Lecture Notes in Computer Science

Conference Paper · January 2010

DOI: 10.1007/978-3-642-16435-4_13 · Source: DBLP

Tomás Pevný
Czech Technical University in Prague

Patrick Bas
French National Centre for Scientific Research

Using High-Dimensional Image Models to Perform Highly Undetectable Steganography

Tomáš Pevný[1], Tomáš Filler[2], and Patrick Bas[3]

[1] Czech Technical University in Prague, Czech Republic
pevnak@gmail.com
[2] State University of New York in Binghamton, NY, USA
tomas.filler@gmail.com
[3] CNRS - LAGIS, Lille, France
patrick.bas@ec-lille.fr

Abstract. This paper presents a complete methodology for designing practical and highly-undetectable stegosystems for real digital media. The main design principle is to minimize a suitably-defined distortion by means of efficient coding algorithm. The distortion is defined as a weighted difference of extended state-of-the-art feature vectors already used in steganalysis. This allows us to "preserve" the model used by steganalyst and thus be undetectable even for large payloads. This framework can be efficiently implemented even when the dimensionality of the feature set used by the embedder is larger than 10^7. The high dimensional model is necessary to avoid known security weaknesses. Although high-dimensional models might be problem in steganalysis, we explain, why they are acceptable in steganography. As an example, we introduce HUGO, a new embedding algorithm for spatial-domain digital images and we contrast its performance with LSB matching. On the BOWS2 image database and in contrast with LSB matching, HUGO allows the embedder to hide $7\times$ longer message with the same level of security level.

1 Introduction

The main goal of a passive-warden steganographic channel [1] (stegosystem) between Alice and Bob is to transmit a secret message hidden in an innocuously looking object without any possibility for the warden Eve to detect such communication. A stegosystem is called *perfectly secure* [2] if the cover distribution exactly matches the stego distribution. Although this problem has been solved by the so-called "cover generation" [3,4,5], this solution requires *exact* knowledge of the probability distribution on cover objects, which is hard (if possible at all) to obtain for real digital media in practice. The most common practical solution is to hide the message by making small perturbations with the hope that these perturbations will be covered by image noise.

One of the most popular embedding methods used with digital images is the Least Significant Bit (LSB) replacement, where the LSBs of individual cover elements are replaced with message bits. It has been quickly realized that the asymmetry in the embedding operation[4] is a potential weakness opening doors to the development of highly accurate targeted steganalyzers (see [6] and references therein) pushing the secure payload almost to zero.

A trivial modification of the LSB replacement method is LSB matching (often called ± 1 embedding). This algorithm randomly modulates pixel values by ± 1 so that the LSBs of pixels match the communicated message. Despite the similarity to LSB replacement, LSB matching is much harder to detect, because the embedding operation is no longer unbalanced. In fact, LSB matching has been shown to be near optimal [7] when only information from a single pixel can be utilized. The biggest weakness of LSB matching is the assumption that image noise is independent from pixel to pixel. It has been shown that this is not true in natural images, which was in different ways exploited by LSB matching detectors [8,9,10].

From the short overview of spatial domain steganography above, it is clearly seen that the embedding algorithms are not secure. This is mainly because their image model is not general enough and some marginal or joint image statistics are not preserved. In this paper, we propose a novel method for designing new steganographic algorithms allowing to use very general and high-dimensional models covering various dependencies in natural images in order to create more secure steganographic algorithms. The method follows and extends the best principles known in steganography and steganalysis so far.

The proposed method relies on the principle of minimal impact embedding [11], which is revisited in Section 2. This principle allows decomposition of the design of steganographic algorithms into the design of the image model and the coder. By virtue of this principle, steganographic algorithms can be improved either by using a better coder, or by using a better model. Thus, the image model becomes one of the most important parts of the design. Section 3 is devoted to this problem. We explain why steganalytic features can be used as a good start to design a steganographic model, if they are extended to avoid overfitting to a particular steganalyzer. Although such steganographic models can be very large (we give an example of a model with dimension 10^7), we argue that for steganographic purposes such large dimension does not pose a problem. In Section 4, we practically demonstrate the presented method by constructing a new steganographic algorithm for the spatial domain based on the SPAM (Subtractive Pixel Adjacency Matrix) features [10]. The security of the proposed scheme and the effect of individual design elements on the security is experimentally verified. The paper is concluded in Section 5.

The ideas presented in this paper can been seen in prior art. (a) Virtually all steganographic algorithms aim to minimize distortion to preserve some image model. The image model is derived either from the image itself (e.g., F5 algorithm [12] and its improvement [13], Model Based Steganography [14], etc.), or the distortion is defined by means of error introduced by quantization. The latter class of algorithms (MMX [15] and its improvement [16], PQ [17], etc.) uses "side information" in the form of a higher quality image, which is not available

[4] Even cover elements are never decreased whereas odd ones are never increased.

to the recipient (and Eve). (b) Many algorithms (F5 [12], nsF5 [13], MMX [15], and [16]) already utilized various coding schemes (matrix embedding) to minimize the distortion. While early schemes (e.g., F5 or LSB matching) used coding to minimize the number of embedding changes, a significant departure was proposed in MMX, which allowed more embedding changes than optimal (with given coding), in order to decrease the overall distortion. Thus, MMX can be interpreted as making local content-adaptive embedding by means of coding, which is close to the proposed scheme.

With respect to the above prior work, the main contributions of this work are as follows. (a) We promote and advocate the use of high-dimensional image models in steganography that cannot be used in steganalysis (yet). (b) We separate the image model from coding, which allows simulating optimal coding and thus comparing image models without the effect of coding. Moreover, the message can be hidden in parts of the image difficult for steganalysis while *considering all pixels simultaneously during the embedding.*

Although the proposed steganographic scheme might be considered as an adaptive, it is not adaptive in the usuall approach, when first good pixels are selected [9,18,19] (e.g. pixels in noisy and textured areas) and than the message is inserted in the image while modifying only the selected pixels (e.g by using wet paper codes). Our scheme always uses all pixels for the embedding, but it changes them with probability inversely proportional to the detectability of their change.

