Rich Models for Steganalysis of Digital Images

Jessica Fridrich, *Member, IEEE* and Jan Kodovský

***Abstract*—We describe a novel general strategy for building steganography detectors for digital images. The process starts with assembling a rich model of the noise component as a union of many diverse submodels formed by joint distributions of neighboring samples from quantized image noise residuals obtained using linear and non-linear high-pass filters. In contrast to previous approaches, we make the model assembly a part of the training process driven by samples drawn from the corresponding cover- and stego-sources. Ensemble classifiers are used to assemble the model as well as the final steganalyzer due to their low computational complexity and ability to efficiently work with high-dimensional feature spaces and large training sets. We demonstrate the proposed framework on three steganographic algorithms designed to hide messages in images represented in the spatial domain: HUGO, edge-adaptive algorithm by Luo et al. [32], and optimally-coded ternary ± 1 embedding. For each algorithm, we apply a simple submodel-selection technique to increase the detection accuracy per model dimensionality and show how the detection saturates with increasing complexity of the rich model. By observing the differences between how different submodels engage in detection, an interesting interplay between the embedding and detection is revealed. Steganalysis built around rich image models combined with ensemble classifiers is a promising direction towards automatizing steganalysis for a wide spectrum of steganographic schemes.**

I. Introduction

Modern feature-based steganalysis starts with adopting an image model (a low-dimensional representation) within which steganalyzers are built using machine learning tools. The model is usually determined not only by the characteristics of the cover source but also by the effects of embedding [4], [6], [14], [18], [21], [33], [35], [38]. For example, the SPAM feature vector [33], which was seemingly proposed from a pure cover model, is in reality, too, driven by a specific case of steganography. The choice of the order of the Markov process as well as the threshold T and the local predictor were "tuned" by observing the detection performance on ± 1 embedding. Had the authors used HUGO [34] instead of ± 1 embedding, the SPAM model might have looked quite different. In particular, in light of the recent work [17], [16], [19], the predictor would have probably employed higher-order pixel differences.

In this paper, we propose a general methodology for steganalysis of digital images based on the concept of a rich model consisting of a large number of diverse submodels. The submodels consider various types of relationships among neighboring samples of noise residuals obtained by linear and non-linear filters with compact supports. The rich model is assembled as part of the training process and is driven by the available examples of cover and stego images. Our design was inspired by the recent methods developed for attacking HUGO [17], [16], [19]. The key element of these attacks is a complex model consisting of *multiple submodels*, each capturing slightly different embedding artifacts. Here, we bring this philosophy to the next level by designing the submodels in a more systematic and exhaustive manner and we let the training data select such a combination of submodels that achieves a good trade-off between model dimensionality and detection accuracy.

Since our approach requires fast machine learning, we use the ensemble classifier as described in [30], [28] due to its low computational complexity and ability to efficiently work with high dimensional features and large training data sets. The rich model is assembled by selecting each submodel based on its detection error estimate in the form of the out-of-bag estimate calculated from the training set. The final steganalyzer for each stego method is constructed again as an ensemble classifier.

Besides the obvious goal to improve upon the state-of-the-art in steganalysis, the proposed approach can be viewed as a step towards automatizing steganalysis to facilitate fast development of accurate detectors for new steganographic schemes. We demonstrate the proposed framework on three steganographic algorithms operating on a fixed cover source. The edge-adaptive algorithm by Luo et al. [32] was included intentionally as an example of a stegosystem that, according to the best knowledge of the authors, has not yet been successfully attacked.

Another promising aspect of rich models is their potential to provide a good general-purpose model for various applications in forensics and in universal blind steganalysis. While the latter may rightfully seem rather out-of-reach due to the fact that steganography can be designed to minimize the disturbance in a fixed model space using the Feature-Correction Method (FCM) [8], [26] or the framework described in [12], [10], model preservation will

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 would like to thank Vojtěch Holub for useful discussions.

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

likely become increasingly more difficult for rich (high-dimensional and diverse) models. Indeed, as shown in Section V, when sufficiently many diverse submodels built from differences between neighboring pixels are combined in the rich model, HUGO becomes quite detectable despite the fact that it was designed to minimize distortion to high-dimensional multivariate statistics computed from the same pixel differences.

The paper is structured as follows. In Section II, we describe the individual submodels built as symmetrized joint probability distributions of adjacent residual samples. The steganalyzer and the experimental setup used in this paper are detailed in Section III. The methodology for assembling the rich model from a sample of cover and stego images while considering the performance–dimensionality trade-off appears in Section IV. The three tested stego methods are described in Section V together with two investigative experiments aimed at analyzing the performance of individual submodels and how it is affected by quantization and steganographic payload. Section VI contains the results of the main experiment in which the full proposed framework is applied to three steganographic methods. Finally, the paper is concluded in Section VII where we elaborate on how the proposed strategy affects future development of steganography and discuss potential applications of rich models outside the field of steganalysis.

Everywhere in this article, lower-case boldface symbols are used for vectors and capital-case boldface symbols for matrices and higher-dimensional arrays. The symbols $\mathbf{X} = (X_{ij}) \in \{0, \dots, 255\}^{n_1 \times n_2}$ and $\bar{\mathbf{X}} = (\bar{X}_{ij})$ always represent pixel values of an 8-bit grayscale cover image with $n = n_1 \times n_2$ pixels and its corresponding stego image. By slightly abusing the language, for compactness we will sometimes say "pixel X_{ij}" meaning pixel located at (i, j) whose grayscale is X_{ij}. A model representation of an image using a feature will always be denoted with the same lower-case letter. For example, image $\mathbf{X}$ is represented with $\mathbf{x}$ and $\bar{\mathbf{X}}$ with $\bar{\mathbf{x}}$. For any vector $\mathbf{x}$ with index set $\mathcal{I}$, $\mathbf{x}_{\mathcal{J}}$, $\mathcal{J} \subset \mathcal{I}$, stands for the vector $\mathbf{x}$ from which all x_i, $i \notin \mathcal{J}$, were removed. We use the symbols $\mathbb{R}$ and $\mathbb{N}$ to represent the set of all real numbers and integers. For any $x \in \mathbb{R}$, the largest integer smaller than or equal to x is $\lfloor x \rfloor$, while the operation of rounding to an integer is denoted $\text{round}(x)$. The truncation function with threshold $T > 0$ is defined for any $x \in \mathbb{R}$ as $\text{trunc}_T(x) = x$ for $x \in [-T, T]$ and $\text{trunc}_T(x) = T\text{sign}(x)$ otherwise. For a finite set $\mathcal{X}$, $|\mathcal{X}|$ denotes the number of its elements.

II. Rich model of noise residual

A good source model is crucial not only for steganography but also, e.g., for source coding and forensic analysis. Indeed, image representations originally designed for steganalysis have found applications in digital forensic to solve difficult problems of a very different nature [24], [7]. However, digital images acquired using a sensor constitute a quite complex source. This is not only due to the richness of natural scenes but also due to the intricate network of dependencies among pixels introduced at the acquisition and during in-camera processing. Model building is further complicated by the enormous diversity among cameras as manufacturers implement more sophisticated processing algorithms as well as specialized hardware components.

Since the focus of this paper is on spatial-domain steganography, our rich model will be constructed in the spatial domain because the best detection is usually achieved by building the model directly in the domain where the embedding changes are localized and thus most pronounced. Since steganography by cover modification makes only small changes to the pixels, we model only the noise component (noise residual) of images rather than their content. This philosophy has been adopted by steganalysts early on (Chapter 2.4 in [22]) and then perfected through a long series of papers examples of which are [2], [1], [9], [18], [33], [38].

A. Submodels

Our overall goal is to capture a large number of different types of dependencies among neighboring pixels to give the model the ability to detect a wide spectrum of embedding algorithms. However, enlarging a *single* model is unlikely to produce good results as the enlarged model will have too many underpopulated bins (e.g., think of the second-order SPAM model with a large truncation threshold T employed by HUGO [34]). Instead, we form the model by merging *many smaller submodels* avoiding thus the problem with underpopulated bins.

1) Computing residuals: The submodels are formed from noise residuals, $\mathbf{R} = (R_{ij}) \in \mathbb{R}^{n_1 \times n_2}$, computed using high-pass filters of the following form:

$$R_{ij} = \hat{X}_{ij}(\mathcal{N}_{ij}) - cX_{ij}, \tag{1}$$

where $c \in \mathbb{N}$ is the residual order, $\mathcal{N}_{ij}$ is a local neighborhood of pixel X_{ij}, $X_{ij} \notin \mathcal{N}_{ij}$, and $\hat{X}_{ij}(.)$ is a *predictor* of cX_{ij} defined on $\mathcal{N}_{ij}$. The set $\{X_{ij} + \mathcal{N}_{ij}\}$ is called the support of the residual. The advantage of modeling the residual instead of the pixel values is that the image content is largely suppressed in $\mathbf{R}$, which has a much narrower dynamic range allowing thus a more compact and robust statistical description. Many steganalysis features were formed in this manner in the past, e.g., [9], [18], [33], [38].

2) Truncation and quantization: Each submodel is formed from a quantized and truncated version of the residual:

$$R_{ij} \leftarrow \text{trunc}_T\left(\text{round}\left(\frac{R_{ij}}{q}\right)\right), \tag{2}$$

where $q > 0$ is a quantization step. The purpose of truncation is to curb the residual's dynamic range to allow their description using co-occurrence matrices with a small T. The quantization makes the residual more sensitive to embedding changes at spatial discontinuities in the image (at edges and textures).

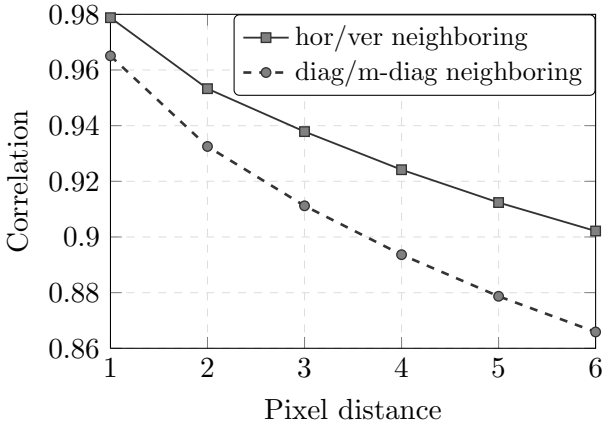


Figure 1. Correlation between pixels based on their distance. The distance of diagonally neighboring pixels is in the multiples of the diagonal of two neighboring pixels. The results were averaged over 100 randomly selected images from BOSSbase ver. 0.92.

