Prior Classification of Stego Containers as a New Approach for Enhancing Steganalyzers Accuracy

Viktor Monarev[1,2] and Andrey Pestunov[3](✉)

[1] Novosibirsk State University, Novosibirsk, Russia
viktor.monarev@gmail.com
[2] Institute of Computational Technologies SB RAS, Novosibirsk, Russia
[3] Novosibirsk State University of Economics and Management, Novosibirsk, Russia
pestunov@gmail.com

Abstract. We introduce a novel "prior classification" approach which can be employed in order to enhance the accuracy of stego detectors as well as to estimate it more subtly. The prior classification is intended for selection a subset of a testing set with such a property that a detection error, calculated over this subset, may be substantially lower than that calculated over the whole set. Our experiments demonstrated that it is possible to select about 30 % of the BOSSbase images for which HUGO 0.4 bpp is detected with the error less than 0.003, while the error over the whole set is 0.141. We also demonstrated that it is possible to find about 5 % of the BOSSbase images which provide the detection error for HUGO 0.1 bpp less than 0.05, while the error, calculated over the whole set, is about 0.37 which is not quite a reliable accuracy.

Keywords: Information hiding · Steganalysis · HUGO · Prior classification · Feature-based steganalysis · SRM features · Ensemble classifier

1 Introduction

A commonly used way for measuring the accuracy of binary stego detectors is to calculate a detection error

$$P_E = \frac{1}{2}(P_{FA} + P_{MD}),$$

where P_{FA} is the probability of false alarms, and P_{MD} is the probability of missed detections (see e.g. [4,5,11–14]). The strategic goal of steganalysis is to make this error as low as possible, therefore those detector is better, whose error is lower.

In practice, the statistical model of covers is unknown, that is why the detection error can not be calculated analytically and the solution of this problem lies in calculating an average error over a given set. For this purpose, steganalysts often utilize several "standardized" sets, like BOSSbase [1], BOWS2 [18] or NRCS [20]. However, the accuracy of the detectors may be different when

S. Qing et al. (Eds.): ICICS 2015, LNCS 9543, pp. 445–457, 2016.
DOI: 10.1007/978-3-319-29814-6_38

calculating over different image sets [14] because of various specific properties of the images (like noisiness, compression rate etc.). Moreover, a certain image set can be heterogenous and might be divided into subsets with different properties and different detection error values.

In this paper we introduce a novel approach to steganalysis, which we call a "prior classification". The idea of this approach is to add a stage before final classification in order to choose those images which will be reliably detected. The size of the subset of these images can be considered as an additional characteristic of the detector. The similar situation can be noticed in cryptography, where some attacks are applicable only for a certain subset of weak keys, and the size of this subset (or the ratio of this subset) is considered as an additional characteristic of the attack [2,10].

We introduce three possible methods of how to implement the prior classification approach: the naive splitting, the single classification, and a combination of these two methods. Our experiments demonstrated that it is possible to select about 30 % of the BOSSbase v1.01 images for which HUGO 0.4 bpp is detected with the error less than 0.003, while the error over the whole set is 0.141. We also demonstrated that it is possible to select about 5 % of the BOSSbase images which provide the detection error for HUGO 0.1 bpp less than 0.05, while the error, calculated over the whole set, is about 0.37 which is not quite a reliable accuracy. In our opinion, the prior classification has several potential practical applications which we discuss at the end of the paper.

2 Binary Classification

2.1 The Problem

In this paper we consider the binary classification problem, which is intended for building the detector which will distinguish between the two classes: empty (H_0) and stego (H_1) containers (see e.g. [11]). There are three assumptions behind binary steganalyzers:

1. The steganalyst has a set of covers which have statistical properties, similar to that of used by the steganographer.
2. The steganalyst knows the embedding algorithm and the payload size.
3. The steganalyst knows which object she must examine.

We do not touch quantitative steganalysis [15] when there is no knowledge about the payload.

The contemporary approaches to solving the binary classification problem are essentially based on image features and machine learning tools. There are two high-level components in binary classifiers: a feature extraction method and a classification algorithm. At first, we need to obtain some amount of empty containers as well as containers with a certain payload and extract features from all of them. Then, the classifier is trained in order to be able to distinguish between the empty containers features and features extracted from the stego containers.

The general scheme of the binary classification is as follows.

