A SIMPLE NEURAL NETWORK FOR DETECTION OF VARIOUS IMAGE STEGANOGRAPHY METHODS

Mikołaj Płachta and Artur Janicki

Warsaw University of Technology, Warsaw, Poland

ABSTRACT

This paper addresses the problem of detecting image steganography based in JPEG files. We analyze the detection of the most popular steganographic algorithms: J-Uniward, UERD and nsF5, using DCTR, GFR and PHARM features. Our goal was to find a single neural network model that can best perform detection of different algorithms at different data hiding densities. We proposed a three-layer neural network in Dense-Batch Normalization architecture using ADAM optimizer. The research was conducted on the publicly available BOSS dataset. The best configuration achieved an average detection accuracy of 72 percent.

KEYWORDS

Steganography, deep machine learning, detection malware, BOSS database, image processing.

1. INTRODUCTION

As the Internet evolves, so do the threats lurking in it. Therefore, cyber security is playing an increasingly important role in our world. Threats are becoming more sophisticated and less obvious, making them more difficult to identify and detect. One such threat is transmissions that do not transmit data overtly. This type of method is called steganography, which aims to hide classified information in unclassified material. In other words, it is possible to hide a message in data that is publicly transmitted without revealing the fact that a secret communication exists. These are very dangerous methods for this reason, as it is hard to protect against them, and they can be used to spread malware or can be exploited by such software, known as stegomalware [1].

Most steganographic methods use multimedia data as a carrier of information, such as images. They are called digital media steganography and image steganography, respectively. Examples of such techniques include the Vawtrak/Neverquest method [2], whose idea was to hide URLs in favicon images, or the Invoke-PSImage tool [3], where developers hid PowerShell scripts in image pixels using the commonly used least significant bit (LSB) approach. Another variation can be hiding information in the structure of GIF files [4], which is quite innovative due to the binary complexity of the GIF structure.

As there are already a lot of ways to hide information in this way and it is a big threat to the ordinary user, there is a great need to develop effective, reliable and fast methods to detect hidden content. For this reason, a number of projects have been set up to improve the ability to warn and prevent attacks of this type. One project aimed at stegomalware detection was Secure Intelligent Methods for Advanced RecoGnition of malware and stegomalware (SIMARGL), which was carried out under the EU's Horizon 2020 program.

David C. Wyld et al. (Eds): ARIA, SIPR, SOFEA, CSEN, DSML, NLP, EDTECH, NCWC - 2022
pp. 281-290, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121522

The experiments presented in this paper are a continuation of this initiative. The goal of this research was to find the most effective automatic methods for detecting digital steganography in JPEG images. JPEG compression is commonly used to store and transmit images, so it can be easily exploited for malicious purposes. For this purpose, different variants of neural networks and shallow learning methods have been studied. Detailed studies and obtained results for such methods are described in [5]. This paper mainly focuses on continuing the search for the best predictive model that would work best for the detection of various steganographic algorithms. The research continues exclusively in the area of deep machine learning. This type of detection method can be integrated with antimalware software or any other system that performs file scanning for security purposes (such as a messaging system).

The first part of the paper will recall the theory of steganography and the algorithms and detection methods used in the research, while the second part will present further research along with a comparison to the original research path.

2. RELATED WORK

Our paper focuses on JPEG images as data storage media for image steganography. The popularity of this file format has resulted in many methods of hiding data, as well as various detection methods. This section will briefly review the basics of JPEG-based image steganography, including the most commonly used algorithms.

2.1. Steganographic Methods in JPEG Images

While many steganographic algorithms operate in the spatial domain, some introduce changes at the level of Discrete Cosine Transform (DCT) coefficients stored in JPEG files. Moreover, some algorithms aim to minimize the probability of detection by exploiting content adaptivity: they embed data mainly in less predictable regions, where changes are harder to identify. Such modifications are the most difficult to detect, which is why they were chosen as the leading ones at the beginning of the ongoing research. After analyzing image collections, e.g. [6], we selected three algorithms: nsF5 [7], JPEG Universal Wavelet Relative Distortion (J-Uniward) [8] and Uniform Embedding Revisited Distortion (UERD) [9]. They are briefly characterized in the following subsections.