3) Co-occurrences: The construction of each submodel continues with computing one or more co-occurrence matrices of neighboring samples from the truncated and quantized residual (2). Forming models in this manner is well-established in the steganalysis literature. The truncation of noise residuals and formation of joint or Markov transition probability matrices as features has appeared for the first time in [38] and in [36]. The key question is how to choose the model parameters – the threshold T, the co-occurrence order, and the spatial positions of the neighboring residual samples. To this end, we analyzed our cover source, which is the BOSSbase ver. 0.92 database [13], and computed the average correlation between neighboring pixels in the horizontal/vertical and diagonal/minor diagonal directions (see Fig. 1). The correlations fall off gradually with increasing distance between pixels and they do so faster for diagonally-neighboring pixels. Thus, we form co-occurrences of pixels only along the horizontal and vertical directions and avoid using groups with diagonally-neighboring pixels.[1] We chose four-dimensional co-occurrences because co-occurrences of larger dimensions had numerous underpopulated bins, which compromised their statistical significance. For a fixed dimension, better results are generally obtained by using a lower value of T and including other types of residuals to increase the model diversity. To compensate for loss of information due to truncating all residual values larger than T, for each residual type we consider several submodels with different values of q, allowing thus our model to "see" dependencies among residual samples whose values lie beyond the threshold.

In summary, our submodels will be constructed from horizontal and vertical co-occurrences of four consecutive residual samples processed using (2) with $T = 2$. Formally, each co-occurrence matrix $\mathbf{C}$ is a four-dimensional array indexed with $\mathbf{d} = (d_1, d_2, d_3, d_4) \in \mathcal{T}_4 \triangleq \{-T, \dots, T\}^4$, which gives the array $(2T+1)^4 = 625$ elements. The $\mathbf{d}$th element of the horizontal co-occurrence for residual $\mathbf{R} = (R_{ij})$ is formally defined as the (normalized) number of groups of four neighboring residual samples with values equal to d_1, d_2, d_3, d_4:

$$\mathbf{C}^{(\mathrm{h})}_{\mathbf{d}} = \frac{1}{Z}\Big|\{(R_{ij}, R_{i,j+1}, R_{i,j+2}, R_{i,j+3}) | R_{i,j+k-1} = d_k, k = 1, \dots, 4\}\Big|, \quad (3)$$

where Z is the normalization factor ensuring that $\sum_{\mathbf{d}\in\mathcal{T}_4} \mathbf{C}^{(\mathrm{h})}_{\mathbf{d}} = 1$. The vertical co-occurrence, $\mathbf{C}^{(\mathrm{v})}$, is defined analogically.

Having fixed T and the co-occurrence order, determining the rest of the rich model involves selecting the local predictors $\hat{X}_{ij}$ for the residuals and the quantization step(s) q, all explained in the following sections.

B. Description of all residuals

All residuals used in this paper are graphically shown in Fig. 2. They are built as locally-supported linear filters whose outputs are possibly combined using minimum and maximum operators to increase their diversity. For better insight, think of each filter in terms of its predictor. For example, in the first-order residual $R_{ij} = X_{i,j+1} - X_{ij}$ the central pixel X_{ij} is predicted as its immediate neighbor, $\hat{X}_{ij} = X_{i,j+1}$, while the predictor in the second-order residual $R_{ij} = X_{i,j-1} + X_{i,j+1} - 2X_{ij}$ assumes that the image is locally linear in the horizontal direction, $2\hat{X}_{ij} = (X_{i,j+1} + X_{i,j-1})$. Higher-order differences as well as differences involving a larger neighborhood correspond to more complicated assumptions made by the predictor, such as locally-quadratic behavior or linearity in both dimensions. Additional motivation for the choice of our filters appears in Section II-B1.

The central pixel X_{ij} at which the residual (1) is evaluated is always marked with a black dot and accompanied with an integer – the value c from (1). If the chart contains only one type of symbol (besides the black dot), we say that the residual is of type 'spam' (1a, 2a, 3a, S3a, E3a, S5a, E5a) by their similarity to the SPAM feature vector [33].

If there are two or more different symbols other than the black dot, we call it type 'minmax'. In type 'spam', the residual is computed as a linear high-pass filter of neighboring pixels with the corresponding coefficients. For example, 2a stands for the second-order $R_{ij} = X_{i,j-1} + X_{i,j+1} - 2X_{ij}$ and 1a for the first-order $R_{ij} = X_{i,j+1} - X_{ij}$ residuals. In contrast, 'minmax' residuals use two or more linear filters, each filter corresponding to one symbol type, and the final residual is obtained by taking the minimum (or maximum) of the filters' outputs. Thus, there will be two minmax residuals – one for the operation of 'min' and one for 'max'. For example, 2b is obtained as $R_{ij} = \min\{X_{i,j-1} + X_{i,j+1} - 2X_{ij}, X_{i-1,j} + X_{i+1,j} - 2X_{ij}\}$ while

[1] A few sample tests confirmed that co-occurrences formed from groups in which pixels do not lie on a straight line have a substantially weaker detection performance across various stego methods and payloads.

1g is $R_{ij} = \min\{X_{i-1,j-1} - X_{ij}, X_{i-1,j} - X_{ij}, X_{i-1,j+1} - X_{ij}, X_{i,j+1} - X_{ij}\}$, etc. The 'min' and 'max' operators introduce non-linearity into the residuals and desirably increase the model diversity. Both operations also make the distribution of the residual samples non-symmetrical, thickening one tail of the distribution of R_{ij} and thinning out the other.

The number of filters, f, is the first digit attached to the end of the residual name. The third-order residuals are computed just like the first-order residuals by replacing, e.g., $X_{i,j+1} - X_{ij}$ with $-X_{i,j+2} + 3X_{i,j+1} - 3X_{ij} + X_{i,j-1}$. The differences along other directions are obtained analogically.

1) Residual classes: As the figure shows, the residuals are divided into six classes depending on the central pixel predictor they are built from. The classes are given the following descriptive names: 1st, 2nd, 3rd, SQUARE, EDGE3x3, and EDGE5x5. The predictors in class '1st' estimate the pixel as the value of its neighbor, while those from class '2nd' ('3rd') incorporate a locally linear (quadratic) model. Such predictors are more accurate in regions with a strong gradient/curvature (e.g., around edges and in complex textures). The class 'SQUARE' makes use of more pixels for the prediction. The 3×3 square kernel S3a has been used in steganalysis before [23] and it also coincides with the best (in the least-square sense) shift-invariant linear pixel predictor on the 3×3 neighborhood for cover images from BOSSbase. The class 'EDGE3x3' predictors, derived from this kernel, were included to provide better estimates at spatial discontinuities (edges). The larger 5×5 predictor in S5a was obtained as a result of optimizing the coefficients of a circularly-symmetrical 5×5 kernel using the Nelder–Mead algorithm to minimize the detection error for the embedding algorithm HUGO [29]. While this (only) predictor was inspired by a specific embedding algorithm, it works very well against other algorithms we tested in this paper. The 'EDGE5x5' residuals E5a–E5d (not shown in Fig. 2) are built from S5a in an analogical manner as E3a–E3d are built from S3a.

2) Residual symmetries: Each residual exhibits symmetries that will later allow us to reduce the number of submodels and make them better populated. If the residual does not change after computing it from the image rotated by 90 degrees, we say that it is non-directional, otherwise it is directional. For instance, 1a, 1b, 2a, 2e, E3c are directional while 1e, 2b, 2c, S3a, E3d are non-directional. Two co-occurrence matrices (3) are computed for each residual – one for the horizontal and one for the vertical scan. We call a residual hv-symmetrical if its horizontal and vertical co-occurrences can be added to form a single matrix (submodel) based on the argument that the statistics of natural images do not change after rotating the image by 90 degrees. Obviously, all non-directional residuals are hv-symmetrical, but many directional residuals are hv-symmetrical as well (e.g, 1c, 1h, 2e, E3b, E3d). In contrast, 1a, 1g, 2a, 2d, E3c are not hv-symmetrical. In general, an hv-symmetrical residual will thus produce a single co-occurrence matrix (sum of both horizontal and vertical matrices), while hv-nonsymmetrical ones will produce two matrices – one for the horizontal and one for the vertical direction. We include this fact into the residual name by appending either 'h' or 'v' to the end. No symbol is appended to hv-symmetrical residuals.

We also define a symmetry index σ for each residual as the number of different residuals that can be obtained by rotating and possibly mirrorring the image prior to computing it. To give an example, 2c, 1b, 1c, and 1g have symmetry indices equal to 1, 2, 4, and 8, respectively. The symmetry index is part of the residual name and it always follows the number of filters, f.

To make the co-occurrence bins more populated, and thus increase their statistical robustness, and to lower their dimensionality, for hv-nonsymmetrical residuals we add all σ co-occurrences. For hv-symmetrical residuals, since we add both the horizontal and vertical co-occurrences, we end up adding 2σ matrices. For example, 1f has symmetry index 4 and because it is hv-symmetrical we can form one horizontal and one vertical co-occurrence for each of the four rotations of the filter, adding together 8 matrices. As another example, 1g has symmetry index 8 and is hv-nonsymmetrical, which means we end up adding 8 matrices.

3) Syntax: The syntax of names used in Fig. 2 follows this convention:

$$name = \{type\}\{f\}\{\sigma\}\{scan\}, \tag{4}$$

where $type \in \{\text{spam}, \text{minmax}\}$, f is the number of filters, σ is the symmetry index, and the last symbol $scan \in \{\emptyset, h, v\}$ may be missing (for hv-symmetrical residuals) or it is either h or v, depending on the co-occurrence scan that should be used with the residual.

In summary, the class '1st' contains 22 different co-occurrence matrices – two for 1a, 1c, 1e, 1f, 1h, and four for 1b, 1d, 1g. The same number is obtained for class '3rd', while '2nd' contains 12 matrices – two for 2a, 2b, 2c, 2e, and four for 2d. There are two matrices in 'SQUARE', S3a, S5a, and ten in 'EDGE3x3' and in 'EDGE5x5' (two for E3a, E3b, and E3d, and four for E3c), giving the total of $22 + 12 + 22 + 2 + 10 + 10 = 78$ matrices, each with 625 elements. These matrices are used to form the final submodels by symmetrization explained next.

