UDC 004.056; 004.415.24

Comparison of Efficiency of Statistical Models Used for Formation of Feature Vectors by JPEG Images Steganalysis

Nataliya Koshkina [1]

[1] *V.M. Glushkov Institute of Cybernetics of National Academy of Sciences of Ukraine*

Abstract

In order to build effective analytical systems for digital covers steganalysis in the given practical conditions, it is necessary to analyze and evaluate the quality of existing methods and components. Thus, it is necessary to compare the baseline characteristics of the available candidates in order to select the optimal components of the system. However, the usage of results from open scientific publications may lead to incorrect comparison due to differences in the conditions of numerical experiments. This study is based on the principle of checking the performance of statistical models for feature vectors formation under the same conditions. The case of JPEG images steganalysis with the usage of machine learning techniques is considered. The performance and detection accuracy of statistical models such as CHEN, CC-CHEN, LIU, CC-PEV, CC-C300, GFR, and DCTR in case of message hiding in the frequency domain of digital images are analyzed. The results of the study are numerical estimates of the performance and accuracy of SVM classification in binary and multilevel steganalysis modes.

Keywords: information security, steganalysis, passive counteraction, training-and-classification methods, comparative analysis, models of feature vectors, SVM.

Introduction

The main condition that must be met by steganographic systems (stegosystems) designed for the hidden data transmission is the impossibility of detecting the fact of operation of the system by third parties. A digital presentation of information and a variety of processing methods play an important role in the modern world. Therefore, the steganography in digital media obtained many new possibilities and applications. Modern steganographic systems offer different ways for organizing of secret (hidden) data communication through public (open) channels do not attract the attention of third-party observers. But this form of communication can be used to carry out unlawful acts and threaten the information security of the state and private organizations. Therefore, along with the evolution of steganography the development of the opposite direction, namely steganalysis, is relevant and important.

Recently, steganalysis (stego-analytical) methods with training and classification have received considerable attention. It is explained by their universality and easy enhancement by applying the latest advances in machine learning theory. Such stego-analytical methods assign to each cover file (container) a feature vector, which elements are sensitive to any alteration of cover and, at the same time, do not depend strongly on the cover's contents. The extracted feature vectors are inputted to the classifier (stego detector) for revealing of stego files. The classifier is built through supervised learning on the cover (blank) and stego (filled) files. Alternatively, the stego detector can be tuned to recognize cover files and marking all "anomalies" as stego files – covers that contain embedded stego data. Thus, the two key components of training and classification methods are the feature vector model and the classification method. The wide range of different models of feature vectors and classifiers suitable for solving of steganalysis tasks are proposed in open scientific publications. Some of the existing solutions are described, for instance, in [1].

In practice, the main criteria for selecting a steganalysis method are its accuracy and speed. However, comparison of methods proposed in open publications is difficult because of differences in settings of numerical experiments. The performance evaluations are influenced not only by the mentioned options, but also by the properties of test sets of cover files, internal parameters of a classifier, parameters of the embedding methods that are attacked by experiments, and so on. Therefore, studies that compare the effectiveness of modern steganalysis methods in the same conditions are relevant, and their results can be used for the practical organization of systems to detect steganographic hiding within digital media. These studies can also serve as a basis for further improvement of steganalysis methods and systems.

Thus, **the purpose of the work** is to perform a comparative analysis of the performance of different statistical models of feature vectors. The results of the analysis can be used for choosing the optimal model for a given classifier and certain practical conditions. By selecting a classic Support Vector Machine (SVM) and fixing the experimental conditions, we estimate the accuracy and speed of steganalysis attacks based on various existing feature vector models for one of the common container types.

Research methodology

The comparative analysis of the performance of different models of feature vectors in the same conditions of steganalysis will be performed according to the following steps:

1) Determination of the set of cover files.
2) Choosing steganographic methods or transformation to be counteracted. Creation of stego files according to chosen methods.
3) Selection of feature vector formation models that can be applied to the covers defined in the first step. Creating feature vectors for empty and filled covers. Estimation of cover processing time and its comparison for different models.
4) Choosing a classifier and defining its parameters. Determination of the training and test subsets of cover files.
5) Classifier tuning. Assessing learning speed and comparing it to different models.
6) Classification of samples from the test subset. Assess the classification accuracy and compare it for different models.
7) Design of a multi-class stego-analytical system. Evaluation of the accuracy of a multiclass classification and its comparison for different models.

