CS-Net: A Deep Learning-Based Analysis for the Cryptography and Steganography

Duk Young Kim, Kyoungbae Jang, Hyunji Kim, Yujin Oh, Seyoung Yoon and Hwajeong Seo

Hansung University, Seoul (02876), South Korea

dudejrdl123@gmail.com, starj1023@gmail.com, khj1594012@gmail.com, sebbang99@gmail.com, hwajeong84@gmail.com

Abstract. Combining cryptography and steganography can provide more reliable and robust security by first concealing the existence of a secret message and then transforming it into ciphertext. While extensive research has been conducted on deep learning for steganalysis, no study to date has focused on a deep learning-based analysis of the combined use of cryptography and steganography.

In this work, we introduce CS-Net, a deep learning-based approach to analyze the combined use of cryptography and steganography. To conceal the data, we used the well-known LSB steganographic algorithm. The concealed data is then encrypted using SPECK (-32/64), a lightweight encryption scheme developed by the NSA.

We propose a novel approach that rotates the stego image containing concealed data to learn robust patterns in various directions of encryption. Further, we apply a state-of-the-art preprocessing technique used in steganalysis and demonstrate that it is still effective even when combined with cryptography.

Indeed, even when the concealed data through steganography is encrypted (using SPECK), CN-Net distinguishes cover and stego images from the ciphertext. Our work generally achieves valid accuracy (greater than 0.5) for steganography/cryptography analysis.

Keywords: Deep learning · Steganalysis · Cryptography · SPECK.

1 Introduction

The exponential growth of digital data, driven by rapid advancements in media and communication technologies, has made the need for robust security measures more critical than ever. Steganography and cryptography are two methods that can be used to securely protect data during communication. Steganography conceals secret messages within digital media (such as images, audio, or video), making it difficult for others to detect their presence without altering the media's structure. In contrast, cryptography prevents unauthorized access by converting the message into ciphertext.

Steganography shows that when an encoding system is exposed, its security relies on the confidentiality of that system, which can be compromised [1,2].

Meanwhile, cryptography encrypts the secret message, making it incomprehensible to anyone except the sender and receiver. Cryptographic algorithms are designed based on mathematical problems to address various aspects of information security, such as data integrity, identification, and authentication.

Traditional steganography [3,4,5] techniques have primarily relied on embedding algorithms without encrypting the secret data, making their security heavily dependent on the performance of these algorithms. This reliance poses a significant limitation, especially as steganography and steganalysis methods improve, making simple data hiding more susceptible to detection.

To improve the detection accuracy of stego images, a deep learning-based steganalysis technique utilizing a Convolutional Neural Network (CNN) has been proposed [6,7,8,9]. The CNN model is designed to automatically learn high-dimensional features from the data, providing a significant improvement over traditional statistical analysis methods.

To address these limitations, various approaches that combine cryptographic algorithms with steganography to enhance the security of hidden data have been proposed [10,11,12,13,14]. Similarly, extensive research on deep learning-based cryptanalysis has been conducted [15,16,17,18,19], inspired by Gohr's neural distinguisher for cryptographic algorithms [20].

To the best of our knowledge, no existing analysis framework combines both steganography and cryptography, despite numerous proposals to merge these techniques to enhance confidentiality. In this paper, we propose a framework, CS-Net, that integrates both steganalysis and cryptanalysis. Table 1 provides a brief comparison with related works in steganalysis. For the steganographic method, we use the Least Significant Bit (LSB) embedding algorithm, one of the simplest and most widely used techniques for hiding information. Additionally, the stego images generated using the LSB algorithm are encrypted with SPECK, a lightweight block cipher developed by the National Security Agency (NSA). The details of the related works [6,7,8,9] are discussed in Section 2.1. We believe that the findings of this work can lay the foundation for developing more secure and efficient steganography systems in the future.

Table 1: Comparison with related works for steganalysis.