2.2. nsF5

The nsF5 algorithm embeds data by modifying the least significant bits of the AC (having at least one non-zero value) of the DCT coefficients of unmodified JPEG objects. The data is hidden using syndrome coding. Having an m p-bit message to embed using n values of AC DCT our task is to obtain a vector y. This vector must satisfy the equation:

$$Dy = m \tag{1}$$

where D is a binary matrix p by n, which is shared by the sending and receiving parties. The embedder must find a solution to the above equation that does not require modification of the zero-value coefficient bits. The solution must minimize the Hamming weight between the modified and unmodified vectors of the least significant bit. The above version is a simple syndrome coding idea, but a more sophisticated coding scheme, Syndrome Trellis Coding (STC) [19], using a parity check matrix instead of D, is usually used. The y-vector represents the path through the trellis built based on the parity check matrix.

2.3. J-Uniward

J-Uniward is a method for modeling steganographic distortion caused by data embedding. The goal is to provide a function that determines which areas of an unmodified object are less predictable and more difficult to recognize. Changes introduced during steganographic data embedding in these areas are more difficult to detect than if they were introduced uniformly across the media. By calculating the relative changes in value based on the directional decomposition of the filter bank, the method is able to detect smooth edges that are easy to recognize. By analyzing in this way which areas may be more susceptible to detection, this method gives a very high efficiency in little-noticed data hiding. As with nsF5, the STC coding scheme is used to create a data hiding algorithm that adapts to the content.

2.4. UERD

UERD is another steganographic embedding model that aims to minimize the probability of detecting the presence of hidden information by reducing the impact of embedding on the statistical parameters of cover information. It achieves this by analyzing the parameters of the DCT coefficients of individual mods, as well as entire DCT blocks and their neighbors. It can then determine whether a region can be considered "noisy" and whether embedding will affect statistical features such as file histograms. "Wet" regions are those where statistical parameters are predictable and where embedding would cause a risk of information detection. The use of values during embedding such as DC mode coefficients or zero DCT coefficients are not excluded. This is because their statistical profiles can make them appropriate from a security perspective. The UERD algorithm evenly distributes the relative changes in statistics resulting from embedding. UERD, like nsF5 and J-Uniward, uses STC to hide message bits in desired values.

3. STEGANOGRAPHY DETECTION

Image steganography is an important topic in cyber security, and so far one can read in the literature about a very large number of attempts to detect it. These methods usually extract certain parameters from analyzed images, and then classification algorithms are applied. They are usually based on machine learning approaches, so shallow or deep methods can be used. The research described in this paper focuses only on the deep ones, so this section first describes the features most commonly used in steganalytic algorithms, and then briefly describes typical examples of detection algorithms based on deep learning.

3.1. Feature Space Extraction

While many feature space analysis methods for image steganalysis have been described in the literature, three of the most effective were selected. The first one analyzed was Discrete Cosine Transform Residuals (DCTR) [10], which analyzes the data resulting from obtaining DCT values for a given image. In the first step, a random 8x8 pixel filter is created, which will be applied later to filter the entire image. Then, iterating step by step over the analyzed image, a histogram is created using the spline function with the previously mentioned filter. The article [11] proposes an example of using DCTR parameters in combination with a multilevel filter. Another variation of this approach is a method based on Gabor filters, or Gabor Filter Residuals (GFR) [12]. It works in a very similar way to DCTR, but instead of a random 8x8 filter, Gabor filters are used. The article [13] describes a successful application of the GFR function in JPEG steganography detection. A third approach to parameterizing the feature space is to use the PHase Aware

pRojection Model (PHARM) [14]. Applying various linear and nonlinear filters, a histogram is created from the projection of values for each residual portion of the image.

3.2. Steganography Detection Methods

Recently, neural networks have been among the most popular machine learning methods used in various task automation applications. Detecting steganographically hidden data in digital images is one of them. Extracted image parameters based on decompressed DCT values, which were pre-filtered and fed into the first weave layer of the network, were usually used as input data.