Definition of the set of test cover files

One of the most common types of cover files to be used for steganographic hiding is JPEG files. These files have a certain amount of redundancy in digital representation that allows them to be used for steganographic related tasks. JPEG images are widely available on social networks, data exchange services and other online resources. The description of the JPEG format itself can be found, for example, in [2]. Typically, the images of relatively small sizes, usual multiples of 8, are used in steganalysis researches. This is primarily due to the need to process a large enough and statistically significant amount of cover images, which will take a long time that increases with the size of the images. Detection accuracy depends not as much as on image sizes but on the percentage of coefficients modified by message embedding. In addition, for graphic covers that containing millions of pixels, steganalysis methods can be applied to separate blocks of the image instead of the whole image. The result is obtained by the fusing of stegdetectors responses for individual blocks. Such an approach allows improving detection accuracy for stego images, which are initially classified as covers (false rejection).

So, the set of 1330 colour JPEG images with size 512 $\times$ 384 and 384 $\times$ 512 pixels, compressed to 75% quality, were selected as the original test set. Image file sizes range from 8 to 82 kbytes. Typical images from the test set are shown in Fig. 1.

Selection of steganographic programs or transformation to be counteracted. Creation of stego files

Three freely available steganographic programs for data hiding into JPEG domain were selected for experiments: Jsteg, Jphide, and Steganos Privacy Suite 2012 (Crypt & Hide module).

The Jsteg application hides information by converting unpacked image files to JPEG format. The embedding method consists in replacement of the least significant bits (LSB) of the discrete cosine transform (DCT) quantized coefficients of the cover image, with the exception of zero and one's values, by bits of the message. If the length of embedding message is more than the number of cover DCT coefficients, only a portion equal to the number of coefficients (with a residual loss) is hidden.

Unlike Jsteg, Jphide application implements a key-based stegosystem and also provides message encryption with the Blowfish algorithm. Key element values depend on the user's password and the length of the embedding message. A fixed lookup table is used for division of applicable DCT coefficients of a cover image into classes. These classes are used for determining the order in which DCT coefficients will be changed. The first class in the lookup table consists of DC coefficients (zero frequency), followed by the classes of AC coefficients (all other frequencies) whose absolute values are greater than the corresponding table values. The LSB-based embedding in the current class continues until the entire message is embedded. In addition, Jphide application can modify first as well as second significant bits of the cover image DCT coefficients. If the message is too large for the selected cover, the program issues an error and does not create a stego image.

Unlike the previous ones, Steganos Privacy Suite 2012 is capable of handling different types of cover images. Message hiding is provided by the Crypt & Hide module of the program. The module implements AES-256 encryption method and embedding of processed messages into images (*.bmp, *.jpg formats) or audio files (*.wav format). For JPEG-compressed cover images, a consistent LSB-embedding into quantized DCT coefficients is implemented with the exception of the DC coefficients for each sub-blocks of the cover image. This step is aimed at increasing the visual invisibility of the steganographic intervention. If the message is too large for this container, the program does not create a container.

Thus, three sets of JPEG-compressed stego images were created from the initial test set. Each created set consists of stego images with 1 kB of hidden information (random text). To automate the process of creating a large number of stego images, shell scripts and the Sikuli visual environment were used. As a result, Jsteg app created 1,330 stego images, Jphide - 1119, and Steganos Privacy Suite 2012 - 1263.

It is clear that if a Jsteg app creates stego image for a given empty container and Jphide as well as Steganos Privacy Suite do not create corresponding stego images, then this Jsteg-based stego image obtains close to 100% payload. Consequently, the set of Jsteg-based stego

