Deep Learning Based Image Steganalysis

Alexandr Kuznetsov

Department of Political Sciences, Communication and International Relations, University of Macerata, Macerata, Italy, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine

[kuznetsov@karazin.ua](mailto:kuznetsov@karazin.ua)

<https://orcid.org/0000-0003-2331-6326>

Luca Romeo

Department Economics and Law, University of Macerata, Macerata, Italy

Department of Information Engineering, Università Politecnica delle Marche, Ancona, Italy

[luca.romeo@unimc.it](mailto:luca.romeo@unimc.it)

<https://orcid.org/0000-0003-1707-0147>

Nicolas Luhanko

School of Physics and Technology, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine

[luhanko2021tya11@student.karazin.ua](mailto:luhanko2021tya11@student.karazin.ua)

<https://orcid.org/0000-0002-6478-0419>

Dmytro

Emanuele Frontoni

Department of Political Sciences, Communication and International Relations, University of Macerata, Macerata, Italy, Università Politecnica delle Marche, Ancona, Italy

[emanuele.frontoni@unimc.it](mailto:emanuele.frontoni@unimc.it)

<https://orcid.org/0000-0002-8893-9244>

Riccardo Rosati

Department of Information Engineering, Università Politecnica delle Marche, Ancona, Italy

[r.rosati@pm.univpm.it](mailto:r.rosati@pm.univpm.it)

<https://orcid.org/0000-0003-3288-638X>

*Abstract*—Various cryptographic and steganographic techniques are used to hide digital information during its processing, storage, and transmission. While cryptography hides the information content of digital data (by converting them into a meaningless set of noise-like sequences), steganography hides the very existence of information messages. In other words, steganographic techniques hide digital messages by embedding them in so-called. containers. Containers are other digital data or physical objects. To do this, containers (covers, media) must be highly redundant data. Revealing the fact of steganographic hiding and detecting an embedded message is usually extremely difficult. In fact, hidden messages are some noise added to the container, and we must, based on the study of this noise, decide on the presence or absence of an embedded message. In this article, we consider deep learning methods for steganoanalysis of digital cover images. We have considered several deep learning models and conduct numerous tests on various datasets. Our experiments show that deep learning does indeed make it possible to design effective stego-detectors, but this requires fine-tuning of model hyperparameters and optimization of the neural network architecture.

Keywords—deep learning; image steganalysis; digital images; styling; convolutional neural network

# Introduction

Digital steganography as a science of hiding information appeared long time ago [1]. Historical examples of the use of various techniques for hiding the fact of the presence of informational messages are known: invisible ink, microfilms, marked cards, stencils and more.

In the modern world, in the era of the Internet and digital technologies, steganography is used much more often [2]–[4]: to hide confidential correspondence between users; in digital watermarks for copyright purposes; for embedding reference and service information and creating related repositories, and much more.

Modern steganography uses various data hiding techniques. For example, works [5]–[8] are devoted to cluster steganosystems. Information messages are hidden here in the structure of the file system, while the amount of data on the storage medium does not change. In [9]–[11], data is hidden in solid objects using 3D modeling techniques. In [12]–[15], messages are hidden in data transfer protocols. There are many other interesting directions in steganography, but most often messages are embedded in digital redundant digital data (photo and video images, audio, etc.) [16]–[18].

The simplest digital steganography techniques use least significant bit (LSB) coding to hide informational messages [19]–[21]. More advanced techniques, such as those in [22], [23], hide information data in audio by encoding the phases of the signal; in [24]–[26] echo signals are used for hiding in audio. The most common are image containers, for which many different steganographic methods are known [2], [4], [16]. As a rule, raster images are used, but several efficient methods have also been proposed for vector graphics [27]–[31]. For digital images and audio, direct-sequence spread spectrum [32]–[36] are also used.

Existing steganalysis techniques have also been investigated in the works of various authors. For example, in [37] methods of statistical analysis of images are studied. In [38], [39] and many other papers, steganalysis techniques using discrete orthogonal transformations are studied. In [40], a universal method of steganalysis based on reference points is proposed, in [41], estimates of local dimensions are used, etc. However, the most promising area of research is deep learning methods.

