Eksplorium *p*-ISSN 0854-1418
Volume 46 No. 1, May 2025: 1192–1206 *e*-ISSN 2503-426X

A Deep Learning Approach to Image Steganography and Steganalysis

Ashish Tripathi[1 (0000-0002-7570-4067)], Anand Bhushan Pandey[2(0000-0003-3117-361X)], Suveg Moudgil[1(0009-0008-5584-416X)], Abha Kiran Rajpoot[1(0000-0002-0643-3646)], Brijesh Kumar Gupta[3], Vineeta Khemchandani[1(0000-0002-7934-8493)]

[1]SCSE, Galgotias University, Greater Noida, U.P., India

[2]JIMS Engineering Management Technical Campus, Greater Noida, U.P., India

[3]GL Bajaj Group of Institutions, Mathura, U.P., India

Email: {ashish.mnnit44, pandey.mymails, suvegmodgil1, akrajpoot, brijanan, vineetakh05}@gmail.com

Corresponding Author: Vineeta Khemchandani (vineetakh05@gmail.com)

Article Received: 22 Feb 2025, Revised: 23 April 2025, Accepted: 05 May 2025

Abstract: Steganography has been and continues to be used for the purpose of hiding the fact that two parties are communicating. Aside from an interesting research problem, steganography has a number of nefarious applications like hiding illegal activity, financial frauds, industrial espionage, and communication among members of criminal organisations. In this work, we are presenting a model to hide two full-size colour images within another image of the same size. Convolutional neural networks have been trained together to create the hiding and revealing processes. The proposed model has been trained on the dataset of images randomly drawn from the Tiny-Images database. Other than presenting the practical application of deep learning for steganography and steganalysis, we carefully examine how the result is affected by the change in the use of various activation functions. Generally, in steganography, the secret images are embedded within the least significant bits of the cover image. Unfortunately, this simple technique would not resist any kind of editing on the stego image nor any attack by steganalysis experts. whereas in this work the secret images have been represented across all the available bits of the cover image. The proposed model gives loss function values of 126547.56, 212927.22, and 283369.93 for single, double, and triple image steganography, respectively, which is very impressive in terms of reconstructing the cover image for retrieving the secret images at the receiver end.

Keywords: Steganography, Steganalysis, Deep learning, Convolutional neural network.

1. INTRODUCTION

The key components of information veiling today are cryptography, watermarking, and steganography, each with different objectives to achieve their goals. Cryptography is the study of processing digital data by encrypting or scrambling the data bits with a key in a way that renders the data incomprehensible to anyone who does not have the key to recover or decode it.

Watermarking digital data is the method of integrating data known as a watermark, tag, or label into a multimedia object in a perceptually invisible manner without degrading the object's quality, such that the watermark can be detected to make an assertion about the object [24].

Likewise, Steganography refers to a technique for secretly inserting messages into various forms of cover media, including text, audio, image, and video. The secret information can be a text, image or video. The secret information is hidden in such a way that it not visible to the human eyes. It is hidden within the noisy regions of a larger image.

Novel approaches in the embedding process have been developed to make detection more difficult, but the presence of the hidden message can still be found. Steganalysis is the term for the science and art of determining whether an embedded message exists. The primary objectives of steganalysis, in addition to the detection of embedded messages, include

estimating the length of embedded messages, estimating the locations of hidden data in the stego data, estimating the embedding algorithm, and extracting hidden messages.

A historical context for steganography as well as examples of software tools that employ steganography to hide data and software tools to detect such hidden files have been properly described in [1]. Steganography hides the secret data but does not hide the fact that the two sides are communicating. The steganography process involves embedding a secret message within some transport medium, called the carrier. The secret message is hidden within the carrier to form the medium. In summary (eqn. 1):

$$\text{transport medium} = \text{cover message} + \text{secret message(s)} \quad (1)$$

Steganography differs from cryptography in the following ways as describes below in Table 1.