Proprietary variants of convolutional networks, such as XuNet [15] or ResNet [16], are most commonly used for this purpose. A common feature of these networks is the combination of Convolution-BatchNormalization-Dense (C-CB-D) structures, i.e. a spline function, a normalization layer and a base layer of neurons with an appropriate activation function. Functions such as Sigmoid, TLU (Threshold Linear Unit) and Gaussian are used, but the most common are Rectified Linear Unit (ReLU) or TanH. Steganography detection models based on feature extraction and the C-BN-D scheme are shown in Figure 1.

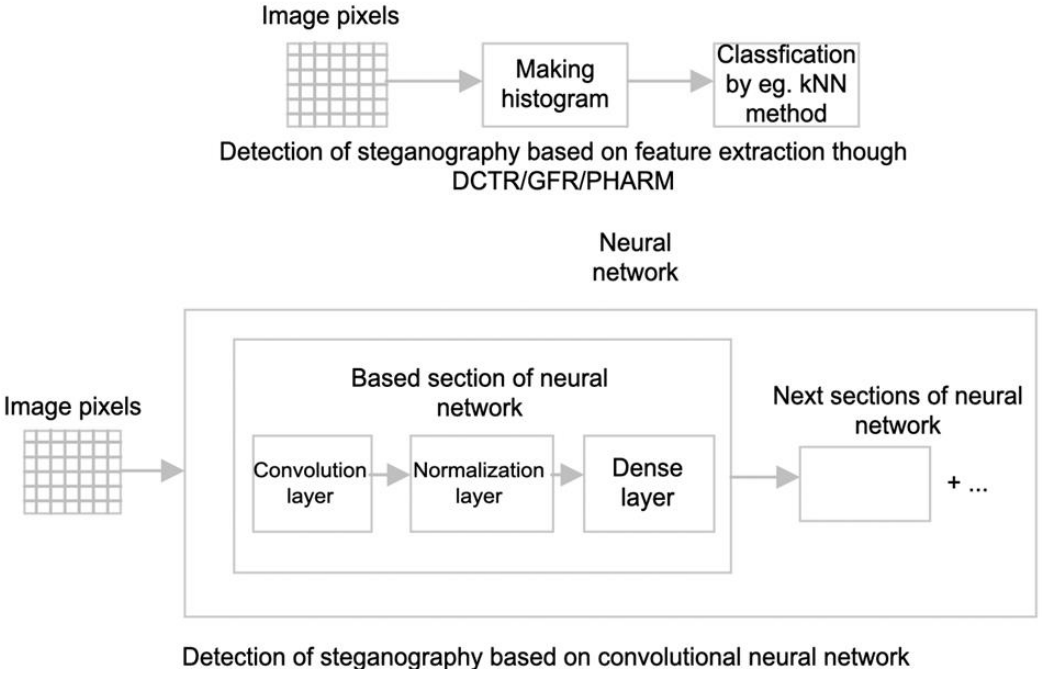

Figure 1. Examples of prediction models

4. RESEARCH METHODOLOGY AND MATERIALS

4.1. Data Set Under Study

The "Break Our Steganograhic System" (BOSS) image set [17], which contains 10,000 black-and-white images (without hidden data), was used for the study. The images were converted to JPEG format with a quality factor of 75. Three other sets of images were then generated, hiding there random data with a density (bpnzac, i.e. the number of bits for each non-zero AC co-factor) of 0.4 or 0.1, using the previously mentioned three steganographic algorithms: nsF5, J-Uniward

and UERD. Each dataset was then divided in parallel into training and test subsets, in a 90:10 ratio.

4.2. The Proposed Detection Method

The neural network environment was based on the Keras library and Tensorflow due to the easy definition of the model. The proposed network architecture was mainly based on the Dense-BatchNormalization structure but did not use the spline part as described in the available literature. A schematic of this concept is shown in Figure 2.