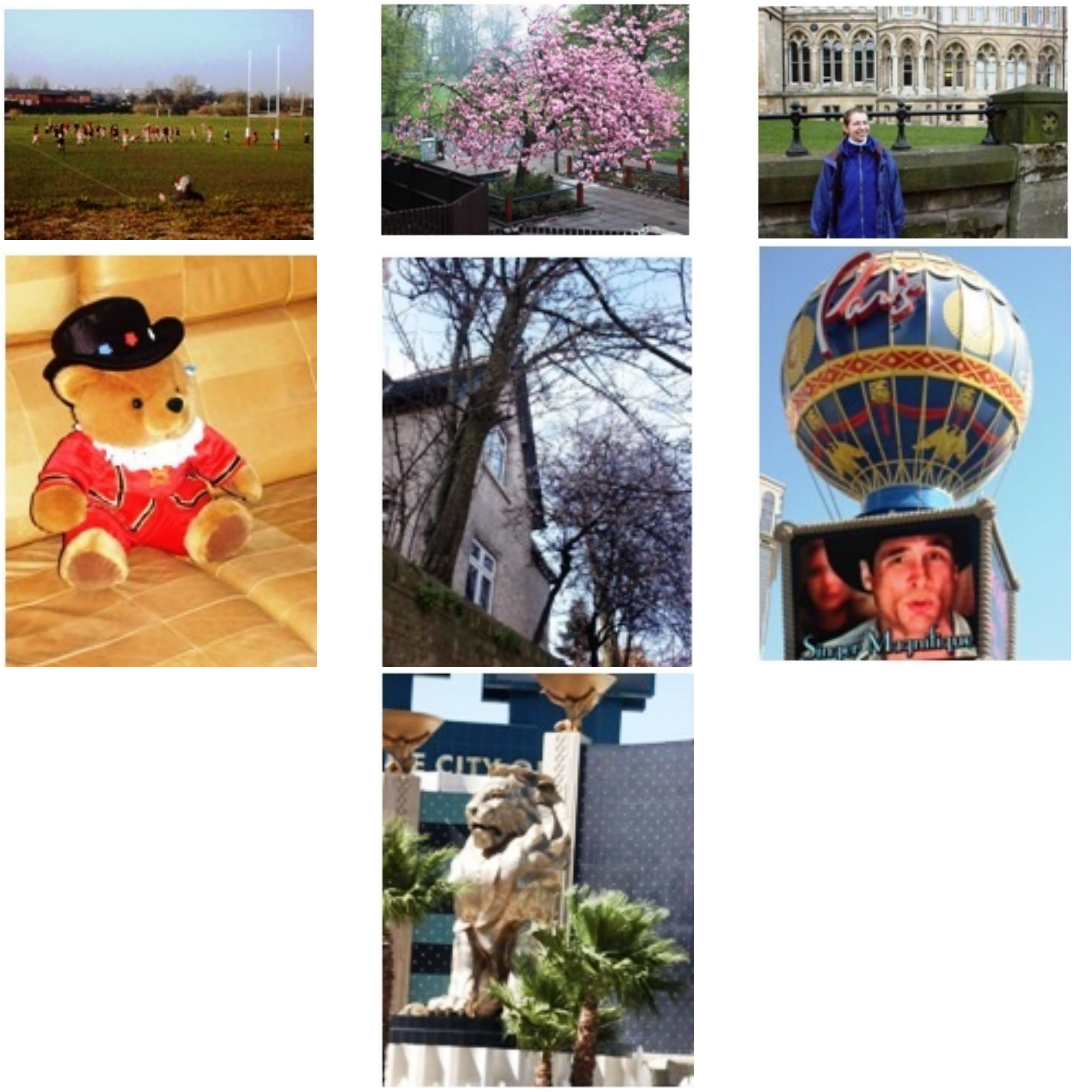

Fig. 1. Typical test images

images contains a higher percentage of high-filled covers than the other two, and because of this, the accuracy of steganalysis for this set would be higher under the same another conditions. Therefore, for further experiments, only 1114 images were left in each of the available sets - exactly how many stego images were created by all three programs from inputted cover images.

The choice of models for the formation of feature vectors. Creating feature vectors. Evaluation of their creation duration and comparison for different models

The following statistical models for the formation of feature vectors for JPEG cover images were selected for the study:

1) CHEN – is a Markov process-based model that uses residuals matrices and corresponding to them transition probability matrices proposed in [3]. The violations of DCT coefficients statistics for fixing intra-block and inter-block DC coefficients are used for the detection of hidden messages.
2) CC-CHEN – is the modification of CHEN model, enhanced by the Cartesian calibration, proposed in [4]. By Cartesian calibration, the feature vector contains the elements calculated for the initial image and for its calibrated version. Calibration, in this case, involves JPEG-decompression of the analyzed image, cropping by 4 pixels in horizontal and vertical directions from the beginning and subsequent compression with the same quality factor.
3) LIU – is a model proposed in [5] and based on the fact that the steganographic hiding changes the total density of adjacent elements. Like the previous ones, the model analyzes the intra-block and inter-block statistics of the quantized DCT coefficients values. Since the total density varies for different images, calibration is applied to reflect its change caused by the message hiding. The average values of the density matrices of the original and calibrated images are estimated and the differential characteristics between the obtained statistics are calculated.
4) CC-PEV - the PEV model presented in [6] and is enhanced by the Cartesian calibration proposed in

