Feature Reduction and Payload Location with WAM Steganalysis

Andrew D. Ker and Ivans Lubenko

Oxford University Computing Laboratory, Parks Road, Oxford OX1 3QD, England.

ABSTRACT

WAM steganalysis is a feature-based classifier for detecting LSB matching steganography, presented in 2006 by Goljan et al. and demonstrated to be sensitive even to small payloads. This paper makes three contributions to the development of the WAM method. First, we benchmark some variants of WAM in a number of sets of cover images, and we are able to quantify the significance of differences in results between different machine learning algorithms based on WAM features. It turns out that, like many of its competitors, WAM is not effective in certain types of cover, and furthermore it is hard to predict which types of cover are suitable for WAM steganalysis. Second, we demonstrate that only a few the features used in WAM steganalysis do almost all of the work, so that a simplified WAM steganalyser can be constructed in exchange for a little less detection power. Finally, we demonstrate how the WAM method can be extended to provide forensic tools to identify the location (and potentially content) of LSB matching payload, given a number of stego images with payload placed in the same locations. Although easily evaded, this is a plausible situation if the same stego key is mistakenly re-used for embedding in multiple images.

Keywords: Steganalysis, LSB Matching, Wavelet Absolute Moments, Feature Reduction, Image Forensics

1. INTRODUCTION

Unlike Least Significant Bit (LSB) replacement, for which highly sensitive detectors exist, LSB matching, also known as ±1 embedding, has proved a difficult steganalysis challenge. Harmsen *et al.* presented a detector based on the Histogram Characteristic Function (HCF)[1] for which a number of performance enhancements have been presented,[2–4] and a detector based on occurrence of histogram extrema was recently proposed,[5] but none of these detectors' sensitivity comes near that of the so-called structural[6,7] and WS[8] detectors for LSB replacement. Another style of detector, for LSB matching as well as some other embedding methods, was suggested by Farid:[9,10] these steganalysers train a learning machine on a feature set. The feature set is chosen to measure the predictability of an image; embedded additive noise is not predictable, whereas the cover should be mostly predictable. This can be seen as the converse of a denoising operation, attempting to remove the cover content to unmask the stego signal. Steganalysis of LSB matching using Wavelet Absolute Moments (WAM)[11] is a detector in this style, with a feature set involving noise residuals in the wavelet domain. Compared with other LSB matching detectors, it is reported in Ref. 11 that WAM exhibits highly superior performance. Since then, however, there has been little published work on WAM steganalysis.

This paper studies and extends the WAM method. We begin by considering the classifier, replacing the original Fisher Linear Discriminator (FLD) by a Support Vector Machine (SVM) and Neural Network (NN). With some statistical techniques, not widely used in the steganalysis literature, we are able to check whether the accuracy improvements are statistically significant or not. We also demonstrate that the performance of WAM steganalysis depends heavily on the type of cover, a difficulty which seems to affect all LSB matching detectors (Sect. 2). Then we consider the features on which the WAM method is based, and demonstrate that they are highly correlated: most of the detection power can be preserved if only a small number of features are used, but again there is large variability according to the cover source (Sect. 3). Finally, we consider the WAM noise residuals themselves as a tool for forensic analysis. It is possible to identify the location (and potentially content) of LSB matching payload, given a number of stego images with the payload in the same locations (Sect. 4). There follows a brief conclusion (Sect. 5).

Further author information:

A. D. Ker: E-mail: adk@comlab.ox.ac.uk, Telephone: +44 1865 283530

1.1 LSB MATCHING AND WAM STEGANALYSIS

LSB replacement is perhaps the simplest digital steganography scheme, which embeds a bitstream payload into a cover by replacing the least significant bits of cover samples with successive payload bits. Such embedding is surprisingly insecure because of structure in the parity of the embedding process,[6] so LSB matching modifies the method to remove all such structure: the payload is still carried in the LSBs of the stego object, but when a cover sample is altered it is incremented or decremented randomly (unless already at the extreme of its range). For byte-valued samples such as grayscale digital images, or colour channels in colour images, the embedding operation can be described by the function

$$x_i \mapsto \begin{cases} x_i - 1, & \text{if } m_i \neq \mathrm{LSB}(x_i) \text{ and } x_i = 255 \\ x_i, & \text{if } m_i = \mathrm{LSB}(x_i) \\ x_i + 1, & \text{if } m_i \neq \mathrm{LSB}(x_i) \text{ and } x_i = 0 \\ x_i \pm 1, & \text{otherwise, equiprobably} \end{cases}$$

where x_i represents the i-th cover byte, and m_i the i-th payload bit. For security, the cover is traversed in a pseudorandom order generated by a secret key presumed shared between sender and recipient.

The WAM steganalyser in Ref. 11 aims to detect the presence of LSB matching payload in a digital image; we now give a brief recapitulation. The method involves computing the residuals from a quasi-Wiener filter: for a two-dimensional signal $\boldsymbol{S}$,

$$\mathcal{R}[\boldsymbol{S}] = \frac{\sigma_0^2 \boldsymbol{S}}{\sigma_0^2 + \boldsymbol{v}}$$

where σ_0^2 is the noise variance (for LSB matching affecting every pixel, 0.5), and v_i is a MAP estimate of the local variance of element i, based on windows of four different sizes:

$$v_i = \max\bigl(0,\ \min(v_i^3, v_i^5, v_i^7, v_i^9) - \sigma_0^2\ \bigr) \quad \text{where} \quad v_i^N = \frac{1}{N^2} \sum_j S_j^2.$$

(for v_i^N, j is summed over the $N \times N$ neighbourhood of pixel i).

The residuals are computed in the wavelet domain: given an input grayscale image $\boldsymbol{X}$ whose pixel locations are a set I, calculate a one-level wavelet decomposition using the 8-tap Daubechies filter. Discard the low-frequency subband, and denote the horizontal, vertical, and diagonal subbands $\boldsymbol{H}$, $\boldsymbol{V}$ and $\boldsymbol{D}$. Write $\mathcal{R}[\boldsymbol{H}]_i$ for the i-th coefficient of the filtered horizontal residual, respectively $\mathcal{R}[\boldsymbol{V}]_i$ and $\mathcal{R}[\boldsymbol{D}]_i$. The WAM features are the absolute central moments of the coefficients in these filtered subbands:

$$A_m^H = \frac{1}{|I|} \sum_{i \in I} \Bigl| \mathcal{R}[\boldsymbol{H}]_i - \overline{\mathcal{R}[\boldsymbol{H}]} \Bigr|^m, \quad A_m^V = \frac{1}{|I|} \sum_{i \in I} \Bigl| \mathcal{R}[\boldsymbol{V}]_i - \overline{\mathcal{R}[\boldsymbol{V}]} \Bigr|^m, \quad A_m^D = \frac{1}{|I|} \sum_{i \in I} \Bigl| \mathcal{R}[\boldsymbol{D}]_i - \overline{\mathcal{R}[\boldsymbol{D}]} \Bigr|^m$$

The first nine moments in each subband, $\{A_1^H, \ldots, A_9^H, A_1^V, \ldots, A_9^V, A_1^D, \ldots, A_9^D\}$ form the 27-dimensional feature vectors used in Ref. 11. WAM features can also be computed for colour images, treating each colour component separately, but we will restrict ourselves to grayscale images in this work.