In this paper, we investigate deep learning methods for detecting steganocontainers and conduct numerous experiments with different datasets. We show that neural network models do enable stego detection, but this requires fine-tuning the hyperparameters and optimizing the architecture. Our goal is to test the basic probabilistic characteristics in stego detection. For this purpose, we consider three implementations of the SRNet model:

* Pytorch Implementation [42];
* TensorFlow Implementation [43];
* Our own implementation presented on the resource [44].

# Materials

Following the basic work [45], which we use as the starting point of our research, we consider several datasets. In particular, we consider the combined dataset BOWS2 [46] and BOSSbase [47]–[49] as described in [45].

## BOWS2 dataset

The BOWS2 dataset [46] consists of 10,000 512\*512 images. Each of the files is a monochrome image. Each image is encoded in portable gray map format using ASCII decimal. The format does not distort the picture or in other words saves every pixel’s intensity .

## BOSSbase dataset

BOSSbase [47]–[49] also consists of 10,000 images. The same transformations were performed with this dataset as with the previous BOWS2 dataset.

## Stego augmentation

Initial images from BOWS2 [46] and BOSSbase [47]–[49] were used to form stego images. For this purpose, we used the S-uniward algorithm [50], [51].

The S-uniward parameters that were used in obtaining the stego-images:

● Bytes per pixel 0.4;

● Random state 42.

Each image from BOWS2 [46] and BOSSbase [47]–[49] was used to obtain the stego-image. This step doubled each of the datasets. Subsequently, each one of these pairs was used in training so that the model was not trained on the features which human eye can detect, but only on finding stego noise.

## Concatenated dataset

Both the BOWS2 [46] and BOSSbase [47]–[49] datasets discussed above were used to create a Concatenated dataset.

# Methods

In the paper [45] describes a deep residual network for stegoanalysis of digital images (SRNet model). The implementation of this model on Pytorch [42] is given on the GitHub resource. In [43], another implementation of SRNet using the TensorFlow library is given. In our work, we tested both of these models, as well as our own implementation of SRNet. In [45], 97 epochs were used to train SRNet, Pytorch Implementation [42] was trained on 500 epochs (this is specified as the default value); TensorFlow Implementation [43] was trained on 200 epochs.

The SRNet model is based on the encoder architecture.

The model has four types of layers as well as a Dense layer (a layer that is deeply connected with its preceding layer which means the neurons of the layer are connected to every neuron of its preceding layer) for final responses.

The layers are arranged in this order:

* Two type 1 layers;
* Five type 2 layers;
* Four type 3 layers;
* One type 4 layers;

Accordingly, each of the layers has the following feature maps (from the beginning to the end of the network): 64, 7, 16, 64, 128, 256, 512, and Dense layer. A feature map, or activation map, is the output activations for a given filter.

Thus, the SRNet deep learning model has the following architecture:

* Layer type 1: convolutional layer, batch-normalization, RELU;
* Layer type 2: layer type 1, convolutional layer, batched normalization;
* Layer type 3: layer type 1, convolution layer, batch normalization, average pooling 3\*3 stride 2;
* Layer type 4: layer type 1, convolution layer, batch normalization, global average pooling.

Only type 1 layers has activation layer. Skip-connections are present in type 2 and type 3 layers. There are no layers in skip-connection in type 2 layer.To layer type 3 Skip-connection was added a 1\*1 convolution for channel mixing and a batch normalization.

This architecture was chosen as the most suitable for noise detection. Although it is impossible to predict exactly what functionality this or that part of the neural network will perform, it is assumed that the neural network can be divided into three parts. The first segment, consisting of type 1 layers, is responsible for noise extraction. The second segment, consisting of layers of type 2, is used to reduce the dimensionality of the feature map. The third segment is a fully connected layer, which ends with a softmax layer, designed to obtain the probabilities of the relation to one of the classes.

