Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2022.027274
Article

Tech Science Press

Image Steganography Using Deep Neural Networks

Kavitha Chinniyan*, Thamil Vani Samiyappan, Aishvarya Gopu and Narmatha Ramasamy

Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, 641004, India
*Corresponding Author: Kavitha Chinniyan. Email: ckk.cse@psgtech.ac.in
Received: 13 January 2022; Accepted: 20 February 2022

Abstract: Steganography is the technique of hiding secret data within ordinary data by modifying pixel values which appear normal to a casual observer. Steganography which is similar to cryptography helps in secret communication. The cryptography method focuses on the authenticity and integrity of the messages by hiding the contents of the messages. Sometimes, it is not only just enough to encrypt the message but also essential to hide the existence of the message itself. As this avoids misuse of data, this kind of encryption is less suspicious and does not catch attention. To achieve this, Stacked Autoencoder model is developed which initially compresses and encodes the data effectively and which finally decodes the data back from the compressed encoded representation to a representation that is more similar to the original input. The secret data is encrypted using Elliptic Curve Cryptography algorithm and transformed into an image before it is encoded by the model which combines the cover and the secret image. The cover and the secret image produce a container image which after decoding and decrypting gives the secret data. The proposed work consists of multiple networks which are trained with Flickr Image Dataset and results in Secret Image with a Loss of 4% and Container Image with a Loss of 14%.

Keywords: Cryptography; data security; image reconstruction; neural networks

1 Introduction

This digital era has witnessed emerging technologies, has seen new inventions being made and has also identified security as the most essential field. The data needs to be kept secure and safe so that it could only be accessed by the authorized people and any unauthorized user cannot have any access to it. Initially, cryptography method was used in which the message was encrypted in a way such that only the sender and receiver knew the ways to decrypt it. The major disadvantage of cryptography was that if an intruder came to know that the message contains secret information, then the probability of message being decrypted by the intruder increases. To overcome this disadvantage, the technique of steganography [1] was introduced.

The technique of steganography was much better than cryptography as the data was concealed inside an image. It has an advantage over cryptography as the intruder will not have any knowledge about the hidden data. The data could be decrypted only from the image by the authorized person using the authorized key that is required to decode the data. This improves the security and reliability of transmission of data.

Steganography technique works by modifying pixel values. A pixel is the smallest unit of a picture and the color in each pixel is a function of the proportions of red, green, and blue (assuming RGB). So, a pixel with a value of 0, 1, and 0 means that it contains 0 parts red, 1 part green, and 0-parts blue; in other words, it is a green pixel. A pixel in an 8-bit system may hold up to 8 digits (zeroes or ones), and the biggest number that can be represented in 8 digits is 11111111, which is 255, and the smallest number that can be represented in 8 digits is 00000000, which is 0. So, in an 8-bit case, every pixel may have a value for each of the color ranging from 0 to 255. The Stacked Autoencoder model [2] works by altering the pixel values of the cover image with the pixel values of the secret image. This would change the colour in the cover image in the least amount for the three pixels, making the secret image practically invisible inside the cover image.

The entire steganography procedure begins with a cover file that contains a carrier-like image. The secret data is encoded inside the cover image. After encoding, the result which is a stego object, also known as the Container image contains the hidden information. The secret message is recovered from the Container picture during the extraction procedure.

Apart from data security, image steganography has various other applications and one among them is Copyright protection in which a watermark [3] can be embedded on an image through steganography techniques. Later, if someone else tries to get the ownership of that image, then the embedded watermark helps the original artist to claim their ownership. Government agencies can use steganography to maintain critical data like criminal records and evidences. Image Steganography can also be used in digital medical imaging [4,5] where patients' details can be embedded in their medical images (like X-ray). If any ambiguity arises about the patients' information, steganography would be extremely helpful as all the information can be decoded from the embedded x-ray image.

The rest of the article is organized as follows: Section 2 presents related works, Section 3 focusses on the System flow, Section 4 discusses the experimental results and finally Section 5 draws the conclusion and discusses the scope for future research.

