Quantitative Steganalysis Using Rich Models

Jan Kodovský and Jessica Fridrich

Department of Electrical and Computer Engineering
Binghamton University, Binghamton, NY 13902, USA

ABSTRACT

In this paper, we propose a regression framework for steganalysis of digital images that utilizes the recently proposed rich models – high-dimensional statistical image descriptors that have been shown to substantially improve classical (binary) steganalysis. Our proposed system is based on gradient boosting and utilizes a steganalysis-specific variant of regression trees as base learners. The conducted experiments confirm that the proposed system outperforms prior quantitative steganalysis (both structural and feature-based) across a wide range of steganographic schemes: HUGO, LSB replacement, nsF5, BCHopt, and MME3.

1. INTRODUCTION

Quantitative steganalysis estimates the number of embedding changes or the message length. Initially, quantitative steganalyzers utilized the knowledge of a specific embedding mechanism and exploited its impact on statistics of cover coefficients. Examples are RS analysis [7], sample pairs (SP) analysis [5], or weighted-stego (WS) steganalysis [6,13]. These so-called *structural* attacks do not require any training phase, however, they are fundamentally limited to a specific type of steganography. Furthermore, it is unlikely that structural attacks could be developed for more advanced steganographic schemes, such as HUGO [22] or BCHopt [24].

An alternative approach was proposed by Pevný *et al.* [23], where the problem of message-length estimation was formulated as a regression in a pre-defined feature space. A quantitative steganalyzer constructed this way can be built for an arbitrary embedding method, and its performance depends on how well the features react to embedding. The price for such flexibility is the need for a training phase in which the regressor is supplied with samples of features extracted from a database of images. In order to make the training computationally feasible, the feature-space dimensionality needs to be adequately low.

The goal of this paper is to extend the feature-based quantitative steganalysis to the recently proposed rich models [8, 17] – high-dimensional statistical image descriptors that have been shown to substantially improve classical (binary) steganalysis. It is reasonable to expect a similar performance improvement when rich models are adopted for quantitative steganalysis.

Rich models require a scalable machine learning algorithm. In the case of binary classification, this can be the FLD-based ensemble classifier [19] or the online ensemble average perceptron [20], both merging a large number of simpler classifiers built on random subspaces of the original feature space. A similar strategy can be implemented for quantitative steganalysis – instead of constructing a regression function in the entire rich model, one could combine a set of simpler regressors (called base learners) built on its random subspaces. However, since the embedding distortion of modern steganographic algorithms is *not* linear in payload (due to coding and adaptivity of embedding changes), the ability to capture non-linear relationships is crucial. Unfortunately, using a complex non-linear regressor at the base-learner level (for example a kernelized support vector regression [25]) would make the overall system computationally infeasible – not only do we need to train a large number of these base learners, but we also need to search for the optimal value of the random subspace dimensionality.

A possible solution to keep the training reasonably fast could be to use simple linear base learners, for example the ordinary least square regression (OLS), and then *inject* the non-linearity into the system by means of a second-layer regression training in which the first-layer estimates would serve as predictors. The

E-mail: {jan.kodovsky, fridrich}@binghamton.edu

success of this strategy, known as regression stacking [2, 26], depends on the ability of the second, non-linear regressor to "correct" for the linear bias introduced in the first layer.

In this paper, we selected a different approach based on Friedman's gradient boosting [11] – the regression function is approximated by a generalized additive model and the quality of the fit is sequentially improved in a gradient-descent manner. In order to utilize the potential of this technique, we introduce a new base-learner that is capable of *local* estimation – it is a variant of a regression tree [3] modified to reflect the specifics of steganalysis. Compared to regression stacking, the proposed solution is simpler (only a single training set is needed) and delivers better results since the non-linearity is handled directly by the base learners.

In the next section, we formally describe the proposed regression framework, introduce its base learner, and relate it to other machine-learning techniques. In Section 3, we discuss and demonstrate the influence of individual system parameters on the performance. Finally, in Section 4 we conduct quantitative steganalysis of five different steganographic methods and contrast the accuracy of the proposed scheme with prior art. The paper is summarized in Section 5.

2. DESCRIPTION OF THE REGRESSION FRAMEWORK

A quantitative steganalyzer is a real-valued mapping $F : \mathcal{F} \to \mathcal{P} \subseteq \mathbb{R}$, where $\mathcal{F} \equiv \mathbb{R}^n$ is a feature-space representation of cover objects (digital images) and $\mathcal{P}$ is a compact subset representing the space of considered message lengths $\alpha \in \mathcal{P}$. In this paper, α is always a relative length expressed in bits per pixel (bpp) or bits per non-zero AC DCT coefficient (bpac), in case of JPEG images, and modeled as a realization of a random variable $P \sim p(x)$, where $p(x)$ is the prior probability distribution over $\mathcal{P}$. Typically, the payloads are assumed to be uniform over $\mathcal{P}$, which is the least informative distribution.