# Results and Discussion

Three cases with different payloads were considered to test the SRNet model [52]:

* Bytes per pixel 0.3;
* Bytes per pixel 0.4;
* Bytes per pixel 0.5.

The obtained test results show that the SRNet deep learning model can indeed be used to solve complex problems of image steganolysis. All three implementations of the model show learning ability for stego recognition. To compare the obtained results, Tables I-IV show Accuracy, Recall, Precision and F1-score.

Recall that in binary classification problems there are 4 possible outcomes:

* True positive () - when the stego-image is correctly classified;
* True negative () - when a clean image is correctly classified;
* False positive () - when the stego is misclassified as a clear image. This outcome is also known as a "Type I error" or "false alarm";
* False negative () - when a pure image is misclassified as an stego. This outcome is also known as "type II error" or "target miss".

Then for each possible result we define the indicator of binary classification efficiency as the frequency of the corresponding events:

* True positive rate () outcomes:

 

* False negative rate () outcomes:

 

* False positive rate () outcomes:

 

* True negative rate () outcomes:

 

where the total number of images is the sum of the number of stego and  cover images participating in the experiment

Accuracy is the proportion of correct predictions (both true positives and true negatives) among the total number of cases considered.

 

Recall characterizes the share of detected stego in their total number:

. (6)

Precision characterizes the share of detected stegos in the total number of evaluations:

. (7)

Recall in this context is also referred to as the true positive rate or sensitivity, and precision is also referred to as positive predictive value.

F1-score is a measure of test accuracy that is calculated as the harmonic mean of Recall and Precision :

 (8)

The maximum value of F1-score is 1, which indicates perfect precision and recall. The smallest value of F1-score is 0 if either precision or recall is zero.

1. Obtained Accuracy values for different SRNet model implementations

|  |  |  |  |
| --- | --- | --- | --- |
| Payload | Our Implementation, 80 epochs | TensorFlow Implementation, 200 epochs | Pytorch Implementation, 500 epochs |
| 0.3 bpp | 0.6860 | 0.7735 | 0.7006 |
| 0.4 bpp | 0.7443 | 0.8335 | 0.7539 |
| 0.5 bpp | 0.7972 | 0.8622 | 0.7874 |

1. Obtained Recall values for different implementations of the SRNet model

|  |  |  |  |
| --- | --- | --- | --- |
| Payload | Our Implementation, 80 epochs | TensorFlow Implementation, 200 epochs | Pytorch Implementation, 500 epochs |
| 0.3 bpp | 0.6642 | 0.7564 | 0.6939 |
| 0.4 bpp | 0.7262 | 0.8085 | 0.7436 |
| 0.5 bpp | 0.7797 | 0.8768 | 0.7835 |

1. Obtained Precision values for different SRNet model implementations

|  |  |  |  |
| --- | --- | --- | --- |
| Payload | Our Implementation, 80 epochs | TensorFlow Implementation, 200 epochs | Pytorch Implementation, 500 epochs |
| 0.3 bpp | 0.7524 | 0.8068 | 0.7178 |
| 0.4 bpp | 0.7844 | 0.874 | 0.775 |
| 0.5 bpp | 0.8284 | 0.8428 | 0.7942 |

1. Obtained F1-score values for different SRNet model implementations

|  |  |  |  |
| --- | --- | --- | --- |
| Payload | Our Implementation, 80 epochs | TensorFlow Implementation, 200 epochs | Pytorch Implementation, 500 epochs |
| 0.3 bpp | 0.7056 | 0.7808 | 0.7057 |
| 0.4 bpp | 0.7542 | 0.84 | 0.759 |
| 0.5 bpp | 0.8033 | 0.8595 | 0.7888 |

We have tested three implementations of the SRNet deep learning model. Our implementation and the Pytorch Implementation show comparable performance. The slight outperformance of some metrics may be due to the longer training time of Pytorch Implementation.

TensorFlow Implementation shows better performance in all the considered indicators. This is explained by the additionally introduced dense layer (512), which improves the classification accuracy.