Framework	Steganography	Cryptography	Descriptions
Xu-Net [6]	✓ (WOW, S-UNIWARD)	✗	HPF and Tanh activation
Ye-Net [7]			SRM filter and TLU activation
Yedroudj-Net [8]			Combination of Xu-Net and Ye-Net
GBRAS-Net [9]			High accuracy and fewer parameters
CS-Net (Ours)	✓ (LSB)	✓ (SPECK)	Rotation and preprocessing strategy

2 Prerequisites

2.1 Steganography

Steganography is a method to conceal secret data within digital media such as images, audio, and video, which is crucial for secure communication. Among the various techniques, Least Significant Bit (LSB) steganography is one of the simplest and most widely used methods for hiding information. Note that we adopt LSB steganography to conceal the secret data in this paper[1].

LSB steganography In digital media, data (such as images or audio) is typically represented as a series of bits (binary digits). For example, an 8-bit grayscale image has pixel values ranging from 0 to 255, represented in binary as 8-bit numbers. LSB steganography involves modifying the least significant bits of the cover file to embed secret data. For instance, if a pixel value is represented in binary as `0b11001010`, replacing the least significant bit with `1` changes it to `0b11001011`. This change is minimal and typically unnoticeable to the human eye or ear.

Steganalysis Using Deep Learning Steganalysis aims to detect secret data embedded in digital media through steganography. The primary goal of steganalysis is to determine whether a given piece of media contains any hidden data and, if possible, to extract that data. Traditional steganalysis methods often rely on statistical analysis or handcrafted features to detect concealed data. However, these methods are limited in their ability to adapt to new or more sophisticated steganography techniques.

In this context, deep learning-based steganalysis has emerged as a powerful alternative to traditional approaches. It leverages the ability of deep neural networks to automatically learn and extract complex patterns and features from large datasets (particularly, CNN is frequently adopted for the architecture).

Xu-Net [6] utilized a high-pass filter (HPF) for preprocessing stego images, followed by feature extraction and classification, taking advantage of the Tanh activation function for improved learning when the input approaches zero.

Ye-Net [7] applied a learnable SRM filter with 30 linear kernels in the initial convolutional layer and used the Truncated Linear Unit (TLU) activation function to classify images into cover and stego categories.

In [8], the authors introduced Yedroudj-Net, which combines techniques from Xu-Net and Ye-Net. Yedroudj-Net uses the SRM filter and TLU activation in the initial layer, followed by Xu-Net's ABS, BN, and TLU activations in subsequent layers, achieving higher accuracy than both Xu-Net and Ye-Net.

GBRAS-Net [9] also used 30 SRM filters and adopted the 3-Tanh activation function, which provides a smoother curve and improved results. This model reduces the number of parameters, enhances computational efficiency, and speeds up training while maintaining performance.

[1] Over the past few decades, many methods have been proposed to enhance the stealthiness, capacity, and efficiency of steganography, such as WOW [3], J-UNIWARD [4], and MIPOD [5].

2.2 Cryptography

Concise Description of SPECK SPECK [21] is a series of lightweight block ciphers introduced by the National Security Agency (NSA) in 2013. The SPECK family employs a Feistel-like architecture and includes ten different versions. There are variants that support different key and block sizes (e.g., SPECK-32/64, -48/72, -128/128 etc.) and SPECK-32/64 is adopted for encryption algorithm in our framework. Figure 1 illustrates the round function and key schedule of SPECK, respectively.

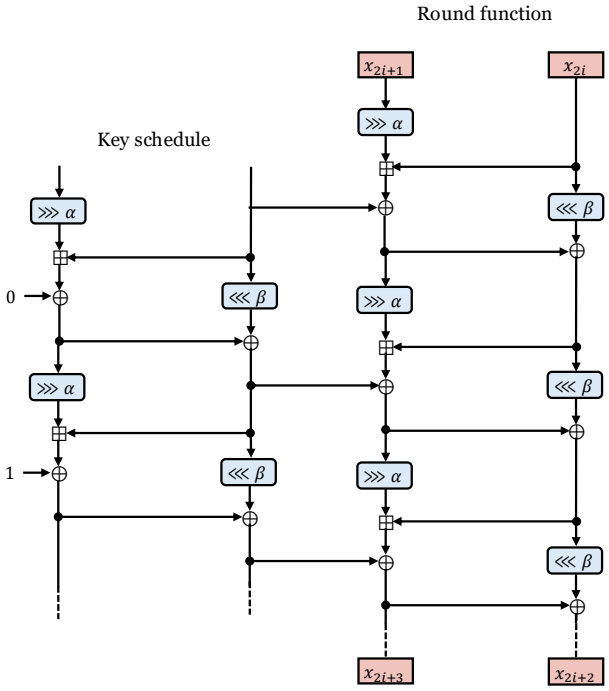

Fig. 1: Schematic of SPECK encryption.