Table 1. Difference between Steganography and Cryptography

Steganography	**Cryptography**
It is about hiding the ‘existence’ of the secret message.	It is about hiding the 'meaning’ of the secret message.
Concern in steganography is regarding the embedding capacity and detectability of the cover image.	Concern in cryptography is regarding the robustness against being able to be deciphered.
Any type of digital media can act as a carrier or a cover.	Depends upon text as the carrier.
Key is optional in steganography.	Key is necessary in cryptography.

There can be some good commercial applications of steganography, as mentioned below:

1. For e-commerce transactions, the current session ID embedded into the fingerprint image of the user can be used to verify if the transaction is being done by the actual card holder or not.
2. Business establishments can have concerns regarding trade secrets or new product data; steganography can be used to accomplish hidden exchanges of such secret data.
3. Steganography can also be used for invisibly watermarking the data to verify its authenticity.

There are quite many ill-intentioned applications of information hiding through, such as perpetrating and coordinating nefarious activities through concealed data in images posted on

public sites, making the recipient difficult to detect [2]. Despite a lot of misuses, a general use for steganography is to conceal authorship information through digital watermarks without effecting the integrity of the content or image.

A convolutional neural network-based model is developed in the proposed work in order to hide two secret images inside a cover image and then retrieve the hidden images from the cover. The dataset was taken from the "Tiny Images Dataset." The Preparation Network, Hiding Network, and Reveal Network are the three primary divisions of the model. The secret images are initially processed by the Preparation Network, which modifies their size to correspond with the cover image's. The carrier or container picture is created in the Hiding Network by combining the cover image with the Preparation Network's outputs. Ultimately, the receiver uses the Reveal Network to extract the secret images from the container image. This approach guarantees:

1. The secrecy and integrity of multiple secret images and protecting them from unauthorised access.
2. The existence of secret images itself gets hidden, unlike cryptography, which hides only the meaning of the data.
3. Minimum loss in reconstruction of the cover image and the hidden images as secret images at the receiver's end are retrieved by means of steganalysis.

1.1 Overview of the Proposed Work

1. A convolutional neural network-based model for concealing two secret images inside a cover image and subsequently extracting the hidden image from the cover image has been built in the proposed work.
2. The dataset used has been picked from the “Tiny images dataset”.
3. The proposed model consists of three sections (Preparation Network, Hiding Network and Reveal Network).
4. The preparation network increases the dimensions of the secret images to match the size of the cover image.
5. The hidden network section takes as input the outputs of the preparation network and the cover image and then combines the three to create the carrier image or container image.
6. The reveal network is the final section of the model. It is used by the receiver of the container image. The section is used by the receiver to extract the secret data in the form of an image from the container image.

2. LITERATURE REVIEW

While remarkable research works have been done in steganography [2-4], there are still some persisting challenges. One of these challenges is that the hiding process during steganography can disturb the appearance of the transport medium. The amount of disturbance in the transport

medium depends upon two factors: The amount of the data to be hidden which is measured in bits per pixel (bpp). The bigger the information, the greater is bpp and the worse the disturbance in the transport medium [4]. Second, the amount of disturbance depends upon the transport medium itself. Hiding information in the noisy, high frequency filled, regions of an image yields less humanly detectable perturbations than hiding in the flat regions. Work on estimating how much information a transport medium can hide can be found in [5]. The generally used techniques for steganography includes manipulating the least significant bits of the images to place the secret data through simple replacement or through some complex schemes [6, 7]. While many significant works have been done by the use of deep neural networks in steganalysis [8-10], the use of deep neural network in the hiding process itself (steganography) still needs more research work [11-14]. Some other state-of-art-research work done in this area is shown in Table 2.

After reviewing all the frameworks available, the methodologies are primarily categorized into three groups, namely, traditional image steganography methods, CNN-based image steganography methods and GAN-based image steganography methods. A brief of these methods is given below.