To compare the obtained simulation results with the data from [45], let us consider one more parameter:

. (9)

This is the arithmetic mean of the errors of the first and second kind.

Table V shows our estimates of the probability (9) when using three different implementations of the SRNet model. In the table we also give the author's estimate from [45].

1. Comparison of simulation results for different SRNet model implementations

|  |  |  |  |
| --- | --- | --- | --- |
| Payload | 0.3 bpp | 0.4 bpp | 0.5 bpp |
| [45], 457 epochs | 0.1432 | 0.1023 | 0.0705 |
| Our Implementation, 80 epochs | 0.1570 | 0.1279 | 0.1014 |
| TensorFlow Implementation, 200 epochs | 0.1133 | 0.0833 | 0.0689 |
| Pytorch Implementation, 500 epochs | 0.1497 | 0.1231 | 0.1060 |

The models results shown in Table V correspond to the case of stego when using the S-uniward concealment algorithm [50], [51]. In general, we can conclude that the characteristics stated in [45] are confirmed. The SRNet model does effectively detect steganoimages, the error probability decreases as Payload increases. The slightly better performance for the TensorFlow Implementation is explained by the additionally introduced dense layer. For our implementation, the worst results are obtained, but they correspond to only 80 epochs of training. The author's paper [45] shows results for 457 epochs. The TensorFlow Implementation was trained on 200 epochs [43], for the Pytorch Implementation a default value of 500 epochs of training is given [42]. Thus, our implementation shows good detection results with relatively short training times.

# Conclusion

Deep learning models find applications in a wide variety of human activities. In this paper, we investigated the problem of image steganalysis. This is an important area of modern cybersecurity, since multimedia content is widely used in inten- tional applications and images are often used for covert transfer of information data. The goal of steganoanalysis is to detect stoic containers when an image is used as a carrier of secret messages. This is a complex problem and our results show that deep learning models produce very efficient steganalysis solutions.

We considered one of the most advanced deep learning models proposed in [45]. It is the SRNet model, which minimizes heuristics and the use of external extras and provides high detection accuracy in both spatial and DCT domains. We considered three independent implementations of this model and conducted numerous tests. All tests established high stego detection performance, with error probability decreasing as Payload increases.

It should be noted that different implementations of the SRNet model show different steganalysis performance. The TensorFlow Implementation trained on 200 epochs showed the highest efficiency. The average probability of stego detection error for this implementation was minimal. The results were even better than in the author's work [45]. This is explained by the additionally introduced layers in the deep learning architecture. The next most efficient implementation is the Pytorch Implementation, trained on 500 epochs (this value is specified by default). Our own implementation showed the lowest efficiency values, but we trained the model only for 80 epochs. Overall, we can conclude that the general conclusions and estimates from the author's work [45] with 457 epochs of training look plausible.

A promising direction for further research is to extend the field of testing to other datasets using various steganoalgorithms. Further improvement of the deep learning architecture for efficient stego detection in various practically important scenarios also seems important.

##### References

[1] F. A. P. Petitcolas, R. J. Anderson, Markus, and G. Kuhn, “Information hiding—A survey,” in *Proc. IEEE*, 1999, pp. 1062–1078.

[2] A. Yahya, “Introduction to Steganography,” in *Steganography* Techniques *for Digital Images*, A. Yahya, Ed. Cham: Springer International Publishing, 2019, pp. 1–7. doi: 10.1007/978-3-319-78597-4\_1.

[3] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography, 2nd Ed.*, 2nd Edition. Amsterdam ; Boston: Morgan Kaufmann, 2007.

[4] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*, Illustrated Edition. Cambridge ; New York: Cambridge University Press, 2009.

[5] H. Khan, M. Javed, F. Mirza, and S. A. Khayam, “Evading Disk Investigation and Forensics using a Cluster-Based Covert Channel,” May 2012.

[6] A. Venčkauskas, N. Morkevičius, G. Petraitis, and J. Čeponis, “Covert Channel for Cluster-based File Systems Using Multiple Cover Files,” *Information Technology and Control*, vol. 42, no. 3, Art. no. 3, Sep. 2013, doi: 10.5755/j01.itc.42.3.3328.