Round Function. The round function of SPECK involves three main operations: modular addition ($\boxplus$), bit-wise rotation ($\ggg \alpha, \lll \beta$), and exclusive-OR ($\oplus$).

$$R_k(x, y) = (((x \ggg \alpha) \boxplus y) \oplus k,\ (y \lll \beta) \oplus ((x \ggg \alpha) \boxplus y) \oplus k) \tag{1}$$

Key Schedule. The round keys for SPECK (T rounds) are generated similarly to the state updates in each round. A key schedule is processed to generate a round key k, which is used in each round. The variable m denotes the number

of words in a key, where m = key size/block size/2. The initial key words are given by $K = (k_0, l_0, \ldots, l_{m-2})$. The following key schedule is performed, and the new key value $K = k_0, k_1, \ldots, k_{T-1}$ is used as the round key.

$$l_{i+m-1} = (k_i \boxplus (l_i \ggg \alpha)) \oplus i, \;\; k_{i+1} = (k_i \lll \beta) \oplus l_{i+m-1} \tag{2}$$

Cryptanalysis Using Deep Learning In Crypto'19 [20], Gohr introduced the first neural distinguisher for a round-reduced version of the SPECK cipher. This neural distinguisher was capable of differentiating cryptographic data from random data up to 7 rounds and extended this capability to 8 rounds using transfer learning. In [22], two neural distinguisher models were proposed, focusing on multi-input differential and single differential analysis, and were applied to the ciphers GIMLI [23], ASCON [24], KNOT [25], and Chaskey [26]. The proposed MLP-based neural distinguisher successfully identified 8-round GIMLI, 3-round ASCON, 10/12-round KNOT (256/512-bit), and 4-round Chaskey. Moreover, several studies on deep learning-based cryptanalysis of various cryptographic algorithms have been conducted, including works in [15,16,17,18,19].

3 CS-Net

We propose CS-Net, a deep learning-based framework for the combined analysis of cryptography and steganography, founded on Remark 1.

Remark 1 *Cryptography can provide confidentiality for steganography by protecting hidden information, but steganography does not enhance the security of cryptography itself.*

To the best of our knowledge, existing steganalysis methods have not considered cryptographic algorithms that ensure confidentiality. In contrast, we introduce a combined analysis for confidentiality-guaranteed steganography with a cryptographic algorithm.

Figure 2 illustrates the overall process of CS-Net. CS-Net integrates various techniques; dataset composition, preprocessing methods, and neural network architecture; to determine whether an image in an unlabeled dataset (i.e., ciphertext) is a cover image or a stego image.

3.1 Dataset: Steganography Combined with Cryptography

In this section, we describe the process of generating a dataset using steganographic and cryptographic algorithms. We convert both cover and stego images into 32×32 grayscale images, considering our experimental environment. The cover images are labeled as 0, while the stego images are labeled as 1. Figure 3 illustrates the entire dataset generation process.

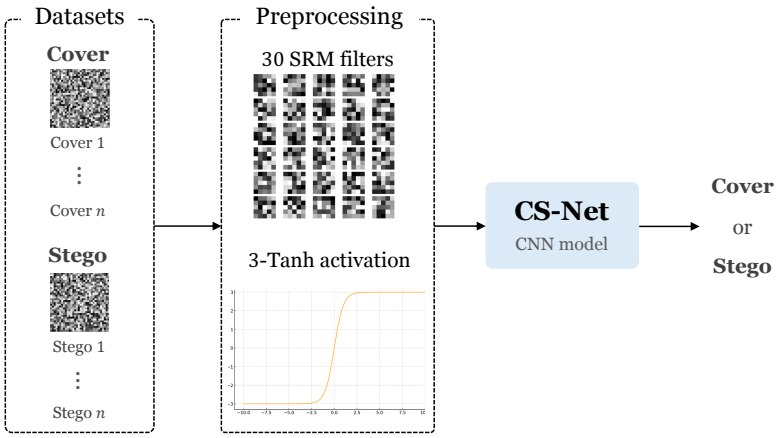

Fig. 2: The overall process of CS-Net.

