- SETUP 2 – Prefeatures: CC-PEV, CC-SHI, and CC-C300 of dimensionality 49,796, $d_{\mathrm{red}} = 2,000$.
- SETUP 3 – Prefeatures: CC-PEV and CC-C300, dimensionality 49,148, $d_{\mathrm{red}} = 2,000$. Even though the *individual* performance of CC-PEV and CC-C300 is similar, their representation in the prefeature set is very uneven – while the 548-dimensional CC-PEV is very compact, CC-C300 was constructed without the low-dimensionality requirement and as a result, it spans a vast majority of the prefeature space. This may prevent the ensemble classifier from leveraging the dependencies among individual elements of CC-PEV. Therefore, we modified the random subspace selection, and instead of taking d_{red} randomly selected components as prefeatures, we took only $d_{\mathrm{red}} - 548$ of them and filled the rest with CC-PEV features. In other words, we *always* included all CC-PEV features as part of the random selections.

- SETUP 4 – Prefeatures: the same as in SETUP 3. Instead of always including all 548 CC-PEV features, in this setup we include only its 300 randomly selected elements. This way, by combining linear decision boundaries on these smaller subsets of CC-PEV, we are capable of learning non-linear decision boundaries. In contrast, in SETUP 3 we always employ only a linear relationship among all CC-PEV features. As Table 3 indicates, this modification improves the performance.

In SETUP 1, the prefeature-space dimensionality is low and the performance of the ensemble classifier quickly saturates. Even though combining as few as 15 classifiers is sufficient, we generously fixed $L = 49$. Because the saturation is slower in SETUPs 2–4, we took $L = 149$.

Table 3 clearly indicates that state-of-the-art steganalysis can be improved and that using high-dimensional features with a scalable machine learning method is a promising direction for future steganalysis. This experiment opens up a number of important questions that we intend to pursue in the future. The first is the issue of selecting a good prefeature set and the second is the random selection of prefeatures for the base learners. As one might expect, when the prefeatures significantly differ in their ablity to classify, random selection is obviously not the best strategy. In general, the optimal selection process will likely depend on mutual dependencies among prefeatures and their classification strength.



	Ensemble classifier				G-SVM
Payload	SETUP 1	SETUP 2	SETUP 3	SETUP 4	CC-PEV
0.05	0.3147	0.3100	0.3021	**0.2952**	0.3316
0.10	0.1575	0.1520	0.1508	**0.1401**	0.1798
0.15	0.0608	0.0571	0.0581	**0.0501**	0.0818
0.20	0.0239	0.0193	0.0232	**0.0175**	0.0349

Table 3. Improving steganalysis of nsF5 by merging different prefeature sets and modifying the random subspace generation. As a reference, we show the performance of G-SVM with CC-PEV features.

5. SPATIAL DOMAIN

In this section, we demonstrate the power of ensemble classifiers by applying them to the recently proposed adaptive spatial-domain steganographic algorithm called HUGO.[23] HUGO preserves a very high-dimensional feature ($d = 10^7$) and current state-of-the-art steganalysis methods cannot detect it reliably. HUGO is significantly more secure than existing non-adaptive schemes, an example of which is the optimally-coded ternary ±1 embedding.

5.1 The prefeatures

Our prefeature set is obtained by modeling the joint distribution of higher-order residuals computed from neighboring pixels. Detailed description and motivation behind the prefeature set appears in.[13] In this paper, due to lack of space we only provide a rather condensed exposition. The prefeatures are built by combining two operators – the residual operator R and a co-occurrence operator C. Given a grayscale image with pixel values $\mathbf{x} = (\mathbf{x}_{ij}),\ i = 1, \ldots, M,\ j = 1, \ldots, N$, the residual operator is defined as

$$R(\mathbf{x}) = (\mathbf{H}_{ij}), \quad \mathbf{H}_{ij} = \mathbf{x}_{ij} - \mathrm{Pred}(i, j, \mathcal{N}_{ij}), \tag{7}$$

where Pred(.) predicts the value of the central pixel $\mathbf{x}_{ij}$ based on its neighborhood $\mathcal{N}_{ij}$, $\mathbf{x}_{ij} \notin \mathcal{N}_{ij}$. Thus, the output of the operator is an array of the same (or slightly smaller) dimensions than $\mathbf{x}$. We will use predictors of order 2–6 listed in Table 4.