In the rest, we use the following notation. Small-case boldface symbols are used for vectors and capital-case boldface symbols for matrices and possibly tensors. Symbols $\mathbf{X} = (x_{ij}) \in \mathcal{X} = \{0, \ldots, 255\}^{n_1 \times n_2}$ and $\mathbf{Y} = (y_{ij}) \in \mathcal{X}$ are exclusively used to represent intensities of $n = n_1 n_2$-pixel cover and stego image. For the sake of simplicity, we sometimes index the pixels with a single number, $\mathbf{X} = (x_i)_{i=1}^{n}$ and similarly for stego image $\mathbf{Y} = (y_i)_{i=1}^{n}$.

2 Minimizing Embedding Impact

Virtually all practical steganographic algorithms for digital media strive to minimize an ad hoc *embedding impact* [11,20], which, if properly defined, is correlated with detectability. In its simplest form, embedding impact is simply the number of changes (known as matrix embedding). However, more general ways, as already suggested by Crandal [21], should be considered. In general, the embedding impact is captured by a non-negative distortion measure $D : \mathcal{X} \times \mathcal{X} \to [0, \infty]$. During embedding, the algorithm should find a stego image $\mathbf{Y}$, which (a) communicates a given message and (b) achieves minimal value of $D(\mathbf{X}, \mathbf{Y})$. Unfortunately, this problem is generally very difficult in practice.

From this reason, we constrain ourselves to a well-studied special (but still powerful enough) case assuming (a) binary embedding changes[5], i.e., $|x_i - y_i| \leq 1$,

[5] Extensions to ternary case can be done by the "$e+1$" construction described in [22].

$i \in \{1, \ldots, n\}$, and (b) additive distortion measure in the form

$$D(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{n} \rho_i |x_i - y_i|. \tag{1}$$

The constants $0 \leq \rho_i \leq \infty$ are fixed parameters expressing costs of (or distortion caused by) pixel changes. The case $\rho_i = \infty$ corresponds to the so-called *wet pixel* not allowed to be modified during embedding. Notice that the additivity of the distortion function D implies that that the embedding changes do not interact between each other. This is a reasonable assumption, especially if we assume low embedding rates and embedding changes being far from each other. Unfortunately, there are cases of important distortion measures which cannot be written in this form. One such case will be introduced in Section 4.

For additive distortion functions (1), the following theorem taken from [11] gives the minimal expected distortion obtained by hiding m bits in an n-pixel cover object.

Theorem 1. *Let $\rho = (\rho_i)_{i=1}^{n}$, $0 \leq \rho_i < \infty$, be the set of constants defining the additive distortion measure* (1) *for $i \in \{1, \ldots, n\}$. Let $0 \leq m \leq n$ be the number of bits we want to communicate by using a binary embedding operation. The minimal expected distortion has the following form*

$$D_{min}(m, n, \rho) = \sum_{i=1}^{n} p_i \rho_i,$$

where

$$p_i = \frac{e^{-\lambda \rho_i}}{1 + e^{-\lambda \rho_i}} \tag{2}$$

is the probability of changing the ith pixel. The parameter λ is obtained by solving

$$-\sum_{i=1}^{n} \Big(p_i \log_2 p_i + (1 - p_i) \log_2 (1 - p_i) \Big) = m. \tag{3}$$

The importance of Theorem 1 is in the separation of the image model (needed for calculating constants ρ_i) and the coding algorithm used in a practical implementation. By virtue of this separation, better steganographic algorithms can be derived by using better coding or by using a better image model. One important consequence is that, in order to study the effect of the image model on steganographic security, no coding algorithm is needed at all! The optimal coding can be simulated by flipping each pixel with probability p_i as defined in (2).

We use this *separation principle* in Section 4 to find a good image model used to derive the costs ρ_i. The study of the loss introduced by a practical coding method is also included.

3 From Steganalysis to Steganography

Almost all state-of-the-art statistical steganalyzers (with the exception of steganalyzers for LSB replacement) are based on a combination of steganalytic features and pattern recognition algorithms. In steganalysis, steganalytic features are used to reduce the dimension of a space of all cover objects, so that the pattern recognition algorithms can learn (if possible) the difference between cover and stego objects in this reduced feature space. Using such a low-dimensional model for designing steganography usually leads to overtraining to a particular feature set (this issue of feature set completeness is discussed in [23,24]). Keeping this in mind, we believe that the features can serve as a good precursor of the image model to determine the embedding costs ρ_i. Although we show this transition from steganalytic features to a steganographic model on spatial domain steganography, we believe that the ideas and tools presented here can be used in other domains and with other steganalytic features as well.

We start by reviewing the recently proposed SPAM features [10] proposed to detect steganographic algorithms in spatial and transformed domains. Then, we discuss the problem of overfitting the steganographic model to steganalytic features as well as the remedy by expanding the model beyond the capabilities of contemporary pattern recognition algorithm. Finally, we propose a simple method to identify parts of the model that are more important for steganalysis.

3.1 SPAM features

It is well known that values of neighboring pixels in natural images are not independent. This is not only caused by the inherent smoothness of natural images, but also by the image processing (de-mosaicking, sharpening, etc.) in the image acquisition device. This processing makes the noise, which is independent in the raw sensor output, dependent in the final image. The latter source of dependencies is very important for steganalysis because steganographic changes try to hide themselves within the image noise.

The SPAM [10] features model dependencies between neighboring pixels by means of higher-order Markov chains. They have been designed to provide a low-dimensional model of image noise that can be used for steganalytic purposes. The calculation of differences can be viewed as an application of high-pass filtering, which effectively suppresses the image content and exposes the noise. The success of SPAM features in detecting wide range of steganographic algorithms [25] suggests this model to be reasonable for steganalysis and steganography.

The SPAM features model transition probabilities between neighboring pixels along 8 directions $\{\leftarrow, \rightarrow, \downarrow, \uparrow, \nwarrow, \searrow, \swarrow, \nearrow\}$. Below, the calculation of the features is explained on horizontal left-to-right direction, because for the other directions the calculations differ only by different indexing. All direction-specific variables are denoted by a superscript showing the direction.