Table 2. Literature Review

Author/Reference	**Method Used**	**Application/Work**
Johnson et al. [15]	Substitution method	The information is converted into a binary form. The cover image is scanned to find out the less significant bits in the distorted area.
Gupta et al. [16]	Substitution method	This method works under the assumption that changing a few pixel values would not show much change that is visible
Das et al. [17]	Substitution method	The substitution method is performed very carefully as overloading the cover image may turn to visible changes
Wu et al. [18-19]	CNN-based method	The substitution method is performed very carefully as overloading

		the cover image may turn to visible changes.
X. Duan [20]	CNN-based method	General Adversarial Networks are a type of deep CNNs introduced by Goodfellow et al. in 2014
Yuan et al. [22]	GAN-based method	An end-to-end image steganographic scheme based on generative adversarial networks (GAN) with adversarial attack and pixel-wise deep fusion has been proposed.

2.1 Traditional Steganography Methods

Least Significant Bits (LSB) substitution method is used to do image steganography. This method works under the assumption that changing a few pixel values would not show much change that is visible. The information is converted into a binary form. The cover image is scanned to find out the less significant bits in the distorted area [15] and [16]. The substitution method is performed very carefully as overloading the cover image may turn to visible changes. [17]. The same method is used to hide secret inside videos as well.

2.2 CNN-Based Steganography Methods

This model is heavily inspired from the encoder-decoder architecture. Cover picture and the secret image are fed into the encoder to create the stego image, which is then fed into the decoder to create the secret image as the output. The size of the cover picture and the secret image must exactly match each other's dimensions in order for every pixel of the secret image to be evenly dispersed throughout the cover image. Wu et al. [18] and [19] have proposed an encoder-decoder architecture. U-Net based encoder-decoder architecture is used for hiding and a CNN with six layers for extraction [20].

2.3 GAN-Based Steganography Methods

Generative Adversarial Networks are a type of deep CNNs, Creswell et al. [21]. A GAN uses the game theory for training a generative model with an adversarial process for image generation tasks. Generator and discriminator network work against each other to produce a perfect image in the GAN architecture. Many variations on GAN have been proposed ever since. The generator model receives the data, and the result is a reasonably accurate

representation of the input image. The discriminator networks categorise generated images as either True or Fake.

3. MATERIAL AND METHODS

3.1 Dataset

1. The dataset used has been picked from "Tiny images dataset" [23].
2. This dataset consists of 4000 images.
3. Each image is of 64x64x3 dimension.
4. We make pairs of secret images and cover images.
5. For single image steganography, we get 2000 such pairs of secret and cover images after pairing.
6. For double image steganography, we get about 1322 such pairs of secret and cover images after pairing.
7. For triple image steganography, we get about 1000 such pairs of secret and cover images after pairing.

Figure 1 shows a sample images of the dataset used in the proposed work.

3.2 Proposed Model

Though parallels are drawn between cryptography and steganography, our work is more like image compression done using auto encoders. We aim to compress the secret data from the covert image into the least noticeable areas of the cover image. Our models for single image steganography, double image steganography and triple image steganography consist of three sections. They are trained together as a single network. Each section's purpose has been described below individually.

3.2.1 Network Preparation

This section of our model serves the following two purposes: First is to increase the dimensions of the secret image(s) to match the size of the cover image in case the secret image's original size is smaller than the size of the cover image. The second purpose is to transform the colour-based pixels to more useful features for succinctly encoding the image. The layers and parameters used in the network preparation is shown in Table 3.