(We also experimented with some additional WAM-style features, including higher and signed moments, but they did not give a significant improvement. For more detail see the second author's MSc thesis.[12])

In Ref. 11 a Fisher Linear Discriminator (FLD) is trained on the features of some cover and stego images, to make a detector for the presence of LSB matching steganography. Extremely sensitive detection is reported, with accuracy of around 90% when the LSB matching payload is 25% of the maximum (0.25 bits per pixel), and near-perfect detection with payloads at 50%. These are compared very favourably with the detectors in Refs. 2 and 13. We will benchmark WAM further in the next section.

2. INFLUENCE OF COVER TYPE

Detectors for LSB matching seem to have uneven performance. For example, the improved HCF COM detectors in Ref. 2 work well on the test images used in that paper, but the first author is now aware that the same performance is not found in many other situations. Similarly, a recent study[14] of three LSB matching steganalysers in three image sets found wide variations in both absolute and relative performance. This raises two questions regarding WAM steganalysis: how much does its performance vary (in particular, are the results in Ref. 11 typical), and can we identify the properties of covers which predict the reliability of LSB matching detectors? So our first set of experiments is to benchmark variants of WAM steganalysis in multiple test cover sets, nine in all; the focus is solely on WAM detectors, and we will postpone comparisons with other LSB matching steganalysers to future work. We will see that the accuracy of WAM steganalysis does vary, greatly, depending on the type of cover, and we briefly investigate whether the performance can be predicted by macroscopic cover properties.

We will test three variants of WAM steganalysis, using the same features with three different classification engines: the original FLD, a Multilayer Perceptron or Neural Network (NN), and a Support Vector Machine (SVM) employing the kernel trick. Each involves training the machine against a set of known cover and stego images, and then testing the trained machine against fresh data.

The FLD is a simple classification method which finds an optimal linear projection of the features: given training data vectors $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$ of cover features and $\boldsymbol{x'_1}, \ldots, \boldsymbol{x'_n}$ of stego features*, it finds the linear projection $\boldsymbol{y} \mapsto \boldsymbol{w} \cdot \boldsymbol{y}$ to maximise the separation between the classes,

$$\frac{\left\| \boldsymbol{w} \cdot \bar{\boldsymbol{x}} - \boldsymbol{w} \cdot \bar{\boldsymbol{x}'} \right\|^2}{S_{\boldsymbol{w},\boldsymbol{x}} + S_{\boldsymbol{w},\boldsymbol{x}'}}$$

where $\bar{\boldsymbol{x}} = \frac{1}{n} \sum_i \boldsymbol{x_i}$ is the class mean, and $S_{\boldsymbol{w},\boldsymbol{x}} = \sum_i (\boldsymbol{w} \cdot \boldsymbol{x_i} - \boldsymbol{w} \cdot \bar{\boldsymbol{x}})^2$ a measure of projected class scatter, similarly $\bar{\boldsymbol{x}'}$ and $S_{\boldsymbol{w},\boldsymbol{x}'}$. Thus the difference between the projected class means, suitably scaled according to the total scatter, is as large as possible (and this is optimal for certain multivariate Gaussian classification problems). Advantages of the FLD method include reasonably fast training (the optimisation problem can be solved by matrix multiplications), very fast use (a single dot product produces the projected value, on which a simple threshold is set), and no training parameters to select. We used our own implementation of FLD for these experiments.

The Multilayer Perceptron, commonly called Neural Network (NN), can find nonlinear classification boundaries. It consists of a number of layers of nodes with nonlinear activation functions, connected by a weighted graph. The weights in the graph can be trained by backpropagation,[15] an iterative algorithm too complex to explain here, which adjusts the weights to reduce the classification errors. Backpropagation has a number of training parameters, which affect the speed of adjustment: a learning rate η and momentum α; we used the WEKA[16] implementation of the Multilayer Perceptron for our experiments, which constructs the perceptron network automatically. The iterative training process can be very time-consuming, but the application of a trained NN to classify a new object is quite fast.

A Support Vector Machine (SVM), in its simple linear form (LVSM), can only find linear class boundaries: it works in a similar way to linear discriminators, but the two classes are defined by two parallel hyperplanes, $\{\boldsymbol{x} \mid \boldsymbol{w} \cdot \boldsymbol{x} > 1\}$ and $\{\boldsymbol{x} \mid \boldsymbol{w} \cdot \boldsymbol{x} < -1\}$. The hyperplanes are chosen to minimize $\boldsymbol{w} \cdot \boldsymbol{w} + Cs$, where s is the sum of the shortest distance of incorrectly-classified points to their correct class region (as a quadratic programming (QP) problem, this can be done with high efficiency). Minimizing the first term is equivalent to maximizing the distance between the classification regions and C is a parameter which adjusts the weight between the two goals of maximum margin and minimum classification errors. An advantage of SVMs is that they can be adapted, very simply, to find nonlinear classification boundaries using the kernel trick,[17] whereby the standard dot product is replaced by a nonlinear function. After some experimentation, we opted to use the polynomial kernel $k(\boldsymbol{x}, \boldsymbol{x'}) = (\boldsymbol{x} \cdot \boldsymbol{x'} + 1)^e$, where e is a constant. For more on SVMs, see Ref. 18. We used the WEKA[16] implementation of soft-margin SVMs for our experiments. Even with efficient QP algorithms it can take hours to perform the optimisation, but it then becomes fast to classify new objects, requiring only the evaluation of a

*More generally, the two training sets need not be the same size, though in our experiments they will always be so.

few kernel functions (how many such evaluations depends on the number of so-called support vectors, which in turn is affected by the parameter C).