Since the choice (construction) of a proper feature space $\mathcal{F}$ has been studied intensively in the past years, it is not the topic of this paper. We assume the space $\mathcal{F}$ to be fixed and to react reasonably well to the embedding changes. Furthermore, as modern steganalysis uses high-dimensional representations [8, 12, 16], we assume that the dimensionality of $\mathcal{F}$ is too large to operate directly in it. Instead, following the suit of [19], we will be extracting the information from $\mathcal{F}$ "by pieces," considering only a small subspace at a time.

Our work is based on the gradient boosting proposed by Friedman [11] where the regression function is defined to minimize the expected error:

$$\hat{F} = \arg\min_{F} \int_{\mathcal{F}\times\mathcal{P}} L(y, F(\mathbf{x}))p(y)\mathrm{d}\mathbf{x}\mathrm{d}y, \tag{1}$$

where $L(a, b)$ is a pre-defined loss function. In this paper we employ the commonly used squared loss function

$$L(a, b) = \frac{1}{2}(a - b)^2. \tag{2}$$

Following the notation of [11], we seek a solution to (1) as a linear combination of some base functions $h(\mathbf{x}; \mathbf{a}) : \mathcal{F} \to \mathcal{P}$ parametrized by a vector $\mathbf{a} \in \mathbb{R}^p$. In other words, we restrict F to be an additive expansion of the form

$$F(\mathbf{x}; \{\beta_m, \mathbf{a}_m\}_1^M) = \sum_{m=1}^{M} \beta_m h(\mathbf{x}; \mathbf{a}_m), \tag{3}$$

which is a common restriction in numerical optimization. The choice of base functions is discussed in Section 2.1.

Given the training set $\{\mathbf{x}_i, y_i\}_1^N$, the optimization procedure is simplified by looking for the parameters $\{\beta_m, \mathbf{a}_m\}_1^M$ in *stages*. For $m = 1, \ldots, M$,

Algorithm 1 Gradient Boosting algorithm with a general loss function $L(a, b)$.

1. $F_0(\mathbf{x}) = \arg\min_\beta \sum_{i=1}^N L(y_i, \beta)$
2. For $m = 1$ to M do:
3. $\quad g_m(\mathbf{x}_i) = \left[\frac{\partial L(y_i, F(\mathbf{x}))}{\partial F(\mathbf{x})}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x}_i)}, \ \ i = 1, \ldots, N$
4. $\quad \mathbf{a}_m = \arg\min_{\beta,\mathbf{a}} \sum_{i=1}^N \left[-g_m(\mathbf{x}_i) - \beta h(\mathbf{x}_i; \mathbf{a})\right]^2$
5. $\quad \beta_m = \arg\min_\beta \sum_{i=1}^N L(y_i, F_{m-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i; \mathbf{a}_m))$
6. $\quad F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \beta_m h(\mathbf{x}; \mathbf{a}_m)$
7. End

$$(\beta_m, \mathbf{a}_m) = \arg\min_{\beta,\mathbf{a}} \sum_{i=1}^N L(y_i, F_{m-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i; \mathbf{a})), \tag{4}$$

and

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \beta_m h(\mathbf{x}; \mathbf{a}_m). \tag{5}$$

This stagewise greedy strategy can be interpreted as an incremental improvement of the current solution where the mth increment $\beta_m h(\mathbf{x}; \mathbf{a}_m)$ is the best step towards minimizing the training error, constrained by our choice of the base learners $h(\mathbf{x}; \mathbf{a}_m)$. In other words, the solution to (4) can be viewed as such a member of the base-function family that is "closest" to the negative gradient of the current loss. In [11], Friedman suggests implementing this strategy in two steps. First, find the parameter $\mathbf{a}_m$ as the least square approximation

$$\mathbf{a}_m = \arg\min_{\beta,\mathbf{a}} \sum_{i=1}^N \left[-g_m(\mathbf{x}_i) - \beta h(\mathbf{x}_i; \mathbf{a})\right]^2 \tag{6}$$

to the (unconstrained) negative gradient giving the steepest-descent direction:

$$-g_m(\mathbf{x}_i) = -\left[\frac{\partial L(y_i, F(\mathbf{x}))}{\partial F(\mathbf{x})}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x}_i)} \tag{7}$$

The coefficient β_m is then found to minimize

$$\beta_m = \arg\min_\beta \sum_{i=1}^N L(y_i, F_{m-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i; \mathbf{a}_m)) \tag{8}$$

which corresponds to the line search implemented in the traditional gradient-descent optimization technique. Algorithm 1 summarizes the gradient boosting procedure.