C. Co-occurrence symmetrization

The individual submodels of the rich image model will be obtained from the 78 co-occurrence matrices computed above by leveraging symmetries of natural images. The symmetries are in fact quite important as they allow us to increase the statistical robustness of the model while decreasing its dimensionality, making it thus more compact and improving the performance-to-dimensionality ratio. We use the sign-symmetry[2] as well as the directional symmetry of images. The symmetrization depends on

[2]Sign-symmetry means that taking a negative of an image does not change its statistical properties.

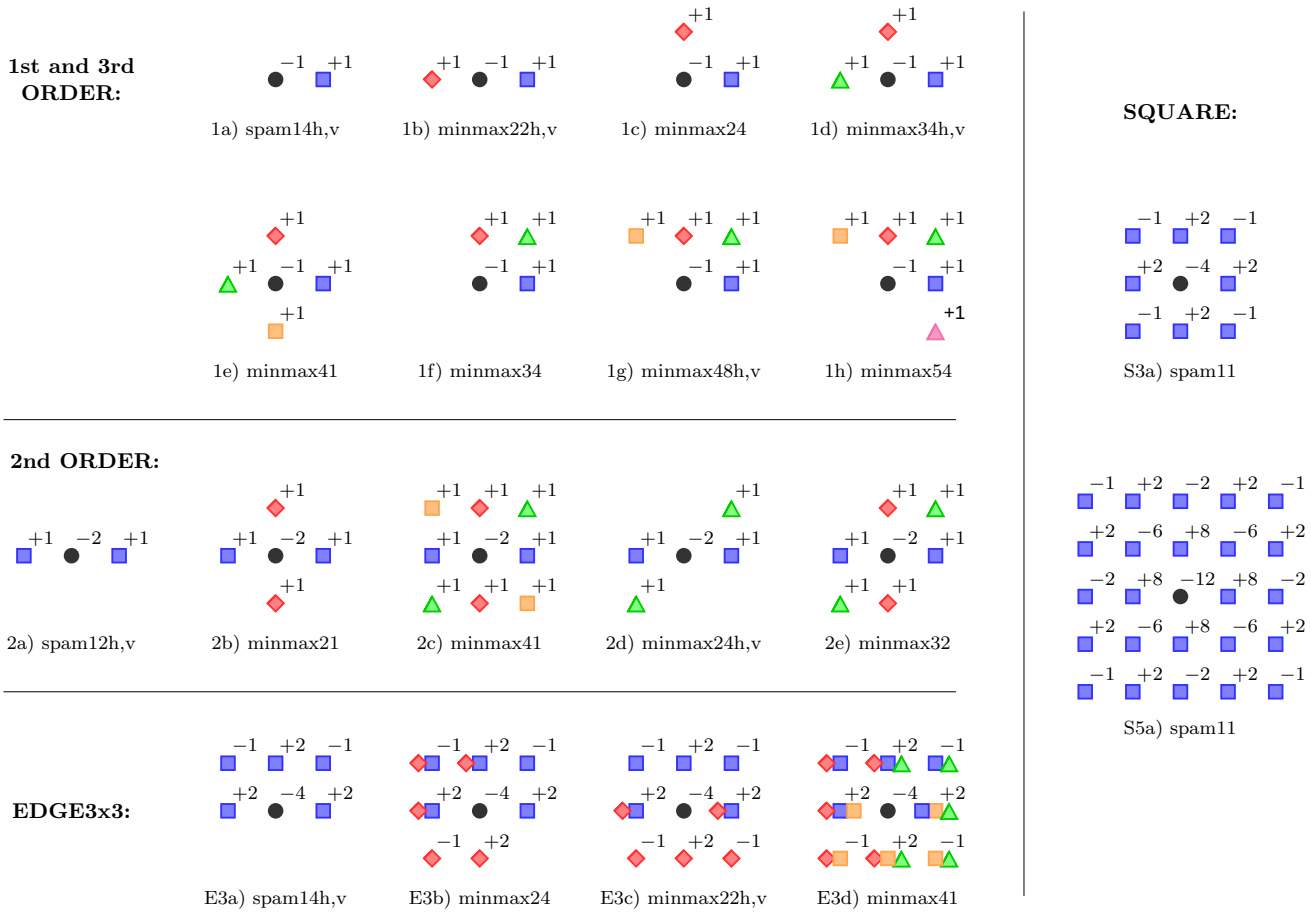


Figure 2. Definitions of all residuals. The residuals 3a – 3h are defined similar to the first-order residuals, while E5a – E5d are similar to E3a – E3d defined using the corresponding part of the 5×5 kernel displayed in S5a. See the text for more details.

the residual type. All 'spam' residuals are symmetrized sequentially by applying the following two rules for all $\mathbf{d} = (d_1, d_2, d_3, d_4) \in \mathcal{T}_4$:

$$\bar{\mathbf{C}}_{\mathbf{d}} \leftarrow \mathbf{C}_{\mathbf{d}} + \mathbf{C}_{-\mathbf{d}}, \tag{5}$$

$$\bar{\bar{\mathbf{C}}}_{\mathbf{d}} \leftarrow \bar{\mathbf{C}}_{\mathbf{d}} + \bar{\mathbf{C}}_{\overleftarrow{\mathbf{d}}}, \tag{6}$$

where $\overleftarrow{\mathbf{d}} = (d_4, d_3, d_2, d_1)$ and $-\mathbf{d} = (-d_1, -d_2, -d_3, -d_4)$. After eliminating duplicates from $\bar{\bar{\mathbf{C}}}$ (which had originally 625 elements), only 169 unique elements remain.

The 'minmax' residuals of natural images also possess the directional symmetry but not the sign symmetry. On the other hand, since $\min(\mathcal{X}) = -\max(-\mathcal{X})$ for any finite set $\mathcal{X} \subset \mathbb{R}$, we use the following two rules for their symmetrization:

$$\bar{\mathbf{C}}_{\mathbf{d}} \leftarrow \mathbf{C}^{(\min)}_{\mathbf{d}} + \mathbf{C}^{(\max)}_{-\mathbf{d}} \tag{7}$$

$$\bar{\bar{\mathbf{C}}}_{\mathbf{d}} \leftarrow \bar{\mathbf{C}}_{\mathbf{d}} + \bar{\mathbf{C}}_{\overleftarrow{\mathbf{d}}}, \tag{8}$$

where $\mathbf{C}^{(\min)}$ and $\mathbf{C}^{(\max)}$ are the 'min' and 'max' co-occurrence matrices computed from the same residual. The dimensionality is thus reduced from 2×625 to 325.

After symmetrization, the total number of submodels decreases from 78 to only 45 as the symmetrization reduces two co-occurrences, one for 'min' and one for 'max', into a single matrix. The number of co-occurrences for the 'spam' type stays the same (only their dimensionality changes). For example, for the class '1st', we will have 12 submodels – one symmetrized spam14h and one spam14v, one minmax22h, one minmax22v, one minmax24, minmax34h, minmax34v, minmax41, minmax34, minmax48h, minmax48v, and one minmax54. There will be 12 submodels from '3rd', seven from '2nd', two from 'SQUARE', and six from each edge class. In total, there are 12 submodels of dimension 169 from 12 'spam' type residuals and 33 of dimension 325 from type minmax. Thus, when all submodels are put together, their combined dimensionality is $12 \times 169 + 33 \times 325 = 12,753$.

We remark that it is possible that the symmetrization might prevent us from detecting steganographic methods that disturb the above symmetries (think of symmetrizing the histogram for Jsteg [37]). Such embedding methods are, however, fundamentally flawed (and easy to detect) as one can likely build accurate quantitative targeted attacks leveraging the symmetry violations.

D. Quantization

Finally, we specify how to select the quantization step q. As mentioned already at the end of Section II-A, it is beneficial to include several versions of submodels with different values of q because residuals obtained with a larger q can better detect embedding changes in textured areas and around edges. Based on sample experiments with different algorithms and submodels, we determined that the best performance of each submodel is always achieved when $q \in [c, 2c]$, where c is the residual order. Thus, we included in the rich cover model all submodels with residuals quantized with q:

$$q \in \begin{cases} \{c, 1.5c, 2c\} & \text{for } c > 1 \\ \{1, 2\} & \text{for } c = 1. \end{cases} \tag{9}$$

The case with $c = 1$ in (9) is different from the rest because quantizing a residual with $c = 1$ and $q = 1.5$ with $T = 2$ leads to exactly the same result as when quantizing with $q = 2$. Thus, each submodel will be built in two versions for residuals in class '1st' and in three versions for the remaining residuals.

The authors acknowledge that the individual performance of each submodel can likely be improved by replacing the simple scalar quantizer with an optimized design. The possibilities worth investigating are non-uniform scalar quantizers and vector quantizers built directly in the four-dimensional space. Due to lack of space for such an investigation in this paper, the authors postpone researching these possibilities to their future work.

E. Discussion

The residuals shown in Fig. 2 were selected using the principle of simplicity and are by no means to be meant as the ultimate result as there certainly exist numerous other possibilities. We view the model building as an open-ended process because, quite likely, there exist other predictors that will further improve the detection after adding them to the proposed model. Having said this, we observed a "saturation" of performance in the sense that further enrichment of the model with other types of predictors lead to an insignificant improvement in detection accuracy for all tested algorithms (see Section V).

In the future, the authors contemplate *learning* the best predictors from the database of cover and stego images, replacing thus the hand design described above. We also note that submodels obtained from residuals computed using denoising filters almost always lead to poor steganalysis results because denoising filters typically put substantial weight to the central pixel being denoised, which leads to a biased predictor $\hat{X}_{ij}$, and, when one computes the residual using (1), the stego signal becomes undesirably suppressed.