Generating Stego Images As described in Section 2.1, we generate stego images from cover images[2] using the LSB embedding method outlined in Algorithm 1. The secret data are embedded into 28,000 images, and using our rotation strategy, the data set is augmented to $112,000$ $(= 28,000 \times 4)$ images. The details of this strategy/augmentation will be described in Section 3.2.

Algorithm 1 LSB embedding algorithm

Input: Cover image I of size 32×32, Secret message M (binary), Threshold T
Output: Stego image I'

Convert secret message S into a binary sequence
Initialize $index \leftarrow 1$
for each pixel p in I **do**
    if $index >$ length(S) **then**
        break
    end if
    if Vaule of $p > T$ **then**
        Replace the LSB of p with $S[index]$
        $index \leftarrow index + 1$
    end if
end for
return Stego image I'

For LSB embedding, pixels are selected to conceal the secret data based on their binary values, which is related to the capacity of the secret data. If a

[2] The image data used in this work is sourced from the ALASKA2 (image steganalysis), available at https://www.kaggle.com/c/alaska2-image-steganalysis.

threshold is set at '0b11110000', the secret data is hidden by replacing the LSB of pixels with values greater than this threshold.

For example, let the secret data be '0b101', and consider pixels with the following values, all greater than the threshold: '0b11111100', '0b11110011', and '0b11110101'. The secret data are concealed in these pixels by replacing their LSBs as follows: '0b11111101', '0b11110010', '0b11110101'.

For steganalysis, the embedding rate of the stego image is crucial. Similarly, in our combined analysis of steganography and cryptography, the embedding rate remains a critical factor (see Table 3). In our frame work, the embedding rate, E_r, of the stego image is given by:

$$E_r = \frac{\textit{the number of pixels with values} > \textit{threshold}}{1024\ (32 \times 32)} \tag{3}$$

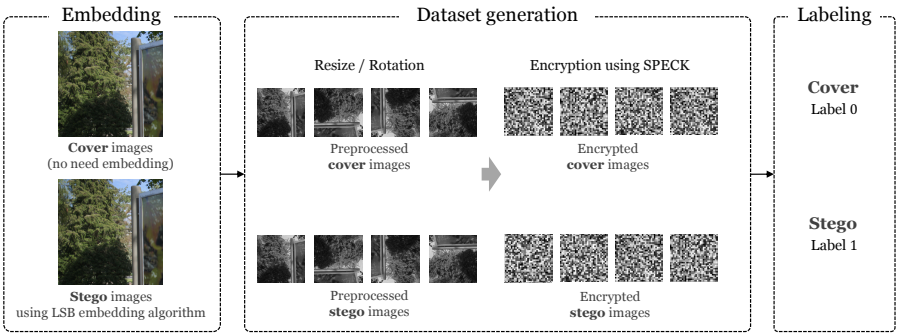

Fig. 3: The process of data set generation.

Encryption Using SPECK We use the lightweight encryption algorithm SPECK-32/64 to encrypt the generated stego images. SPECK-32/64 supports a 64-bit key to encrypt a 32-bit block. For encryption, all cover images are resized to 32×32 pixels to fit the 32-bit block size. Note that this also affects the input dimensions of the deep learning model. Thus, 32 ciphertexts are generated for each stego image (of size 32×32).

We train and test the model on both the full and reduced encryption rounds of SPECK-32/64 (resp. 22 and 11 rounds). In cryptanalysis, reducing the number of rounds in cryptographic algorithms is often adopted to adjust robustness [20,22].

3.2 Novel Rotation Strategy

We propose a novel rotation strategy for stego images that considers various encryption directions. For a stego image, the directions of left, right, down, and up can be selected to encrypt the 32-bit blocks within the image. By considering

these different directions, we augment the training dataset by rotating each stego image by 90, 180, and 270 degrees. Figure 4 shows the stego images (including encryption) with the rotation strategy.