[7] K. Y. Shekhanin, A. O. Kolhatin, E. E. Demenko, and A. A. Kuznetsov, “ON HIDING DATA INTO THE STRUCTURE OF THE FAT FAMILY FILE SYSTEM,” *TRE*, vol. 78, no. 11, 2019, doi: 10.1615/TelecomRadEng.v78.i11.50.

[8] A. Kuznetsov, K. Shekhanin, A. Kolhatin, I. Mikheev, and I. Belozertsev, “Hiding data in the structure of the FAT family file system,” in *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, May 2018, pp. 337–342. doi: 10.1109/DESSERT.2018.8409155.

[9] A. Kuznetsov, O. Stefanovych, K. Kuznetsova, M. Pastukhov, and D. Prokopovych-Tkachenko, “Method of 3D-steganography,” *CS&CS E-journal*, no. 4, Art. no. 4, 2018.

[10] A. Kuznetsov, O. Stefanovych, Y. Gorbenko, O. Smirnov, V. Krasnobaev, and K. Kuznetsova, “Information Hiding Using 3D-Printing Technology,” in *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, Sep. 2019, vol. 2, pp. 701–706. doi: 10.1109/IDAACS.2019.8924352.

[11] A. A. Kuznetsov, O. O. Stefanovych, D. I. Prokopovych-Tkachenko, and K. O. Kuznetsova, “3D STEGANOGRAPHY INFORMATION HIDING,” *TRE*, vol. 78, no. 12, Art. no. 12, 2019, doi: 10.1615/TelecomRadEng.v78.i12.30.

[12] K. Szczypiorski, “HICCUPS: Hidden communication system for corrupted networks,” Jan. 2003.

[13] W. Mazurczyk and K. Szczypiorski, “Steganography of VoIP Streams,” *arXiv:0805.2938 [cs]*, Aug. 2008, Accessed: Sep. 30, 2021. [Online]. Available: http://arxiv.org/abs/0805.2938

[14] W. Fraczek, W. Mazurczyk, and K. Szczypiorski, “Stream Control Transmission Protocol Steganography,” in *2010 International Conference on Multimedia Information Networking and Security*, Nov. 2010, pp. 829–834. doi: 10.1109/MINES.2010.176.

[15] W. Fraczek, W. Mazurczyk, and K. Szczypiorski, “How Hidden Can be Even More Hidden?,” in *2011 Third International Conference on Multimedia Information Networking and Security*, Nov. 2011, pp. 581–585. doi: 10.1109/MINES.2011.97.

[16] A. Yahya, “Steganography Techniques,” in *Steganography Techniques for Digital Images*, A. Yahya, Ed. Cham: Springer International Publishing, 2019, pp. 9–42. doi: 10.1007/978-3-319-78597-4\_2.

[17] J. Qin, Y. Luo, X. Xiang, Y. Tan, and H. Huang, “Coverless Image Steganography: A Survey,” *IEEE Access*, vol. 7, pp. 171372–171394, 2019, doi: 10.1109/ACCESS.2019.2955452.

[18] N. Menon and Vaithiyanathan, “A survey on image steganography,” in *2017 International Conference on Technological Advancements in Power and Energy ( TAP Energy)*, Dec. 2017, pp. 1–5. doi: 10.1109/TAPENERGY.2017.8397274.

[19] K. Akbay, M. Z. Konyar, S. İlkın, and A. Sondaş, “Data hiding using shuffle algorithm and LSB method,” in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, 2018, pp. 1–4. doi: 10.1109/SIU.2018.8404172.

[20] H. A. Nassrullah, W. N. Flayyih, and M. A. Nasrullah, “Enhancement of LSB Audio Steganography Based on Carrier and Message Characteristics,” *Journal of Information Hiding and Multimedia Signal Processing*, vol. 11, no. 3, pp. 126–137, 2020.