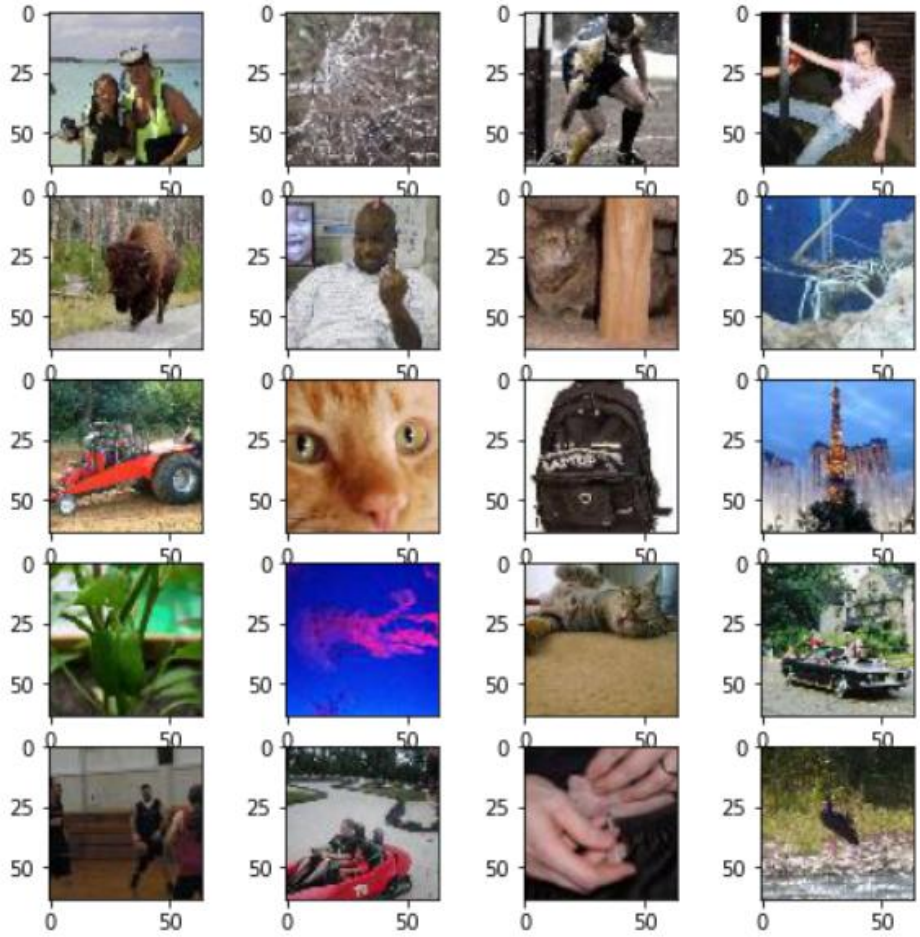


Figure 1. Sample Images from the Dataset [22].

Table 3. Layers and Parameters in Preparation Network.

Layers	No. of filters	Kernal Size	Strides	Padding	Activation
conv prep0_3x3	50	(3, 3)	(1, 1)	same	SELU
conv prep0_4x4	10	(4, 4)	(1, 1)	same	SELU
conv prep0_5x5	5	(5, 5)	(1, 1)	same	SELU

Algorithm for Network Preparation

Input: Secret image (input_secret_image)

Output: Prepared image (x)

Step 1: Apply ‘conv_prep0_3x3’ to input secret image and store it in x1.

Step 2: Apply ‘conv_prep0_4x4’ to input secret image and store it in x2.

Step 3: Apply ‘conv_prep0_5x5’ to input secret image and store it in x3.

Step 4: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 5: Apply ‘conv_prep0_3x3’ to input x and store it in x1.

Step 6: Apply ‘conv_prep0_4x4’ to input x and store it in x2.

Step 7: Apply ‘conv_prep0_5x5’ to input x and store it in x3.

Step 8: Concatenate x1, x2 and x3 and store the resultant image in x.

Algorithm for Hiding Network

This section takes as input the output of the preparation network(s) and the cover image and then combines the two to create the carrier image or container image. Table 4 shows the layers and parameters used in the hidden network.