Let $\mathbf{I} \in \mathcal{X}$ be an image of size $n_1 \times n_2$. The calculation starts by computing the difference array $\mathbf{D}^\bullet$, which is for a horizontal left-to-right direction

$$\mathbf{D}_{ij}^{\rightarrow} = \mathbf{I}_{ij} - \mathbf{I}_{i,j+1},$$

for $i \in \{1, \dots, n_1\}$, $j \in \{1, \dots, n_2 - 1\}$. Depending on the desired order of the features, either the first-order Markov process is used,

$$\mathbf{M}^{\rightarrow}_{d_1 d_2} = Pr(\mathbf{D}^{\rightarrow}_{i,j+1} = d_1 | \mathbf{D}^{\rightarrow}_{ij} = d_2), \tag{4}$$

or the second-order Markov process is used,

$$\mathbf{M}^{\rightarrow}_{d_1 d_2 d_3} = Pr(\mathbf{D}^{\rightarrow}_{i,j+2} = d_1 | \mathbf{D}^{\rightarrow}_{i,j+1} = d_2, \mathbf{D}^{\rightarrow}_{ij} = d_3), \tag{5}$$

where $d_i \in \{-T, \dots, T\}$. The calculation of the features is finished by separate averaging of the horizontal and vertical matrices and the diagonal matrices to form the final feature sets. With a slight abuse of notation, this averaging can be written as

$$\begin{aligned} \mathbf{F}^{\bullet}_{1,\dots,k} &= \frac{1}{4}\left[\mathbf{M}^{\rightarrow}_{\bullet} + \mathbf{M}^{\leftarrow}_{\bullet} + \mathbf{M}^{\downarrow}_{\bullet} + \mathbf{M}^{\uparrow}_{\bullet}\right], \\ \mathbf{F}^{\bullet}_{k+1,\dots,2k} &= \frac{1}{4}\left[\mathbf{M}^{\searrow}_{\bullet} + \mathbf{M}^{\nwarrow}_{\bullet} + \mathbf{M}^{\swarrow}_{\bullet} + \mathbf{M}^{\nearrow}_{\bullet}\right], \end{aligned} \tag{6}$$

where $k = (2T + 1)^2$ for the first-order features and $k = (2T + 1)^3$ for the second-order features. In [10], the authors used $T = 4$ for the first-order features (leading to 162 features) and $T = 3$ for the second-order features (leading to 686 features).

3.2 Decomposing SPAM features

Although the second-order SPAM features use conditional probabilities to model pixel differences, their essential components are actually co-occurrence matrices

$$\mathbf{C}^{\rightarrow}_{d_1 d_2} = Pr(\mathbf{D}^{\rightarrow}_{ij} = d_1, \mathbf{D}^{\rightarrow}_{i,j+1} = d_2), \tag{7}$$
$$\mathbf{C}^{\rightarrow}_{d_1 d_2 d_3} = Pr(\mathbf{D}^{\rightarrow}_{ij} = d_1, \mathbf{D}^{\rightarrow}_{i,j+1} = d_2, \mathbf{D}^{\rightarrow}_{i,j+2} = d_3). \tag{8}$$

It is easy to show that the second order SPAM features with $T = 3$ can be directly obtained[6] from the set $\{\mathbf{C}^{k}_{d_1 d_2}, \mathbf{C}^{k}_{d_1 d_2 d_3} | k \in \{\rightarrow, \uparrow, \nwarrow, \nearrow\}, -3 \le d_i \le 3\}$. In fact, we observed that this set of $4 \times (343+49) = 1568$ co-occurrence features has only slightly inferior performance in detecting LSB matching, which we attribute to a smaller ratio of training samples per dimension (known as curse of dimensionality). From this point of view, the distortion measure used to derive embedding costs ρ_i should be designed to preserve the co-occurrence matrices (7) and (8), because their preservation implies the preservation of second-order SPAM features.

Although the idea of preservation of SPAM features is tempting, the distortion measure would not be general enough. The new scheme would be so tied to a particular steganalytic method that it can be expected to be detectable by a slight modification of the features. This problem of "overfitting" the distortion

[6] Observe that $\mathbf{C}^{\rightarrow}_{d_1 d_2 d_3} = \mathbf{C}^{\leftarrow}_{-d_3,-d_2,-d_1}$, and $\mathbf{M}^{\rightarrow}_{d_1 d_2 d_3} = \mathbf{C}^{\rightarrow}_{d_3 d_2 d_1} / \mathbf{C}^{\rightarrow}_{d_2 d_1}$.

measure to a particular steganalytic method together with the need for a complete feature set has been already described [23,24] for the DCT domain. Here, we propose to resolve the issue of overfitting to a particular model by expanding it beyond practical limits of steganalysis (for this model). This can be easily done in the case of co-occurrence matrices by increasing the range of covered differences T.

At this point, it is important to clarify the difference between the effects of model dimensionality for steganography and for steganalysis. The high-dimensional models in steganalysis present a serious problem for subsequent machine learning due to the curse of dimensionality and related overfitting. Although the actual ratio between the number of training samples and the model dimensionality depends on the used machine learning algorithm and the problem, the rule of thumb is to have ten times more samples than the model dimensionality (number of features). These drawbacks prevent the use of high-dimensional models in steganalysis. By contrast, high-dimensional models in steganography do not cause problems, because there is no statistical learning involved. The cover image provides the exact model to be preserved and, consequently, there is no curse of dimensionality, which justifies the use of high-dimensional models in steganography.

An additional important practical detail is that updating the co-occurrence matrices to reflect one pixel change is much easier than updating the conditional probabilities (the former involves only addition and subtraction of a few items of the matrices, while the latter involves division of the large part of the matrices). The efficient update of co-occurrence matrices enables modeling a wide range of differences between pixels (the use of large T) resulting in modeling most differences (and pixels) in the image (and better preservation of the SPAM features).

3.3 Identification of detectable parts of the models

Unfortunately, the ideal case, when the image model is fully preserved during the embedding, is virtually impossible to realize in practice. It is therefore important to identify parts of the model important for steganalysis and set appropriate costs of pixel changes ρ_i.