We will test WAM features, classified by FLD, NN, and SVM, in each of nine image sets, chosen to represent a variety of types. These sets are not entirely independent because some of them come from the same sources but have been subject to different preprocessing. Where necessary the images have been cropped to 400×300 from larger sizes (the crop regions chosen randomly); this removes cover size as a factor in the results. Each set (except one) contains 2000 grayscale images. When the parent set was smaller than 2000, more than one region was cropped out of each image to make the total up to 2000. The sets are labelled A to I:

Set A: RAW images taken with a Minolta DiMAGE A1 camera, originally 5Mpixels in size; converted to grayscale, but not denoised, using the Minolta software, and subsequently denoised using Photoshop. This avoids any colour filter array interpolation.

Set B: Identical images to set A, but converted from RAW to *colour* images, and also denoised, using the Minolta software with default options. Subsequently converted to grayscale.

Set C: Images from a mixture of digital cameras supplied by the researchers at Binghamton University, a subset of those used in Ref. 11. The images were 1.5Mpixels in size, from 16 different camera models. They have never been compressed, and were converted from colour to grayscale.

Set D: Images from a photo library CD, all stored as colour JPEGs with quality factor 50 and sized 540Kpixels. Decompressed and converted to grayscale.

Set E: Scanned images downloaded from the NRCS website,[19] originally around 3Mpixels but downsampled to around 300Kpixels to reduce scanner noise, then converted to grayscale.

Set F: High-quality JPEG images taken with a Fuji Finepix 6900 camera, originally around 6Mpixels; downsampled to 400×300 and converted to grayscale. There are only 1225 images in this set.

Set G: The same images as set F, but cropped down to 400×300 instead of being downsampled.

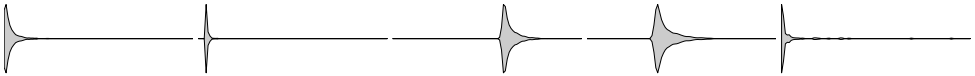
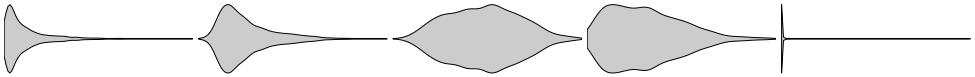
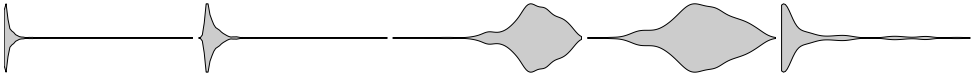
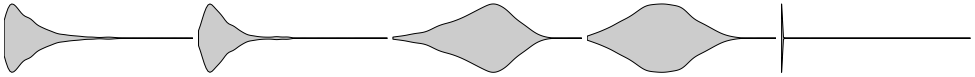
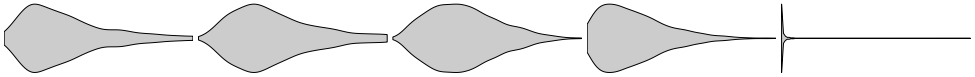
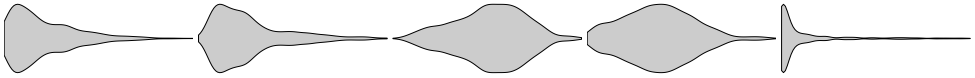
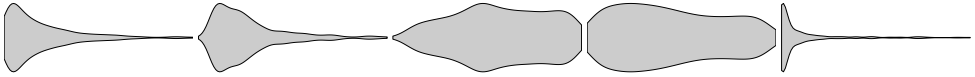
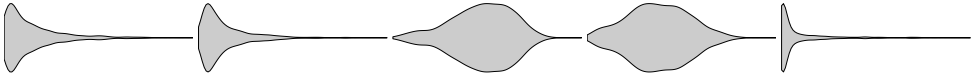
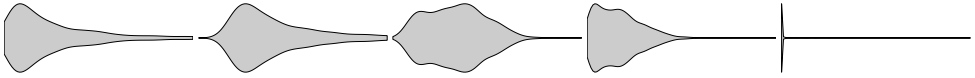
Set H: A mixed set of JPEG files downloaded from a variety of photo sharing websites. The images were a mixture of sizes, uploaded by many different users, and the preprocessing is completely unknown.

Set I: Images downloaded from the McGill Calibrated Colour Image Database.[20] They have been downsampled, and there is some preprocessing related to colour calibration.

The FLD has no training parameters, but the SVM has two (the polynomial kernel exponent e and the misclassification weight C) and the NN has two (learning rate η and momentum α). A grid search was employed to find parameters that yielded best accuracy for both algorithms: to make a fair comparison, the same amount of computational effort was used in tuning SVM and NN. In the grid search, we tried five values for each parameter: $C \in \{0.01, 0.1, 1, 10, 100\}$, $e \in \{1, 2, 3, 4, 5\}$, $\eta \in \{0.03, 0.1, 0.3, 0.6, 1\}$, $\alpha \in \{0.02, 0.2, 0.4, 0.8, 1\}$. Each algorithm was benchmarked by its ten-fold cross-validation *minimum error probability*: we generated ROC curves for each of the ten cross-validation folds for each machine and found the point on the curve to maximize the *accuracy* value $1 - P_E = 1 - \frac{1}{2}\min(P_{FP} + P_{FN})$, where P_{FP} and P_{FN} are the observed false positive and negative error proportions. Fixing on a payload of 0.5 bits per pixel, 50% of maximum, the results for each classifier and each image set are displayed in Tab. 1. We also chart the distribution of some macroscopic properties of these image sets: a measure of noisiness (local variance), unpredictability of pixels in the spatial domain (residuals under the WS steganalysis method[8]), the noisiness in the high-frequency wavelet subbands (mean absolute coefficient in $\mathcal{R}[\boldsymbol{D}]$), one of the key WAM features A_2^D, and the proportion of saturated (0 or 255) pixels. These properties are displayed as density estimates and no scales are included, but the scales are constant in each column of the table: this is sufficient to allow comparison.

These results demonstrate that WAM steganalysis is hugely variable in its detection power. This variation is equally observable for each of the three WAM classifiers in question: depending on the source of cover image, the performance changes from being almost perfect (set A) to barely better than random guessing (set I). Moreover, inaccuracy is not simply (as one might expect) due to noisy or unpredictable images, because the images in set C are not as noisy as those in set D yet show lower accuracy, and images in set H are less predictable

Table 1. Performance of WAM steganalysers (measured via ten-fold cross-validation) in nine sets of images. Training and testing was performed with 50% payload (0.5bpp). All images are the same size. Violin plots (density estimates) of five macroscopic image properties are displayed for each set.