The residual is a signal with a smaller spread when compared with the range of the original pixel values. Since we will be forming co-occurrence matrices from adjacent values of the residual, all residuals will be truncated to the interval $[-T, T]$ as in Section 4.1.

Order r	$\mathbf{H}_{ij}$
2	$\frac{1}{2}(-\mathbf{x}_{i,j-1}+2\mathbf{x}_{ij}-\mathbf{x}_{i,j+1})$
3	$\frac{1}{3}(-\mathbf{x}_{i,j-1}+3\mathbf{x}_{ij}-3\mathbf{x}_{i,j+1}+\mathbf{x}_{i,j+2})$
4	$\frac{1}{6}(\mathbf{x}_{i,j-2}-4\mathbf{x}_{i,j-1}+6\mathbf{x}_{ij}-4\mathbf{x}_{i,j+1}+\mathbf{x}_{i,j+2})$
5	$\frac{1}{10}(\mathbf{x}_{i,j-2}-5\mathbf{x}_{i,j-1}+10\mathbf{x}_{i,j}-10\mathbf{x}_{i,j+1}+5\mathbf{x}_{i,j+2}-\mathbf{x}_{i,j+3})$
6	$\frac{1}{20}(-\mathbf{x}_{i,j-3}+6\mathbf{x}_{i,j-2}-15\mathbf{x}_{i,j-1}+20\mathbf{x}_{ij}-15\mathbf{x}_{i,j+1}+6\mathbf{x}_{i,j+2}-\mathbf{x}_{i,j+3})$

Table 4. Higher-order residuals for building prefeatures.

Notice that suppressing the image content in $\mathbf{H}_{ij}$ means that the SNR between the stego signal and the remnant of the content is increased. The predictor must not depend on the central pixel value, otherwise the stego signal would become undesirably attenuated in the residual. We do not consider first-order residuals as HUGO preserves their joint higher-order statistics. Also, the residuals are computed only for pixels ij whose entire neighborhood $\mathcal{N}_{ij}$ falls inside the image. This is why the output of the residual operator may be slightly smaller than the input pixel array. The residuals shown in Table 4 are computed along the horizontal direction. The vertical (v), diagonal (d) and minor diagonal (m) residuals are defined analogically.

We next define four types of co-occurrence operators: C^{h}, C^{v}, C^{d}, and C^{m}. We explain them on the example of C^{h} as it should be obvious how to define the rest. For any matrix $\mathbf{A}_{ij}$, $C^{\mathrm{h}}(\mathbf{A})$ is a 3D array whose elements are the co-occurrences:

$$C^{\mathrm{h}}(\mathbf{A})_{d_1d_2d_3}=|\{(i,j)|\mathbf{A}_{ij}=d_1\ \&\ \mathbf{A}_{i,j+1}=d_2\ \&\ \mathbf{A}_{i,j+2}=d_3\}|,\quad d_1,d_2,d_3\in[-T,T]. \tag{8}$$

Having introduced the co-occurrence operators, we form two types of prefeatures (sample joint probability matrices): the MINMAX type and the MARKOV type. We first describe the MARKOV type, for which the co-occurrences are formed as in the SPAM feature set.[22] For each $r\in\{2,3,4,5,6\}$, two 3D arrays, $C^{\mathrm{h}}(\mathbf{H})+C^{\mathrm{v}}(\mathbf{V})$ and $C^{\mathrm{d}}(\mathbf{D})+C^{\mathrm{m}}(\mathbf{M})$, are formed, each of dimensionality $(2T+1)^3$. Thus, we have in total $2\times5\times(2T+1)^3$ prefeatures of type MARKOV.