1. Extract the features from images in the training set which contains empty and stego images.
2. Train the classifier on this set to distinguish between features of empty and stego images.
3. Extract the features from the testing containers and classify them via the trained classifier.

2.2 Ensemble Classifier and Base Learners

The methods, which we introduce in this paper, are based on the idea of applying ensemble classifiers to steganalysis [12]. J. Fridrich et al. call them "a great alternative to support vector machines" because of their good performance and competitive accuracy [6]. The ensemble classifiers has been already applied for breaking HUGO during the public BOSS competition [6] by the winners. The ensemble classifier, as it was introduced in [12], works as follows.

1. Take d features (like SPAM [14], SRM [4], PSRM [9] etc.).
2. Obtain L random subsets of all the features, each of which of $d_{sub} < d$ features.
3. Train L base-leaners on the training set.

Let $N_{votes}(z)$ be the number of the base learners which voted in favor of the fact that z contains information:

$$N_{votes}(z) = \sum_{l=1}^{L} B_l(z).$$

Each base learned works as follows:

$$B_l(z) = \begin{cases} 0 & \text{the base learner } l \text{ votes that } z \text{ is empty;} \\ 1 & \text{the base learner } l \text{ votes that } z \text{ contains information;} \end{cases}$$

The final decision is made by the majority of voters according to the Algorithm 1.

```
ENSEMBLE-RULE($L, N_{votes}$)
    $L$ — the number of the base learners,
    $N_{votes}$ — the number of voices in favor of $H_1$.
    Estimate
        $B = \begin{cases} 0 & \text{if } N_{votes} < L/2; \\ 1 & \text{if } N_{votes} > L/2; \\ random(0,1) & \text{otherwise.} \end{cases}$
    Result: $B$ — the class number.
```

Algorithm 1. Ensemble classifier decision rule

3 Common Background and Designations

3.1 Images

We utilized the well-known standardized image database BOSSbase v1.01 [1,19] which contains 10000 images captured by seven different cameras in RAW format (CR2 or DNG). These images had been converted into 8-bit grayscale format, resized and cropped to the size 512 x 512 pixels.

In order to prepare the training set $\mathcal{X}^p$ and the testing set $\mathcal{Y}^p$, where p identifies the embedding rate in bpp, the whole BOSSbase set was divided into two subsets $\mathcal{X}_0$ and $\mathcal{Y}_0$, where $|\mathcal{X}_0| = 8000$ and $|\mathcal{Y}_0| = 2000$. Then by random embedding p bpp into all the images from $\mathcal{X}_0$ and $\mathcal{Y}_0$ we obtained $\mathcal{X}_1^p$ and $\mathcal{Y}_1^p$ correspondingly. The training set was $\mathcal{X}^p = \mathcal{X}_0 \cup \mathcal{X}_1^p$ and the testing set $\mathcal{Y}^p = \mathcal{Y}_0 \cup \mathcal{Y}_1^p$. Thus, $|\mathcal{X}^p| = 16000$ and $|\mathcal{Y}^p| = 4000$. Both sets contain a half of empty images and a half of stego images.

Further in the paper we omit the payload index p (it will not confuse the reader) and designate the training set as $\mathcal{X}$ and the testing set as $\mathcal{Y}$.

3.2 Features

We utilize Spatial Rich Model (SRM) features [4] as one of the state-of-the-art instruments for steganalysis. The newer Projection Spatial Rich Model features (PSRM) [9] provide only slight improvement but require substantially greater complexity. SRM features have a total dimension of 34,671 and we took the extractor provided by [17].

3.3 Base Learners

There are several variants for choosing the base learners, but in our experiments we follow the recommendations of Kodovsky et al. [12] and exploit the Fisher Linear Discriminant [3] due to its low training complexity. There will be two types of the base learners in our paper, which we designate as B_l, $l = 1, \ldots, L$ and B'_m, $m = 1, \ldots, M$ correspondingly. Thus, the number of base learners is L or M. Each base learner is always assigned with 800 randomly chosen SRM features.

3.4 Embedding Algorithm

We used Highly Undetectable Steganography (HUGO) as an embedding algorithm [16]. This method is one of the hardest steganography to detect (see e.g. results in [9] where HUGO is compared to WOW [7] and UNIWARD [8] — the other content adaptive embedding algorithms). HUGO is based on the LSB matching but chooses the places for embedding probabilistically according to the SPAM-features [16] rather than randomly as LSB matching. This modification allows to lengthen the hidden message by 7 times comparing to LSB matching preserving the security level (the error).