Table 4. Layers and Parameters in Hiding Network

Layers	No. of filters	Kernal Size	Strides	Padding	Activation
conv prep0_3x3	50	(3, 3)	(1, 1)	same	SELU
conv prep0_4x4	10	(4, 4)	(1, 1)	same	SELU
conv prep0_5x5	5	(5, 5)	(1, 1)	same	SELU

Algorithm: Secret Image Preparation and Transformation

Input: Image x

Output: Output_Cprime

Step 1: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 2: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 3: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 4: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 5: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 6: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 7: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 8: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 9: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 10: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 11: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 12: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 13: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 14: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 15: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 16: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 17: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 18: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 19: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 20: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 21: Apply a Conv layer with 3 filters and kernel size= (3, 3) on x and store the result in Output_Cprime.

3.2.1 Reveal Network

This is the final section of the model. It is used by the receiver of the container image. This section is used by the receiver to extract the secret data in the form of two images from the container image. The working models of the single image steganography, double image steganography and triple image steganography is depicted in Figure 2(a), 2(b), and 2(c) respectively. The layers and parameters related data is shown in Table 5. The description of the symbols Figure 2(b) and 2(c) is shown in Table 6 and 7 respectively.

Table 5. Layers and Parameters in Reveal Network

Layers	No. of filters	Kernal Size	Strides	Padding	Activation
conv prep0_3x3	50	(3, 3)	(1, 1)	same	SELU
conv prep0_4x4	10	(4, 4)	(1, 1)	same	SELU
conv prep0_5x5	5	(5, 5)	(1, 1)	same	SELU

Algorithm for Revealing Network

Input: Image x with Noise

Output:

Step 1: Add Gaussian Noise to reveal image and store the result in input_with_noise.

Step 2: Apply ‘conv_prep0_3x3’ to input_with_noise and store it in x3.

Step 3: Apply ‘conv_prep0_4x4’ to input_with_noise and store it in x4.

Step 4: Apply ‘conv_prep0_5x5’ to input_with_noise and store it in x5.

Step 5: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 6: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 7: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 8: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 9: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 10: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 11: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 12: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 13: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 14: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 15: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 16: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 17: Concatenate x1, x2 and x3 and store the resultant image in x.

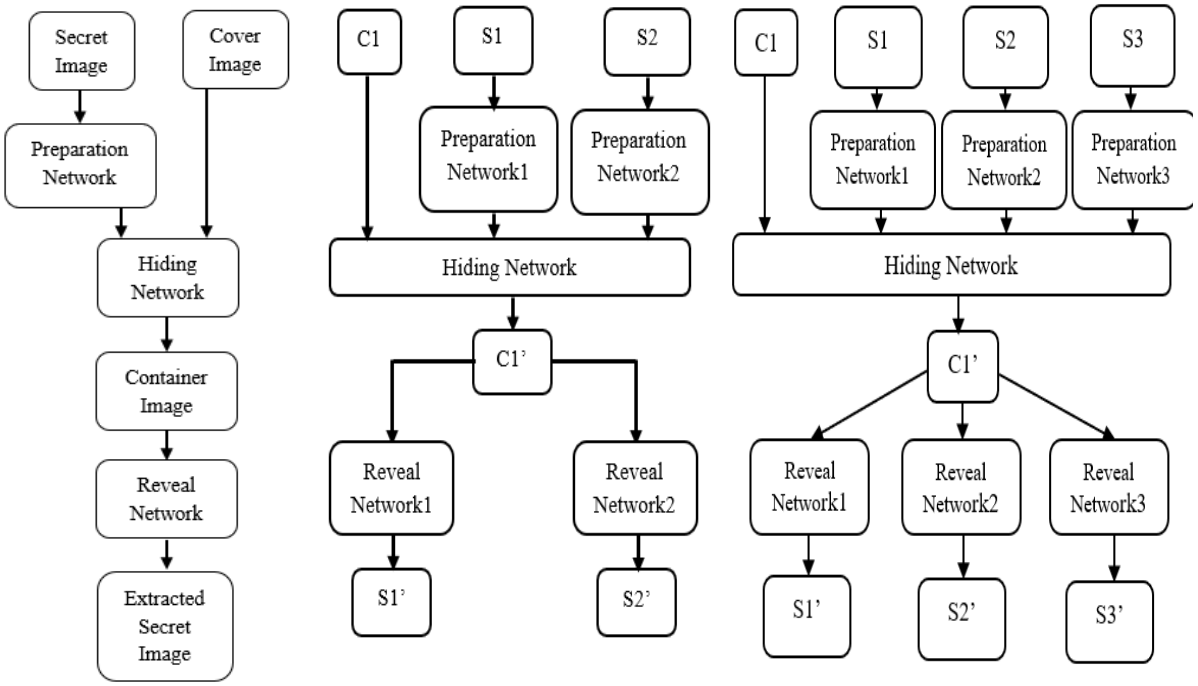
Step 18: Apply ‘conv_prep0_3x3’ to input x and store it in x3.