III. Ensemble classifier

Our strategy for constructing steganography detectors involves a training phase in which we not only build the steganalyzer but also form the rich image model from available cover and stego images for a given steganographic algorithm. Thus, most experiments will be built from the following experimental unit that starts by randomly splitting the available image database into a training and testing set $\mathcal{X}^{\text{trn}}$ and $\mathcal{X}^{\text{tst}}$, each with N^{trn} and N^{tst} cover images and the same number of their corresponding stego images. Using only $\mathcal{X}^{\text{trn}}$, we assemble the rich model, construct the steganalyzer, and then test it on $\mathcal{X}^{\text{tst}}$. All images will be represented using d-dimensional features, $\mathbf{x} \in \mathbb{R}^d$, where d could stand for the dimension of a given submodel or their arbitrary union.

Everywhere in this paper, we use the ensemble classifier as described in [27], [28]. Since these two references obtain a detailed description of the tool and since this paper is not about ensemble classification, we repeat only the most essential details here, referring the reader to the original publications for more details.

The ensemble classifier is essentially a random forrest consisting of L binary classifiers called base learners, B_l, $l = 1, \dots, L$, each trained on a different d_{sub}-dimensional subspace of the feature space selected uniformly at random. Each random subspace will be described using an index set $\mathcal{D}_l \subset \{1, \dots, d\}$, $|\mathcal{D}_l| = d_{\text{sub}}$. The ensemble reaches its decision by fusing all L decisions of individual base learners using majority voting.

Following the investigation reported in the original publications, we use Fisher Linear Discriminants (FLDs) as base learners due to their simple and fast training and due to the fact that in steganalysis we are unlikely to encounter a small feature set responsible for majority of detection accuracy. Denoting the cover and stego features from the training set as $\mathbf{x}^{(m)}$ and $\bar{\mathbf{x}}^{(m)}$, $m = 1, \dots, N^{\text{trn}}$, respectively, each base learner B_l is trained as the FLD on the set $\mathcal{X}_l = \{\mathbf{x}^{(m)}_{\mathcal{D}_l}, \bar{\mathbf{x}}^{(m)}_{\mathcal{D}_l}\}_{m \in \mathcal{N}^{\text{b}}_l}$, where $\mathcal{N}^{\text{b}}_l$ is a bootstrap sample of $\{1, \dots, N^{\text{trn}}\}$ with roughly 63% of unique training examples.[3] The remaining 37% are used for estimating the classifier's testing error as each B_l provides a single vote $B_l(\mathbf{x}_{\mathcal{D}_l}) \in \{0, 1\}$ (cover = 0, stego = 1) for each $\mathbf{x} \notin \mathcal{X}_l$. After all L base learners are trained, each training sample $\mathbf{x} \in \mathcal{X}^{\text{trn}}$ will thus collect on average $0.37 \times L$ predictions that are fused using the majority voting strategy into the final prediction, which we denote $B^{(L)}(\mathbf{x}) \in \{0, 1\}$.

The optimal number of base learners, L, and the dimensionality of each feature subspace, d_{sub}, are determined automatically during ensemble training. The training makes use of the so-called "out-of-bag" (OOB) error estimate:

$$E^{(L)}_{\text{OOB}} = \frac{1}{2N^{\text{trn}}} \sum_{m=1}^{N^{\text{trn}}} \left(B^{(L)}(\mathbf{x}^{(m)}) + 1 - B^{(L)}(\bar{\mathbf{x}}^{(m)}) \right), \tag{10}$$

which is an unbiased estimate of the testing error [3]. The parameter d_{sub} is determined by a simple one-dimensional direct-search derivative-free technique inspired by the

[3] The threshold of all base learners is set to minimize the total detection error with equal priors determined again from examples from the training set only.

compass search [31] that minimizes (10). During the search, for each tested d_{sub} the number of base learners, L, is gradually increased one-by-one while monitoring the gradual decrease of the OOB error estimate (10). This process is stopped, and L is fixed, when (10) starts showing signs of saturation.

The OOB error estimate is a very convenient by-product of training. Since it is an unbiased estimate of the testing error, it is ideally suited for evaluating the detection performance of the submodel on unseen data without having to reserve a portion of the training set to assess the testing error as is commonly done in cross-validation.[4] In the next section, we will assemble the rich model using the OOB error estimates of all submodels.

IV. Assembling the rich model

Fundamentally, the model assembly is a feature selection problem as the steganalyst strives to achieve the best performance for a given feature dimensionality. As already pointed out above, our approach will combine a feature subset selection heuristic with a classification feedback. The union of all submodels (co-occurrence matrices), including their differently quantized versions, has a total dimension of $2\times(2\times 169+10\times 325)+3\times(10\times 169+23\times 325)=34,671$.[5] We apply our feature selection strategies to the *entire submodels* based on the estimate of their individual performance (in terms of the OOB error (10)) as this allows us to interpret the results, relate the selected submodels to the steganographic algorithms, and provide interesting feedback to steganographers (Sections V and VI), which would not be possible if we were selecting individual features. The learning process is applied for a given stegochannel as defined in [15], which entails a specific steganographic algorithm, message source (payload size), and a sample of cover and stego images.

Since the dimensionality of submodels that originated from the 'spam' type residual is 169 ('minmax' residuals have dimensionality 325), to make the ranking by OOB fair, we merge the vertical and horizontal 'spam' submodels into a single $2\times 169=338$-dimensional submodel (merging spam14h with spam14v from classes '1st', '3rd', 'EDGE3x3', and 'EDGE5x5', and also spam12h with spam12v from class '2nd'). Additionally, we merge the two spam11 submodels from class 'SQUARE' into one 338-dimensional submodel. Now, all submodels have approximately the same dimension (338 or 325) and can thus be fairly ranked by their individual detection performance. Note that after merging the horizontal and vertical 'spam' submodels, we end up with 11 submodels in class '1st' and '3rd', six in '2nd', one in 'SQUARE', and five in each edge class (total of 39 submodels or 106 if counting quantized versions as different). A short acronym will be used for each submodel consisting of the number of filters, f, the symmetry index, σ, and the co-occurrence scan direction. Since for submodels of type 'spam' both scan directions were merged, we use the scan string 'hv' in their acronym. These acronyms are used in Fig. 3.

To formally and unambiguously explain the feature selection strategies, we introduce additional notation. Let $\mathcal{I}_1,\dots,\mathcal{I}_6\subset\{1,\dots,39\}$ be the index sets corresponding to submodels from '1st', '2nd', '3rd', 'EDGE3x3', 'EDGE5x5', and 'SQUARE' classes, respectively. The cardinalities of these six index sets are 11, 6, 11, 5, 5, and 1, respectively. The reader is advised to follow Fig. 3 for better clarity. We denote by $\mathcal{M}_i^{(q)}$ the ith submodel, $i\in\{1,\dots,39\}$, quantized with q ($q\in\{1c,2c\}$ for $i\in\mathcal{I}_1$ and $q\in\{1c,1.5c,2c\}$ otherwise). The OOB error estimate (10) of the ith submodel quantized with q will be denoted $E_i^{(q)}$.

The following six submodel selection strategies will be explored in the next section:

- ALL. This strategy is a simple forward feature selection applied to submodels. The idea is to merge the M best individually performing submodels (based on their OOB errors) into one model and omit the rest. Formally, we form the model by merging M submodels with the M smallest OOB error estimate $E_i^{(q)}$ out of all 106 submodels.
- BEST-q. Since two differently quantized versions of one submodel provide less diversity than two submodels built from different residuals, in this strategy we force diversity among the selected submodels while hoping to improve the performance-to-dimensionality ratio. First, we determine the best quantization step q for each submodel i: $q_i=\arg\min_q E_i^{(q)}$. We remind that the argument minimum is taken over two values for q in class '1st' and over three values for all other classes. As a result, we obtain 39 submodels, $\mathcal{M}_i^{(q_i)}$, with the corresponding OOB error estimates $E_i^{(q_i)}$. Now, we apply the forward feature selection to these 39 submodels as in ALL. In particular, we merge M submodels $\mathcal{M}_i^{(q_i)}$ with M lowest OOB errors $E_i^{(q_i)}$.
- BEST-q-CLASS. In this strategy, we force diversity even more than in BEST-q by first selecting one (the best) submodel from each class before selecting a different submodel from the same class again. As in BEST-q, we begin with only 39 submodels and proceed in rounds. In the first round, we find $i_1=\arg\min_{i\in\mathcal{I}_1}E_i^{(q_i)}$, …, $i_6=\arg\min_{i\in\mathcal{I}_6}E_i^{(q_i)}$ and merge $\mathcal{M}_6=\cup_{k=1}^6\mathcal{M}_{i_k}^{(q_{i_k})}$. Then, we remove the submodels from their classes, $\mathcal{I}_k\leftarrow\mathcal{I}_k-\{i_k\}$ and proceed to the second round. The second round is the same as the first one but with each class being by one submodel smaller, etc. If, at some point, the class becomes empty, we remove it from consideration. The idea of this strategy is to force diversity even more than in BEST-q as the first six submodels are guaranteed to be selected from six different classes, so are the next six, etc.
- Q1. Merge $\mathcal{M}_i^{(1c)}$, $i=1,\dots,39$. We want to compare a fixed quantization $q=1c$ with the optimized quan-

[4] Realize that we do not (and cannot!) use any feedback from the actual testing set $\mathcal{X}^{\text{tst}}$ for this purpose.

[5] We provide the extractor of all 34,671 features at http://dde.binghamton.edu/download/feature_extractors.

tization of the BEST-q (or BEST-q-CLASS) strategy after merging all 39 submodels. Formally, this strategy is a simple union $\cup_{j=1}^{39}\mathcal{M}_i^{(1c)}$.

- CLASS-q. The goal here is to see how successful each residual type is in detecting a given stego algorithm. To this end, we form the model by merging all submodels with the best quantization step q from a fixed class. There will be total of six models here, one for each class ($k = 1, \ldots, 6$): $\cup_{i\in\mathcal{I}_k}\mathcal{M}_i^{(q_i)}$.
- ITERATIVE-BEST-q. This strategy is the only one that considers mutual dependencies among submodels. The submodels are selected sequentially one by one based on how much they improve the detection w.r.t. the union of those already selected. We start with 39 submodels just like in strategies BEST-q and BEST-q-CLASS. The first submodel selected is the one with the lowest OOB error. We denote it $\mathcal{M}_{i_1}^{(q_{i_1})}$, $i_1 = \arg\min_i E_i^{(q_i)}$. Having selected $k \geq 1$ submodels, the $k+1$st submodel $\mathcal{M}_{i_{k+1}}^{(q_{i_{k+1}})}$ is selected as

$$i_{k+1} = \arg\min_{i\notin\{i_1,\ldots,i_k\}} E_{\text{OOB}}\left(\cup_{j=1}^{k}\mathcal{M}_{i_j}^{(q_{i_j})}\cup\mathcal{M}_i^{(q_i)}\right), \tag{11}$$

  where $E_{\text{OOB}}(\mathcal{M})$ is the OOB error estimate when training model $\mathcal{M}$. In other words, we add the one submodel among the $39-k$ remaining submodels that leads to the biggest drop in the OOB estimate when the union of all $k+1$ submodels is used as a model. Determining the first k submodels requires $39 + 38 + \cdots + 39 - k + 1$ trainings, which makes this method rather expensive for increasing k. In this paper, we applied this strategy for $k \leq 10$.