Even though gradient boosting is a general technique that can be applied to an arbitrary differentiable loss function $L(a, b)$, we restrict ourselves to the squared loss (2) for which Algorithm 1 simplifies – the gradient (7) reduces to

$$-g_m(\mathbf{x}_i) = y_i - F_{m-1}(\mathbf{x}_i) \tag{9}$$

and the iterative procedure can thus be implemented as a continuous update of the training responses y_i to reflect the actual estimation error, $y_i - F_{m-1}(\mathbf{x}_i)$, before executing the least square regression in Step 4. Furthermore, Steps 4 and 5 become identical since

$$L(y_i, F_{m-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i; \mathbf{a}_m)) = \left[y_i - F_{m-1}(\mathbf{x}_i) - \beta h(\mathbf{x}_i; \mathbf{a}_m)\right]^2 = \left[-g_m(\mathbf{x}_i) - \beta h(\mathbf{x}_i; \mathbf{a}_m)\right]^2 \tag{10}$$

which makes it unnecessary to treat the parameter β separately from the remaining base-learner parameters $\mathbf{a}_m$. The simplified variant of the gradient boosting with the squared loss is summarized in Algorithm 2.

Algorithm 2 Gradient Boosting algorithm with the squared loss function (2).

1. $F_0(\mathbf{x}) = \arg\min_\beta \sum_{i=1}^N L(y_i, \beta)$
2. For $m = 1$ to M do:
3. $\quad y_i \leftarrow y_i - F_{m-1}(\mathbf{x}_i)$, $i = 1, \ldots, N$
4. $\quad \mathbf{a}_m = \arg\min_{\mathbf{a}} \sum_{i=1}^N \left[y_i - h(\mathbf{x}_i; \mathbf{a})\right]^2$
5. $\quad F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + h(\mathbf{x}; \mathbf{a}_m)$
6. End

2.1 Base functions $h(\mathbf{x}; \mathbf{a})$

The choice of the base learners has a crucial impact on the quality of the resulting regression function. The function $h(\mathbf{x}; \mathbf{a})$ needs to be non-linear, it should be able to estimate *locally*, and there are also computational considerations. Regression trees [3] are simple yet powerful base learners, due to their flexibility and low computational complexity. Unfortunately, regression trees would not work well in steganalysis because they create decisions in every node based on a single feature and in steganalysis, there is typically no single feature, or a small group of features, that would be responsible for a reasonably accurate estimation [8].* Instead, every group of features adds a small contribution to the overall performance, and the power of feature spaces in steganalysis appears to be in their entirety, which holds especially for high-dimensional rich models.

To keep the flexibility of regression trees, but also to reflect the steganalysis-specific nature of our feature spaces, we modified the tree-splitting procedure as follows. First, choose a random subspace $\mathcal{F}_{\text{sub}} \subset \mathcal{F}$ of a fixed dimension d_{sub}, uniformly equiprobably, and train a regressor in $\mathcal{F}_{\text{sub}}$ (recall that we assume that the full dimension of $\mathcal{F}$ is too high to operate directly in $\mathcal{F}$). We need to consider a regression algorithm that utilizes all d_{sub} dimensions *at once* and that is computationally inexpensive. Thus, we use a linear OLS regression for this purpose. Since OLS is a well established technique, we refrain from giving further technical details, and denote the resulting linear regressor $B : \mathcal{F}_{\text{sub}} \to \mathcal{P}$.

The base learners will be constructed iteratively from the following step function

$$B'(\mathbf{x}; T, c_L, c_R) = \begin{cases} c_L & \text{if } B(\mathbf{x}) \leq T, \\ c_R & \text{otherwise,} \end{cases} \tag{11}$$

whose "orientation" is determined by B. The function B' divides the training set into two regions where the target variable is approximated by the constants c_L and c_R, respectively. This binary splitting can be executed recursively to both parts of the training set until a predefined depth of the tree is reached, similarly as in classical binary regression trees. We will denote the tree depth l. Higher values of l result in more accurate regression functions, but they are more expensive to train and more prone to overtraining. The proper choice of l is discussed in Section 3.2.

The parameter vector $\mathbf{a} = (T, c_L, c_R)$ is determined through the least squares in Step 4 of Algorithm 2:

$$\mathbf{a} = \arg\min_{T, c_L, c_R} \sum_{i=1}^N \left(y_i - h(\mathbf{x}_i; T, c_L, c_R)\right)^2. \tag{12}$$

For a fixed threshold T, the optimal values of c_L and c_R in equation (12) are trivially the averages of y_i over the training samples for which $B(\mathbf{x}_i) \leq T$ and $B(\mathbf{x}_i) > T$, respectively. Thus, solving (12) requires only a one-dimensional search for the optimal threshold T.

*This could happen for insecure and faulty stego-schemes for which there usually exist accurate targeted attacks [18]. In this paper, we prefer to focus on modern, difficult-to-detect algorithms.

2.2 Regularization

Minimizing the training error in Step 4 of Algorithm 2 can result in overtraining – while still decreasing the training error, the error on the testing set stops decreasing and starts to increase. There are several ways how to prevent this from happening – for example by decreasing the number of base learners M or by decreasing the tree depth l. Both strategies lower the degree of "complexity" of the resulting regression functions $F_M(\mathbf{x})$. The optimal values of M and l could be determined by means of cross-validation.