Step 19: Apply ‘conv_prep0_4x4’ to input x and store it in x4.

Step 20: Apply ‘conv_prep0_5x5’ to input x and store it in x5.

Step 21: Concatenate x1, x2 and x3 and store the resultant image in x.

Step 22: Apply a Conv layer with 3 filters and kernel size= (3, 3) on x and store the result in Output_Sprime.



a. Single Image b. Double Image c. Tripple Image

Figure 2. Working Model of Image Steganography

Table 6: Description of Symbols Used in Figure 2 (b)

Symbol	Description
C1	Cover Image
S1	First Secret image
S2	Second Secret image
C1’	Container image
S1’	First Secret image after retrieval
S2’	Second Secret image after retrieval

Table 7: Description of Symbols Used in Figure 2 (c)

Symbol	Description
C1	Cover Image
S1	First Secret Image
S2	Second Secret Image
S3	Third Secret Image
C1’	Container image
S1’	First secret image after retrieval
S2’	Second secret image after retrieval
S3’	Third secret image after retrieval

4. EXPERIMENTAL WORK

4.1 Implementation Details

The training details that have been used in each case have been explained below:

1. Adam optimizer has been used for optimization purpose.

2. Learning rate has been set tot with 0.001 till first 100 epochs, then decreasing to 0.0003 from 100 epochs to 250 epochs.

3. Models have been trained for 250 epochs.

4. Tiny Image Dataset has been used, where images are 64x64.

5. The loss used is for the full model is represented as:

k1||c1 − c1'||^2 + k2||s1 − s1'||^2 + k3||s2-s2'||^2+k4||s3-s3'||

Here, C1: Original cover image, C1': Cover image after steganography, S1: First Original Secret Image, S1': First Secret Image obtained after steganalysis, S2: Second Original Secret Image, S2': Second Secret Image obtained after steganalysis, S3: Third Original Secret Image, S3': Third Secret Image obtained after steganalysis.

6. Initially we take k1, k2, k3 and k4 as 1.0.

4.2 Results and Discussion

The performance of the model is evaluated by the value of the loss function.

Loss Function

The loss function is composed of two terms:

- **a.** Loss in the reconstruction of cover image is **||C − C'||^2,**
- **b.** Loss in the reconstruction loss of secret images is **||S-S'||^2.**

Here, C: Original cover image, C': Cover image after steganography, S: Original Secret Image, S': Secret Image obtained after steganalysis.

Loss For Single Images Steganography:

||c1 – c1'||^2 + |s1 − s1'||^2

Loss For Double Images Steganography:

||c1 − c1'||^2 + ||s1 − s1'||^2 + ||s2-s2'||^2

Loss For Triple Images Steganography:

||c1 − c1'||^2 + ||s1 − s1'||^2 + ||s2-s2'||^2+ ||s3-s3'||^2

(C1: Original cover image, C1': Cover image after steganography, S1: First Original Secret Image, S1': First Secret Image obtained after steganalysis, S2: Second Original Secret Image, S2': Second Secret Image obtained after steganalysis, S3: Third Original Secret Image, S3': Third Secret Image obtained after steganalysis;)

Table 8: Description of the Losses for Single Image Steganography

1. Loss of the model	126547.56
2. Loss secret1	61734.72
3. Loss cover	64812.84

The equations for calculating the losses of single image steganography shown in Table 8 are discussed below in eqns. (2), (3), (4).