3.5 State-of-the-Art

In our experiments we needed our own implementation of the ensemble classifier. There are many possible parameters of this classifier that is why we implemented it ourselves with some certain parameters and in the Table 1 we compare the original implementations of state-of-the-art classifiers from [9] with our implementation. We see that the detection errors of our implementation is rather close to original ones. These errors values will be used for comparison with our prior classification results.

Let $\mathcal{Y}^{good}$ be the set of "good" images, which were selected after the prior classification stage, and $P_E(\mathcal{Y}^{good})$ be the detection error calculated over this set. In our experiments we compare $P_E(\mathcal{Y}^{good})$ and $P_E(\mathcal{Y})$ and try to make $|\mathcal{Y}^{good}|/|\mathcal{Y}|$ as large as possible.

Table 1. State-of-the-art results on HUGO detection (ensemble classifier).

	Detection error $P_E(\mathcal{Y})$ over the whole BOSSbase 1.01	
Payload	Results from [9], various features and parameters	Our implementation, SRM features, $L = 500$
0.05 bpp	-	0.44
0.10 bpp	0.3564–0.3757	0.37
0.20 bpp	0.2397–0.2701	-
0.40 bpp	0.1172–0.1383	0.141

4 Prior Classification

4.1 Basic Idea

The basic idea behind all our methods consists in quite a natural assumption that if for an image z we have $N_{votes}(z)$ rather close to 0 or to L than we can be more sure in the decision. This idea is directly implemented in our first naive splitting method, which selects "good" images as images, for which

$$N_{votes}(z) \leq T^{left} \text{ or } N_{votes}(z) \geq T^{right}$$

for some fixed thresholds T^{left} and T^{right}.

The next, single classification method, consists in training an additional classifier to recognize the "good" images and use this classifier as the prior classification stage. At last, we give an algorithm of how to combine these two methods.

4.2 Method 1: Naive Splitting

Our first idea was to define some thresholds T^{left} and T^{right}, such that T^{left} is close to 0 and T_{right} is close to L (the number of the base learners), and split the testing set $\mathcal{Y}$ into "good" and "bad" subsets, according to the thresholds as follows

$$\mathcal{Y} = \mathcal{Y}^{good} \cup \mathcal{Y}^{bad}, \text{where}$$
$$\mathcal{Y}^{good} = \{y \in \mathcal{Y} \,|\, N_{votes}(y) \leq T^{left} \text{ or } N_{votes}(y) \geq T^{right}\},$$
$$\mathcal{Y}^{bad} = \mathcal{Y} \setminus \mathcal{Y}^{good}.$$

This idea is implemented in the Algorithm 2.

```
NAIVE-SPLITTING($\mathcal{Z}, t^{left}, t^{right}$)
    $\mathcal{Z}$ — the set to be splitted,
    $t^{left}$ — the left threshold,
    $t^{right}$ — the right threshold.

  1. Train the base learners $B_l, \ldots, B_L$ on the sets $\mathcal{X}_0$ and $\mathcal{X}_1$ to distinguish
     between the empty/stego containers.
  2. For each $z \in \mathcal{Z}$ calculate the number of base learners votes
     $N_{votes}(z) = \sum_{l=1}^{L} B_l(z)$.
  3. Obtain the set $\mathcal{Z}^{good} = \{z \in \mathcal{Z} \,|\, N_{votes}(z) \leq t^{left} \text{ or } N_{votes}(z) \geq t^{right}\}$.

Result: $\mathcal{Z}^{good} \subseteq \mathcal{Z}$ — the subset of the "good" containers.
```

Algorithm 2. Naive Splitting

Our hypothesis was that the detection error, calculated over $\mathcal{Y}^{good}$, would be smaller than that calculated over the whole $\mathcal{Y}$. The results were as we expected, but for 0.40 bpp they were really impressing. Prior classification allowed to select about one third of images with very low error — less than 0.003 (see Table 2). This error is approximately by 50 times lower than over the whole set (see Table 1). For the other two payloads the error lowered, but not so dramatically. Moreover, the sizes of the filtered subsets were rather small.

4.3 Method 2: Single Classification

In order to make $\mathcal{Y}^{good}$ bigger we built one more classifier (this method we call the simple classification) for distinguishing between the "good" and the "bad" images (see Algorithm 3). This classification gave a drastic increase in the size of $\mathcal{Y}^{good}$, but the detection error $P_E(\mathcal{Y}^{good})$ was rather high comparing to the simple splitting (see Table 3). However, it was lower than that calculated over the whole set (see Table 1).