Set	Source	Local Variance	Mean Absolute WS Residual	A_2^D	Mean Absolute $\boldsymbol{D}$-Residual	Saturated Pixels	WAM Accuracy		
							FLD	NN	SVM
A	digital camera RAW never-compressed, pre-processed as grayscale						100%	100%	100%
B	digital camera RAW never-compressed, pre-processed as colour						69.7%	73.4%	75.8%
C	digital cameras RAW never-compressed, unknown pre-processing						80.6%	89.2%	90.4%
D	photo library decompressed JPEGs, quality factor 50						95.5%	97.7%	97.5%
E	scanner TIFF never-compressed, downsampled						60.9%	64.3%	64.7%
F	digital camera high quality JPEGs, downsampled						74.2%	77.2%	77.1%
G	digital camera high quality JPEGs, cropped						69.6%	71.7%	71.4%
H	internet mixed JPEGs, downsampled & cropped						97.3%	98.0%	98.1%
I	image database unknown pre-processing, includes colour calibration						53.2%	56.1%	55.1%

than those in set C yet show better accuracy. Rather it seems to be because some sets of covers have very *consistent* predictability or noise. The proportion of saturated pixels does not appear to predict accuracy, either. Unfortunately, this means that it is difficult to know whether WAM steganalysis is appropriate for a given image. We have been unable to find any macroscopic properties of images which predict whether WAM steganalysis will be near-perfect (as for sets A and H) or near-worthless (as for sets E and I), and this limits its applicability. It is particularly surprising that set H gave such good results, given the eclectic nature of its source, but perhaps this is related to JPEG compression.

When WAM was introduced in Ref. 11, it was noted that additional information about the cover source can increase detection accuracy, but none of the image sets tested in Ref. 11 exposes the weakness in performance. What is most curious is the discrepancy between our image sets A and B. They contain *the same photographs*, i.e. derived from the same set of RAW digital camera images. But while image set A was produced by converting the RAW images to grayscale TIFF files then applying a mild denoising filter, image set B was produced by converting the RAW images to colour TIFFs (which involves a different denoising filter, and demosaicing of the camera's colour filter array) and then converting to grayscale. Clearly the precise parameters of the denoising and/or grayscale conversion processes can make a huge difference to the reliability of WAM steganalysis. Further research is needed to explain this.

We also performed some similar experiments for other payloads, with consistent results.

The figures in Tab. 1 suggest that WAM's detection accuracy varies when it is tested on the same image set using a different classifier. To establish whether these discrepancies are meaningful (and ultimately, which classifier is better for this problem), we find the statistical significance of differences using Student's t-test. The data for the test come from the performance of the classifiers in the ten individual folds of the ten-fold benchmark, but note that these figures cannot be independent because they involve substantially overlapping training data. Therefore we employ the *corrected resampled Student's t-test*,[16] a variation that was designed to take account of the dependency between the samples and is ideal for cross-validation data. In the case of k-fold cross-validation, the adjusted t-statistic is given by

$$t = \frac{\bar{x} - \bar{y}}{\sqrt{\left(\frac{1}{k} + \frac{1}{k-1}\right)\left(\frac{\sigma_x^2}{k} + \sigma_y^2\right)}}, \tag{1}$$

where $\bar{x}$ and $\bar{y}$ are the mean reliability scores across the k folds for algorithms A and B, and σ_x^2 and σ_y^2 the sample variances within the k folds. (When the reliability scores are close to 100% it may be wiser to convert them to log odds, but there seems no need to do so for our data.) When the null hypothesis—that algorithms A and B give rise to the same reliability—holds then the t-statistic follows a t-distribution with $k-1$ degrees of freedom, and thereby significance tests can be constructed.

Table 2. Results of adjusted t-tests for comparison of detector performance, in each image set. Data from ten-fold cross validation. For two-sided tests, $|t| > 2.26$ indicates significance at the 5% level, $|t| > 3.25$ at the 1% level, and $|t| > 4.78$ at the 0.1% level.

Image Set	WAM accuracy			adjusted t scores for	
	FLD	NN	SVM	NN > FLD	SVM > NN
A	100%	100%	100%	–	–
B	69.7%	73.4%	75.8%	3.57	3.94
C	80.6%	89.2%	90.4%	22.7	1.56
D	95.5%	97.7%	97.5%	13.3	-0.58
E	60.9%	64.3%	64.7%	9.70	0.62
F	74.2%	77.2%	77.1%	5.85	-0.12
G	69.6%	71.7%	71.4%	6.33	-0.49
H	97.3%	98.0%	98.1%	3.79	0.14
I	53.2%	56.1%	55.1%	3.65	-0.77

The t scores arising from our experiments, comparing the performance of FLD and NN, and then NN and SVM, are displayed in Tab. 2. Because there is no *a priori* reason to suppose that one machine learning method is better than another, the critical values correspond to two-sided hypothesis tests. It is clear that the FLD (as originally used in Ref. 11) is significantly inferior to the other two machines, but the difference between NN and SVM is only significant in the case of set B (when it favours the SVM classifier). More generally, this table indicates that authors should be most cautious about interpreting performance improvements of, say, 1% in classification accuracy: depending on the context, it is entirely possible for such a difference to be due to statistical noise instead of a genuine improvement. If cross-validation has already been performed, the corrected resampled t-test is a cheap way to check the significance of any difference.

3. WAM FEATURE REDUCTION

The WAM features are far from independent: a high value of A_1^H ensures a high value of A_i^H, for all i, for example. One way to quantify the dependency is to perform Principal Component Analysis (PCA) on the feature data. We observed that between 1 and 3 eigenvectors (depending on the set) were sufficient to account for 90% of the total eigenvalues, and between 3 and 5 eigenvectors for 99%; this is true for cover and stego images. We interpret these numbers as indicating that the true dimensionality of the features is effectively only in low single figures. Hence we suspect that the WAM feature set can be reduced, without removing too much information. PCA can, of course, provide a feature reduction, but (apart from the fact that it only finds linear dependencies) it transforms the entire feature set into another with lower dimensionality and this does not reduce the computational complexity of extracting the features in the first place, so we take another approach.

Greedy forward and backward search are feature selection methods that consider the fitness of a subset of features using the underlying classifier. Forward search begins with zero features, then iteratively scans through the feature set adding each new feature that gives the best accuracy. Backward search starts with the full set and iteratively eliminates the weakest feature. Both strategies are greedy and consider only local changes in performance through the addition or elimination of competing features to or from the current best feature set: this is a weakness, because it is possible that two features *in combination* provide good class separation, but neither separately does so. Both methods require $O(N^2)$ train-and-test iterations, where N is the number of features. The optimal feature selection method, of course, is to try every possible subset, but this requires $O(2^N)$ iterations and is infeasible even for our modest feature set with $N = 27$.