In addition, we can extract residual information from multiple directions by using rotated stego images. This rotation technique provides a more diverse range of inputs, helping the model learn more robust features to distinguish between them. Thanks to this approach, our framework effectively learns patterns corresponding to various directions of encryption. Further, this strategy increases the amount of training data (i.e., data augmentation).

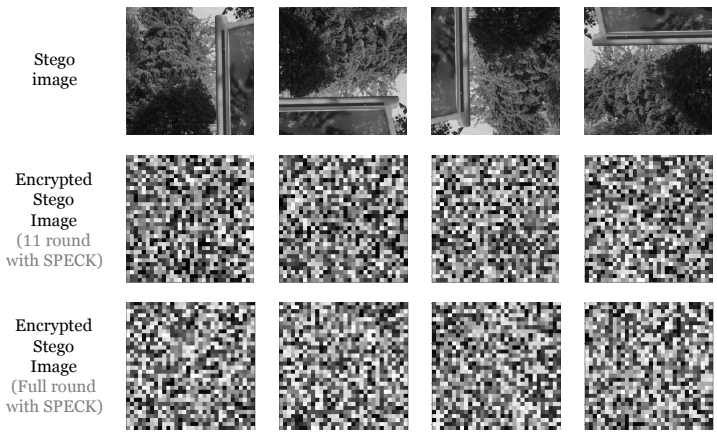

Fig. 4: Stego and encrypted 32×32 images using the rotation strategy.

3.3 Preprocessing on the Encrypted Stego Images

Interestingly, we found that preprocessing techniques for steganalysis are still effective for combined analysis of steganography and cryptanalysis, even when applied to encrypted stego images. By applying the preprocessing technique, the accuracy was improved by $2 \sim 8$ percent.

We utilized the 30 SRM filter, a preprocessing technique that has been commonly used since Ye-Net [7]. In our work, preprocessing is performed by applying 30 SRM filters as initial parameters of the untrainable convolutional layer [3]. There are 30 high-pass filters (HPF) in 30 SRMs. HPF is used to remove useless noise from images while retaining information in complex areas [27]. This prevents the neural network for steganalysis from being sensitive to unnecessary information (noise) in stego images by over-extracting unnecessary features. Additionally, we apply 3-Tanh, an activation function with a smoother curve (the range is -3 to 3), which is commonly used in steganalysis.

[3] For this, we use the $'trainable = false'$ option of the tensorflow convolution layer.

$$3\tanh(x) = 3 \times \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

3.4 Network Architecture

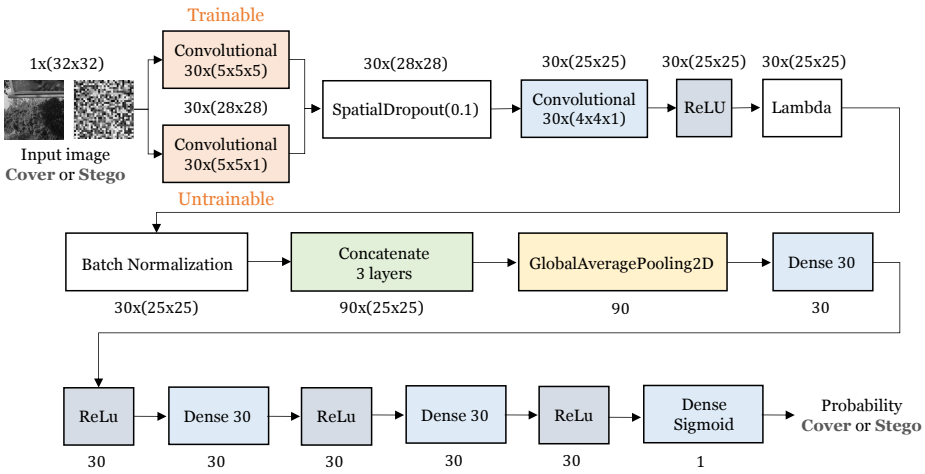

Fig. 5: The detail architecture of CS-Net.