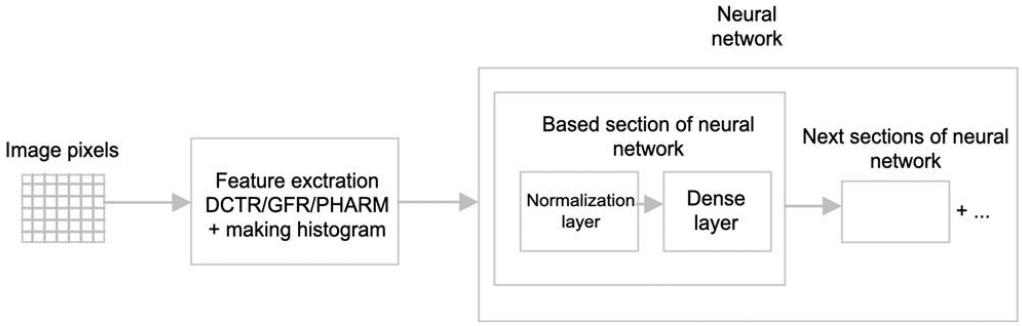

Figure 2. Proposed model for research

We also tested different activation functions for the dense layer, but the best results were obtained for the ReLU function. After extensive research, one optimizer was selected: Adaptive Moment Estimation (ADAM) [18], which gave better results than the others like Stochastic Gradient Descent etc. The last parameter that significantly affected the model's learning efficiency was the learning rate. During the study, it turned out that lowering it gave very promising results without changing the network architecture and optimizer.

4.3. Neural network learning environment

The research consisted of two parts. The first part was similar to the preliminary research described in the article [5]. First, two neural network model architectures were selected:

- The first with three layers with ReLU activation function, with 250 neurons in the first, 120 in the second and 50 in the third, used in four neural network models;
- The second with two layers also with ReLU function, having 500 neurons in the first layer and 250 in the second, used in the last (fifth) neural network model.

No spline layers were used, while additional normalization layers (BatchNormalization) were applied between the simple layers. All models use the ADAM optimizer. The SGD optimizer was also tested in a previous phase of research described here [5], but it gave relatively poor results and can be ignored. The learning rate used values of $1e^{-4}$ or $1e^{-5}$ for ADAM. In this way, three configurations were prepared. The results are shown in Table 1

Table 1. Results of accuracy for all three configurations for matching conditions, i.e., detection model was dedicated to given steganographic algorithm (a dash means that network learning did not successfully converge)

Network Architecture	Learning Rate	Parameters	J-Uniward		nsF5		UERD		Average
			0.1	0.4	0.1	0.4	0.1	0.4	
250 x BN x 120 x BN x 50 (3 layers)	$1e^{-4}$	DCTR	-	83.1	**76.3**	98.8	**66.5**	**94.5**	**78.3**
		GFR	-	86.5	68.3	95.5	63.4	92.9	76.1
		PHARM	-	74.7	62.3	95.9	51.4	88.5	70.5
	$1e^{-5}$	DCTR	-	83.0	74.2	**99.7**	64.7	93.1	77.5
		GFR	-	**88.4**	68.0	98.2	62.6	92.5	76.6
		PHARM	-	76.1	66.1	93.4	55.5	89.4	71.8
500 x BN x 250 (2 layers)	$1e^{-5}$	DCTR	-	80.8	73.5	99.6	61.9	93.5	76.6
		GFR	**53.6**	86.4	67.6	97.4	64.2	91.9	76.9
		PHARM	-	75.0	54.1	94.2	54.0	87.9	69.2

Next, the best model configuration was selected, i.e., a three-layer model with the ADAM optimizer at a learning rate of $1e^{-4}$, and six separate models were taught for this configuration, one for each version of the set. Next, cross-testing was carried out, that is, each model was tested to see how it performed in detecting all six harvests. These results are shown in Table 2.

4.3. Evaluating the Effectiveness of Models

To assess the effectiveness of the resulting models, popularly used metrics were applied. The first type is accuracy, which defines what percentage of the entire examined data set is correctly classified. The second metric is precision, which defines what proportion of the images indicated by the classifier as belonging to a given class actually do.

Table 2. Accuracy results of cross-testing DCTR model. Best results in columns are shown in bold.