[4]. The feature vector in the PEV model includes a number of statistics for the frequency domain, including the values of the global histogram of the distribution of DCT coefficients, dual histograms, local histograms of the distribution of the first five AC coefficients, the averaged variation, the elements of the block, the elements of the matrix of the common coefficients of the coefficient. In addition, the vector is supplemented by averaged values of the intra-block transition probability matrices calculated by the CHEN model algorithm.

5) CC-C300 - the model is based on the usage of multidimensional vectors to capture the most dependencies between DCT coefficients of analyzed images. The model is proposed in [7]. The elements of the feature vectors represent the values of the matrices of the joint appearance of DCT coefficients pairs. In order to make the construction of individual matrices more systematic, all possible pairs of DCT coefficients are sorted by importance and the matrices are calculated in order from the most important to the least important pairs (the mutual information is used as a measure of importance). This model combines the 300 most important co-occurrence matrices.
6) GFR – is a model of feature vectors constructed as histograms of quantized residuals obtained using two-dimensional Gabor Filters. Gabor 2D filters describe the texture features of images on different scales and orientations. The model is presented in [8]. Unlike the previous ones, it is being built in a spatial area of the cover image.
7) DCTR – is a phase-based model of Discrete Cosine Transform Residual vectors. It was proposed in [9] and is built in the spatial domain as the GFR model. The elements of the feature vectors are formed from histograms of residuals obtained using basic DCT templates. The model is based on the calculation of 64 convolutions of inputted JPEG image, decompressed into a spatial region without rounding to integers, with 64 kernels with size 8x8. Then, normalized histograms similar to those used in the previous model are calculated.

Table 1 shows the dimensionality of the feature vector for each of the above models as well as the duration of one vector estimation, estimated on a PC (3.33 GHz Intel Core i5-661 CPU, 8 GB RAM) and averaged over 100 images.

It is obvious that the duration of the feature vectors calculation affects the speed of operations of the stego-analytical system both at the stage of its training and at the stage of detecting the stego images. This feature is particularly critical for real-time systems. As can be seen from Table 1, the difference in the calculation duration of feature vectors for different models is quite noticeable. The vectors are most quickly calculated for the CHEN model, the most slowly for the LIU: when only one file is processed in the LIU model, there will be more than 200 such files in the CHEN model.

However, in the first place, the stego-analytical system must have high accuracy, so estimating the performance of a model based only on calculation duration alone is not appropriate.

Choosing a classifier and defining its parameters. Determination of the training and test sets of cover images

As a classifier, we chose SVM that refers to boundary classification methods. We have used it in a number of previous studies and described, for example, in [10]. Among the possible kernels for this classifier, the simplest linear kernel was chosen to be used. In terms of geometry, the linear classifier corresponds to some separating hyperplane, where the object belongs to the first class if it lies on the positive side of the hyperplane, and to the second class otherwise. In this work, the open library LIBSVM was used, which embodies the method of sequential minimal optimization to construct a separating hyperplane. Classifier training was done with the svmtrain function of the library and was performed in nu-SVC mode.

Note that it is desirable for classifier training with usage of pairs of cover and corresponding to it stego images. This simple step usually improves the accuracy of the stego-analytical system.

Using a random number generator, we split each test set into two parts with 557 containers in each. The first part was used to train the classifier, the second one – for tuned classifier testing. To obtain sufficiently stable results in the sense of independence of accuracy estimations from division into training and control sets, the experiments were repeated 10 times, each time changing the starting number of the generator. The resulting accuracy was calculated as the average of each such series of tests. Several experiments were repeated 100 or 1000 times, and the obtained average accuracy coincided with the accuracy of 10 repetitions at least to the whole values.

Classifier training. Assessing learning speed and comparing it to different models

The learning speed of the classifier is directly influenced by the dimensionality of a feature vector: the more elements are in vector, the slower the separating hyperplane is built. In the same way, the learning speed is affected by the number of images in the training set. At the same time, an insufficient number of training samples results in fast learning but reduced accuracy. And too much of training samples leads to slow learning, which does not improve the accuracy of further steganalysis. According to the results of numerical experiments, it is advisable to select the number of training samples within 500-2000 pieces. Such a wide recommended range results from the using of different models of feature vectors formation, different steganographic methods and other experimental parameters.