The MINMAX type works with a different set of residuals

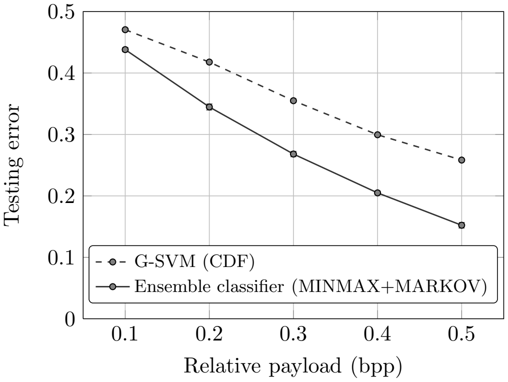
$$\mathbf{R}^{\mathrm{MIN}}_{ij}=\min\{\mathbf{H}_{ij},\mathbf{V}_{ij},\mathbf{D}_{ij},\mathbf{M}_{ij}\}, \tag{9}$$
$$\mathbf{R}^{\mathrm{MAX}}_{ij}=\max\{\mathbf{H}_{ij},\mathbf{V}_{ij},\mathbf{D}_{ij},\mathbf{M}_{ij}\}. \tag{10}$$

Two co-occurrences are defined for each r: $C^{\mathrm{h}}(\mathbf{R}^{\mathrm{MIN}})+C^{\mathrm{v}}(\mathbf{R}^{\mathrm{MIN}})$ and $C^{\mathrm{h}}(\mathbf{R}^{\mathrm{MAX}})+C^{\mathrm{v}}(\mathbf{R}^{\mathrm{MAX}})$. The dimensionality of the MINMAX type prefeature set is again $2\times5\times(2T+1)^3$. Thus, the union of both types of prefeatures has dimensionality $20\times(2T+1)^3$. In this paper, we used $T=4$, which leads to a $14,580$-dimensional prefeature set.

5.2 Experiments

In this section, we steganalyze HUGO using current state-of-the-art steganalyzer for spatial domain and the ensemble classifier using the prefeature set from the previous section. We used the BOSSbase 0.92 database offerred for the BOSS competition.[11] It consists of $9,074$ cover and the same number of stego images, all originally obtained using seven different digital cameras in the RAW format (CR2, DNG). The images were converted to grayscale and then resampled so that the shorter size was 512 pixels and then cropped to 512×512 pixels. The database was divided randomly into two halves, one used for training and the other for testing. The performance is averaged over ten random splits.

The performance of our ensemble classifier is contrasted with a Gaussian SVM implemented using five-fold cross-validation trained on the $1,234$-dimensional CDF set. The ensemble classifier was implemented with $L=51$ and $d_{\mathrm{red}}=1,600$. The comparison is carried out separately for cover images versus stego images embedded with relative payload $\alpha\in\{0.1,0.2,0.3,0.4,0.5\}$ bpp, i.e., a separate classifier was trained for each payload. The results are rendered in a chart in Figure 4. The ensemble classifier on a high-dimensional prefeature space outperforms the state-of-the-art by a large margin.



	G-SVM	Ensemble classifier
Payload	CDF	MINMAX+MARKOV
0.1	0.4705	0.4381
0.2	0.4178	0.3447
0.3	0.3549	0.2681
0.4	0.2995	0.2049
0.5	0.2583	0.1522

Figure 4. Steganalyzer error P_{E} for a G-SVM using CDF features (dashed line) and our ensemble classifier using MINMAX+MARKOV features (solid line) for five different payloads for HUGO.

6. CONCLUSIONS

Modern steganographic systems are designed in feature spaces of very high dimension and can thus preserve many complex dependencies among individual elements of the cover. This makes them particularly difficult to steganalyze. Steganalysis needs to follow the suit and build detectors in feature spaces that can capture as many complex relationships among pixels as possible. This leads to two non-trivial problems – namely how to form useful features and then how to build classifiers in high-dimensional spaces in an efficient manner without encountering the curse of dimensionality.

The high dimensionality itself, however, is not sufficient to improve detection. For example, one can foolishly attempt to classify the entire images without any preprocessing or dimensionality reduction. Because digital media objects, such as digital images, exhibit dependencies mostly across small distances, a good set of prefeatures that encapsulate various types of relationships among neighboring elements of the cover may suffice. In this paper, we looked at several ways how to form good high-dimensional prefeatures – by merging existing feature sets and by forming joint probability distributions (co-occurrences) among those cover elements that exhibit strong relationship (because embedding will likely violate at least some of the relationships).