The association of costs ρ_i to the modification of the model is in general very difficult because we do not know which parts of the model are important. Here, we suggest to evaluate the individual elements of the model independently of each other (any method for feature ranking can be used [26]) and set the costs ρ_i to reflect this ranking. The advantage of individual evaluation is that it can be done quickly even for a large number of features. On the other hand, the individual evaluation of the model elements is certainly not optimal, especially from the machine learning point of view. However, we believe (and our experiments confirm that) that the costs derived this way can be used as a good starting point. There is no doubt that other (and better) methods of deriving costs ρ_i exist.

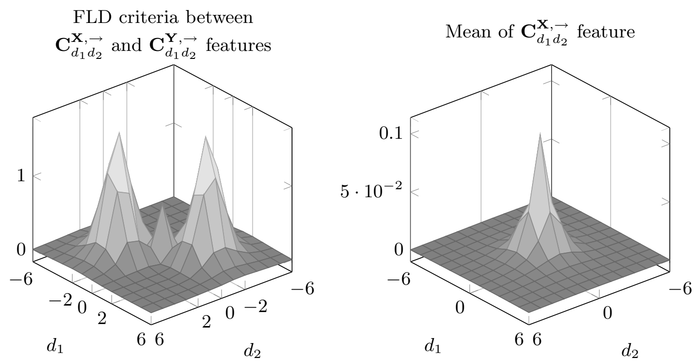


Fig. 1: Left: Values of FLD criteria (9) between the feature $\mathbf{C}^{\rightarrow}_{d_1d_2}$ calculated from cover images and stego images obtained by LSB matching with full payload. Right: mean of the feature $\mathbf{C}^{\rightarrow}_{d_1d_2}$ over the set of cover images from the BOWS2 database.

Our approach works as follows. First, we create a set of images embedded with a simulated maximum payload by a given embedding operation (in our case of spatial domain steganography, this amounts to randomly increase or decrease the pixel value by one with probability 50%). Then, we use the criteria optimized in Fisher Linear Discriminant (FLD criteria) (9) to evaluate, how good are individual features for detecting given embedding changes. The values of FLD criteria (9) of individual elements may be either used directly to set the costs of embedding changes ρ_i, which might be dangerous due to the already discussed problem of overfitting. Alternatively, they can be used to obtain insight into the problem and set the costs heuristically, which is recommended. In the rest of this section, we use the analysis of the FLD criteria to identify parts of the co-occurrence model that can be used for embedding.

For co-occurrence matrices introduced in the previous subsection, the values of FLD criteria for a single feature $\mathbf{C}^{\rightarrow}_{d_1d_2}$ (for fixed d_1 and d_2) can be written as

$$\frac{\left(E[\mathbf{C}^{\mathbf{X},\rightarrow}_{d_1d_2}] - E[\mathbf{C}^{\mathbf{Y},\rightarrow}_{d_1d_2}]\right)^2}{E\big[\mathbf{C}^{\mathbf{X},\rightarrow}_{d_1d_2} - E[\mathbf{C}^{\mathbf{X},\rightarrow}_{d_1d_2}]\big]^2 + E\big[\mathbf{C}^{\mathbf{Y},\rightarrow}_{d_1d_2} - E[\mathbf{C}^{\mathbf{Y},\rightarrow}_{d_1d_2}]\big]^2}, \tag{9}$$

where $E[\cdot]$ stands for the empirical mean (obtained in our case over all images in the BOWS2[7] image database), and $\mathbf{C}^{X,\rightarrow}_{d_1d_2}$, $\mathbf{C}^{Y,\rightarrow}_{d_1d_2}$ stand for a single element of the co-occurrence matrix $\mathbf{C}^{\rightarrow}_{d_1d_2}$ calculated from the cover and stego image, respectively. The higher the value, the better the feature when used alone for detecting the LSB matching algorithm. Figure 1 shows the values estimated from cover and stego images obtained by embedding a full payload with LSB matching. We can see that the most influential features are $\mathbf{C}^{\rightarrow}_{-2,2}$ and $\mathbf{C}^{\rightarrow}_{2,-2}$ corresponding to regions containing noisy pixels in a smooth area. Also, it is interesting to see that regions having the same color (such as saturated pixels)

[7] See `http://bows2.gipsa-lab.inpg.fr/BOWS2OrigEp3.tgz`

represented by $\mathbf{C}^{\rightarrow}_{0,0}$, or pixels in smooth transitions represented by $\mathbf{C}^{\rightarrow}_{d,d}$, do not constitute a good *single* feature. This is most probably caused by their high variance, which makes features $\mathbf{C}^{\rightarrow}_{-2,2}$ and $\mathbf{C}^{\rightarrow}_{2,-2}$ more stable and more suitable for steganalysis. Although not easy to visualize, similar results and interpretation can be obtained from higher-order co-occurrence matrices $\mathbf{C}^{\bullet}_{d_1 d_2 d_3}$.

This analysis shows which parts of the image model should be preserved. We stress again that this analysis was performed from the evaluation of a single feature and its direct application may lead to overtraining. As was already mentioned above, we consider this analysis as a good guide to derive heuristics to build the embedding costs ρ_i.

4 From Theory to Practice

In this section, all pieces and ideas presented above are put together, in order to give life to a new steganographic algorithm called HUGO (Highly Undetectable steGO). The individual steps of this algorithm are depicted in Figure 2.

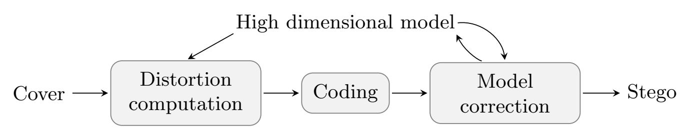


Fig. 2: High-level diagram of HUGO.

4.1 Evaluation setting

The scheme was assessed using the BOWS2 image database, containing approximately 10800 images of fixed size 512×512. Thanks to the fixed size, all images have the same number of usable elements, which means that we do not have to take the Square Root Law [27,28] into the account. Prior to all experiments, the images were divided into two sets of equal size, one used exclusively for training, the other exclusively for evaluation of the accuracy. The chosen accuracy measure is the minimal average decision error under equal probability of cover and stego images, defined as

$$P_{\mathrm{E}} = \min \frac{1}{2}\left(P_{\mathrm{Fp}} + P_{\mathrm{Fn}}\right),$$

where P_{Fp} and P_{Fn} stand for the probability of false alarm or false positive (detecting cover as stego) and probability of missed detection (false negative). To observe the effect of over-fitting for a particular feature set, we create blind steganalyzers employing four different feature sets (first- and second-order SPAM features [10] with $T = 4$ and $T = 3$ respectively, WAM [9], and recently proposed Cross Domain Features[8] (CDF) [25]).