A further complication with feature selection is the tuning of training parameters: potentially, different parameters might be needed for each different combination of features. The means that some sort of parameter optimization must be carried out for each step of the feature reduction process, and this may require too much computation to be feasible. Some experiments with WAM feature reduction in SVMs and NNs can be found in Ref. 12, but we will avoid the tuning problem by using the FLD classifier for feature selection: it requires no learning parameters. It is also conveniently fast both to train and to test.

Keeping to a payload of 0.5 bpp, ten-fold cross-validation, and the minimum error probability metric, we ran forward and backward selection algorithms on the WAM features in each of the nine image sets. We report the results of the forward selection, for sets A to D, in Fig. 1 (backward selection results were rather similar, and the other image sets all showed similar shape to one of those displayed). In all cases, except set B, the performance reaches nearly that of the full 27 features within about four or five features: unfortunately, it is a different set of features which are selected for each cover set, so this does not suggest a universally-useful reduced feature set.

It is just about feasible, using the FLD, to test all $\binom{27}{4}$ combinations of four features: we tested them against all image sets and ranked the selections by aggregate performance. The result of this experiment was that the attractively-symmetrical set $\{A_1^H, A_1^V, A_1^D, A_6^D\}$ gave the best overall performance, so we tried more experiments with the individual cover sets, benchmarking FLD and tuned SVM (NN was omitted) steganalysers based on these four features. The results are displayed in Tab. 3. They are somewhat disappointing: although a simplified WAM steganalyser can be constructed (and it is computationally less complex, because there is no need to compute so many moments) it does lose a significant amount of detection power. Once again the variability of WAM, with different cover sources, weakens the value of this detector.

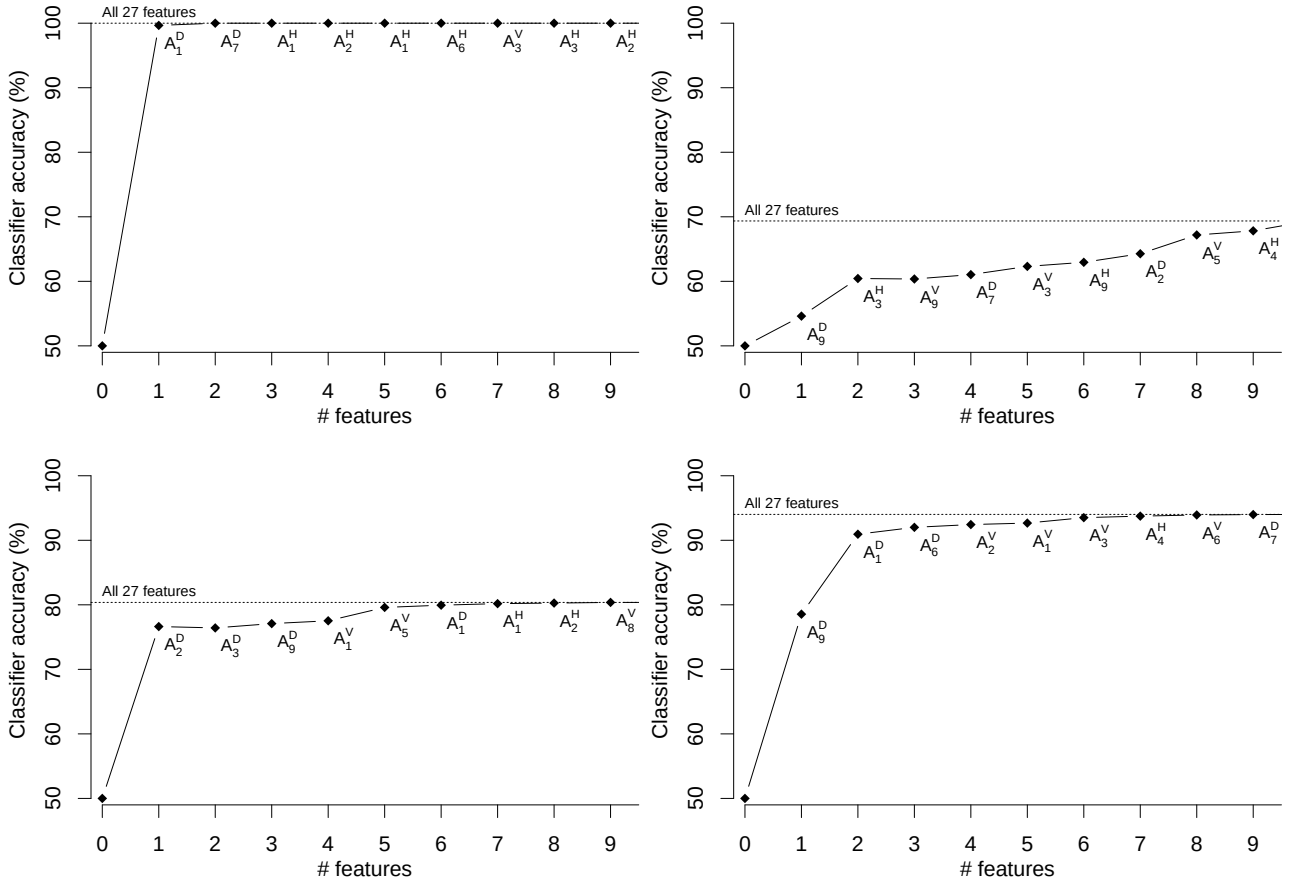


Figure 1. Classification accuracy (ten-fold cross-validation) in four sets of cover and stego objects (set A, top left, set B, top right, set C, bottom left, set D, bottom right), as the first 9 features are added by greedy forward selection. The features added at each stage are marked on the charts.

Table 3. Accuracy of classifiers (measured via ten-fold cross validation) for FLD and SVM engines, comparing the full 27-dimensional feature set with a 4-dimensional subset.

Image Set	27 features		4 features	
	FLD	SVM	FLD	SVM
A	100%	100%	100%	100%
B	69.7%	75.8%	62.7%	67.6%
C	80.6%	90.4%	76.2%	83.2%
D	95.5%	97.5%	92.1%	94.3%
E	60.9%	64.7%	55.5%	57.1%
F	74.2%	77.1%	67.5%	68.9%
G	69.6%	71.4%	65.4%	65.5%
H	97.3%	98.1%	91.0%	93.5%
I	53.2%	55.1%	53.6%	54.1%

4. LOCATING PAYLOAD

In recent work[21] we used the residuals of a pixel predictor to perform forensic analysis on stego images. The scenario is as follows: we assume that the steganalyst has a number of stego images; the payloads in each image are different, but *placed in the same locations every time.* Of course this assumption is quite strong, and requires the steganographer to have made a mistake in their embedding process, but it is not implausible. If either the embedding software neglects to randomise the payload locations, or the embedder re-uses the same embedding key for multiple cover images, then this situation arises.