Table 2. Naive splitting ($\% = 100 \cdot |\mathcal{Y}^{good}|/|\mathcal{Y}|$ — the percent of the "good" images, $T^{left} = 1$, $T^{right} = L - 1$).

	HUGO 0.05 bpp			HUGO 0.10 bpp			HUGO 0.40 bpp		
L	$\|\mathcal{Y}^{good}\|$	%	$P_E(\mathcal{Y}^{good})$	$\|\mathcal{Y}^{good}\|$	%	$P_E(\mathcal{Y}^{good})$	$\|\mathcal{Y}^{good}\|$	%	$P_E(\mathcal{Y}^{good})$
100	50	1.25	0.260	271	6.76	0.140	1651	41.28	0.0042
200	31	0.76	0.258	176	4.40	0.125	1462	36.55	0.0041
300	23	0.56	0.304	137	3.43	0.124	1384	34.60	0.0022
400	16	0.40	0.313	117	2.93	0.120	1323	33.08	0.0023
500	14	0.35	0.286	106	2.65	0.123	1285	32.13	0.0016

Table 3. Single classification ($\% = 100 \cdot |\mathcal{Y}^{good}|/|\mathcal{Y}|$ — the percent of the "good" images).

		HUGO 0.05 bpp			HUGO 0.1 bpp			HUGO 0.4 bpp		
T^{left}	T^{right}	$\|\mathcal{Y}^{good}\|$	%	$P_E(\mathcal{Y}^{good})$	$\|\mathcal{Y}^{good}\|$	%	$P_E(\mathcal{Y}^{good})$	$\|\mathcal{Y}^{good}\|$	%	$P_E(\mathcal{Y}^{good})$
1	499	292	7	0.353	1198	30	0.244	1903	48	0.0189
2	499	280	7	0.346	1139	28	0.227	2022	51	0.0218
3	499	455	11	0.365	1158	29	0.225	2061	52	0.0213
1	498	232	6	0.332	1510	38	0.273	1911	48	0.0167
2	498	337	8	0.359	1189	30	0.230	2132	53	0.0225
3	498	528	13	0.371	1302	33	0.247	2009	50	0.0184
1	497	284	7	0.357	1171	29	0.243	2045	51	0.0220
2	497	340	9	0.362	1204	30	0.236	2091	52	0.0210
3	497	347	9	0.378	1182	30	0.228	2048	51	0.0215

```
SINGLE-CLASSIFICATION($\mathcal{Z}, t^{left}, t^{right}$)

  1. Obtain the set of "good" containers
     $\mathcal{X}^{good} :=$ NAIVE-SPLITTING($\mathcal{X}, t^{left}, t^{right}$).
  2. Obtain the set of "bad" containers
     $\mathcal{X}^{bad} := \mathcal{X} \backslash \mathcal{X}^{good}$.
  3. Train the base learners $B'_1, \ldots, B'_M$ on the sets $\mathcal{X}^{good}$ and $\mathcal{X}^{bad}$
     to distinguish between the "good"/"bad" containers.
  4. Apply the ensemble classifier for each $z \in \mathcal{Z}$ in order to classify it
     as a "good" or a "bad" container.
     (a) Calculate the number of voices $N'_{votes}(z) = \sum_{m=1}^{M} B'_k(m)$.
     (b) Obtain the set
         $\mathcal{Z}^{good} = \{z \in \mathcal{Z} \mid$ ENSEMBLE-RULE($M, N'_{votes}(z)$) $= 1\}$.

Result: $\mathcal{Z}^{good} \subseteq \mathcal{Z}$ — the subset of the "good" containers.
```

Algorithm 3. Single classification

```
COMBINED-CLASSIFICATION($\mathcal{Y}, t_1^{left}, t_1^{right}, t_2^{left}, t_2^{right}$)
    $\mathcal{Z}_1^{good}$ = NAIVE-SPLITTING($\mathcal{Y}, t_1^{left}, t_1^{right}$);
    $\mathcal{Z}_2^{good}$ = SINGLE-CLASSIFICATION($\mathcal{Y}, t_2^{left}, t_2^{right}$);
    $\mathcal{Y}^{good} = Z_1^{good} \cap Z_2^{good}$;
Result: $\mathcal{Y}^{good} \subseteq \mathcal{Y}$ — the subset of the “good” containers.
```