All steganalyzers were realized as soft-margin SVMs [29] with Gaussian kernel[9], $k(x, y) = exp(-\gamma \|x - y\|^2)$. The parameters γ and C were set to values corresponding to the least error estimated by five-fold cross-validation on the training set on the grid $(C, \gamma) \in \{(10^k, 2^j) | k \in \{-3, \ldots, 4\}, j \in \{-d-3, -d+3\}\}$, where d is the logarithm at the base 2 of the number of features.

Besides the SVM-based blind steganalyzers, we also use the Maximum Mean Discrepancy [30] (MMD) to quickly compare the security of different versions of the algorithm.

4.2 Co-occurrence model in steganography

Section 3.2 motivated the use of co-occurrence matrices (SPAM features) as a reliable model for steganography and explained, why the distortion function D (not just constants ρ_i) is derived directly from them. In order to stress those parts of the co-occurrence matrices that are more important for steganalysis, the distortion function D is defined as a weighted sum of differences

$$D(\mathbf{X}, \mathbf{Y}) = \sum_{d_1, d_2, d_3 = -T}^{T} \Bigg[w(d_1, d_2, d_3) \left| \sum_{k \in \{\rightarrow, \leftarrow, \uparrow, \downarrow\}} \mathbf{C}^{\mathbf{X},k}_{d_1 d_2 d_3} - \mathbf{C}^{\mathbf{Y},k}_{d_1 d_2 d_3} \right| + \\ + w(d_1, d_2, d_3) \left| \sum_{k \in \{\searrow, \nwarrow, \swarrow, \nearrow\}} \mathbf{C}^{\mathbf{X},k}_{d_1, d_2, d_3} - \mathbf{C}^{\mathbf{Y},k}_{d_1, d_2, d_3} \right| \Bigg], \tag{10}$$

where $w(d_1, d_2, d_3)$ is a weight function quantifying the detectability of the change in the co-occurrence matrix[10]. The weight function $w(d_1, d_2, d_3)$ has the following simple form

$$w(d_1, d_2, d_3) = \frac{1}{\left[\sqrt{d_1^2 + d_2^2 + d_3^2} + \sigma\right]^{\gamma}}, \tag{11}$$

where $\sigma, \gamma > 0$ are parameters that can be tuned in order to minimize the detectability. This very conservative choice mimics the average number of samples available to Eve to estimate the individual features $\mathbf{C}^{\bullet}_{d_1 d_2 d_3}$ from a single image (see the right part of Figure 1). Motivated by the analysis performed in Section 3.3, the rationale of this choice is simple: the more samples Eve has, the better estimate of individual feature she can obtain and the more she can

[8] CDF combines second-order SPAM features ($T = 3$) and cartesian calibrated features proposed originally for DCT domain. To extract the DCT domain features, we compressed the image with quality factor 100.

[9] We did some experiments with linear SVMs and never obtained better results. For a discussion related to linear SVMs, see [10].

[10] If the $w(d_1, d_2, d_3) = 1$ for all d_i and $T = 255$, then all ρ_i would be the same and the whole scheme would just minimize the number of embedding changes.

```
                     HUGO embedding algorithm
1  for (i,j) in PIXELS {           //function D is taken from (10)
2     Yp = X; Yp(i,j)++; rho_p(i,j) = D(X,Yp);  //calculate emb. impact
3     Ym = X; Ym(i,j)--; rho_m(i,j) = D(X,Ym);  //for each pixel
4  }
5  rho_min = min(rho_p, rho_m);    //elementwise; use minimum for embedding
6  PIXELS_TO_CHANGE = minimize_emb_impact(LSB(X), rho_min, message)
7  Y = X;                          //start making changes in cover image
8  for (i,j) in PIXELS_TO_CHANGE { //order given by the MC visit. strategy
9     if ( model_correction_step_enabled ) {
10       Yp = Y; Yp(i,j)++; dp = D(X,Yp); Ym = Y; Ym(i,j)--; dm = D(X,Ym);
11       if ( dp<dm ) { Y(i,j)++; } else { Y(i,j)--; }
12    } else {
13       if ( rho_p(i,j)<rho_m(i,j) ) { Y(i,j)++; } else { Y(i,j)--; }
14    }
15 }
```

Fig. 3: Pseudo-code of the HUGO embedding algorithm as described in Section 4.3.

utilize it for steganalysis. By penalizing highly-populated features (in this case features extracted from pixels with low differences d_1, d_2, and d_3), we drive the algorithm to hide the message into parts of the image difficult for Eve to model. In practice, our choice of $w(d_1, d_2, d_3)$ correlates the distribution of the message bits with the local texture of the image.

Note that the distortion measure (10) is not additive in the sense of (1). This is a significant deviation from the assumptions of Theorem 1, because for this more general case near-optimal practical algorithms for minimizing such embedding impact do not exist yet. To make this measure additive, we approximate the costs of embedding change as

$$\rho_{i,j} = D(\mathbf{X}, \mathbf{Y}^{i,j}), \tag{12}$$

where $\mathbf{Y}^{i,j}$ is the stego image obtained by changing the (i,j)th pixel of cover image $\mathbf{X}$. As will be seen later, this approximation has a crucial impact on the detectability of the scheme.

4.3 Implementation details of HUGO

Figure 3 shows the pseudo-code of our implementation. On lines 1–5, the algorithm calculates distortions corresponding to modifying each pixel by ± 1 and sets the embedding cost of pixel change ($\rho_{i,j}$) to the minimum of these two numbers (for saturated pixels, there is only one choice).

Once the positions of pixel changes are determined (either by simulating the embedding by virtue of Theorem 1, or by using a practical algorithm, such as the syndrome-trellis codes [20], (function `minimize_emb_impact` on line 6 of the code)), there are two ways to ensure that the pixel's LSB communicates the message.