From the machine-learning point of view, the first three strategies, ALL, BEST-q, and BEST-q-CLASS, could be classified as filters [20]. They are based solely on the initial OOB ranking of every submodel and thus ignore dependencies among the submodels. They differ mainly in how they enforce diversity. The ITERATIVE-BEST-q strategy, on the other hand, continuously utilizes classification feedback of the ensemble as it greedily minimizes the OOB error in every iteration, taking thus the mutual dependencies among individual submodels into account. This is an example of a *wrapper* [20], which is a feature selection method using a machine-learning tool as a black-box and is thus classifier-dependent. Filters and wrappers are both examples of forward feature selection methods, here applied to the whole submodels rather than to individual features.

The CLASS-q strategy corresponds to merging all submodels with the best q from one chosen class, while the Q1 strategy corresponds to merging all 39 submodels with a fixed quantization $q = 1c$. The purpose of these two simple heuristic merging strategies is rather investigative, see Section V-C.

V. Investigative experiments

All experiments in this paper are carried out on three steganographic algorithms with contrasting embedding mechanisms:

1) Non-adaptive ±1 embedding (also called LSB Matching) implemented with ternary matrix embedding that is optimally coded to minimize the number of embedding modifications. In particular, the relative payload α bpp (bits per pixel) is embedded with change rate $H_3^{-1}(\alpha)$, where $H_3^{-1}(x)$ is the inverse of the ternary entropy function $H_3(x) = -x\log_2 x - (1-x)\log_2(1-x) + x$. (For more details, see, e.g., Chapter 8 in [15].)
2) HUGO [34], which was designed to minimize embedding distortion in a high-dimensional feature space computed from differences of four neighboring pixels. We used the embedding simulator available from the BOSS website [13] with $\sigma = 1$ and $\gamma = 1$, for the parameters of the distortion function, and the switch –T 255, which means that the distortion function was computed with threshold $T = 255$ instead of the default value $T = 90$ used in the BOSS challenge [13]. We did it to remove a weakness of HUGO with $T = 90$ that makes the algorithm vulnerable to first-order attacks due to an artifact present in the histogram of pixel differences [29].
3) Edge-Adaptive (EA) algorithm, due to Luo et al. [32] confines the embedding changes to pixel pairs whose difference in absolute value is as large as possible (e.g., around edges). Both HUGO and the EA algorithm place the embedding changes to those parts of the image that are hard to model and are thus expected to be more secure than the non-adaptive ±1 embedding.

A. Image source

The image source used for all experiments is the BOSSbase ver. 0.92 consisting of 9,074 cover images taken with seven digital cameras in their RAW format, converted to grayscale, and resized/cropped to 512×512 using the script provided by the BOSS organizers. The reason for constraining our investigation to a single cover source was our desire to apply the proposed framework to several different algorithms for a wide range of relative payloads, which by itself is extremely computationally demanding and required use of a high-performance computer cluster for an extensive period of time. Moreover, the focus of this paper is on the methodology rather than benchmarking steganography in different cover sources.

We structure our investigation into three experiments, two of which are in this section, while the third one appears in the next section. The goal of Experiment 1 is to obtain insight about the detection performance for each submodel, stego algorithm, and quantization step. We also assess the statistical spread of the results w.r.t. the randomness in the ensemble. Experiment 2 was designed to evaluate the efficiency of each model-assembly strategy listed in Section IV.

B. Experiment 1

We start by computing the OOB estimates (10) for each submodel, including its differently quantized versions, for each stego method and for one small and one large payload (0.1 and 0.4 bpp).[6] The intention is to investigate how the submodel ranking is affected by the stego algorithm, quantization factor, and payload. Note that the ensemble classifier is built using random structures (randomness enters the selection of subspaces for base learners and the bootstrap sample formation), which is why we repeated each run five times and report the average values of OOB estimates. Table I shows that the variations are in general rather negligible. They can also be made arbitrarily small by the user by increasing the number of base learners L.

All results are summarized in Fig. 3 showing the average OOB error estimates $E_i^{(q)}$ for all $i = 1, \dots, 39$ and for all values of q. The dashed lines separate the 'spam' submodels from submodels of type 'minmax'. The dots were connected by a line to enable a faster visual interpretation of the results.

1) Evaluating individual algorithms: By comparing the patterns for a fixed algorithm, we see that there is a great deal of similarity between the performance of submodels across payloads even though the actual rankings may be different. Remarkably, ±1 embedding with small payload shows by far the largest sensitivity to the quantization factor than any other combination of algorithms and payloads. This effect is caused by the non-adaptive character of ±1 embedding. For small payloads, the amount of changes in edges and textures is so small that detection essentially relies on smooth parts where the finest quantization discerns the embedding far better in comparison to other quantizations. On the contrary, both adaptive algorithms are much less sensitive to the quantization step because small payloads are more concentrated in textures and edges.

Notice that, for HUGO, submodels built from first-order differences have worse performance than submodels obtained from third-order differences, which is due to the fact that HUGO approximately preserves statistics among first-order differences; the higher-order differences thus reach "beyond" the model. Also, features of type 'spam' seem to be consistently better than 'minmax' for this algorithm.

The OOB estimates for the EA algorithm exhibit a remarkable similarity for both payloads. This property can probably be attributed to the much more selective character of embedding. While HUGO makes embedding changes even in less textured areas albeit with smaller probability, the EA algorithm limits the embedding only to those pairs of adjacent pixels whose difference is above a certain threshold, eliminating a large portion of the image from the embedding process.

2) Universality of submodels: The best individual submodels for the larger payload and ±1 embedding, HUGO, and EA achieve OOB error estimates around 0.1, 0.21, and 0.12, indicating that HUGO is by far the best algorithm among the three. While there exist clear differences among the performance of each submodel across algorithms, it is worth noting that certain submodels rank the same w.r.t. each other for all three algorithms, both payloads, and all quantization steps. For example, 'minmax22h' is always worse than 'minmax22v' for class '1st' as well as '3rd'. In other words, it is better to form co-occurrences in the direction that is perpendicular to the direction in which the pixel differences are computed. This is most likely because the perpendicular scan prevents overlaps of filter supports and thus utilizes more information among neighboring pixels. The universality of submodels is further supported by the fact that pair-wise relationships between submodels are largely invariant to stego method and payload – for 40% of all pairs (i, j), $i > j$, $i, j \in \{1, \dots, 39\}$ the numerical relationship between errors of models $\mathcal{M}_i^{(q_i)}$ and $\mathcal{M}_j^{(q_j)}$ does not depend on the algorithm or payload.

To further investigate the universality of submodels, in Fig. 4 we plot for each submodel its OOB error estimate averaged over all three stego algorithms and five payloads, 0.05, 0.1, 0.2, 0.3, and 0.4 bpp. The fact that submodels from '3rd' consistently provide lower OOBs than the corresponding submodels from '1st' allowed us to overlap the results and thus compare both classes visually. The best overall submodel is minmax24 in class 'EDGE3x3'. Note that 'h' versions of submodels built from residuals that are not hv-symmetrical are almost always worse than 'v' versions as most residuals are defined in Fig. 2 in their horizontal orientation. This supports the rule that forming co-occurrence matrices in the direction perpendicular to the orientation of the kernel support generally leads to better detection as the co-occurrence bin utilizes more pixels. Fig. 3 also nicely demonstrates that submodels built from first-order differences are in general worse than their equivalents constructed from third-order differences. Finally, observe that the best submodels are in general from hv-symmetrical non-directional residuals.

Table I
Mean Absolute Deviation (MAD) of OOB estimates ($\times 10^{-3}$) over five database splits. The table reports the average and maximal values over all 106 submodels, $\mathcal{M}_i^{(q)}$, for all three tested stego algorithms and two payloads.

Algorithm	±1 embedding		HUGO		EA	
Payload	0.10	0.40	0.10	0.40	0.10	0.40
avg. MAD	0.649	0.679	0.656	0.526	0.601	0.484
max. MAD	1.640	1.500	2.840	1.220	1.200	0.940

C. Experiment 2

The purpose of this experiment is to investigate the efficiency of the submodel-selection strategies explained in Section IV. It is done for a fixed payload of 0.4 bpp for all three algorithms on the training set for one fixed split of BOSSbase into 8074 training and 1000 testing images.

1) Evaluation by submodel selection strategies: Fig. 5 shows the OOB error estimate as a function of model

[6] Experiment 1 was carried out on the training set for one fixed split of BOSSbase into 8,074 training and 1000 testing images.