Another way of regularization is through shrinkage [4], where the amount of improvements in every stage of the gradient boosting is constrained by an additional parameter $\eta > 0$. This parameter has the role of a "learning rate," and has been successfully used in many statistical learning algorithms in the past, including neural networks or ridge regression. Step 5 of Algorithm 2 is simply replaced by

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \eta h(\mathbf{x}; \mathbf{a}_m), \quad 0 < \eta \leq 1. \tag{13}$$

A smaller value of η typically results in more accurate regression function F_M, however, the convergence is slower (higher M is needed). Thus, the choice of η is often a trade-off between the regression accuracy and the computation time. In Section 3, we illustrate the effects of different choices of M, l, and η on steganalysis of HUGO [22].

2.3 Validation

In order to determine the proper value of the random subspace dimensionality d_{sub}, as well as optimal values of parameters M, l, and η, we need a continuous performance feedback. The simplest way is to set a small portion of the training set aside and use it to evaluate the accuracy of the regression function $F_m(\mathbf{x})$ after every stage of the gradient boosting. We denote this validation set $\mathcal{X}_{\text{val}} = \{\mathbf{x}_j, y_j\}_1^{N_{\text{val}}}$ and define the validation (squared) error

$$E_{\text{val}}(F_m) = \frac{1}{N_{\text{val}}} \sum_{j=1}^{N_{\text{val}}} (y_j - F_m(\mathbf{x}_j))^2. \tag{14}$$

Throughout this paper, we use 20% of the original training set for validation purposes, leaving the remaining 80% for the actual regression training.

3. INFLUENCE OF INDIVIDUAL PARAMETERS

The choice of the parameters d_{sub}, M, l, and η affects the performance of the proposed steganalysis methodology. In this section, this influence is illustrated on the steganographic algorithm HUGO [22]. We used images from BOSSbase ver. 0.92 [1], a publicly available database that has been recently adopted for benchmarking in steganalysis.

All images were embedded using the publicly available simulator of HUGO with parameters $\gamma = 4$, $\sigma = 10$, and $T = 255$ and with payloads distributed uniformly over $[0, 1]$ bits per pixel (bpp). Images were then randomly divided in two halves, one of which was used for training and the other one for testing. One fifth of the training part was further reserved for validation purposes as described in Section 2.3. As for the feature space $\mathcal{F}$, we used the $12,753$-dimensional spatial-domain rich model (SRMQ1) proposed in [8].

We executed our gradient boosting regression with different choices of the parameters d_{sub}, l, and η, and monitored the progress of the validation error $E_{\text{val}}(F_m)$ with the growing number of base learners m.

3.1 Optimal number of base learners M

While the training error keeps decreasing with every new stage of gradient boosting, the validation error $E_{\text{val}}(F_m)$ reliably corresponds to the expected testing error because none of the images from the validation set were used for the regression training (see Figure 1 for illustration). Therefore, the optimal number of base learners M can be determined directly by observing the validation error $E_{\text{val}}(F_m)$ – once it stops improving, the training can be halted. In the rest of the experiments in this paper, the value of M is thus always determined automatically.

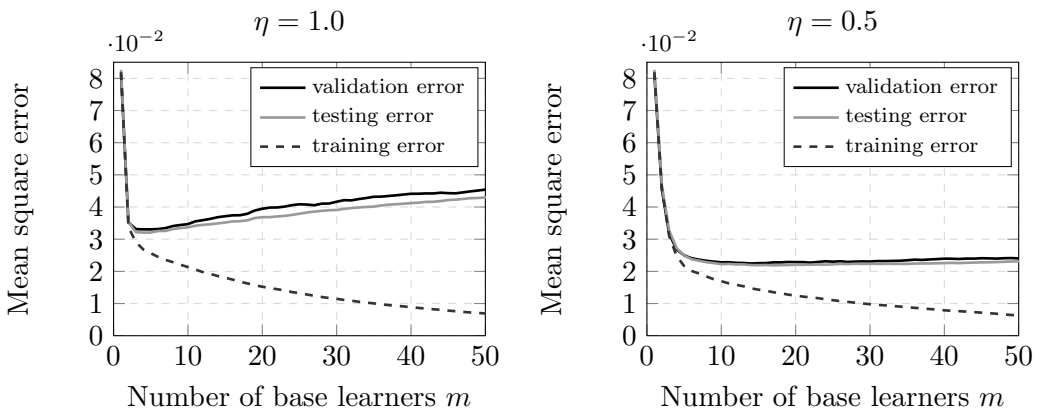


Figure 1. Progress of the validation, testing, and training error for two different values of η. Steganalysis of HUGO, fixed $d_{\text{sub}} = 100$.