Table 2 shows the learning speed of the binary SVM classifier using 1114 cover images in the training set and training on the pairs of cover and stego images

Table 1. Parameters of feature vectors for different models

№	Parameters / Model	Vector dimensionality	Classifier learning speed, sec	The ratio of the number of elements to the speed of learning
1	CHEN	486	3.8	128
2	CC-CHEN	972	7.2	135
3	LIU	216	0.9	240
4	CC-PEV	548	33.5	16
5	CC-C300	48600	403.9	120
6	GFR	17000	137.9	123
7	DCTR	8000	61.9	129

Table 2. Estimated training speed of the classifier for different models

№	Parameters / Model	Vector dimensionality	Classifier learning speed, sec	The ratio of the number of elements to the speed of learning
1	CHEN	486	3.8	128
2	CC-CHEN	972	7.2	135
3	LIU	216	0.9	240
4	CC-PEV	548	33.5	16
5	CC-C300	48600	403.9	120
6	GFR	17000	137.9	123
7	DCTR	8000	61.9	129

formed by Jsteg application. As we can see, as the dimensionality of the feature vector increases, the learning speed decreases. The only exception is the CC-PEV model. Further experiments show that the reason for the decreasing of learning speed, in this case, may lie in the not sufficiently good class separability for this model (see Table 3 classification accuracy).

In general, the optimal model for the criterion of the ratio of the dimensionality of a feature vector to the learning speed is the LIU model. All other models, except CC-PEV model, are characterized by a nearly identical ratio (close to 120-135). However, if we recall the results of Table 1, the LIU model has a significantly slower computational rate than the other models.

Note also that the authors of CC-C300, GFR, and DCTR models propose to use them with ensemble classifier as an alternative to SVM, which scales well with increasing of dimensionality of the feature vector.

Classification of test set images. Assess the classification accuracy and compare it for different models

The accuracy of stego image detection on the basis of the studied statistical models of feature vectors is shown in Table 3. For each model, the average classification accuracy is represented in percentages. And the average number of false-positive and false-negative errors, are indicated in brackets. For example, a value of 92.8% (26; 54) means that in a series of 10 tests out of 1,114 test samples, in average 1034 were classified correctly (92.8% accuracy), 26 cover images were classified as stego ones and 54 stego images in opposite marked as covers.

As a result of these experiments, the LIU, CC-C300 and DCTR models were in the top (see Table 4). The best detection accuracy for Jsteg and Steganos Privacy Suite 2012 applications was provided by the LIU model, and for detecting Jphide-created stego images - the DCTR model. The worst classification accuracy for all three embedding methods was obtained for the CC-PEV model. Also, regardless of the steganographic program, the CC-CHEN model takes advantage over the original CHEN model.

Note that different ratios of performance estimations can be obtained by changing of classifier internal parameters, in particular moving from a linear to a Gaussian kernel, as well as usage another classifier instead of the SVM one. Therefore, the CC-PEV model may be quite competitive in another combination of components of the stego-analytical system.

In cases where performance reductions are acceptable, the detection accuracy may be improved. Yes, it is possible not to choose any one model of formation of feature vectors, but to use several effective ones for this type of cover and stego images. You can either combine or average the results based on different models (for example, Bayesian averaging), and train a separate model what exactly from the available models to use to predict (for example, the decision tree).

In particular, by using the LIU and CC-C300 models together under the same conditions of steganalysis, we have achieved an improvement in the detection accuracy for Jsteg-created stego images to 99.9%. Using LIU and DCTR in conjunction, we increased the detection accuracy of Jphide-created stego images from 93.1% to 97.3%, and of those created by Steganos – from 98% to 99.1%.