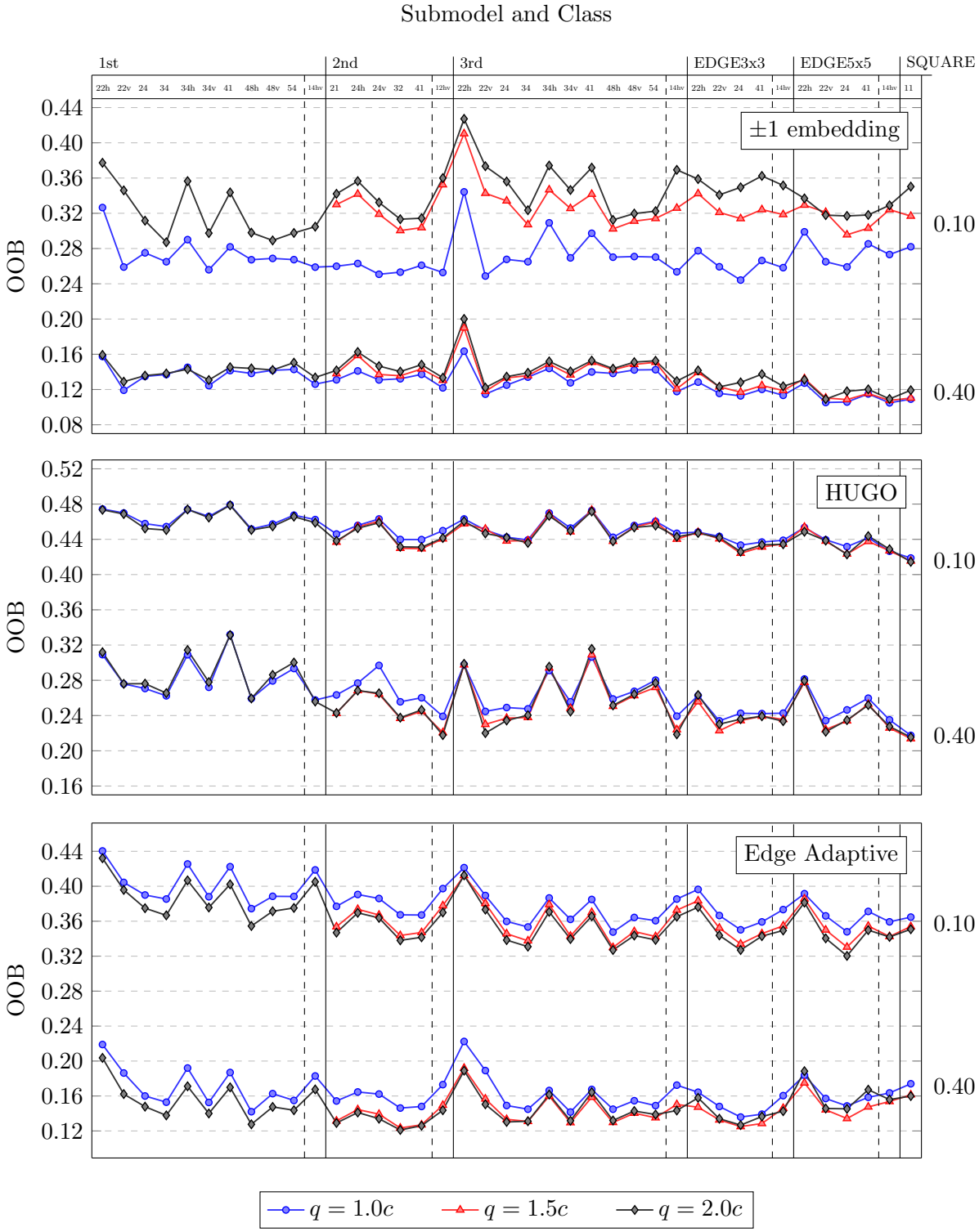


Figure 3. OOB error estimates (10) for all 106 submodels, $\mathcal{M}_i^{(q)}$, for three stego algorithms and two payloads. The values were averaged over five runs of the ensemble for a fixed split of the BOSSbase.

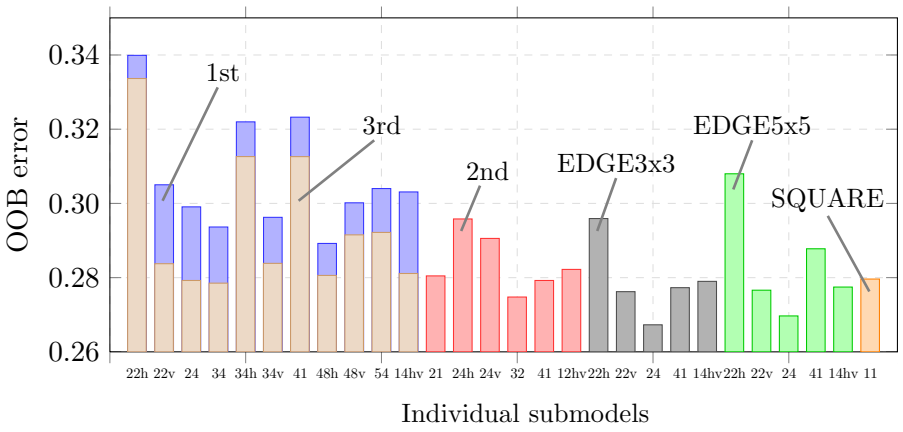


Figure 4. OOB error estimates averaged over all three stego methods and five payloads (0.05, 0.1, . . ., 0.4 bpp). Individual classes are shown in different shades of gray.

dimensionality for all assembly strategies. Diversity-boosting strategies (BEST-q and BEST-q-CLASS) clearly achieve better results than the simple ALL.

As expected, ITERATIVE-BEST-q outperforms all other strategies but its complexity limited us to merging only ten submodels. A little over 3000 features are in general sufficient to obtain detection accuracy within $0.5-1\%$ of the result when the entire 34,761-dimensional rich model is used. When all ten submodels are selected using this strategy, the best "dependency-unaware" strategy, BEST-q-CLASS, needs roughly double the dimension for comparable performance. This seems to suggest that further and probably substantial improvement of the performance-dimensionality trade-off is likely possible using more sophisticated feature-selection methods.

Overall, the lowest OOB error estimate is indeed obtained when all 106 (dimension 34,671) submodels are used. The gain between using $M = 39$ submodels of BEST-q-CLASS (dimensionality 12,753) and all 106 quantized submodels is however rather negligible, indicating a saturation of performance.

Models assembled from a specific class (CLASS-q) also provide interesting insight. We obtain another confirmation that third-order residuals have better detection accuracy than first-order residuals across all stego algorithms. Remarkably, despite its lower dimension, the model assembled from class '2nd' for HUGO is better than class '1st'. This is not true for the other two algorithms and is due to the fact that HUGO preserves complex statistics computed from first-order differences among neighboring pixels. Curiously, while 'EDGE5x5' is better than 'EDGE3x3' for ±1 embedding and HUGO, the opposite is true for EA. The 'EDGE5x5' class appears to be particularly effective against ±1 embedding.

Strategy Q1 (the single black cross at dimensionality 12,753 in Fig. 5) does not optimize w.r.t. the quantization factor q, and thus it is not surprising that its performance is generally inferior to the performance of the equally-dimensional BEST-q-CLASS strategy with $M = 39$ merged submodels. The loss is however rather small (and there is almost no loss for ±1 embedding). Additionally, Q1 allows the steganalyst to reduce the feature extraction time roughly to 1/3 as only 39 submodels with $q = 1c$ (out of 106) need to be calculated.

Finally, it is rather interesting that at this payload (0.4 bpp) the EA algorithm is less secure than the simple non-adaptive ±1 embedding.

VI. Testing the full framework

The purpose of this last experiment is to test the proposed framework in a way it is customary in research works on steganalysis. In particular, for each split of BOSSbase into 8,074 training images and 1000 testing images, and for each payload (0.05, 0.1, 0.2, 0.3, and 0.4 bpp) and stego method, we use the BEST-q-CLASS strategy and assemble the rich model as well as the final steganalyzer using only the training set.[7] Here, the training set serves the role of an available sample from the cover source in which secret messages are to be detected for a known stego method.

We evaluate the performance using the detection error on the testing set:

$$P_{\mathrm{E}} \triangleq \min_{P_{\mathrm{FA}}} \frac{1}{2}(P_{\mathrm{FA}} + P_{\mathrm{MD}}(P_{\mathrm{FA}})), \tag{12}$$

as a function of the payload expressed in bpp, where P_{FA} and P_{MD} are the probabilities of false alarm and missed detection. Fig. 6 shows the median values, $\bar{P}_{\mathrm{E}}$, together with detection errors for the CDF set [30] implemented with a Gaussian SVM – the state-of-the-art approach before introducing rich models and ensemble classifiers.[8] The figure contains the results for several models depending on how many submodels in the BEST-q-CLASS strategy are used; TOPM means that the first M submodels as reported in Fig. 5 were used.

[7] It is entirely possible that the submodel ranking is slightly different on each split.

[8] To save on processing time, we report the results for the CDF set with a G-SVM for only a single split as the variations over different splits are rather small and similar to those of the ensemble.