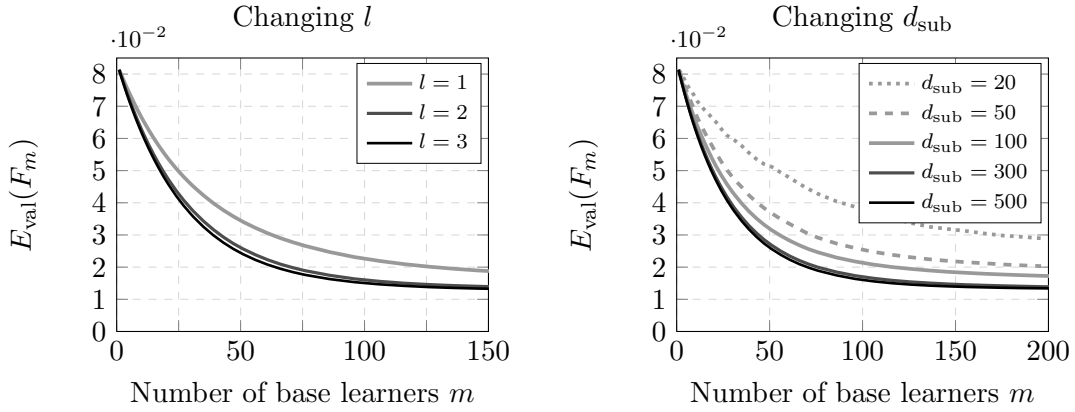


Figure 2. Influence of the parameters l and d_{sub} on detection accuracy of HUGO. Left: fixed $\eta = 0.02$, $d_{\text{sub}} = 500$, changing l; Right: fixed $\eta = 0.02$, $l = 2$, changing d_{sub}.

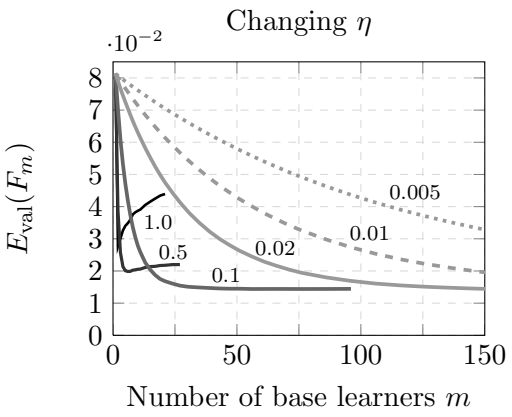
3.2 Tree depth l and subspace dimensionality d_{sub}

As shown in Figure 2 (left), switching from $l = 1$ (a single step function) to $l = 2$ (one recursive call, 3 OLS regressions) increases the estimation accuracy – because of the gained ability of local estimation. Further increase of l, however, does not seem to bring any additional improvement. Also, the complexity starts hurting due to the exponentially growing number of OLS regressions involved. Thus, we recommend to use the tree depth $l = 2$.

In Figure 2 (right), we fixed $l = 2$ and varied the random subspace dimensionality d_{sub}. With growing d_{sub}, the performance quickly improved and then saturated around $d_{\text{sub}} = 500$. For a fixed steganographic channel, one can always search for the best value of d_{sub} by wrapping the gradient boosting algorithm in a one-dimensional search loop and utilizing the validation feedback. Note that the computational complexity of a single base-learner training increases quadratically in d_{sub}.

3.3 Learning rate η

The learning rate η (shrinking) is an important regularization factor. As can already be seen in Figure 1, decreasing η from 1.0 to 0.5 significantly reduced the rate of overtraining. There is one more reason for keeping the value of η rather low, however, which is specific to our scenario and the way the base learners are formed. Because small values of η cause the regression function $F_m(\mathbf{x})$ to converge at a slower rate, we need to accumulate more base learners before the validation error saturates (we are also "equalizing" the



η	M	$E_{\text{val}}(F_M)$	Time (sec)
1.0	2	$2.77 \cdot 10^{-2}$	5
0.5	6	$1.98 \cdot 10^{-2}$	8
0.1	75	$1.44 \cdot 10^{-2}$	60
0.02	335	$1.35 \cdot 10^{-2}$	255
0.01	636	$1.33 \cdot 10^{-2}$	476
0.005	1,104	$1.32 \cdot 10^{-2}$	828

Figure 3. Influence of the learning rate η on the accuracy and speed of the training. Fixed parameters: $d_{\text{sub}} = 500$, $l = 2$.

weights of individual base learners). And since every base learner is trained on a different random subspace of the original feature space $\mathcal{F}$, this effectively allows us to collect more evidence and in turn obtain more accurate regression function.

Figure 3 illustrates this phenomenon. When $\eta = 1$ (no shrinking), after as few as $M = 3$ stages of gradient boosting the validation error starts increasing and the training stops. Three subspaces of dimension $d_{\text{sub}} = 500$, however, clearly cannot extract enough information from the whole $12,753$-dimensional space. But decreasing η necessitates more base learners, gradient boosting utilizes a larger portion of $\mathcal{F}$, and the final validation error thus decreases. There is an apparent trade-off between the accuracy and the computation time.