[21] M. C. Lee and C. Y. Lau, “Three Orders Mixture Algorithm of Audio Steganography Combining Cryptography,” *Journal of Information Hiding and Multimedia Signal Processing*, vol. 9, no. 4, pp. 959–969, 2018.

[22] T. T. Manunggal and D. Arifianto, “Data protection using interaural quantified-phase steganography on stereo audio signals,” in *2016 IEEE Region 10 Conference (TENCON)*, 2016, pp. 3817–3821. doi: 10.1109/TENCON.2016.7848778.

[23] R. R. Ginanjar, S. Bhardwaj, D.-S. Kim, and J.-M. Lee, “Rounding Modulation for Transparent Data-Hiding Scheme in High-Quality Audio File,” in *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 2019, pp. 1–2. doi: 10.1109/ICCE.2019.8662015.

[24] R. Kh. Latypov and E. L. Stolov, “Ternary Echo Hiding in Audio Files,” in *2020 28th Telecommunications Forum (℡FOR)*, 2020, pp. 1–4. doi: 10.1109/℡FOR51502.2020.9306575.

[25] S. Wang, W. Yuan, and M. Unoki, “Multi-Subspace Echo Hiding Based on Time-Frequency Similarities of Audio Signals,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2349–2363, 2020, doi: 10.1109/TASLP.2020.3013785.

[26] B. Tabara, J. Wojtuń, and Z. Piotrowski, “Data hiding method in speech using echo embedding and voicing correction,” in *2017 Signal Processing Symposium (SPSympo)*, Sep. 2017, pp. 1–6. doi: 10.1109/SPS.2017.8053697.

[27] O. Kinzeryavyy, I. Kinzeriava, A. Olenyuk, and K. Sulkowsky, “Steganographic Method of Bitwise Information Hiding in Point-Defined Curves of Vector Images,” in *Advances in Computer Science for Engineering and Education*, Cham, 2019, pp. 478–486. doi: 10.1007/978-3-319-91008-6\_48.

[28] F. Peng, Z.-X. Lin, X. Zhang, and M. Long, “Reversible Data Hiding in Encrypted 2D Vector Graphics Based on Reversible Mapping Model for Real Numbers,” *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 9, pp. 2400–2411, Sep. 2019, doi: 10.1109/TIFS.2019.2899520.

[29] A. Kuznetsov and A. Kononchenko, “Data Hiding in Vector Images,” in *2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)*, Sep. 2021, vol. 1, pp. 171–176. doi: 10.1109/CSIT52700.2021.9648652.

[30] A. Kuznetsov and A. Kononchenko, “Software Implementation of Data Hiding in Vector Images,” in *2021 IEEE 3rd Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, Aug. 2021, pp. 572–577. doi: 10.1109/UKRCON53503.2021.9575240.

[31] A. Kuznetsov and A. Kononchenko, “Message Concealing in Vector Images,” in *2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, Sep. 2021, vol. 1, pp. 465–470. doi: 10.1109/IDAACS53288.2021.9660899.

[32] O. I. Sheluhin, D. I. Magomedova, and S. Y. Rybakov, “Data Hiding and Transmission Over Occupied Audio Channels Using Deterministic Chaos,” in *2021 Systems of Signals Generating and Processing in the Field of on Board Communications*, Mar. 2021, pp. 1–7. doi: 10.1109/IEEECONF51389.2021.9415979.

[33] A. Kuznetsov, A. Onikiychuk, O. Peshkova, T. Gancarczyk, K. Warwas, and R. Ziubina, “Direct Spread Spectrum Technology for Data Hiding in Audio,” *Sensors*, vol. 22, no. 9, Art. no. 9, Jan. 2022, doi: 10.3390/s22093115.

[34] A. A. Krishnan, C. S. Chandran, S. Kamal, and M. H. Supriya, “Spread spectrum based encrypted audio steganographic system with improved security,” in *2017 International Conference on Circuits, Controls, and Communications (CCUBE)*, 2017, pp. 109–114. doi: 10.1109/CCUBE.2017.8394128.