Table 3. Accuracy of detection of stego images on the basis of different statistical models

№	Target App / Model	Jstag	Jphide	Steganos
1	CHEN	99.7%(0;24)	83.1%(89;99)	83.1%(67;50)
2	CC-CHEN	98.0%(0;23)	88.2%(61;71)	94.9%(32;25)
3	LIU	98.8%(0;2)	88.8%(57;68)	98.0%(5;17)
4	CC-PEV	84.0%(75;104)	76.9%(79;179)	76.9%(81;177)
5	CC-C300	99.1%(4;6)	91.6%(51;43)	95.7%(25;24)
6	GFR	96.3%(19;22)	91.6%(66;27)	92.0%(40;50)
7	DCTR	98.6%(13;3)	93.1%(45;32)	97.1%(14;18)

Table 4. Sorting models by detection accuracy of stego images

№	Jstag	Jphide	Steganos
1	LIU	DCTR	LIU
2	CC-C300	CC-C300 , GFR	DCTR
3	DCTR	LIU	CC-C300
4	CC-CHEN	CC-CHEN	CC-CHEN
5	CHEN	CHEN	GFR
6	GFR	CC-PEV	CHEN
7	CC-PEV		CC-PEV

Design a multi-class stego-analytical system. Evaluate the accuracy of a multiclass classification and compare it for different models

In the general case, the stego-analytical system should be able to operate in multi-class mode. Note that the problem of effectively extending a binary SVM classifier to solve multi-class recognition problems is still a matter of research. Today, there are two approaches to solving it:

1) Reducing the problem of multi-class classification to binary. The basic strategies are: One against all (OVA), One against one (OVO), directed acyclic graph SVM (DAGSVM), error-correcting output coding (ECOC);
2) Building a multi-class SVM in one step (these are the so-called "All Together" strategies).

Research and comparative analysis of all strategies are beyond the scope of this paper. Due to the high computational complexity of implementation, comparative analysis of different strategies is usually performed on small samples of data [11]. In the general case, it is computationally more difficult to solve the problem of multi-class classification in one step, that is, to directly consider all the output data in one optimization task. In view of the above, we describe only two of the first binary classification reduction strategies that can be applied to construct stego-analytical systems.

Let us learn the classifier $p\colon Q \to X$, where $X = \{1, 2, \ldots, k\}$ and k is the number of classes. The One Against All (or One Against All) strategy involves training k binary classifiers, each separating one class from all others. From the training set $(Q, X) = (\vec{\delta}_n, \chi_n)_{n=1}^N$, k binary samples (Q_1, X_1), (Q_2, X_2), …, (Q_k, X_k) are formed, where in the sample (Q_i, X_i) the cover is marked with a «+1» if it was marked as i in the original sample. In all other cases, the container is marked with a «−1». Thus, for each $i = 1, 2, \ldots, k$ we study the binary classifier $p_i\colon Q \to \{\pm 1\}$ on the basis of the sample (Q_i, X_i). Further, having a set of k binary classifiers, we construct a multiclass classifier according to the rule $p(Q) = \underset{i \in X}{\operatorname{argmax}}(p_i(Q))$.

To avoid ambiguity when multiple classes are assigned to one sample, this strategy requires that classifiers p_i produce not just a class label, but a label with an estimate of its validity (confidence in prediction). Moreover, by estimation of reliability, it should be taken into account that the scale of their values for the basic binary classifiers may differ. In addition, even if the distribution of class samples in the original training set is balanced (the number of samples for each class is less than an order of magnitude), in most cases, binary samples (Q_i, X_i) of samples with a mark -1 will be much larger than $+1$.

An alternative strategy is One-to-One (or All-pairs). This strategy involves pairwise training of $k(k-1)/2$ binary classifiers. In this case, for any $1 \le i < j \le k$ from the original training set $(Q, X) = (\vec{\delta}_n, \chi_n)_{n=1}^N$ a binary sample (Q_{ij}, X_{ij}) is formed, containing only those samples (feature vectors) whose label is i or j. Samples that were labeled i in the original sample are binary $+1$, and those with j are -1. The binary classifier $p_{ij}\colon Q \to \{\pm 1\}$ is trained on the basis of the obtained set. The multiclass classifier as a result outputs the label of the class with the highest number of $+1$:

$$p(Q) = \underset{i \in X}{\operatorname{argmax}} \sum_{\substack{j=1..k, \\ j \neq i}} p_{ij}(Q).$$

It should be noted that when forming the initial training set it is necessary to control that the number of samples for all classes is large enough and representative enough to prevent the problem of overfitting [12]. This strategy like the previous one suffers from ambiguity when the sample receives an equal number of votes in several classes. However, applying it to the SVM is a better choice than One Against All because it has less computational complexity in the learning process and requires less memory resources [13].