Figure 5 and Table 2 illustrate the model architecture of CS-Net. This model includes multiple convolutional layers, activation functions, normalization techniques, dropout mechanisms, and dense layers. First, it takes grayscale images of 32×32 size as input data. The network begins with two initial convolutional layers: the first employs untrainable weights, while the second utilizes trainable weights. This technique allows the model to benefit from both predefined filters and adaptive learning, thereby enhancing the initial stage of feature extraction. In particular, a large number of zero values can impede feature extraction, and square filters are commonly used in image processing. Based on the above considerations, we have decided to set the kernel size to 4×4 (except for the initial layer). This size is optimal for training without padding (e.g., zero or same padding) on our input image of 32×32 pixels.

To mitigate overfitting, the network applies a dropout layer that randomly deactivates a portion of neurons during training, and residual network and batch normalization is also applied. The residual network is effective in reusing features and preventing overfitting by propagating the extracted features to multiple layers instead of just one layer. For similar reasons, an ensemble structure is used through layer concatenation. Our network is also designed with a concatenated residual network structure for effective learning with fewer parameters.

Table 2: Hyperparameters of CS-Net.

Hyperparameters	Descriptions
Epochs	10
Loss function	binary cross-entropy
Optimizer	SGD (learning rate=0.005, momentum=0.95)
Activation function	3-Tanh (Initial), ReLu (Hidden), Sigmoid (Output)
Batch size	32
Parameters	20701 (trainable), 840 (untrainable)
Initial parameters	30 SRM filters and bias

In addition, we chose the SGD [28] optimizer over Adam because, while Adam tends to converge faster, SGD often provides better generalization in the final model. For our work, we prioritized the model's ability to generalize well on unseen data rather than achieving faster convergence. Furthermore, SGD's simpler update rule can help avoid overfitting, especially when combined with techniques like momentum, making it a suitable choice for our problem, where stability and generalization are key concerns. Additionally, we use average pooling to preserve as much information from the image data as possible. It is important to note that, although max pooling is faster, it can result in the loss of valuable features in the ciphertext because some information within the window is discarded and only the maximum value is retained.

The final layer employs a sigmoid activation function, producing a probability that indicates whether the input is classified as a cover or stego image. Ultimately, our network ensures stable generalization performance by integrating techniques such as dropout, regularization, residual structure, and initial preprocessing.

4 Evaluation

4.1 Experiment Environment

This experiment is performed on Google Colab, a cloud computing platform supporting Ubuntu 20.04.5 LTS and Tesla T4 (GPU) 12GB RAM. As a programming environment, TensorFlow 2.12.0 and Python 3.9.16 are used.

4.2 Results

Table 3 shows the results of CS-Net based on the threshold T, embedding rate E_r, and the number of rounds in SPECK-32/64. The accuracy depends on E_r, which is determined by T according to Equation 3. For $T = 210, 220$ (resp. $E_r = 0.185, 0.175$), the accuracy is not valid (around 0.5) for the full rounds of SPECK-32/64. The accuracy is influenced by the embedding rate E_r because a

Table 3: Result of CS-Net.

T	E_r	Full round (22)			Reduced round (11)		
		Train accuracy	Test accuracy	Reliability	Train accuracy	Test accuracy	Reliability
140	0.255	0.6408	0.6364	0.1364	0.7657	0.7512	0.2512
150	0.245	0.6301	0.6298	0.1298	0.7386	0.7201	0.2201
160	0.235	0.5930	0.5872	0.0872	0.7223	0.7192	0.2192
170	0.225	0.5721	0.5692	0.0692	0.7179	0.7028	0.2028
180	0.215	0.5649	0.5669	0.0669	0.6781	0.6702	0.1702
190	0.205	0.5489	0.5487	0.0487	0.6647	0.6531	0.1531
200	0.195	0.5271	0.5295	0.0295	0.6325	0.6208	0.1208
210	0.185	0.5017	0.5016	0.0016	0.6024	0.5974	0.0974
220	0.175	0.5015	0.5012	0.0012	0.5825	0.5734	0.0734

higher E_r makes it easier to detect the differences caused by embedding within the ciphertext.