The technique in Ref. 21 was to average the residuals across all stego images, for each pixel individually. It uses so-called *WS residuals*, which derive from WS steganalysis[8,22] and are specific to LSB replacement embedding: because of the parity structure in LSB replacement, LSB flipped pixels tend to have a higher WS residual than those not flipped. LSB matching has no such structure, but we can adapt the WAM residuals for the same purpose.

If the WAM filter succeeds in removing the cover content, we would expect to see larger absolute values for residuals corresponding to payload locations, than for residuals corresponding to unaltered cover locations: this is because the stego noise is unmasked. However the residuals whose moments are computed as WAM features, $\mathcal{R}[\boldsymbol{H}]$, $\mathcal{R}[\boldsymbol{V}]$, and $\mathcal{R}[\boldsymbol{D}]$, do not correspond to individual pixel locations because they are in the wavelet domain. Indeed, some simple experiments do show an association between residual magnitude and payload location, but convolved by the wavelet filter. In order to locate payload, we must convert the WAM residuals back into the spatial domain: this suggests inverting the wavelet filter, while suppressing as much of the predictable cover content as possible.

So for each input grayscale image $\boldsymbol{X}$ we propose the following procedure: calculate the one-level wavelet decomposition to produce low-frequency, horizontal, vertical, and diagonal subbands $\boldsymbol{L}$, $\boldsymbol{H}$, $\boldsymbol{V}$, and $\boldsymbol{D}$; zero out the low-frequency component, and apply the WAM filter to the other subbands; obtain the spatial domain residuals $\boldsymbol{X'}$ as the inverse wavelet transform of $\mathbf{0}, \mathcal{R}[\boldsymbol{H}], \mathcal{R}[\boldsymbol{V}], \mathcal{R}[\boldsymbol{D}]$.

Now we estimate the size of the residuals in cover images, compared with stego noise. Figure 2 shows a histogram of residuals across all cover images in sets A and B (these were chosen as examples of images where WAM steganalysis is particularly easy, and difficult, respectively). We can estimate how much of the stego noise survives the filter by embedding payload in flat gray images: because the noise interacts the precise response depends on the payload size, but we see a good estimate on the right of Fig. 2 which is the result of embedding random payloads of 50%. Observe that the stego noise is not sufficiently high, compared with residual cover noise, for us to identify exactly which pixels contain payload in a single image. But in the scenario when multiple stego images locate payload in the same place, following Ref. 21, we can compute the average absolute residual $|\boldsymbol{X'}|$, across the set of stego images, for each pixel location. Given enough images, we expect the cover noise to wash out, and to see detectably higher values for pixels subject to LSB matching embedding, than for those not used for embedding.

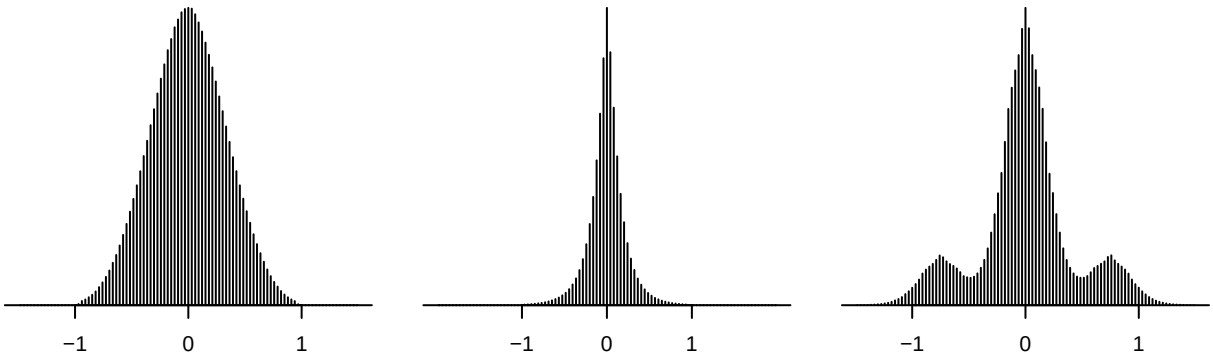


Figure 2. Histograms of WAM residuals for cover images in sets A, left, and B, center, as well as an estimate of the distribution of filtered stego noise, right.

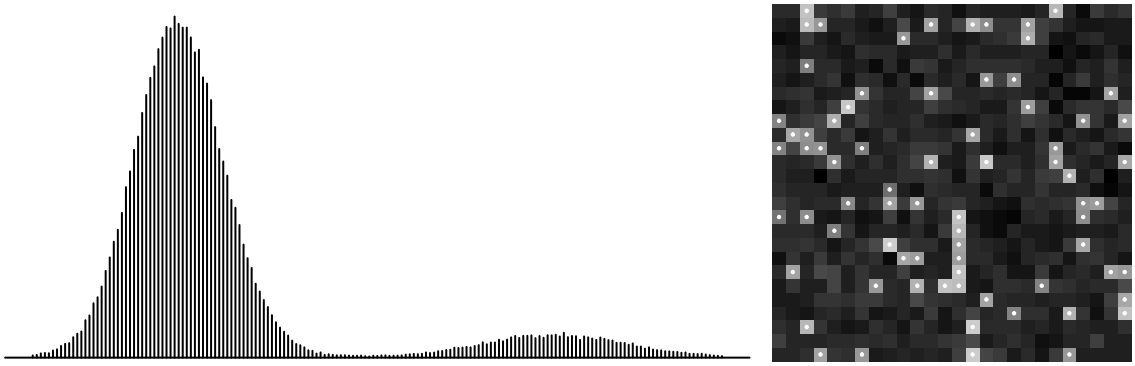


Figure 3. Left, histogram of spatial domain absolute residuals averaged across 100 images containing random payloads at a fixed set of 10% of locations. Right, a 25×25 region of the average absolute residuals (light pixels indicate larger absolute residuals) with the payload-carrying locations marked by a white dot.