Without model correction: This version assumes that the assumption of the Theorem 1 holds, which means that we cannot do any better than change pixels to values determined in lines 1–5 (line 13 of the pseudo-code). The order in which the pixels are changed does not matter.

With model correction (MC): Since our distortion measure D (10) does not satisfy the assumptions of Theorem 1, we can further decrease the distortion by changing pixels to values (remember that there are two ways to match pixels' LSB to the desired bit) minimizing the overall distortion $D(\mathbf{X}, \mathbf{Y}^i)$, where $\mathbf{Y}^i$ denotes the cover image $\mathbf{X}$ after changing the ith pixel (see lines 10–11 in the pseudo-code). As will be seen in the experimental part below, the impact of model correction on the security is significant. In this case of model correction, the order in which the pixels requiring change of LSB are processed is important. In the next subsection, we experimentally evaluate the following strategies: (S1) top left to bottom right, (S2) from highest $\rho_{i,j}$ to lowest $\rho_{i,j}$, (S3) from lowest $\rho_{i,j}$ to highest $\rho_{i,j}$, (S4) random order.

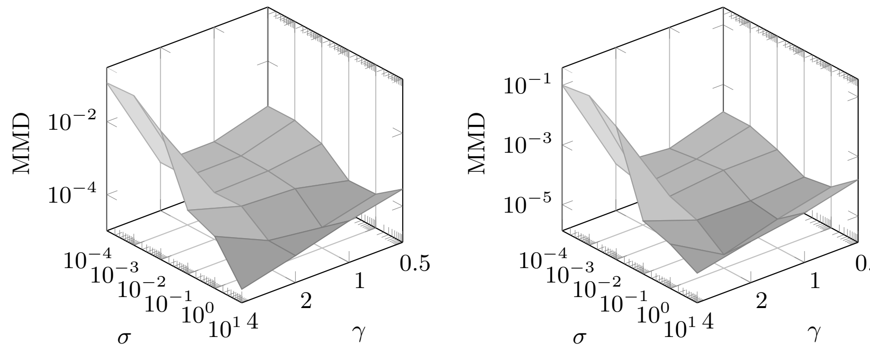
Finally we note that our implementation of HUGO in C++ with $T = 90$, the model correction step, and practical Syndrome-Trellis Code (STC) embeds message with relative length 0.25bpp to image of size 512×512 in approximately 5s on Intel Core 2 Duo 2.8 GHz processor. We consider this time more than suitable for real applications. In practice, the algorithm may need to communicate a small number of parameters in order to be able to decode the message correctly. In HUGO, we need to communicate the size of the message in order to construct the same STC code at the receiver side. This is usually done by reserving a small portion of the image based on the stego key, where a known code is used for embedding.

4.4 HUGO's maturing

The HUGO algorithm has several parameters: the range of modeled differences T, the parameters of the weight function γ and σ, and utilization of the model correction step. All these parameters need to be set before the actual use of the algorithm. Since we are not aware of any general guidance, we set them experimentally while comparing different versions of the algorithm by blind steganalysis. Although it can be argued that the parameters will be tied to the database, we prefer to see this step as tuning the algorithm to image source used by Alice and Bob.

The parameter setting proceeds as follows: (a) set the parameter T, (b) find suitable values of σ and γ in (11), (c) set the the strategy of pixel visits. In all experiments aimed to tune HUGO, the coding was simulated by virtue of Theorem 1.

The parameter T was set to $T = 90$ (the model has more then 10^7 features), causing more than 99% of the co-occurrences in the typical image to be covered by the model. By this choice of T, we strongly believe that the detectability of HUGO by SPAM features cannot be improved by increasing the range of modeled differences. In fact, our experiments showed that the increase of the



(a) Second-order SPAM (b) WAM

Fig. 4: Value of MMD (lower is better) plotted against parameters γ and σ for HUGO with model correction and S1 visiting strategy. Results for other features and even when MC step was not used were similar and are omitted due to space constraints.

range of modeled differences was not followed by a decrease of the classifier error (most probably due to the curse of dimensionality).

The search for suitable parameters of the weight function (11) was performed on a grid $(\sigma, \gamma) \in \left\{(10^k, 2^j) | k \in \{-3, \ldots, 1\}, j \in \{-1, 2\}\right\}$ for both versions of the algorithm (with and without MC). The embedding payload was fixed to 0.25bpp. In order to reduce the complexity of the search, the detectability was evaluated by means of the Maximum Mean Discrepancy [30]. Figure 4 shows the MMD values for HUGO with the MC step and S1 visiting strategy. Due to space constraints, we report graphs only for SPAM and WAM features with MC step S1. All other graphs even for the case of Hugo without MC step were of similar shape suggesting the choice parameters γ and σ to be reasonable. For all experiments presented in the rest of this section, we chose $\gamma = 4$ and $\sigma = 10$.

As we have already mentioned, the effect of the model correction on the security is substantial. For fixed classification error $P_{\mathrm{E}} = 40\%$ of an SVM-based steganalyzer utilizing second-order SPAM features, HUGO with model correction step increases the secure payload from 0.25bpp to 0.4bpp. This difference is entirely due to the fact that our distortion measure is not additive. Since we do not know yet how to do optimal coding for non-additive measures, the model correction step is in this case a reasonably good remedy.

Finally, we have compared the strategies of pixel visits S1–S4 in the model correction step by training SVM-based steganalyzer utilizing second order SPAM features. From Figure 5 (a), strategy S2 seems to be the most secure wrt the SPAM features. Model correction strategies S3 and S4 were performing slightly worse than S2 and are not displayed. These results show that the model correction step should perform embedding changes from pixels causing the largest distortion to pixels causing the least distortion.