4. EXPERIMENTAL RESULTS

In this section, we experimentally demonstrate the performance of the proposed methodology on several qualitatively different algorithms operating in both the spatial and JPEG domains. As in the previous section, we used BOSSbase image database randomly divided into training and testing sets. The performance is now measured in terms of the mean square error achieved on the testing set. To increase the statistical significance of the results, all reported values are medians over ten experiments repeated with different training/testing database splits.

4.1 Spatial domain setup

First, we attacked HUGO [22] (with the same parameter setup as in Section 3). The second spatial domain algorithm we tested was the LSB replacement (LSBR) simulated with optimal binary coding. In both cases, stego images were generated with payloads uniformly distributed over $[0, 1]$ bits per pixel (bpp). The best prior art (quantitative) steganalysis of HUGO was achieved by training the kernelized Support Vector Regression (kSVR) in the SPAM feature space [21]. In case of LSBR, we compare to the best structural attack – the weighted-stego (WS) steganalysis [13].[†] For benchmarking purposes, we also include the performance of the optimized constant regression – a dummy estimator that always outputs the middle value (0.5 bpp).

For steganalysis of HUGO, we used the $12,753$-dimensional SRMQ1 [8]. In order to capture the parity, for steganalysis of LSBR we used the 50,856-dimensional parity-aware variant of SRMQ1 proposed in [9].

[†]Since WS estimates the relative number of changes, γ, its outputs were transformed into the actual payload values prior the MSE calculation through the binary entropy function, $\alpha = h(\gamma)$.

	Const.	kSVR	WS	Proposed	Improvement
HUGO	$8.33 \cdot 10^{-2}$	$4.84 \cdot 10^{-2}$		$1.49 \cdot 10^{-2}$	69.2%
LSBR	$8.33 \cdot 10^{-2}$	$1.48 \cdot 10^{-2}$	$1.15 \cdot 10^{-3}$	$1.01 \cdot 10^{-3}$	12.2%
BCHopt	$7.50 \cdot 10^{-3}$	$6.46 \cdot 10^{-3}$		$5.45 \cdot 10^{-3}$	15.6%
MME3	$7.50 \cdot 10^{-3}$	$2.35 \cdot 10^{-3}$		$1.31 \cdot 10^{-3}$	44.3%
nsF5	$7.50 \cdot 10^{-3}$	$3.10 \cdot 10^{-3}$		$1.91 \cdot 10^{-3}$	38.4%

Table 1. Quantitative steganalysis of five different steganographic methods. The last column shows the improvement over the prior art. The parameters of the proposed method were fixed to $d_{\text{sub}} = 500$, $\eta = 0.02$, and $l = 2$ in all cases.

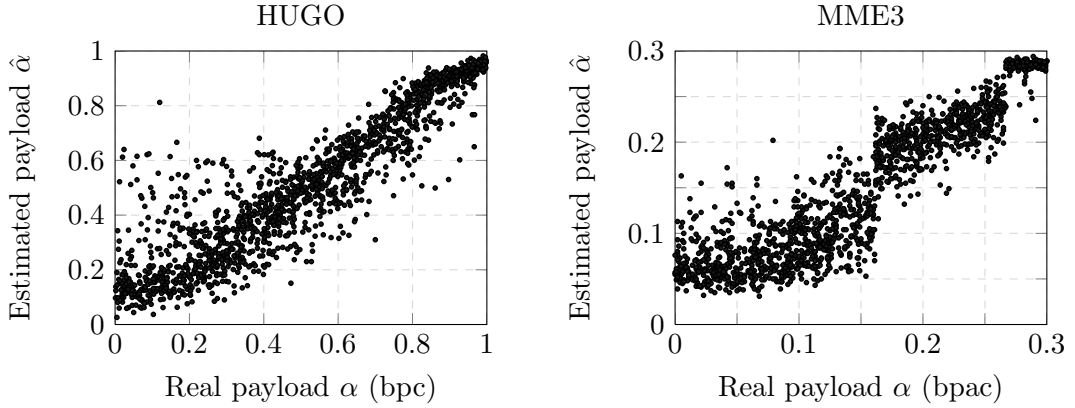


Figure 4. Quantitative steganalysis of HUGO and MME3 – estimated payload $\hat{\alpha}$ versus real payload α.

4.2 JPEG domain setup

For experiments in JPEG domain, we compressed all BOSSbase images using the Matlab's command `imwrite` using the quality factor 75. The tested algorithms were: BCHopt [24], MME3 [14], and nsF5 [10]. BCHopt and MME3 are side-informed schemes minimizing different distortion measures. We used the $22,510$-dimensional Cartesian-calibrated JPEG domain rich model (CCJRM) proposed in [17]. The feature extractors of all spatial- and JPEG-domain rich models are available at `http://dde.binghamton.edu/download`.