Algorithm 4. Combined classification

4.4 Method 3: Combined Classification

In our aspiration to build a better prior classifier, we decided to combine the two introduced methods: the naive splitting and the single classification. According to our hypothesis, using appropriate thresholds T_1^{left} and T_1^{right} — for the naive splitting, and T_2^{left} and T_2^{right} — for the single classification, would allow to distinguish a reasonably large set $\mathcal{Y}^{good}$ such that the detection error $P_E(\mathcal{Y}^{good})$ would be noticeable smaller. The formal description of the combined classification is shown in the Algorithm 4.

Due to the big amount of parameters, there are several possible schemes can be used for testing this method. We tried the following. The goal was to search for the combination of the thresholds for getting the largest subset with a fixed detection error P_E^*.

In our experiments, we fixed the detection error P_E^* and searched for those thresholds $(T_2^{left}, T_2^{right})$ which provide the largest set

$$\mathcal{Y}^{good}(T_1^{left}, T_1^{right}, T_2^{left}, T_2^{right})$$

such that the detection error calculated over this set does not exceed P_E^*. More formally,

$$(T_2^{left}(P_E^*), T_2^{right}(P_E^*)) = \underset{t^{left}, t^{right}}{argmax}\, |\mathcal{Y}^{good}(T_l^{left}, T_l^{right}, t^{left}, t^{right})|,$$

under the limitation that

$$P_E(\mathcal{Y}^{good}(T_l^{left}, T_l^{right}, t^{left}, t^{right})) \le P_E^*.$$

Table 4. Combined prior classification (HUGO 0.1 bpp, L=500, M=1).

	$T_1^{left} = 1,\ T_1^{right} = 499$				$T_1^{left} = 2,\ T_1^{right} = 498$			
P_E^*	$\lvert\mathcal{Y}^{good}\rvert$	%	T_2^{left}	T_2^{right}	$\lvert\mathcal{Y}^{good}\rvert$	%	T_2^{left}	T_2^{right}
0.04	187	5	1	489	202	12	2	490
0.05	230	6	2	485	251	12	2	481
0.06	252	6	4	485	303	12	17	490
0.07	346	8	19	483	391	12	27	481
0.08	401	10	33	489	463	12	30	464

Table 5. Combined prior classification (HUGO 0.05 bpp, L=500, M=11, $T_1^{left} = 10$, $T_1^{right} = 490$).

P_E^*	$\|\mathcal{Y}^{good}\|$	%	T_2^{left}	T_2^{right}
0.15	21	0.5	0	487
0.18	28	0.7	3	487
0.21	58	1.5	20	486
0.24	92	2.3	42	470

Table 6. Combined prior classification (HUGO 0.4 bpp, L=500, M=11, $T_1^{left} = 20$, $T_1^{right} = 480$).

P_E^*	$\|\mathcal{Y}^{good}\|$	%	T_2^{left}	T_2^{right}
0.00000	1481	37	1	492
0.00125	1655	41	9	492
0.00225	1680	42	9	490
0.00325	1853	46	99	492

In Tables 4, 5, 6 and 7 there are the results. In the Table 7 it is demonstrated that it is possible to find about 5 % of the BOSSbase images which provide the detection error for HUGO 0.1 bpp less than 0.05, while the error, calculated over the whole set, is about 0.37 (see Table 1) which is not quite a reliable accuracy. Thus, here we see that not a very reliable detector turns into a more reliable one.

Table 7. Combined prior classification (HUGO 0.1 bpp, L=500, M=11)

	$T_1^{left} = 1$, $T_1^{right} = 499$				$T_1^{left} = 2$, $T_1^{right} = 498$				$T_1^{left} = 10$, $T_1^{right} = 490$			
P_E^*	$\|\mathcal{Y}^{good}\|$	%	T_2^{left}	T_2^{right}	$\|\mathcal{Y}^{good}\|$	%	T_2^{left}	T_2^{right}	$\|\mathcal{Y}^{good}\|$	%	T_2^{left}	T_2^{right}
0.01	0	0	-	-	0	0	-	-	0	0	-	-
0.02	157	3.9	1	471	0	0	-	-	0	0	-	-
0.03	181	4.5	2	464	175	4.4	1	485	0	0	-	-
0.04	191	4.8	2	455	227	5.7	1	464	203	5.1	2	490
0.05	209	5.2	3	438	255	6.4	2	455	284	7.1	2	470
0.06	267	6.7	27	455	336	8.4	28	464	334	8.4	27	490
0.07	293	7.3	45	464	388	9.7	45	455	460	11.5	27	455
0.08	354	8.9	90	438	456	11.4	93	442	518	13.0	33	434
0.09	378	9.5	90	403	482	12.1	96	418	567	14.2	33	403
0.10	386	9.7	98	403	499	12.5	98	401	626	15.7	93	453
0.11	388	9.7	98	401	499	12.5	98	401	710	17.6	95	403