Loss secret1= ||Original Secret image – Secret image obtained post steganalysis||^2 **(2)**

Loss Cover= ||Original cover image – Cover image after steganography||^2 **(3)**

Loss of the model= Loss secret1 + Loss Cover) **(4)**

Table 9: Description of the Losses for Double Image Steganography

1. Loss of the model	212927.22
2. Loss secret1	71435.27
3. Loss secret2	62181.72
4. Loss Cover	79310.23

Equations (5), (6), (7) and (8) are discussed here, which are used to calculate the losses of double image steganography as shown in Table 9.

Loss secret1= ||Original Secret image1 – Secret image1 obtained post steganalysis||^2 **(5)**

Loss secret2= ||Original Secret image2 – Secret image2 obtained post steganalysis||^2 **(6)**

Loss Cover= ||Original cover image – Cover image after steganography||^2 **(7)**

Loss of the model= Loss secret1 + Loss secret2 + Loss Cover) **(8)**

Table 10: Description of the Losses for Triple Image Steganography

1. Loss of the model	283369.93
2. Loss secret1	73016.59
3. Loss secret2	65051.05
4. Loss secret3	70651.48
5. Loss Cover	74651.29

Equations (9), (10), (11), (12) and (13) below provide the formulas used to determine the losses of triple image steganography, which are displayed in Table 10.

Loss secret1= ||Original Secret image1 – Secret image1 obtained post steganalysis||^2 **(9)**

Loss secret2= ||Original Secret image2 – Secret image2 obtained post steganalysis||^2 **(10)**

Loss secret3= ||Original Secret image3 – Secret image3 obtained post steganalysis||^2 **(11)**

Loss Cover= ||Original cover image – Cover image after steganography||^2 **(12)**

Loss of the model= Loss secret1 + Loss secret2 + Loss secret3 + Loss Cover **(13)**

5 CONCLUSION

Through the proposed work, a method for effective steganography and steganalysis has been presented using deep learning. The proposed model has further improved the single image steganography proposed by (Baluja, 2017) by designing two and three reveal networks for double and triple images steganography and steganalysis. The performance of the models for the single image, double image, and triple image steganography and steganalysis have been evaluated based upon the value of the loss function. For single image steganography, the value of the loss function obtained after training process is 126547.56, for double images steganography, the value of the loss function obtained is 212927.22, and for triple images steganography, it is 283369.93. The model is successfully being able to embed the secret images at the sender's end and being able to successfully retrieve them at the receiver's end without much distortion in either the secret images or the cover image.

REFERENCES

[1] Kessler, G.C., 2004. An overview of steganography for the computer forensics examiner. *Forensic science communications*, *6*(3), pp.1-27.

[2] Fridrich, J. and Goljan, M., 2002. Practical steganalysis of digital images: state of the art. *security and Watermarking of Multimedia Contents IV*, *4675*, pp.1-13.

[3] Subramanian, N., Elharrouss, O., Al-Maadeed, S. and Bouridane, A., 2021. Image steganography: A review of the recent advances. *IEEE access*, *9*, pp.23409-23423.

[4] Ozer, H., Avcibas, I., Sankur, B. and Memon, N.D., 2003, June. Steganalysis of audio based on audio quality metrics. In *Security and Watermarking of Multimedia Contents V* (Vol. 5020, pp. 55-66). SPIE.

[5] Yaghmaee, F. and Jamzad, M., 2010. Estimating watermarking capacity in gray scale images based on image complexity. *EURASIP Journal on Advances in Signal Processing*, *2010*, pp.1-9.

[6] Fridrich, J., Goljan, M. and Du, R., 2001. Detecting LSB steganography in color, and gray-scale images. *IEEE multimedia*, *8*(4), pp.22-28.