The prior distribution of relative payloads is now set to be uniform over $[0, 0.3]$ bits per nonzero AC DCT coefficient (bpac). We restricted ourselves to this range because of the limitations of the BCHopt implementation we used that didn't allow us to embed larger messages (due to the involved coding mechanism). Similarly to the spatial domain, we used two benchmark performances. The first one is the optimal value benchmark always outputting the middle value 0.15 bpac, and the second one is obtained by training the kSVR in the CCPEV feature space [15]. We are not aware of any structural attacks on any of these algorithms.

4.3 Results and discussion

Table 1 shows the mean square errors (MSE) calculated on the testing set for all five tested algorithms while in Figure 4 we show the results graphically for two selected algorithms. The proposed scheme clearly outperforms prior quantitative steganalysis across all tested algorithms.

The most significant improvement in estimation accuracy in comparison to the prior art was observed for HUGO – the testing error decreased by 69%. This is consistent with binary steganalysis [17, 19] where the detection improvement was also the biggest for HUGO due to the ability of the rich models to detect embedding changes in more complex content. Figure 4 (left) indicates that the accuracy of estimation increases with growing payload.

The best structural steganalysis of LSBR (WS) was improved by more than 10%, which is also not surprising and is consistent with the results of [9]. For many applications, however, WS may still be the better choice since it does not require any training phase and keeps the high accuracy.

In the JPEG domain, switching from kSVR with CCPEV features to the proposed gradient-boosting system utilizing rich models improved the testing error roughly by 33% on average. BCHopt is the most secure algorithm out of the three tested while the performance of MME3 and nsF5 is comparable, which is again consistent with prior work on binary steganalysis. Quantitative steganalysis of MME3 is more accurate than nsF5, probably due to the sharp "jumps" in distortion at certain payloads caused by suboptimal Hamming codes which aids the steganalyzer to assign the stego-image into the corresponding payload range (see also Figure 4).

In all experiments, we fixed $d_{\text{sub}} = 500$, $\eta = 0.02$, and $l = 2$. Even though the performance of the system could likely be improved by optimizing w.r.t. these choices for individual algorithms separately, we expect only marginal improvements. Also, the goal of this paper is mainly to demonstrate the proof of concept – Table 1 clearly indicates that rich models improve quantitative steganalysis.

4.4 General thoughts on quantitative steganalysis

We would like to conclude this section with a short discussion about quantitative steganalysis as a practical tool for forensic steganalysis. Certainly, any additional knowledge about the payload distribution $p(x)$ should help a forensic investigator to construct a better estimator. In this paper, we assumed the payloads to be uniformly distributed on a fixed interval, and both training and testing images were generated accordingly. The role of $p(x)$, however, reaches farther than that. Uniform distribution of payloads or, more generally, any continuous $p(x)$ on a fixed interval, *a priori assumes* that we are dealing with stego images, and that the probability of encountering a cover image is zero. This assumption, hardly valid in practice, is implicitly reflected in the behavior of the estimator around edges, see Figure 4 – since we know that the image is certainly stego (and because we minimize the squared loss), it pays off for the estimator to overshoot around zero.

We believe that any quantitative steganalysis for *practical* applications should consider not only the distribution of stego-payloads, but also the probability of seeing a cover/stego image. Perhaps, the best solution might be a compound system consisting of a binary classifier and an additional stage of a message-length estimation, provided the binary decision is "stego."

However, even in this layered detector, many other design decisions still need to be made that will all have a major effect on the overall performance. For example, what payload distribution should be used for training the binary classifier? Should we minimize the total classification error or rather adjust the decision thresholds for a fixed false alarm rate? Answers to these questions will obviously depend on the knowledge about the steganographic channel available to the Warden as well as on practical design requirements driven by costs and available resources. For the quantitative part, one could rightfully challenge the squared loss as being overly sensitive to outliers and replace it with a more robust loss, such as the Huber loss function. It would even make sense to use a loss defined as the *relative* estimation error, which would put more emphasis on estimating small payloads. Undoubtedly, these are all valid questions that deserve further research, especially when designing a steganalysis system for a practical application rather than for the purpose of a pure academic research.

5. SUMMARY

We propose a novel machine-learning framework for quantitative steganalysis in high-dimensional feature spaces. The framework combines the ideas of [23], where the authors formulated quantitative steganalysis as a regression problem in an appropriate feature space, and [19], where an ensemble framework allowed the employment of high-dimensional feature spaces for binary steganalysis. The proposed system assembles a series of regressors via the process of gradient boosting [11]. The individual base learners are variants of regression trees with a modified splitting criterion to reflect the specific nature of feature spaces in steganalysis. Our experiments confirmed that, when employing rich models, the improvement in message

length estimation roughly mimics the improvement in binary classification accuracy across five different steganographic schemes embedding in both the spatial and JPEG domains.