4.5 Prior Classification On-The-Fly

Above, the prior classification algorithms were described in the terms of the sets, nevertheless, all of them can be easily applied to single images. Instead of creating the set $\mathcal{Z}^{good}$ we can test the next image via the classifiers, and, in such a way, the prior classifiers will work on-the-fly.

5 Possible Applications and Future Work

5.1 Increasing the Practical Significance of Weak Detectors

The prior classification can be used for making weak detectors more practically significant. If some detector is not reliable, i.e. when its detection error over the whole set is close to 0.5, maybe the prior classification will select some set of images such that the error over this subset will be lower. For instance, in our results with HUGO 0.05 bpp, the error over the whole BOSSbase was 0.37 (see Table 1) while the prior classification turned it into a lower one albeit over a small subset (see Table 5).

5.2 Spreading Images Between Different Detectors

The prior classification might be used in order to select the most accurate detector for a given image or an image subset. For instance, if there are several different detectors available to the steganographer, she can use the prior classification for the given image, then select those detectors which will classify this image as "good", and test the image only via them. The accuracy of such a testing scheme might be higher comparing to single detectors.

5.3 Splitting Image Sets into Subsets with Different Properties

There are at least two factors which may impact the detector's accuracy: image properties and the detector itself. Therefore, if one detector has a lower detection error than another on a certain set of images, it is not necessary that it will be always the case. There are no guarantee, that the second detector will not be more accurate on some other image set with specific properties. Thus, when comparing several detectors, it might be reasonable to estimate the detection error on such sets which are obtained by collecting images with common (in some sense) properties.

The introduced method for splitting an image set into subsets of "good" and "bad" images is suitable for obtaining such sets with common properties, and, moreover, it can be used for splitting the set into more that two parts, thereby creating layers of, for example, "very good", "good", "bad" and "very bad" images. Moreover, the prior classification can be used in order to provide a ceratin size of the "good" subset via adjusting detector's parameters. It will allow to pick out a certain (prescribed) percent of images where it will be possible to detect (the absence) of steganography reliably.

5.4 Potential Enhancement of Steganalytic Detectors

If portion of "good" images is significant (certainly, it depends on the image set), then adding the prior classification phase may be used for increasing the effectiveness (accuracy or throughput) of traditional classifiers. For instance, if there is some slow but highly accurate detector (like SVM), then it can be preceded by a quicker classifier (like ensemble) which will filter out "bad" images leaving only "good" ones for the slow detector.

5.5 Extended Definition of the Accuracy

Estimating the detectors accuracy over only a certain subset of the testing set is similar to that is used in cryptography when cryptanalysts develop attacks under the assumption of the weak keys [2,10]. Such attacks are characterized not only by their complexity or success probability, but by the cardinality of the weak keys class. Similarly, stego detectors can be characterized by the detection error and the size of the subset over which this error is estimated.

6 Conclusion

In this paper we introduced a new approach to steganalysis which we call the "prior classification". This approach assumes that there is an additional prior classification stage before the final classification, which allows to select those images which would be detected with the smaller detection error comparing to the error calculated over the whole containers set. There can be various ways of how to implement the prior classification in a given particular case. In such a way we presented the three possible methods of the prior classification: the naive splitting, the simple classification, and the combination of these two methods which we call the combined classification.

According to our experiments, the prior classifiers are sensitive to the choice of the parameters (the thresholds), therefore we presented our results via the tables, where the parameters varied. But, in our opinion, the most impressing results are as follows. We demonstrated that it is possible to select about 30 % of the BOSSbase images for which HUGO 0.4 bpp steganography is detected with the error less than 0.003, while the error over the whole set is 0.141. So the error decreased by almost 50 times for the rather large subset.