(It is strange that the cover residual noise in set B is lower than in set A—and this is confirmed by the data in Tab. 1—yet set B gives so much worse steganalysis performance. Indeed, from Fig. 2 one might expect almost-perfect detection and location of payload, but this does not happen. It is possibly a consequence of heavy tails in the residual noise in set B, so that averaging across many images does not reduce noise as one would expect: more research is needed to investigate the anomalous results from set B.)

To validate the concept, we started with experiments using the most well-behaved cover set A. We took 100 cover images and selected 10% of locations to receive payload, with a different random payload embedded in each image. The spatial domain residual images were computed and their absolute values averaged. A histogram of the resulting values is displayed in Fig. 3: it is visible that there are 10% of locations with higher residuals, and the second part of Fig. 3 confirms that these locations are those where the payload was placed.

Using the mean absolute WAM residuals as a forensic tool is a bit more complicated than for the WS residuals in Ref. 21; in the latter case there was a canonical threshold, above which a pixel can be considered as payload-carrying, and below which as unused. For WAM residuals, the threshold is image-dependent. For the purposes of testing we will assume that the steganalyst already knows the size of the payload p, and therefore estimates that it is carried in the p pixels with the highest mean absolute residual. In that case, the number of false positive and false negative pixel classifications will be equal. We must admit that when the payload size is unknown it must first be estimated by a quantitative steganalyser,[23] and this will cause errors of both types to be more frequent.

An investigation of how the accuracy of payload location depends on the number of stego images and the cover image set (for sets A to D only) is displayed in Tab. 4. 50% of pixels were used for embedding, i.e. a payload size of 60000 bits, and the accuracy with which these locations are identified is displayed in the second column, as the number of stego objects N increases. As expected because of its consistent WAM residuals, cover set A shows the best performance: 90% accuracy of payload location is achieved with fewer than 20 stego images, 99% accuracy within a hundred images, and with a few hundred the detector is near-perfect. On the other extreme, set B gives very poor performance even with thousands of stego images. Sets C and D are intermediate, with a few hundred stego images required for about 99% accuracy.

It is notable that, even with an excess of images, there are still some stubborn errors in the diagnosis. Further examination showed these to be exclusively near the edges of the images, with residuals within 8 pixels of the edge exhibiting more noise than the rest: this should not be surprising, when one considers that pixels near the edge are more difficult to predict than those near the middle because they have fewer near neighbours, and the size of the wavelet filter. We display the same benchmarks, but excluding all pixels within 8 of the image edges, in the third column of Tab. 4; with the exception of the very difficult set B, the error rate drops faster and all the way to zero with enough evidence. However, it may be unsatisfactory to exclude edge areas and we searched for ways to expand the successful detection zone nearer to the edges. After some trial-and-error, we found that most of the accuracy can be maintained by reflecting at least 8 pixels of the image into its borders (the edge row must not be duplicated) and then discarding the extra regions. This makes up for some of the unpredictability

Table 4. Incorrectly classified pixels, using the average absolute spatial domain residual to locate payload, as a function of the number of stego images N.

N	full image	excluding 8-pixel edge regions	with reflected borders excluding 1-pixel edges
Set A	(60000 payload locations)	(54588 payload locations)	(59322 payload locations)
1	23202 (38.67%)	20997 (38.55%)	23064 (38.91%)
10	9656 (16.09%)	8548 (15.69%)	9289 (15.67%)
20	5295 (8.82%)	4531 (8.32%)	5053 (8.52%)
50	1345 (2.24%)	938 (1.72%)	1035 (1.75%)
100	362 (0.60%)	102 (0.19%)	142 (0.24%)
200	146 (0.24%)	3 (0.01%)	6 (0.01%)
500	52 (0.09%)	0 (0.00%)	0 (0.00%)
1000	48 (0.08%)	0 (0.00%)	0 (0.00%)
2000	43 (0.07%)	0 (0.00%)	0 (0.00%)
Set B	(60000 payload locations)	(54500 payload locations)	(59298 payload locations)
1	29795 (49.66%)	27041 (49.57%)	29337 (49.47%)
10	28014 (46.69%)	25407 (46.57%)	27520 (46.40%)
20	26775 (44.63%)	24241 (44.43%)	26408 (44.53%)
50	25243 (42.07%)	22787 (41.77%)	24898 (41.98%)
100	21244 (35.41%)	18947 (34.73%)	20890 (35.22%)
200	18595 (30.99%)	16302 (29.88%)	18170 (30.64%)
500	13701 (22.84%)	11442 (20.97%)	13037 (21.98%)
1000	11235 (18.73%)	9018 (16.53%)	10361 (17.47%)
2000	8628 (14.38%)	6350 (11.64%)	7817 (13.18%)
Set C	(60000 payload locations)	(54491 payload locations)	(59263 payload locations)
1	25686 (42.81%)	23288 (42.68%)	25563 (43.08%)
10	15436 (25.73%)	13897 (25.47%)	15024 (25.32%)
20	10306 (17.18%)	9165 (16.80%)	9919 (16.72%)
50	4835 (8.06%)	4043 (7.41%)	4493 (7.57%)
100	1799 (3.00%)	1253 (2.30%)	1408 (2.37%)
200	637 (1.06%)	207 (0.38%)	238 (0.40%)
500	511 (0.85%)	13 (0.02%)	24 (0.04%)
1000	749 (1.25%)	2 (0.00%)	12 (0.02%)
2000	695 (1.16%)	0 (0.00%)	1 (0.00%)
Set D	(60000 payload locations)	(54559 payload locations)	(59302 payload locations)
1	27802 (46.34%)	25257 (46.35%)	27431 (46.26%)
10	21264 (35.44%)	19301 (35.42%)	20893 (35.23%)
20	16636 (27.73%)	14874 (27.29%)	16093 (27.14%)
50	11432 (19.05%)	9857 (18.09%)	10800 (18.21%)
100	5343 (8.90%)	4095 (7.51%)	4491 (7.57%)
200	2198 (3.66%)	1176 (2.16%)	1308 (2.21%)
500	595 (0.99%)	22 (0.04%)	18 (0.03%)
1000	579 (0.96%)	2 (0.00%)	4 (0.01%)
2000	618 (1.03%)	1 (0.00%)	1 (0.00%)

of the image near the edge, but it still leaves the outermost rows and columns poorly predicted and this single line of pixels must still be excluded from the analysis. (Steganalysis at the very edge of an image appears to be difficult.) These results appear in the final column of Tab. 4.