However, for the round-reduced (11 rounds) SPECK-32/64, the accuracy increases due to the weakened encryption strength of SPECK, as more differences can be detected in the less complex ciphertexts (a generally expected result in cryptanalysis). For the reduced rounds, a valid accuracy (greater than 0.5) is achieved even for $T = 210, 220$ (corresponding to $E_r = 0.185, 0.175$, respectively).

Reliability is calculated by subtracting the random probability of the classification task from the accuracy. In binary classification tasks like ours, reliability is determined by subtracting 0.5 from the accuracy, with 0.5 representing the highest possible random probability. A higher reliability indicates a stronger and more robust neural network. Conversely, a network with a reliability of 0 has not learned effectively and only classifies correctly by chance. Therefore, models with a reliability of at least 0.005 (the reliability threshold) are considered valid for distinguishing models.

In terms of reliability, CS-Net for the full round of SPECK-32/64 achieves valid reliability for encrypted stego images when E_r is between 0.195 and 0.255. However, when E_r falls below 0.185, the reliability drops below the threshold. This result indicates that as E_r decreases, it becomes more challenging to distinguish between stego and cover images.

For the round-reduced SPECK-32/64, our approach achieves significantly higher accuracy and reliability than for the full round. In this case, we can successfully classify even when E_r is 0.175. This demonstrates that distinguishing embedding methods for stego data, where confidentiality is ensured by encryption, is much more difficult than for unencrypted cases.

5 Concluding Remarks (and Note on Limitation)

Cryptography can enhance steganography by providing the security property of confidentiality. However, steganography cannot add security properties to cryp-

tography itself. This suggests that if a trapdoor exists in the cryptography used within steganography, then steganalysis remains effective.

In this paper, we presented a deep learning-based steganalysis combined with cryptanalysis to counter it. In our approach, we use the same key for encryption with SPECK in ECB mode (considered a trapdoor in our work), and CS-Net successfully distinguishes between cover and stego images from ciphertexts. This demonstrates the importance of Remark 1 (Section 3) and also highlights a limitation of our work. If there is no trapdoor in the cryptographic implementation (i.e., strict use), steganalysis may become ineffective.

However, advanced deep learning-based cryptanalysis techniques (e.g., differential cryptanalysis) are being developed. With these powerful cryptanalysis methods, we can anticipate more effective analysis when combining steganography and cryptography.

For our future work, we plan to incorporate recent cryptanalysis techniques and replace the LSB algorithm with more advanced steganographic methods, such as WOW and J-UNIWARD.

References

1. R. J. Anderson and F. A. Petitcolas, "On the limits of steganography," *IEEE Journal on selected areas in communications*, vol. 16, no. 4, pp. 474–481, 1998. 1
2. J. Fridrich and M. Goljan, "Practical steganalysis of digital images: state of the art," *security and Watermarking of Multimedia Contents IV*, vol. 4675, pp. 1–13, 2002. 1
3. V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," in *2012 IEEE International workshop on information forensics and security (WIFS)*, pp. 234–239, IEEE, 2012. 2, 3
4. V. Holub and J. Fridrich, "Digital image steganography using universal distortion," in *Proceedings of the first ACM workshop on Information hiding and multimedia security*, pp. 59–68, 2013. 2, 3
5. V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 221–234, 2015. 2, 3
6. G. Xu, H.-Z. Wu, and Y.-Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, 2016. 2, 3
7. J. Ye, J. Ni, and Y. Yi, "Deep learning hierarchical representations for image steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2545–2557, 2017. 2, 3, 8
8. M. Yedroudj, F. Comby, and M. Chaumont, "Yedroudj-net: An efficient cnn for spatial steganalysis," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 2092–2096, IEEE, 2018. 2, 3
9. T.-S. Reinel, A.-A. H. Brayan, B.-O. M. Alejandro, M.-R. Alejandro, A.-G. Daniel, A.-G. J. Alejandro, B.-J. A. Buenaventura, O.-A. Simon, I. Gustavo, and R.-P. Raul, "Gbras-net: a convolutional neural network architecture for spatial image steganalysis," *IEEE Access*, vol. 9, pp. 14340–14350, 2021. 2, 3
10. H. Sharma, K. K. Sharma, and S. Chauhan, "Steganography techniques using cryptography-a review paper," *International Journal of Recent Research Aspects, Special Issue: Engineering Research Aspects ISSN*, pp. 2349–7688, 2015. 2