DCTR model version	J-Uniward		nsF5		UERD		Average
	0.1	0.4	0.1	0.4	0.1	0.4	
J-Uniward 0.1	50.2	51.4	50.4	54.3	50.4	52.3	51.5
J-Uniward 0.4	**53.6**	**83.1**	64.2	88.7	57.6	84.5	**72.0**
nsF5 0.1	52.9	70.7	**76.3**	85.9	56.7	82.7	70.9
nsF5 0.4	50.2	55.5	53.5	**98.8**	50.6	63.0	61.9
UERD 0.1	53.3	71.2	63.3	76.9	**66.5**	76.8	68.0
UERD 0.4	50.8	66.1	54.9	95.8	54.1	**94.5**	69.3

The next metric is recall, which determines what fraction of images of a given class will be detected by the model. The fourth metric analyzed is F1-score, which is the harmonic mean of precision and recall. The last metric we used to test the effectiveness of the model is the area under the ROC curve (AUC). ROC curves will also be presented, as they can show the effectiveness of a given model very well. The larger the area under the ROC curve (i.e., AUC), the more effective the model is. In the first and second phases of the study, the main metric used was accuracy, while for the best model, which was selected after cross-testing, the other metrics will also be calculated. Since the test set is perfectly balanced, the accuracy score is not biased and reflects well the detection ability of a given classifier.

5. OBTAINED RESULTS

Table 1 shows the accuracy results for each configuration tested in the first phase of the study. The two-layer model was noticeably worse than the three-layer versions. The best results were obtained for the ADAM optimizer at a learning rate of 1e-4 with the three-layer neural network model. For the matching conditions (detection model trained on data generated with the same image steganographic method) the worst accuracy results were obtained for J-Uniward sets, better on UERD, and the best for nsF5 sets. When analyzing feature spaces, PHARM-based models were the least accurate, while GFR and DCTR were very close to each other with a slight advantage for DCTR. Therefore, a model with the DCTR feature space was selected for further testing.

In the second part of the study as for cross-testing, one can notice that obviously the best scores are mostly on the diagonal of Table 2, meaning the matching condition. However, it can be seen that the model trained on the J-Uniward 0.4 set performed best as for classification of images. Additional metrics shown in Table 3 were calculated for this configuration.

Table 3. All metrics for best cross-testing DCTR model

	J-Uniward		**nsF5**		**UERD**		
Metrics	**0.1**	**0.4**	**0.1**	**0.4**	**0.1**	**0.4**	**Average**
Accuracy	53.6	83.1	64.2	88.7	57.6	84.5	72.0
Precision	57.1	80.2	69.9	82.1	63.0	80.8	72.2
Recall	28.8	87.7	50.1	99.1	36.8	90.7	65.5
F1-score	38.3	83.8	58.4	89.8	46.5	85.4	67.0
AUC	57.3	91.6	70.2	99.1	63.2	93.3	79.1

As one can observe the precision and recall parameters, for sets with a density of 0.4 the results are close to the precision, which means that the model is well balanced. There is a larger difference for a density of 0.1 due to the greater hiding of data in the files. In Figure 3, we can see that the values of accuracy, F1-score and AUC are close to each other at each harvest. It can also be visually observed that the weakest results were obtained for J-Uniward, and the best for nsF5. Figure 4 shows the ROC curves, which illustrate the efficiency in detection of a given harvest. They are consistent with previous results for this model.

6. SUMMARY

Comparing the results obtained with previous results from the article [5], it can be seen that using a single model instead of separate six models for universal detection is feasible. The difference in the average accuracy score between the two concepts is on the order of 10 percent relative, which, with the possible complexity of interpreting data from six separate models plus a high chance of false-positive cases, is a very good and promising result.