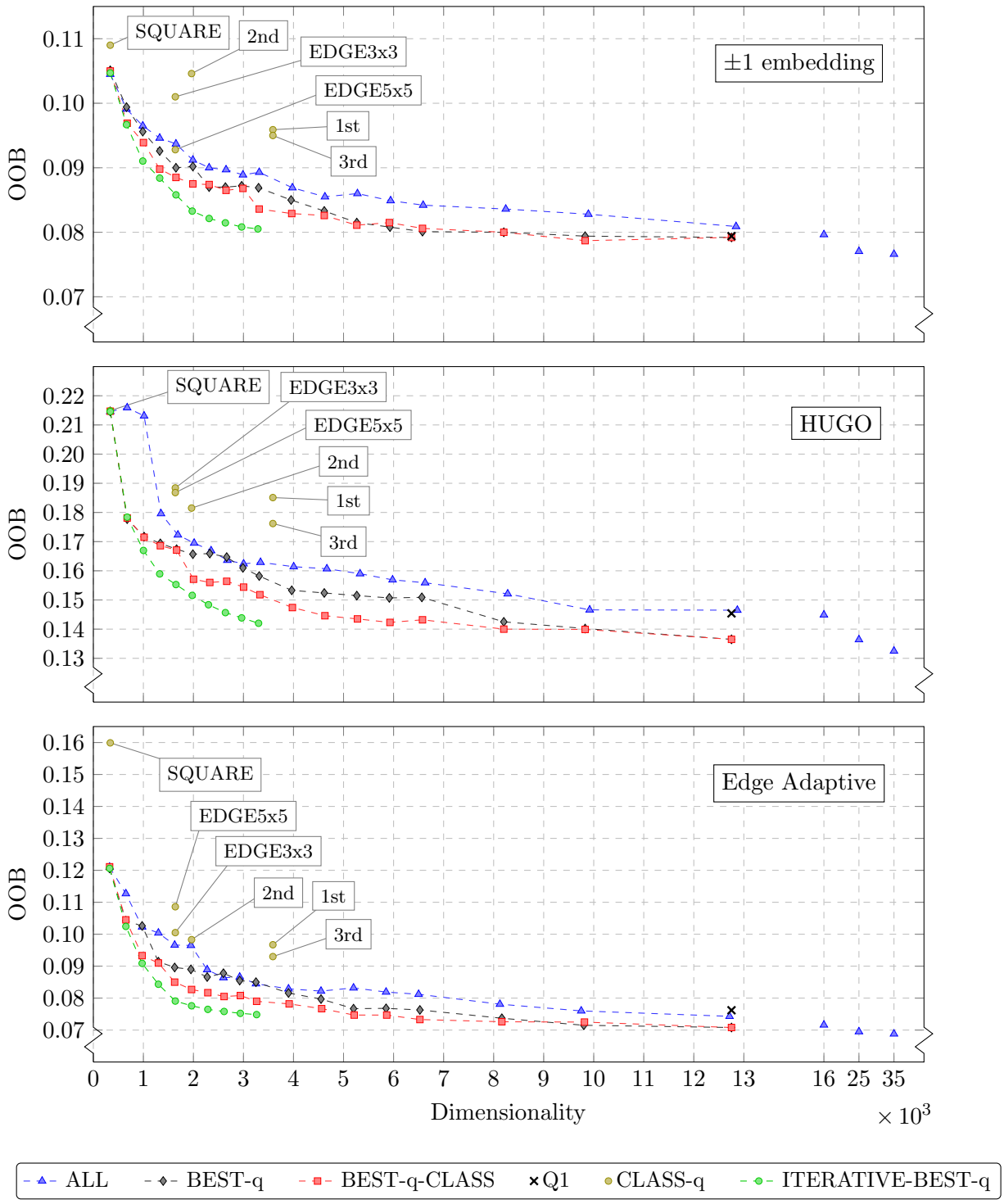
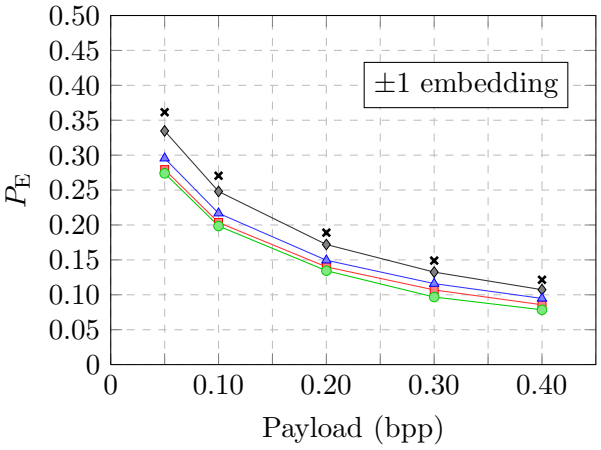


Figure 5. Performance-to-model dimensionality trade-off for five different submodel selection strategies for three algorithms and a fixed relative payload of 0.4 bpp. The performance is reported in terms of OOB error estimates. The last three tics on the x axis for strategy ALL are not drawn to scale. The last point corresponds to a model in which all quantized versions of all 106 submodels are merged.

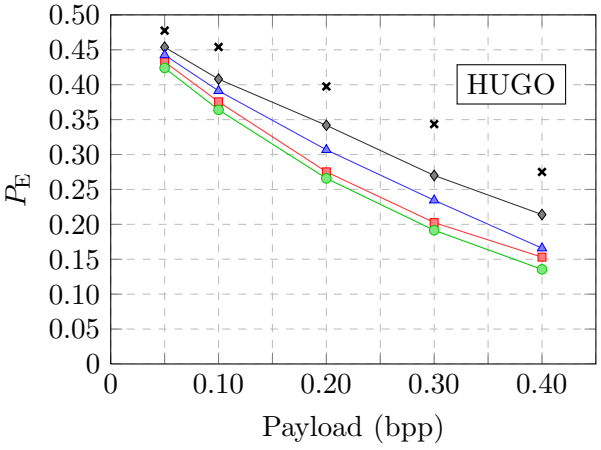
±1 embedding

payload (bpp)	CDF (G-SVM) (single split)	TOP 39 MED	TOP 39 MAD
0.05	0.3615	0.2740	0.0065
0.10	0.2705	0.1985	0.0057
0.20	0.1890	0.1345	0.0035
0.30	0.1490	0.0968	0.0038
0.40	0.1215	0.0785	0.0035



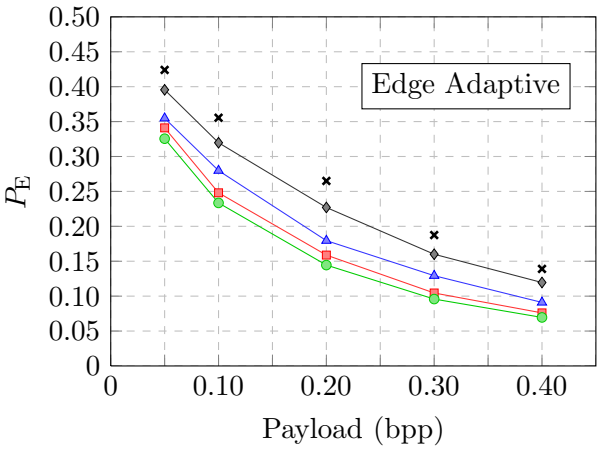
HUGO

payload (bpp)	CDF (G-SVM) (single split)	TOP 39 MED	TOP 39 MAD
0.05	0.4775	0.4240	0.0045
0.10	0.4540	0.3640	0.0023
0.20	0.3975	0.2658	0.0053
0.30	0.3435	0.1915	0.0033
0.40	0.2750	0.1355	0.0035



Edge Adaptive

payload (bpp)	CDF (G-SVM) (single split)	TOP 39 MED	TOP 39 MAD
0.05	0.4240	0.3255	0.0028
0.10	0.3555	0.2335	0.0067
0.20	0.2650	0.1445	0.0037
0.30	0.1875	0.0958	0.0038
0.40	0.1390	0.0695	0.0020



CDF	...	dimension $d = 1234$
TOP 39	...	dimension $d = 12753$

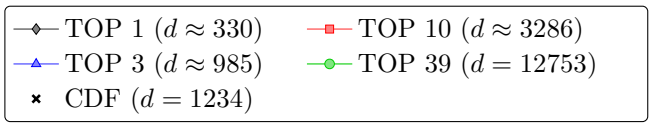


Figure 6. Detection error for three stego algorithms as a function of payload for several rich models. $\bar{P}_E$ is the median detection error P_E over ten database splits into 8074/1000 training/testing images. The models as well as the classifiers were constructed for each split. The model assembly strategy was BEST-q-CLASS. The tables on the left contain the numerical values and a comparison with a classifier implemented using Gaussian SVM with the CDF set.

A. Evaluation by steganographic methods

The results confirm that HUGO is by far the best algorithm of all three capable of hiding the payload 0.05 bpp with $P_{\text{E}} \approx 0.42$. Surprisingly, the security of the EA algorithm is comparable with that of ± 1 embedding for payloads larger than 0.3 bpp. We observed that at higher payloads the EA algorithm loses much of its adaptivity and embeds with higher change rate than ± 1 embedding due to its less sophisticated syndrome coding. For smaller payloads, the EA algorithm is only slightly more secure than ± 1 embedding. Overall, the detection of both adaptive stego methods benefits more from the rich model than ± 1 embedding, which is to be expected and was commented upon already in Section V-B.

B. Evaluation w.r.t. previous models

The proposed detectors provide a substantial improvement in detection accuracy over the 1234-dimensional CDF set with a Gaussian SVM even when the smallest model (TOP1 with dimensionality slightly above 300) is used. This improvement is again much higher for the two adaptive stego algorithms. With regards to the more recent publications on detection of HUGO, we note that the results of [19] were reported for HUGO implemented with $T = 90$, which introduces artifacts that make the steganalysis significantly more accurate and thus incomparable with the HUGO algorithm run with $T = 255$ tested here (see Section V and [29] for more details). Even though the attacks on HUGO reported in [17], [16], [27], [19] did not explicitly utilize the above-mentioned weakness, they, too are likely affected by the weakness and are thus not directly comparable. Having said this, the best results of [17] on HUGO (with $T = 90$) achieved with model dimensionality of 33,930 can now be matched with our rich model with dimensionality 30–100 times smaller. The decrease in the detection error P_{E} ranges from roughly 6% (for payload 0.1 bpp) to about 3% for payload 0.4 bpp. For ± 1 embedding, the improvement is smaller and ranges from $1 - 2\%$.

VII. Conclusion

Recent developments in digital media steganalysis clearly indicate the immense importance of accurate models that are relevant for steganalysis. The accuracy of steganalyzers and their ability to detect a wide spectrum of embedding methods in various cover sources strongly depends on the quality and generality of the cover model. It appears that any substantial progress is only possible when steganalysts incorporate more complex models that capture a large number of dependencies among pixels. This paper introduces a novel methodology for constructing rich models of the noise component of digital images, rich in the sense that they consider numerous qualitatively different relationships among pixels. The model is assembled for a given sample of the cover source and stego method. Both the model-building and the construction of the final steganalyzer use ensemble classifiers because of their good performance that can be achieved with very low complexity. Symmetries of natural images are heavily utilized to compactify the model and increase the statistical significance of individual co-occurrence bins forming the model. Several simple submodel-selection strategies are tested to improve the trade-off between detection accuracy and model dimensionality.

The framework is demonstrated on three stego algorithms operating in the spatial domain: ± 1 embedding and two content-adaptive methods – HUGO and an edge-adaptive method by Luo et al. [32]. Ensemble classifiers with the rich model significantly outperform previously-proposed detectors especially for the two adaptive methods as they place embedding changes in hard-to-model regions of images where the rich model better discerns the embedding changes. Remarkably, the rich model is capable of achieving the same level of statistical detectability with dimensions 30 to 100 times smaller than for the early versions of rich models [17], [16], [27].

The rich models are built using the philosophy of maximizing the diversity of submodels while keeping all their elements (co-occurrence bins) well populated and thus statistically significant. This is quite different from the model used in HUGO, where the authors simply increased the truncation threshold to obtain a high-dimensional model. Besides steganalysis, the rich model could be used for steganography as well by endowing the model space with an appropriate distortion function using, e.g., the method described in [11]. The authors, however, hypothesize that steganographic methods based on minimizing distortion in a rich model space, such as [10], may no longer be able to embed large payloads undetectably as it will become increasingly harder to preserve a large number of statistically significant quantities. This statement stems from an observation made in this paper, namely that submodels built from first-order differences among pixels are able to detect HUGO relatively reliably despite the fact that its distortion function minimizes perturbations to joint statistics built from such differences.