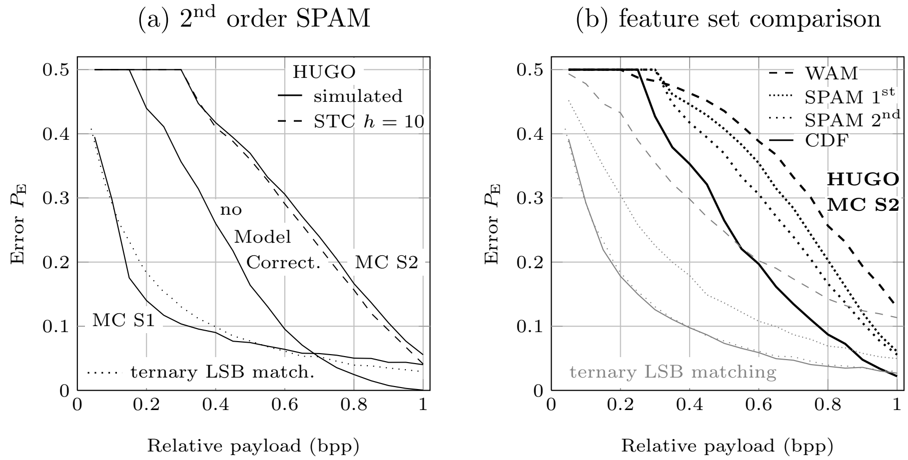


Fig. 5: (a) Comparison of security of different versions of HUGO by means of error P_E of steganalyzers utilizing second-order SPAM features with $T = 3$. (b) Comparison of different steganalytic features for detecting ordinary LSB matching with optimal ternary coding and HUGO with MC step S2. All steganalyzers are targeted to a given algorithm and message length.

4.5 HUGO's security

Figure 5 (a) compares the security of HUGO with simulated optimal coding utilizing different model correction strategies. For S2, which seems to be the best, we also report its practical implementation using syndrome-trellis code with constraint height $h = 10$ (STC) [20]. All algorithms are compared to ordinary LSB matching with optimal (simulated) ternary matrix embedding. The reported quantity P_{E} is the error of SVM-based steganalyzers. We did not compared HUGO to adaptive ternary LSB matching [9], or to MPSteg [31], because the reported improvement in the security of both schemes over standard LSB matching were not significant.

The impact of switching from the optimal (simulated) coding to the STC coder (STC) on the detectability of HUGO is also interesting and interpretable. Ideally, we would like to have code which would change each pixel with probability (2). To compare the effect of a practical coder for fixed distortion d, we evaluate the *coding loss* $l(d) = (\alpha_{\mathrm{OPT}} - \alpha_{\mathrm{ACT}})/\alpha_{\mathrm{OPT}}$, where α_{OPT} is the payload embedded by the optimal coder and α_{ACT} is the payload embedded by a practical algorithm while both of them achieve the same distortion d. Coding loss $0 \leq l(d) \leq 1$ tells us what portion of the ideal payload we are loosing due to practical embedding algorithm. For STCs, $l(d)$ was often around $3\% - 7\%$ depending on ρ and h. This finding is consistent with Figure 5 (a).

According to Figure 5 (b), HUGO offers very high security. Even for payloads as large as 0.30bpp, the error of all four steganalyzers targeted to detect HUGO with optimal coding and MC step is above 40%. It is expected that secure payload may be higher for cover sources without such strong pixel dependencies as present in BOWS2 database from scaling the original images.

Even though the improvement obtained from CDF features is significant when compared to second-order SPAM, the relative payload for which the scheme remains undetectable stays essentialy the same. This threshold may point to amount of pixels that are not modeled by either feature set (SPAM or DCT based). However, including such pixels in the steganalytic model may not be as beneficial as including them into steganographic model due to the statistical learning problem. Such pixels are expected to be part of very noisy end textured areas which will be challenging for steganalysis.

Last, but not least, if we compare HUGO with MC step S2 to the state-of-the-art LSB matching with optimal ternary coding, we can see that by using HUGO, Alice gains more than 700% of the capacity at $P_{\mathrm{E}} = 40\%$ on the BOWS2 database.

5 Conclusion

This paper presented a complete method for designing practical and secure steganographic schemes for real digital media. The main design principle is to minimize a suitably-defined distortion caused by the embedding. Since the distortion function is an essential input of the method, a large part of the paper was devoted to its design. We recommended to use weighted difference of *extended* state-of-the-art feature vectors already used in steganalysis. The extension of the feature sets, which can contain even 10^7 features, is important to avoid overfitting to a particular steganalyzer. The use of such large feature sets was justified by explaining the fundamental difference of their role in steganography and steganalysis.

The whole approach was demonstrated by designing a new steganographic algorithm for spatial domain (called HUGO), where the image model was derived from SPAM features. Parts of the model, i.e., the weights, responsible for detection of LSB matching were identified using criteria optimized in Fisher Linear Discriminant, which motivated the construction of an ad hoc distortion measure. The coding itself was performed using the syndrome-trellis codes which enable very fast implementation of the scheme in practice for arbitrary set of embedding costs ρ.

The security of HUGO was verified and compared to prior art (LSB matching) on a wide range of payloads for four different features sets. In contrast with LSB matching, HUGO allows the embedder to hide $7\times$ longer message with the same level of security level. By concrete numbers, the payload of HUGO at detection error 40% is 0.3bpp, while for LSB matching it is 0.04bpp.

6 Acknowledgements

Tomáš Filler was supported by Air Force Office of Scientific Research under the research grant FA9550-08-1-0084. 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.

Tomáš Pevný and Patrick Bas are supported by the National French project Nebbiano ANR-06-SETIN-009. Tomáš Pevný is also supported by Czech Ministry of Education grant 6840770038.

Special thanks belong to Jessica Fridrich for many interesting discussions and for proofreading the paper.