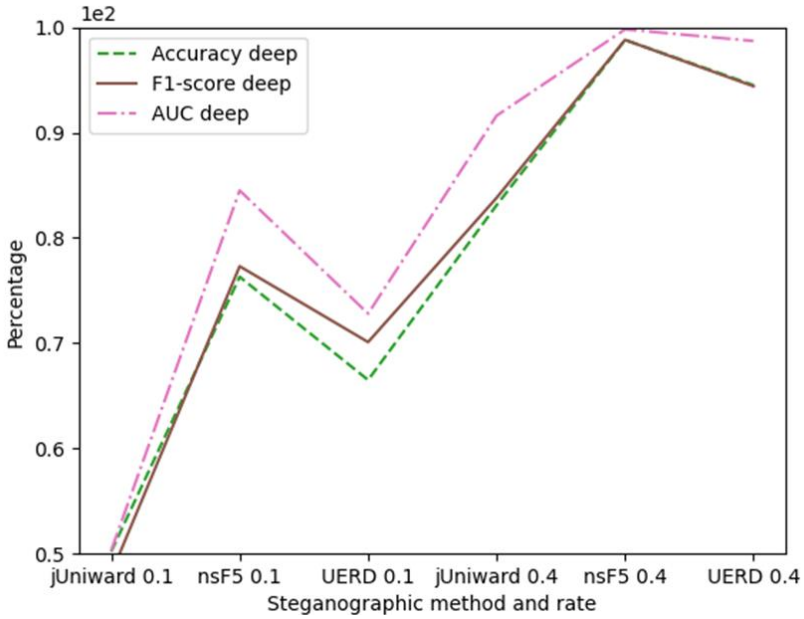

Figure 3. Accuracy for the best DCTR model

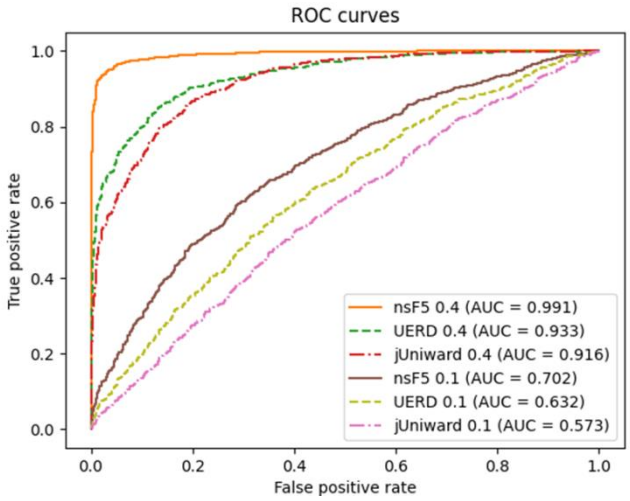

Figure 4. ROC curves for the best DCTR model

During the study, it was also noted that adding a normalization layer was able to improve the results significantly. For example, for the nsF5 sets, t he difference was on the order of 15-20 percent relative. This means that the normalization layer from the C-BN-D model is indispensable over the splicing part, which can be dispensed with. Also, it was noted that the difference between two and three layers of dense networks is practically imperceptible, and there is no good reason to use much more complicated networks. Regardless of the phase one or phase two tests conducted, the J-Uniward algorithm was the most difficult to detect, and the easiest was nsF5. Also, for sets of 0.1 there is noticeably worse detection than for sets of 0.4, which means that the less data we hide in an image, the lower the chance of discovering it.

This paper analyzed the effectiveness of image steganography detection based solely on a single neural network model. The effectiveness depended on the algorithm used as well as the data density used. Analyzing the results, we can see that one network model using the DCTR feature

space did quite well in detecting most threats. This gives hope for further potential improvements to this model using some combination of two or three base models. Further research on this issue will be conducted in the next iteration.

ACKNOWLEDGEMENTS

The study has been partially supported by the SIMARGL Project with the support of the European Commission and the Horizon 2020 Program, under Grant Agreement No. 833042 and also by the IDUB program from the Warsaw University of Technology.

REFERENCES

[1] Caviglione, L.; Choraś, M.; Corona, I.; Janicki, A.; Pawlicki, M,.Mazurczyk.W.; Wasielewska, K. Tight Arms Race: Overview of Current Malware Threats and Trends in Their Detection. IEEE Access 2021, 9, 5371–5396

[2] Cabaj, K.; Caviglione, L.; Mazurczyk, W.; Wendzel, S.; Woodward, A.; Zander, S. The New Threats of Information Hiding: The Road Ahead. IT Professional 2018, 20, 31–3

[3] Encodes a PowerShell script in the pixels of a PNG file and generates a oneliner to execute. https://github.com/peewpw/ Invoke-PSImage. Accessed: 2022-01-18

[4] Puchalski, D.; Caviglione, L.; Kozik, R.; Marzecki, A.; Krawczyk, S.; Choraś, M. Stegomalware Detection through Structural Analysis of Media Files. Proc. 15th International Conference on Availability, Reliability and Security; Association for Computing Machinery: New York, NY, USA, 2020; ARES '20.