We expect that the proposed rich models of the noise component might find applications beyond steganography and steganalysis in related fields, such as digital forensics, for problems dealing with imaging hardware identification, media integrity, processing history recovery, and authentication. A similar framework based on rich models can likely be adopted for other media types, including audio and video signals.

The steganalyzers and models proposed in this paper consist of several procedures and modules whose design certainly deserves further investigation and optimization that might bring further performance improvement. This concerns, for example, the quantizer of the multi-dimensional co-occurrences. Replacing the uniform scalar quantizers with non-uniform or vector quantizers may help us further improve the performance for a fixed model dimensionality. Additional boost can likely be obtained by applying more sophisticated feature selection algorithms for choosing the submodels and/or their individual bins.

Finally, the local pixel predictors could be parametrized and optimized w.r.t. a specific stego method and cover source.

As our final remark, we note that one could view the model-building process independently of the final classifier design. It is certainly possible to use the speed and convenience of the ensemble to assemble the model and then use it to build a classifier using a different machine-learning tool that may provide a better separation between classes when a highly non-linear boundary exists that may not be well captured by the ensemble equipped with linear base learners. In fact, we observed that features built as co-occurrences of neighboring noise residuals often lead to non-linear boundaries that are better captured by Gaussian SVMs than the ensemble as implemented in this paper. This has already been observed in our previous work [17] and is confirmed for our rich model as well.[9]

To demonstrate the potential of this approach, we included one more final experiment. We used our best rich model (best in terms of the OOB error estimate vs. dimensionality) assembled using the strategy ITERATIVE-BEST-q with ten submodels merged (dimension approximately 3300) and trained a G-SVM for all three algorithms using the same experimental setup as described in Section VI. This was the largest model we could afford to use with a G-SVM given our computing resources. Calculating the median detection error over ten splits, in Table II we compare the results with the detection error of classifiers implemented as ensembles using the 12,753-dimensional TOP39 rich model. We only show the results for the 0.4 bpp payload as carrying out these types of experiments under our experimental setting (feature dimensionality and training set size) is rather expensive with a G-SVM. Interestingly, the smaller model with a G-SVM as the final classifier provided our best detection results. The improvement is roughly by 0.5–1% over all three steganographic methods, in terms of the median testing error, with a similar level of statistical variability over the splits. The running time of a G-SVM classifier with 3,300-dimensional features, however, was on average 30–90 times higher than the running time of the ensemble classifier with 12,753-dimensional features, as reported in Table III. The measured running times correspond to the full training and testing, including the parameter-search procedures of both types of classifiers. In case of the ensemble, this is the search for the optimal value of $d_{\rm sub}$, and in case of a G-SVM it is 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 $\mathcal{G}_C \times \mathcal{G}_\gamma$, $\mathcal{G}_C = \{10^a\},\ a \in \{0,\dots,4\}$, $\mathcal{G}_\gamma = \left\{\frac{1}{d}\cdot 2^b\right\},\ b \in \{-4,\dots,3\}$, where d is the feature space dimensionality. We used our Matlab implementation of the ensemble classifier[10] and the publicly available package LIBSVM [5] (with manually implemented cross-validation [25]) to conduct the G-SVM experiments.

[9] In contrast, in the JPEG domain co-occurrences between quantized DCT coefficients appear to react in a more linear fashion to embedding, causing the ensemble to perform equally well as G-SVMs [27], [28].

[10] available at http://dde.binghamton.edu/download/ensemble

Table II
Detection error $P_{\rm E}$ for three algorithms for payload 0.4 bpp when the ensemble is used with the rich 12,753-dimensional TOP39 model and when a G-SVM is combined with the $\sim$ 3300-dimensional best ITERATIVE-BEST-q model. The reported numbers are achieved over ten splits of BOSSbase.

Algorithm	Ensemble		G-SVM	
	MED	MAD	MED	MAD
±1 Emb.	0.0785	0.0035	0.0683	0.0042
HUGO	0.1355	0.0035	0.1310	0.0065
EA	0.0695	0.0020	0.0643	0.0030

Table III
The average running time (for the training and testing together) of the experiments in Table II if executed on a single computer with the AMD Opteron 275 processor running at 2.2 GHz.

Algorithm	Ensemble	G-SVM
±1 Emb.	1 hr 20 min	4 days 22 hr 37 min
HUGO	4 hr 35 min	8 days 15 hr 31 min
EA	3 hr 09 min	3 days 23 hr 50 min

References

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

[2] I. Avcibas, N.D. Memon, and B. Sankur. Steganalysis using image quality metrics. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security and Watermarking of Multimedia Contents III*, volume 4314, pages 523–531, San Jose, CA, January 22–25, 2001.

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

[4] G. Cancelli, G. Doërr, I.J. Cox, and M. Barni. Detection of ±1 LSB steganography based on the amplitude of histogram local extrema. In *Proceedings IEEE, International Conference on Image Processing, ICIP 2008*, pages 1288–1291, San Diego, CA, October 12–15, 2008.

[5] 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.

[6] 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.

[7] C. Chen, Y.Q. Shi, and Wei Su. A machine learning based scheme for double JPEG compression detection. In *19th International Conference on Pattern Recognition (ICPR 2008)*, pages 1–4, Tampa, FL, 2009.

[8] V. Chonev and A.D. Ker. Feature restoration and distortion metrics. 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 0G01–0G14, San Francisco, CA, January 23–26, 2011.

[9] 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.

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

Table IV
List of symbols.

$\mathbf{X}, \bar{\mathbf{X}}$	cover/stego image	$\mathbf{C}_{\mathbf{d}}^{(\mathrm{h})}$,$\mathbf{C}_{\mathbf{d}}^{(\mathrm{v})}$	hor./ver. cooc.	N^{trn},N^{tst}	no. of train/test pts	$E_i^{(q)}$	OOB of $\mathcal{M}_i^{(q)}$
$\mathcal{I}, \mathcal{J}$	index sets	$\mathbf{R} = (R_{ij})$	noise residual	B_l	lth base learner	$E_{\mathrm{OOB}}^{(L)}$	OOB with L learners
T	threshold	$\hat{X}_{ij}$	pixel predictor	$B^{(l)}$	decision of B_l	$E_{\mathrm{OOB}}(\mathcal{M})$	OOB with model $\mathcal{M}$
q	quant. step	c	residual order	$\mathcal{N}_l^{\mathrm{b}}$	B_l bootstrap set	$\mathcal{M}_i^{(q)}$	ith submodel quant. q
trunc_T	truncation	f	no. of filters	$\mathcal{D}_l$	lth rand. subspace	q_i	gives smallest OOB for $\mathcal{M}_i^{(q)}$
$P_{\mathrm{E}}, P_{\mathrm{E}}$	(median) tst. error	σ	symmetry index	d_{sub}	rand.subspace dim.	OOB	out of bag estimate
$H_3(x)$	ternary entropy	$\mathcal{N}_{ij}$	pixel neighb.	L	no. of base learners	$\bar{\bar{\mathbf{C}}}_{\mathbf{d}}$	symmetrized $\mathbf{C}_{\mathbf{d}}$

[11] T. Filler and J. Fridrich. Design of adaptive steganographic schemes for digital images. 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 OF 1–14, San Francisco, CA, January 23–26, 2011.

[12] T. Filler, J. Judas, and J. Fridrich. Minimizing additive distortion in steganography using syndrome-trellis codes. *IEEE Transactions on Information Forensics and Security*, 6(3):920–935, 2010.

[13] T. Filler, T. Pevný, and P. Bas. BOSS (Break Our Steganography System). http://boss.gipsa-lab.grenoble-inp.fr, July 2010.

[14] 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.

[15] J. Fridrich. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2009.

[16] 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, pages 85–101, Prague, Czech Republic, May 18–20, 2011.

[17] 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, pages 102–117, Prague, Czech Republic, May 18–20, 2011.

[18] 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.

[19] 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.

[20] I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors. *Feature Extraction, Foundations and Applications*. Physica-Verlag, Springer, 2006.

[21] J. J. Harmsen and W. A. Pearlman. Steganalysis of additive noise modelable information hiding. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security and Watermarking of Multimedia Contents V*, volume 5020, pages 131–142, Santa Clara, CA, January 21–24, 2003.

[22] S. Katzenbeisser and F. A. P. Petitcolas, editors. *Information Hiding Techniques for Steganography and Digital Watermarking*. New York: Artech House, 2000.

[23] 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 1–5 17, San Jose, CA, January 27–31, 2008.

[24] M. Kirchner and J. Fridrich. On detection of median filtering in images. In *Proc. SPIE, Electronic Imaging, Media Forensics and Security XII*, volume 7542, pages 10 1–12, San Jose, CA, January 17–21 2010.

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

[26] J. Kodovský and J. Fridrich. On completeness of feature spaces in blind steganalysis. In A.D. Ker, J. Dittmann, and J. Fridrich, editors, *Proceedings of the 10th ACM Multimedia & Security Workshop*, pages 123–132, Oxford, UK, September 22–23, 2008.

[27] 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.

[28] J. Kodovský, J. Fridrich, and V. Holub. Ensemble classifiers for steganalysis of digital media. *IEEE Transactions on Information Forensics and Security*, 2011. Under review.

[29] 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*, Niagara Falls, NY, September 29–30, 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] 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.

[33] 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.

[34] 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.

[35] 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.

[36] K. Sullivan, U. Madhow, S. Chandrasekaran, and B.S. Manjunath. Steganalysis of spread spectrum data hiding exploiting cover memory. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VII*, volume 5681, pages 38–46, January 16–20,.

[37] D. Upham. Steganographic algorithm JSteg. Software available at http://zooid.org/~paul/crypto/jsteg.

[38] D. Zou, Y. Q. Shi, W. Su, and G. Xuan. Steganalysis based on Markov model of thresholded prediction-error image. In *Proceedings IEEE, International Conference on Multimedia and Expo*, pages 1365–1368, Toronto, Canada, July 9–12, 2006.