Thus, using the one-to-one strategy, we obtained the precision of the multi-class classification shown in Table 5. In this case, the training and test sets contained

Table 5. Detection accuracy of stego images for multi-class SVM

№	Model	Overall accuracy	Number of correctly classified images			
			Covers	Jstag	Jphide	Steagons
1	CHEN	84.3%	422	530	435	492
2	CC-CHEN	89.4%	475	531	467	519
3	LIU	93.23%	487	554	496	540
4	CC-PEV	60.5%	453	416	287	193
5	CC-C300	90.1%	496	549	500	463
6	GFR	88.5%	477	524	510	462
7	DCTR	92.7%	497	553	516	500

557 cover images from each of the four available sets. The estimates were also averaged over a series of 10 experiments. In addition to the overall percentage of accuracy, the table indicates the number of correctly classified covers for each set, by which the accuracy of the detection of the stego images formed by each steganographic programs, can be compared.

As we can see, the accuracy of a multiclass classification is generally lower than the accuracy of a binary one. The sorting of the models by accuracy gives us the same order that was obtained in the binary classification with the attack on Steganos Privacy Suite 2012: LIU, DCTR, CC-C300, CC-CHEN, GFR, CHEN, CC-PEV. For these test image sets, the highest precision of the multi-class classification that was achieved through model sharing was 97.6%. This accuracy was achieved with a combination of four models: LIU, DCTR, CC-C300 and CC-CHEN.

Further research directions

Continuing the analysis and comparison of methods and components for building effective stego-analytical systems, one should investigate how changing a classifier influences on the basic characteristics of steganalysis. In particular, analysis of stego-analytical systems performance by switching from SVM to ensemble classifiers, based on Fisher Linear Discriminants or Generalized Likelihood Ratio Test with replacing majority rule by generalized credibility criterion. In addition, to get a broader picture of the possibilities of steganalysis, it is necessary to supplement the list of studied steganographic methods by the nsF5, YASS, HUGO, WOW, UNIWARD algorithms, etc.

References

[1] N. Koshkina, "Overview and classification of steganoanalysis method," *Control systems and machines*, no. 3, pp. 3–12, 2015.

[2] N. Koshkina, "Steganoanalysis of jpeg images based on control injection attack," *Control systems and machines*, no. 4, pp. 3–17, 2014.

[3] C. Chen and Y. Shi, "Jpeg image steganalysis utilizing both intrablock and interblock correlations," *IEEE ISCAS, International Symposium on Circuits and Systems*, pp. 3029–3032, 2008.

[4] J. Kodovsky and J. Fridrich, "Calibration revisited," *Proceedings of the 11th ACM Multimedia and Security Workshop*, pp. 63–74, 2009. In J. Dittmann, S. Craver, and J. Fridrich, editors.

[5] Q. Liu, "Steganalysis of dct–embedding based adaptive steganography and yass," *Proceedings of the 11th ACM Multimedia and Security Workshop*, pp. 77–86, 2011. In J. Dittmann, S. Craver, and C. Heitzenrater, editors.

[6] T. Pevny and J. Fridrich, "Merging markov and dct features for multiclass jpeg steganalysis," *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX*, vol. 6505, pp. 301–314, 2007. In E. J. Delp, P. W. Wong, editors.

[7] J. Kodovsky and J. Fridrich, "Steganalysis in high dimensions: fusing classifiers built on random subspaces," *8th SPIE Electronic Imaging, Media, Watermarking, Security and Forensics*, vol. 7880, pp. 1–13, 2011.

[8] C. Y. X. L. X. Song, F. Liu and Y. Zhang, "Steganalysis of adaptive jpeg steganography using 2d gabor filters," *Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security. ACM*, pp. 15–23, 2015.

[9] V. Holub and J. Fridrich, "Low complexity features for jpeg steganalysis using undecimated dct," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 219–228.

[10] N. Koshkina, "Steganoanalysis of mic-steganography based on the adjacency matrix and the method of reference vectors," *Artificial Intelligence*, no. 4, pp. 567–577, 2012.

[11] C. Hsu and C. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[12] A. Syrota, *Methods and algorithms for data analysis and their modeling in MATLAB.* BHV-Peterburg, 2017. P. 384.

[13] R. Rifkin, "Multiclass classification." Electronic resource. Access mode: `https://www.mit.edu/~9.520/spring09/Classes/multiclass.pdf`.