References

1. Fridrich, J.: Steganography in Digital Media: Principles, Algorithms, and Applications. Cambridge University Press (2009)
2. Cachin, C.: An information-theoretic model for steganography. In: Information Hiding, 2nd International Workshop. Volume 1525 of Lecture Notes in Computer Science., Portland, OR (April 14–17, 1998) 306–318
3. Anderson, R.: Stretching the limits of steganography. In: Information Hiding, 1st International Workshop. Volume 1174 of Lecture Notes in Computer Science., Cambridge, UK (May 30–June 1, 1996) 39–48
4. Wang, Y., Moulin, P.: Perfectly secure steganography: Capacity, error exponents, and code constructions. IEEE Transactions on Information Theory, Special Issue on Security **55**(6) (June 2008) 2706–2722
5. Ryabko, B., Ryabko, D.: Asymptotically optimal perfect steganographic systems. Problems of Information Transmission **45**(2) (2009) 184–190
6. Ker, A.D., Böhme, R.: Revisiting weighted stego-image steganalysis. In: Proceedings SPIE, EI, Security, Forensics, Steganography, and Watermarking of Multimedia Contents X. Volume 6819., San Jose, CA (January 27–31, 2008) 5 1–5 17
7. Filler, T., Fridrich, J.: Fisher information determines capacity of ϵ-secure steganography. In: Information Hiding, 11th International Workshop. Volume 5806 of Lecture Notes in Computer Science., Darmstadt, Germany (June 7–10, 2009) 31–47
8. Harmsen, J.J., Pearlman, W.A.: Steganalysis of additive noise modelable information hiding. In: Proceedings SPIE, EI, Security and Watermarking of Multimedia Contents V. Volume 5020., Santa Clara, CA (January 21–24, 2003) 131–142
9. Goljan, M., Fridrich, J., Holotyak, T.: New blind steganalysis and its implications. In: Proceedings SPIE, EI, Security, Steganography, and Watermarking of Multimedia Contents VIII. Volume 6072., San Jose, CA (2006) 1–13
10. Pevný, T., Bas, P., Fridrich, J.: Steganalysis by subtractive pixel adjacency matrix. In: Proceedings of the 11th ACM Multimedia & Security Workshop, Princeton, NJ (September 7–8, 2009) 75–84
11. Fridrich, J., Filler, T.: Practical methods for minimizing embedding impact in steganography. In: Proceedings SPIE, EI, Security, Steganography, and Watermarking of Multimedia Contents IX. Volume 6505., San Jose, CA (January 29–February 1, 2007) 02–03
12. Westfeld, A.: High capacity despite better steganalysis (F5 – a steganographic algorithm). In: Information Hiding, 4th International Workshop. Volume 2137 of Lecture Notes in Computer Science., Pittsburgh, PA (April 25–27, 2001) 289–302

13. Fridrich, J., Pevný, T., Kodovský, J.: Statistically undetectable JPEG steganography: Dead ends, challenges, and opportunities. In: Proceedings of the 9th ACM Multimedia & Security Workshop, Dallas, TX (September 20–21, 2007) 3–14
14. Sallee, P.: Model-based steganography. In: Digital Watermarking, 2nd International Workshop. Volume 2939 of Lecture Notes in Computer Science., Seoul, Korea (October 20–22, 2003) 154–167
15. Kim, Y., Duric, Z., Richards, D.: Modified matrix encoding technique for minimal distortion steganography. In: Information Hiding, 8th International Workshop. Volume 4437 of Lecture Notes in Computer Science. (2006) 314–327
16. Sachnev, V., Kim, H.J., Zhang, R.: Less detectable JPEG steganography method based on heuristic optimization and BCH syndrome coding. In: Proceedings of the 11th ACM Multimedia & Security Workshop. (September 7–8, 2009) 131–140
17. Fridrich, J., Goljan, M., Soukal, D.: Perturbed quantization steganography. ACM Multimedia System Journal **11**(2) (2005) 98–107
18. Franz, E., Rönisch, S., Bartel, R.: Improved embedding based on a set of cover images. In: Proceedings of the 11th ACM Multimedia & Security Workshop, Princeton, NJ (September 7–8, 2009) 141–150
19. Franz, E.: Steganography preserving statistical properties. In: Information Hiding, 5th International Workshop. Volume 2578 of Lecture Notes in Computer Science., Noordwijkerhout, The Netherlands, Springer-Verlag (October 7–9, 2002) 278–294
20. Filler, T., Fridrich, J., Judas, J.: Minimizing embedding impact in steganography using Trellis-Coded Quantization. In: Proceedings SPIE, EI, Media Forensics and Security XII, San Jose, CA (January 18–20, 2010) 05 1–05 14
21. Crandall, R.: Some notes on steganography. Steganography Mailing List, available from `http://os.inf.tu-dresden.de/~westfeld/crandall.pdf` (1998)
22. Zhang, X., Zhang, W., Wang, S.: Efficient double-layered steganographic embedding. Electronics Letters **43** (April 2007) 482–483
23. Kodovský, J., Fridrich, J.: On completeness of feature spaces in blind steganalysis. In: Proceedings of the 10th ACM Multimedia & Security Workshop, Oxford, UK (September 22–23, 2008) 123–132
24. Ullerich, C., Westfeld, A.: Weaknesses of MB2. In Shi, Y.Q., Kim, H.J., Katzenbeisser, S., eds.: Digital Watermarking. 6th International Workshop, IWDW 2007, Guangzhou, China (December 3–5 2007) 127–142
25. Kodovský, J., Pevný, T., Fridrich, J.: Modern steganalysis can detect YASS. In: Proceedings SPIE, EI, Media Forensics and Security XII, San Jose, CA (2010)
26. Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.A.: Feature Extraction, Foundations and Applications. Springer (2006)
27. Ker, A.D., Pevný, T., Kodovský, J., Fridrich, J.: The Square Root Law of steganographic capacity. In: Proceedings of the 10th ACM Multimedia & Security Workshop, Oxford, UK (September 22–23, 2008) 107–116
28. Filler, T., Ker, A.D., Fridrich, J.: The Square Root Law of steganographic capacity for Markov covers. In: Proceedings SPIE, EI, Security and Forensics of Multimedia XI. Volume 7254., San Jose, CA (January 18–21, 2009) 08 1–08 11
29. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag, New York (1995)
30. Pevný, T., Fridrich, J.: Benchmarking for steganography. In: Information Hiding, 10th International Workshop. Volume 5284 of Lecture Notes in Computer Science., Santa Barbara, CA (June 19–21, 2008) 251–267
31. Cancelli, G., Barni, M., Menegaz, G.: Mpsteg: hiding a message in the matching pursuit domain. In: Proceedings SPIE, EI, Security, Steganography, and Watermarking of Multimedia Contents VIII. Volume 6072., San Jose, CA (2006) 60720P