The increased dimensionality of the feature space leads to numerous problems in machine learning that are commonly recognized as the curse of dimensionality – the failure of the classifier to generalize to previously unseen data and cover sources and a rapidly increasing complexity of training. A standard way to implement classifiers today is to train an SVM with a Gaussian kernel on a large database of cover and stego images. However, SVMs with a Gaussian kernel can be very expensive to train on a large number of images for high feature dimensions. Although very efficient implementations of linear SVMs exist (e.g., the LIBLINEAR package[10]), the training can still take a substantial amount of time. Moreover, linear SVMs are often suboptimal w.r.t. their Gaussian counterparts.

The main contribution of our paper is a scalable, fast, and simple classification methodology based on ensemble classifiers. The low computational complexity and scalability allowed us to routinely work with features of dimensionality over 40,000 on a database of almost 90,000 images. A comparison with Gaussian SVMs on smaller-scale problems seems to indicate that our ensemble classifier is as accurate as or even slightly more accurate than the much more costly Gaussian SVMs. The ensemble classifier is built by fusing decisions of weak classifiers constructed on random subspaces of the high-dimensional prefeature space. In practice, we fuse about 100 decisions obtained from a simple Fisher linear discriminant on subspaces of dimensionality 400–2000. The classifier is very simple to implement and scales well with both the number of features and images. It also allows fast design and development of steganalysis detectors, a property which we used conveniently in the BOSS competition.

This paper is the first step in a direction that appears very promising and highly relevant to steganalysis of modern embedding algorithms. As such, the work presented here is far from complete and leaves many problems open and questions unanswered. The two most pressing issues that have a strong effect on the classifier's performance are the formation of prefeatures and the random selection of subspaces. We explored a method for computing the prefeatures as sample joint distributions of pairs of cover elements that we intuitively expect to be dependent (e.g., neighboring DCT modes or coefficients or adjacent pixels). This strategy worked very well for the nsF5 algorithm and HUGO. The second issue concerns the process of selecting the random subspaces. Our experiments with combining a high-dimensional prefeature set with existing hand-designed feature sets clearly suggest that the selections should be somehow determined by the mutual dependencies among the prefeatures and their ability to classify instead of being completely random.

7. ACKNOWLEDGEMENTS

This work was supported by Air Force Office of Scientific Research under the research grant number FA9550-08-1-0084 and 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.

REFERENCES

1. N. Ailon and B. Chazelle. Faster dimension reduction. *Communications ACM*, 53(2):97 – 104, 2010.
2. A. Blum. Random projection, margins, kernels, and feature-selection. In M. Grobelnik, S. Gunn, and J. Shawe-Taylor, editors, *Subspace, Latent Structure and Feature Selection, Statistical and Optimization Perspectives Workshop*, volume 3940 of Lecture Notes in Computer Science, pages 52 – 78, Bohinj, Slovenia, February 2005.
3. L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, August 1996.
4. L. Breiman. Random forests. *Machine Learning*, 45:5–32, October 2001.
5. G. Cancelli, G. Doërr, I. J. Cox, and M. Barni. A comparative study of ±1 steganalyzers. In *Proceedings IEEE International Workshop on Multimedia Signal Processing*, pages 791–796, Cairns, Australia, October 8–10, 2008.
6. Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
7. 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.
8. A. Cutler and G. Zhao. PERT - perfect random tree ensembles. *Computing Science and Statistics*, 33:490–497, 2001.
9. R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* New York: John Wiley & Sons, Inc., 2nd edition, 2001.
10. Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen 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`.
11. T. Filler, T. Pevný, and P. Bas. BOSS (Break Our Steganography System). `http://boss.gipsa-lab.grenoble-inp.fr`, July 2010.
12. 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.
13. 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*, Lecture Notes in Computer Science, Prague, Czech Republic, May 18–20, 2011.
14. 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.

15. 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.
16. A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 513–520. MIT Press, Cambridge, MA, 2007.
17. 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.
18. 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.
19. L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms.* Wiley-Interscience, 2004.
20. 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.
21. 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.
22. 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.
23. 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.
24. 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.
25. 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.
26. 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.
27. 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.
28. D. Upham. Steganographic algorithm JSteg. Software available at `http://zooid.org/~paul/crypto/jsteg`.