11. A. Dhamija and V. Dhaka, "A novel cryptographic and steganographic approach for secure cloud data migration," in *2015 international conference on green computing and internet of things (ICGCIoT)*, pp. 346–351, IEEE, 2015. 2
12. P. Vijayakumar, V. Vijayalakshmi, and G. Zayaraz, "An improved level of security for dna steganography using hyperelliptic curve cryptography," *Wireless Personal Communications*, vol. 89, pp. 1221–1242, 2016. 2
13. B. Karthikeyan, A. C. Kosaraju, and S. Gupta, "Enhanced security in steganography using encryption and quick response code," in *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pp. 2308–2312, IEEE, 2016. 2
14. B. Pillai, M. Mounika, P. J. Rao, and P. Sriram, "Image steganography method using k-means clustering and encryption techniques," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1206–1211, IEEE, 2016. 2
15. A. Benamira, D. Gerault, T. Peyrin, and Q. Q. Tan, "A deeper look at machine learning-based cryptanalysis," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 805–835, Springer, 2021. 2, 5
16. Y. Chen and H. Yu, "A new neural distinguisher model considering derived features from multiple ciphertext pairs.," *IACR Cryptol. ePrint Arch.*, vol. 2021, p. 310, 2021. 2, 5
17. Z. Hou, J. Ren, and S. Chen, "Cryptanalysis of round-reduced Simon32 based on deep learning," *Cryptology ePrint Archive*, 2021. 2, 5
18. T. Yadav and M. Kumar, "Differential-ml distinguisher: Machine learning based generic extension for differential cryptanalysis," in *International Conference on Cryptology and Information Security in Latin America*, pp. 191–212, Springer, 2021. 2, 5
19. A. Baksi, J. Breier, V. A. Dasu, X. Hou, H. Kim, and H. Seo, "New results on machine learning-based distinguishers," *IEEE Access*, 2023. 2, 5
20. A. Gohr, "Improving attacks on round-reduced speck32/64 using deep learning," in *Annual International Cryptology Conference*, pp. 150–179, Springer, 2019. 2, 5, 7
21. R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK families of lightweight block ciphers." Cryptology ePrint Archive, Report 2013/404, 2013. https://eprint.iacr.org/2013/404. 4
22. A. Baksi, "Machine learning-assisted differential distinguishers for lightweight ciphers," in *Classical and Physical Security of Symmetric Key Cryptographic Algorithms*, pp. 141–162, Springer, 2022. 5, 7
23. D. J. Bernstein, S. Kölbl, S. Lucks, P. M. C. Massolino, F. Mendel, K. Nawaz, T. Schneider, P. Schwabe, F.-X. Standaert, Y. Todo, *et al.*, "Gimli: a cross-platform permutation," in *Cryptographic Hardware and Embedded Systems–CHES 2017: 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, pp. 299–320, Springer, 2017. 5
24. C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1. 2: Lightweight authenticated encryption and hashing," *Journal of Cryptology*, vol. 34, pp. 1–42, 2021. 5
25. W. Zhang, T. Ding, B. Yang, Z. Bao, Z. Xiang, F. Ji, and X. Zhao, "Knot: algorithm specifications and supporting document," *Submission to NIST lightweight cryptography project*, 2019. 5
26. N. Mouha, B. Mennink, A. Van Herrewege, D. Watanabe, B. Preneel, and I. Verbauwhede, "Chaskey: an efficient mac algorithm for 32-bit microcontrollers," in

Selected Areas in Cryptography–SAC 2014: 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers 21, pp. 306–323, Springer, 2014. 5

27. X. Duan, C. Zhang, Y. Ma, and S. Liu, "Preprocessing enhancement method for spatial domain steganalysis," *Mathematics*, vol. 10, no. 21, p. 3936, 2022. 8
28. S.-i. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993. 10