[35] A. Kuznetsov, O. Smirnov, D. Kovalchuk, and T. Kuznetsova, “New technique for data hiding in cover images using adaptively generated pseudorandom sequences,” in *Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019), Kyiv, Ukraine, November 30, 2019.*, 2020, vol. 2654, pp. 1–14. [Online]. Available: http://ceur-ws.org/Vol-2654/paper1.pdf

[36] G. Hua, “Over-Complete-Dictionary-Based Improved Spread Spectrum Watermarking Security,” *IEEE Signal Processing Letters*, vol. 27, pp. 770–774, 2020, doi: 10.1109/LSP.2020.2986154.

[37] R. Böhme, *Advanced Statistical Steganalysis*, vol. 0. Berlin, Heidelberg: Springer, 2010. doi: 10.1007/978-3-642-14313-7.

[38] X. Luo, F. Liu, C. Yang, and D. Wang, “Image universal steganalysis based on best wavelet packet decomposition,” *Sci. China Inf. Sci.*, vol. 53, no. 3, pp. 634–647, Mar. 2010, doi: 10.1007/s11432-010-0044-6.

[39] A. Rani, M. Kumar, and P. Goel, “Image Modelling: A Feature Detection Approach for Steganalysis,” in *Advances in Computing and Data Sciences*, Singapore, 2017, pp. 140–148. doi: 10.1007/978-981-10-5427-3\_15.

[40] X. Hou and T. Zhang, “Universal blind steganalysis via reference points-based local outlier factor,” in *2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN)*, 2017, pp. 1287–1291. doi: 10.1109/ICCSN.2017.8230317.

[41] X. Quan, “JPEG Steganalysis Based on Local Dimension Estimation,” in *2021 6th International Conference on Image, Vision and Computing (ICIVC)*, Jul. 2021, pp. 306–311. doi: 10.1109/ICIVC52351.2021.9527021.

[42] B. Singh, “Deep Residual Network for Steganalysis of Digital Images (SRNet model) Pytorch Implementation.” Aug. 10, 2022. Accessed: Aug. 18, 2022. [Online]. Available: https://github.com/brijeshiitg/Pytorch-implementation-of-SRNet

[43] D. G. González, “SRNet Deep Learning steganalyzer TensorFlow implementation.” Jul. 03, 2022. Accessed: Aug. 18, 2022. [Online]. Available: https://github.com/davidggz/SRNet-Tensorflow-Implementation

[44] N. Luhanko, “SRNET.” Sep. 04, 2022. Accessed: Sep. 04, 2022. [Online]. Available: https://github.com/nick-leman/SRNET

[45] M. Boroumand, M. Chen, and J. Fridrich, “Deep Residual Network for Steganalysis of Digital Images,” *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1181–1193, 2019, doi: 10.1109/TIFS.2018.2871749.

[46] P. Bas and T. Furon, “BOWS-2 Web page.” http://bows2.ec-lille.fr/ (accessed Aug. 18, 2022).

[47] P. Bas, T. Filler, and T. Pevný, “”Break Our Steganographic System”: The Ins and Outs of Organizing BOSS,” in *Information Hiding*, Berlin, Heidelberg, 2011, pp. 59–70. doi: 10.1007/978-3-642-24178-9\_5.

[48] “Starter: bossbase 14c7b9a4-0.” https://kaggle.com/code/kerneler/starter-bossbase-14c7b9a4-0 (accessed Aug. 18, 2022).

[49] “BOSS Web page.” http://agents.fel.cvut.cz/boss/index.php?mode=VIEW&tmpl=about (accessed Aug. 18, 2022).

[50] T. Denemark, J. Fridrich, and V. Holub, *Further study on the security of S-UNIWARD*, vol. 9028. 2014. doi: 10.1117/12.2044803.

[51] “Steganographic Algorithms.” http://dde.binghamton.edu/download/stego\_algorithms/ (accessed Sep. 03, 2022).

[52] N. Luhanko, “S-Uniward 0.3 0.4 0.5.” https://www.kaggle.com/datasets/nicolasluhanko/suniward-03-04-05 (accessed Sep. 20, 2022).