[5] Płachta, M.; Krzemień, M.; Szczypiorski, K.; Janicki, A. Detection of Image Steganography Using Deep Learning and Ensemble Classifiers. Electronics 2022, 11, 1565. https://doi.org/10.3390/electronics11101565

[6] Yang, Z.; Wang, K.; Ma, S.; Huang, Y.; Kang, X.; Zhao, X. IStego100K: Large-scale Image Steganalysis Dataset. International Workshop on Digital Watermarking. Springer, 2019

[7] Fridrich, J.; Pevný, T.; Kodovský, J. Statistically undetectable JPEG steganography: Dead ends, challenges, and opportunities. the 9th ACM Multimedia & Security Workshop. Association for Computing Machinery, 2007, p. 3–14.

[8] Holub, V.; Fridrich, J.; Denemark, T. Universal distortion function for steganography in an arbitrary domain. EURASIP Journal on Multimedia and Information Security 2014

[9] Guo, L.; Ni, J.; Su, W.; Tang, C.; Shi, Y.Q. Using Statistical Image Model for JPEG Steganography: Uniform Embedding Revisited. IEEE transactions on information forensics and security 2015

[10] Holub, V.; Fridrich, J. Low-complexity features for JPEG steganalysis using undecimated DCT. IEEE Transactions on Information Forensics and Security 2015

[11] Wang, C.; Feng, G. Calibration-based features for JPEG steganalysis using multi-level filter. 2015 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), 2015,

[12] Song, X.; Liu, F.; Yang, C.; Luo, X.; Zhang, Y. Steganalysis of adaptive JPEG steganography using 2D Gabor filters. Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security; Association for Computing Machinery: New York, NY, USA, 2015; IH&MMSec '15,

[13] Xia, C.; Guan, Q.; Zhao, X.; Xu, Z.; Ma, Y. Improving GFR Steganalysis Features by Using Gabor Symmetry and Weighted Histograms. Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security; Association for Computing Machinery: New York, NY, USA, 2017; IH&MMSec '17, p. 55–66.

[14] Holub, V.; Fridrich, J. Phase-aware projection model for steganalysis of JPEG images. Media Watermarking, Security, and Forensics 2015; Alattar, A.M.; Memon, N.D.; Heitzenrater, C.D., Eds. International Society for Optics and Photonics, SPIE, 2015, pp. 259 – 269

[15] Xu, G.; Wu, H.Z.; Shi, Y.Q. Structural Design of Convolutional Neural Networks for Steganalysis. IEEE Signal Processing Letters 2016, 23, 708–712

[16] He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recongnition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[17] Break Our Steganographic System Base webpage (BossBase). http://agents.fel.cvut.cz/boss/. Accessed: 2022-01-18

[18] Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization, 2014.
[19] Filler, T.; Judas, J.; Fridrich, J. Minimizing Embedding Impact in Steganography using Trellis-Coded Quantization. Media Forensics and Security II; Memon, N.D.; Dittmann, J.; Alattar, A.M.; III, E.J.D., Eds. International Society for Optics and Photonics, SPIE, 2010

AUTHORS

Mikołaj Płachta doctoral student at the Warsaw University of Technology, mobile application developer by profession, research area mainly focused on the study of the operation of neural networks in the application of steganography detection.

Artur Janicki university professor at the Cybersecurity Division of the Institute of Telecommunications, Warsaw University of Technology. His research and teaching activities focus on signal processing and machine learning, mostly in cybersecurity context. Member of technical program committees of various international conferences, reviewer for international journals in computer science and telecommunications. Author or co-author of over 80 conference and journal papers.