We also demonstrated that it is possible to select about 5 % of the BOSSbase images which provide the detection error for HUGO 0.1 bpp less than 0.05, while the error, calculated over the whole set, is about 0.37 which is not quite a reliable accuracy. Here we see an other application of the prior classification — it allowed to turn the unreliable detector into the reliable one, albeit for the small subset.

At the end of the paper several additional potential applications of the prior classification has been discussed, among them, the extended definition of the accuracy, ability to choose the size of the "good" subset by adjusting the parameters, splitting containers sets into several subsets with different properties.

Acknowledgment. This research has been supported by the Russian Foundation of Basic Research, grant no. 14-01-31484.

References

1. Bas, P., Filler, T., Pevný, T.: Break our steganographic system: the ins and outs of organizing BOSS. In: Filler, T., Pevný, T., Craver, S., Ker, A. (eds.) IH 2011. LNCS, vol. 6958, pp. 59–70. Springer, Heidelberg (2011)
2. Biryukov, A., Nakahara Jr., J., Preneel, B., Vandewalle, J.: New weak-key classes of IDEA. In: Deng, R.H., Qing, S., Bao, F., Zhou, J. (eds.) ICICS 2002. LNCS, vol. 2513, pp. 315–326. Springer, Heidelberg (2002)
3. Duda, R., Hart, P., Stork, D.: Pattern classification, 2nd edn. Wiley, New York (2001)
4. Fridrich, J.: Rich models for steganalysis of dugital images. IEEE Trans. Inf. Forensics Secur. **7**(3), 868–882 (2012)
5. Fridrich, J., Kodovský, J., Holub, V., Goljan, M.: Steganalysis of content-adaptive steganography in spatial domain. In: Filler, T., Pevný, T., Craver, S., Ker, A. (eds.) IH 2011. LNCS, vol. 6958, pp. 102–117. Springer, Heidelberg (2011)
6. Fridrich, J., Kodovský, J., Holub, V., Goljan, M.: Steganalysis of content-adaptive steganography in spatial domain. In: Filler, T., Pevný, T., Craver, S., Ker, A. (eds.) IH 2011. LNCS, vol. 6958, pp. 102–117. Springer, Heidelberg (2011)
7. Holub, V., Fridrich, J.: Designing steganographic distortion using directional filters. In: Proceedings 4th IEEE International Workshop on Information Forensics and Security, pp. 234–239. IEEE (2012)
8. Holub, V., Fridrich, J.: Digital image steganography using universal distortion. In: Proceedings 1th ACM Workshop Information Hiding and Multimedia Security, pp. 59–68 (2013)
9. Holub, V., Fridrich, J.: Random projections of residuals for digital image steganalysis. IEEE Trans. Inf. Forensics Secur. **8**(12), 1996–2006 (2013)
10. Kara, O., Manap, C.: A new class of weak keys for blowfish. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 167–180. Springer, Heidelberg (2007)
11. Ker, A., et al.: Moving steganography and steganalysis from the laboratory into the real world. In: Proceedings 1st ACM Workshop on Information Hiding and Multimedia Security, pp. 45–58. ACM (2013)
12. Kodovsky, J., Fridrich, J., Holub, V.: Ensemble classifiers for steganalysis of digital media. IEEE Trans. Inf. Forensics Secur. **7**(2), 434–444 (2011)
13. Monarev, V., Pestunov, A.: A known-key scenario for steganalysis and a highly accurate detector within it. In: Proceedings IEEE 10th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 175–178. IEEE (2014)
14. Pevny, T., Bas, P., Fridrich, J.: Steganalysis by subtractive pixel adjacency matrix. IEEE Trans. Inf. Forensics Secur. **5**(2), 215–224 (2010)
15. Pevny, T.: Detecting messages of unknown length. In: Proceedings 8th Media Watermarking, Security and Forensics, pp. 1–12 (2011)
16. Pevný, T., Filler, T., Bas, P.: Using high-dimensional image models to perform highly undetectable steganography. In: Böhme, R., Fong, P.W.L., Safavi-Naini, R. (eds.) IH 2010. LNCS, vol. 6387, pp. 161–177. Springer, Heidelberg (2010)
17. Feature Extractors for Steganalysis. http://dde.binghamton.edu/download/feature_extractors/

18. Break our watermarking system 2nd ed. http://bows2.ec-lille.fr/
19. Break our steganographic system. http://www.agents.cz/boss/
20. NRCS photo gallery. http://photogallery.nrcs.usda.gov/