The proposed system should be seen as a practical way of utilizing rich models for quantitative steganalysis – if we had an extra computational power, we could simply train a single (non-linear) regressor directly in the rich model. Also, it should be noted that even though the proposed system is more scalable than, for example, the kernelized SVR, it would still run into difficulties with extremely large amount of training samples (>50k). In this case, one would likely need to switch to an online regression, similar to the steps taken in [20], where the authors managed to operate on data sets containing 1.6M images.

REFERENCES

1. P. Bas, T. Filler, and T. Pevný. Break our steganographic system – the ins and outs of organizing BOSS. In T. Filler, T. Pevný, A. Ker, and S. Craver, editors, *Information Hiding, 13th International Workshop*, volume 6958 of Lecture Notes in Computer Science, pages 59–70, Prague, Czech Republic, May 18–20, 2011.
2. L. Breiman. Stacked regressions. *Machine Learning*, 24:49–64, 1996.
3. L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.
4. J. B. Copas. Regression, prediction and shrinkage. *Journal of the Royal Statistical Society. Series B (Methodological)*, 45(3):311–354, 1983.
5. S. Dumitrescu, X. Wu, and Z. Wang. Detection of LSB steganography via Sample Pairs Analysis. In F.A.P. Petitcolas, editor, *Information Hiding, 5th International Workshop*, volume 2578 of Lecture Notes in Computer Science, pages 355–372, Noordwijkerhout, The Netherlands, October 7–9, 2002. Springer-Verlag, New York.
6. J. Fridrich and M. Goljan. On estimation of secret message length in LSB steganography in spatial domain. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VI*, volume 5306, pages 23–34, San Jose, CA, January 19–22, 2004.
7. J. Fridrich, M. Goljan, and R. Du. Detecting LSB steganography in color and gray-scale images. *IEEE Multimedia, Special Issue on Security*, 8(4):22–28, October–December 2001.
8. J. Fridrich and J. Kodovský. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, June 2012.
9. J. Fridrich and J. Kodovský. Steganalysis of LSB replacement using parity-aware features. In *Information Hiding, 14th International Workshop*, Lecture Notes in Computer Science, Berkeley, CA, May 15–18, 2012. To appear.
10. 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.
11. J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.
12. 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, pages 71–84, Prague, Czech Republic, May 18–20, 2011.
13. A.D. Ker and R. Böhme. Revisiting weighted stego-image 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 5 1–17, San Jose, CA, January 27–31, 2008.
14. Y. Kim, Z. Duric, and D. Richards. Modified matrix encoding technique for minimal distortion 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 314–327, Alexandria, VA, July 10–12, 2006. Springer-Verlag, New York.

15. 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.
16. 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.
17. J. Kodovský and J. Fridrich. Steganalysis of JPEG images using rich models. In A. Alattar, N. D. Memon, and E. J. Delp, editors, *Proceedings SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics of Multimedia XIV*, volume 8303, pages 0A 1–13, San Francisco, CA, January 23–26, 2012.
18. J. Kodovský, J. Fridrich, and V. Holub. On dangers of overtraining steganography to incomplete cover model. In J. Dittmann, S. Craver, and C. Heitzenrater, editors, *Proceedings of the 13th ACM Multimedia & Security Workshop*, pages 69–76, Niagara Falls, NY, September 29–30, 2011.
19. J. Kodovský, J. Fridrich, and V. Holub. Ensemble classifiers for steganalysis of digital media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, April 2012.
20. I. Lubenko and A. D. Ker. Going from small to large data sets in steganalysis. In A. Alattar, N. D. Memon, and E. J. Delp, editors, *Proceedings SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics of Multimedia XIV*, volume 8303, pages OM 1–10, San Francisco, CA, January 23–26, 2012.
21. T. Pevný, P. Bas, and J. Fridrich. Steganalysis by subtractive pixel adjacency matrix. In J. Dittmann, S. Craver, and J. Fridrich, editors, *Proceedings of the 11th ACM Multimedia & Security Workshop*, pages 75–84, Princeton, NJ, September 7–8, 2009.
22. 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.
23. T. Pevný, J. Fridrich, and A.D. Ker. From blind to quantitative steganalysis. In N.D. Memon, E.J. Delp, P.W. Wong, and J. Dittmann, editors, *Proceedings SPIE, Electronic Imaging, Security and Forensics of Multimedia XI*, volume 7254, pages 0C 1–14, San Jose, CA, January 18–21, 2009.
24. V. Sachnev, H. J. Kim, and R. Zhang. Less detectable JPEG steganography method based on heuristic optimization and BCH syndrome coding. In J. Dittmann, S. Craver, and J. Fridrich, editors, *Proceedings of the 11th ACM Multimedia & Security Workshop*, pages 131–140, Princeton, NJ, September 7–8, 2009.
25. B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. The MIT Press, 2001.
26. D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.