5. CONCLUSIONS

WAM steganalysis is a powerful detector which can also be adapted to forensic ends. But our investigations have revealed discrepancies in its performance on images from different sources: in some types of image its accuracy is very low. The phenomenon seems to depend heavily on the image processing operations which had been applied to the cover image, but we have not yet identified macroscopic properties which can predict whether WAM is appropriate for a given image. This is an important avenue for future research, since LSB matching is a high-capacity and simple embedding method, and no detectors have yet proven universally reliable. We stress that this variability is not a flaw particular to WAM: other LSB matching detectors (particularly those based on the HCF) have similarly uneven performance. The results in Sect. 2 sound a cautionary note to authors, as it would be possible to misunderstand the performance of a detector if only tested on one set of covers. To have credibility, any further work on detection of LSB matching must test on multiple sets of images.

Because a small number of WAM features contain almost all of the information about the presence of hidden payload, the feature set can be reduced. But, again, the best reduced feature set is heavily cover-dependent. Although we found a set of four features preserving most of the detection power (and which can be computed more quickly than the full set of 27), there is a performance penalty associated with this feature reduction.

We have also demonstrated that, by extending the WAM method to transform residuals back to the spatial domain, it is possible to determine the location of hidden payload when enough stego images have payload embedded in the same pixels: depending on the character of the covers, this might require tens or hundreds of images. This is a valuable forensic tool, the next step towards decoding the hidden message or at least to gaining information about the embedding algorithm. Of course the detector is defeated if different embedding locations are used every time, and the moral is that steganographers, like cryptographers and watermarkers, must not re-use their embedding keys, a conclusion particularly applicable to batch steganography.[24] Some of the results in this forensic work, in particular Fig. 2, suggest that that the WAM filter is not optimal for separating stego noise from cover content, and it might be valuable to revisit this question. This work has also demonstrated that it can be more difficult to steganalyse pixels near the edge of an image, because those pixels are more difficult to predict. Perhaps there is another lesson for steganographers here, and an interesting tradeoff to explore: if the steganalyst knows that payload is more likely to be located in the edges, does this outweigh their difficulty in predicting edge regions?

ACKNOWLEDGMENTS

The first author is a Royal Society University Research Fellow.

REFERENCES

[1] Harmsen, J. and Pearlman, W., "Higher-order statistical steganalysis of palette images," in [*Security and Watermarking of Multimedia Contents V*], *Proc. SPIE* **5020**, 131–142 (2003).

[2] Ker, A., "Steganalysis of LSB matching in grayscale images," *IEEE Signal Processing Letters* **12**(6), 441–444 (2005).

[3] Xuan, G., Shi, Y., Gao, J., Zou, D., Yang, C., Zhang, Z., Chai, P., Chen, C., and Chen, W., "Steganalysis based on multiple features formed by statistical moments of wavelet characteristic functions," in [*Proc. 7th Information Hiding Workshop*], *Springer LNCS* **3727**, 262–277 (2005).

[4] Li, X., Zeng, T., and Yang, B., "A further study on steganalysis of LSB matching by calibration," in [*Proc. IEEE International Conference on Image Processing*], 2072–2075 (2008).

[5] Cancelli, G., Doërr, G., Cox, I., and Barni, M., "Detection of ±1 LSB steganography based on the amplitude of histogram local extrema," in [*Proc. IEEE International Conference on Image Processing*], 1288–1291 (2007).

[6] Ker, A., "A general framework for the structural steganalysis of LSB replacement," in [*Proc. 7th Information Hiding Workshop*], *Springer LNCS* **3727**, 296–311 (2005).

[7] Ker, A., "A fusion of maximum likelihood and structural steganalysis," in [*Proc. 9th Information Hiding Workshop*], *Springer LNCS* **4567**, 204–219 (2007).

[8] Ker, A. and Böhme, R., "Revisiting WS steganalysis," in [*Security, Forensics, Steganography and Watermarking of Multimedia Contents X*], *Proc. SPIE* **6819**, 0401–0417 (2008).

[9] Farid, H. and Lyu, S., "Detecting hidden messages using higher-order statistics and support vector machines," in [*Proc. 5th Information Hiding Workshop*], *Springer LNCS* **2578**, 340–354 (2002).

[10] Lyu, S. and Farid, H., "Steganalysis using color wavelet statistics and one-class support vector machines," in [*Security, Steganography, and Watermarking of Multimedia Contents VI*], *Proc. SPIE* **5306**, 35–45 (2004).

[11] Goljan, M., Fridrich, J., and Holotyak, T., "New blind steganalysis and its implications," in [*Security, Steganography and Watermarking of Multimedia Contents VIII*], *Proc. SPIE* **6072**, 0101–0113 (2006).

[12] Lubenko, I., "Spatial domain steganalysis using statistical learning techniques," (2008). MSc thesis, Oxford University.

[13] Holotyak, T., Fridrich, J., and Voloshynovskiy, S., "Blind statistical steganalysis of additive steganography using wavelet higher order statistics," in [*9th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security*], *Springer LNCS* **3677**, 273–274 (2005).

[14] Cancelli, G., Doërr, G., Cox, I., and Barni, M., "A comparative study of ±1 steganalyzers," in [*Proc. IEEE International Workshop on Multimedia Signal Processing*], 791–796 (2008).

[15] Rojas, R., [*Neural Networks — A Systematic Introduction*], Springer-Verlag (1996).

[16] Witten, I. and Frank, E., [*Data Mining: Practical Machine Learning Tools and Techniques*], Morgan Kaufmann, 2nd ed. (2005).

[17] Aizerman, M., Braverman, E., and Rozonoer, L., "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control* **25**, 821–837 (1964).

[18] Schölkopf, B. and Smola, A., [*Learning with Kernels*], MIT Press (2001).

[19] "NRCS Photo Gallery." `http://photogallery.nrcs.usda.gov/`.

[20] "McGill Calibrated Colour Image Database." `http://tabby.vision.mcgill.ca/`.

[21] Ker, A., "Locating steganographic payload via WS residuals," in [*Proc. 10th ACM Workshop on Multimedia Security*], 27–32 (2008).

[22] Fridrich, J. and Goljan, M., "On estimation of secret message length in LSB steganography in spatial domain," in [*Security, Steganography, and Watermarking of Multimedia Contents VI*], *Proc. SPIE* **5306**, 23–34 (2004).

[23] Soukal, D., Fridrich, J., and Goljan, M., "Maximum likelihood estimation of secret message length embedded using $\pm k$ steganography in spatial domain," in [*Security, Steganography, and Watermarking of Multimedia Contents VII*], **5681**, 595–606 (2005).

[24] Ker, A., "Batch steganography and pooled steganalysis," in [*Proc. 8th Information Hiding Workshop*], *Springer LNCS* **4437**, 265–281 (2006).