[7] Tamimi, A.A., Abdalla, A.M. and Al-Allaf, O., 2013. Hiding an image inside another image using variable-rate steganography. *International Journal of Advanced Computer Science and Applications (IJACSA)*, *4*(10).

[8] Qian, Y., Dong, J., Wang, W. and Tan, T., 2015, March. Deep learning for steganalysis via convolutional neural networks. In *Media Watermarking, Security, and Forensics 2015* (Vol. 9409, pp. 171-180). SPIE.

[9] Lionel Pibre, Pasquet Jérôme, Dino Ienco, and Marc Chaumont. Deep learning for steganalysis is better than a rich model with an ensemble classifier and is natively robust to the cover source-mismatch. arXiv preprint arXiv:1511.04855, 2015.

[10] Kheddar, H., Hemis, M., Himeur, Y., Megías, D. and Amira, A., 2024. Deep learning for steganalysis of diverse data types: A review of methods, taxonomy, challenges and future directions. *Neurocomputing*, p.127528.

[11] Sabah Husien and Haitham Badi. Artificial neural network for steganography. Neural Computing and Applications, 26(1):111–116, 2015.

[12] Reshma, V.K., Vinod Kumar, R.S., Shahi, D. and Shyjith, M.B., 2022. Optimized support vector neural network and contourlet transform for image steganography. *Evolutionary Intelligence*, *15*(2), pp.1295-1311.

[13] V Kavitha and KS Easwarakumar. Neural based steganography. PRICAI 2004: Trends in Artificial Intelligence, pages 429–435, 2004.

[14] Alexandre Santos Brandao and David Calhau Jorge. Artificial neural networks applied to image steganography. IEEE Latin America Transactions, 14(3):1361–1366, 2016

[15] N. F. Johnson and S. Jajodia, ‘‘Exploring steganography: Seeing the unseen,’’ Computer, vol. 31, no. 2, pp. 26–34, Feb. 1998.

[16] S. Gupta, G. Gujral, and N. Aggarwal, ‘‘Enhanced least significant bit algorithm for image steganography,’’ Int. J. Comput. Eng. Manage., vol. 15, no. 4, pp. 40–42, 2012.

[17] R. Das and T. Tuithung, ‘‘A novel steganography method for image based on Huffman encoding,’’ in Proc. 3rd Nat. Conf. Emerg. Trends Appl. Comput. Sci., Mar. 2012, pp. 14–18.

[18] P. Wu, Y. Yang, and X. Li, ‘‘Image-into-image steganography using deep convolutional network,’’ in Proc. Pacific Rim Conf. Multimedia. Cham, Switzerland: Springer, 2018, pp. 792–802.

[19] P. Wu, Y. Yang, and X. Li, ‘‘StegNet: Mega image steganography capacity with deep convolutional network,’’ Future Internet, vol. 10, no. 6, p. 54, Jun. 2018.

[20] X. Duan, K. Jia, B. Li, D. Guo, E. Zhang, and C. Qin, ‘‘Reversible image steganography scheme based on a U-Net structure,’’ IEEE Access, vol. 7, pp. 9314–9323, 2019.

[21] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B. and Bharath, A.A., 2018. Generative adversarial networks: An overview. *IEEE signal processing magazine*, *35*(1), pp.53-65.

[22] Yuan, C., Wang, H., He, P. *et al.* GAN-based image steganography for enhancing security via adversarial attack and pixel-wise deep fusion. *Multimed Tools Appl* **81**, 6681–6701 (2022). https://doi.org/10.1007/s11042-021-11778-z

[23] Dataset:https://www.kaggle.com/datasets/aryashah2k/mit-80-million-tiny-images-datase

[24] Kumar, U., Karn, S. and Shivahare, B.D., 2023, September. Image Watermarking Techniques and Applications. In *2023 3rd International Conference on Innovative Sustainable Computational Technologies (CISCT)* (pp. 1-4). IEEE.