2 Literature Review

Juneja et al. [6] proposed an improved Least Significant Bit (LSB) based Steganography technique for images conveying better data security. The method used an embedding algorithm for hiding encrypted messages in nonadjacent and random pixels in the images where the area was smooth and had edges. Their main aim was to evaluate and design a new and improved information concealing technology based on LSB. The improved LSB technique was both resilient and successful, as well as one that makes it extremely difficult for the naked eye to predict and identify the presence of any hidden data inside the host image.

Halder et al. [7] proposed a new method of image steganography by encoding the encrypted data or message using Hash-LSB with Rivest–Shamir–Adleman (RSA), which provides more security to data. The main objective is to combine one steganographic technique and one cryptographic technique to enhance data security. The algorithm uses a hash function which is used to generate a pattern for concealing secret data bits into LSB of cover image pixel values, but if the hash function is figured out, the secret data can easily be decoded.

Rehman et al. [8] proposed a deep learning-based encoder-decoder model for encoding of images as payload. The method directly used images to encode and recover from the cover image unlike earlier methods which used the binary representation of secret data. This is a novel technique which showed excellent results on wide range of image datasets. Though the technique is new, the visual loss in the data is high.

Pal et al. [9] proposed a steganography technique in which the secret data was encoded in jpeg pixels of the cover image. The image underwent Discrete Cosine Transformation (DCT) followed by quantization process which transformed the cover image. Their main objective was to increase the embedding capacity which was done using the modified quantization process and to maintain the visual quality of the reconstructed image which was achieved by embedding the secret data into middle band quantized DCT coefficients. In this scheme, rather than embedding the secret data bits directly into the coefficients, an appropriate indirect approach is adopted to conceal two bits of the secret message into some selected DCT coefficients, but the method was tested only on grayscale images.

Rasras et al. [10] proposed a method which combined cryptography and LSB based Image Steganography. This paper is similar to some of the previously mentioned papers, as it combines cryptography and steganography. The security level is increased by using two private keys but the disadvantage here is that the LSB is an unsecure technique for concealing messages since it can be easily detected.

Zhang et al. [11] proposed Generative Adversarial Networks (GAN) model which can completely hide a gray secret image into a RGB cover image of same size. Their main aim was to improve data security by the use of adversarial training. The model was experimented with different activation functions and different optimizers which speeded up the training and produced better results but was not suitable for applications which require the secret image to be accurately revealed since the secret image gets distorted.

Baluja [12] presented a Neural Network model based on autoencoders to hide a full color image inside another color image of similar size with a minimal pixel loss in both the images. Unlike the widely used steganography techniques which encode the secret image inside the least significant bits of the cover image, the method compresses and distributes the secret image's representation across all of the available bits. The system was trained with images from ImageNet dataset and numerous transformations were applied on the image to analyse the quality of cover and secret images.

Duan et al. [13] proposed a new high-capacity image steganography method based on deep learning which used SegNet Deep Neural Network with a set of Hiding and Extraction networks. The SegNet Deep Neural Network with a collection of Hiding and Extraction networks enabled steganography and retrieval of complete pictures to boost steganographic ability. The DCT was used for converting the secret image, and then the modified image was encoded by Elliptic Curve Cryptography (ECC) but the amount of data that can be hidden inside the cover image was minimised when compared with other techniques.

Das et al. [14] performed multi-image steganography by hiding two or more secret images in a single cover image. Unlike the previous works which focused on hiding single image inside the cover image, this work focused on hiding multiple images inside a cover image. The hidden secret images were decoded with minimum visual loss. Several previous methods have attempted to use neural networks to either augment or replace a tiny portion of an image. This multi-image steganography method creates a fully trainable neural network model based on autoencoders that provided good results visually. But the system relied heavily on visual loss for overall loss and didn't analyse different losses which could have better suited the models' performance.

Oktavianto et al. [15] primarily discussed the application of steganography on Portable Network Graphics (PNG) image media with the Spread Spectrum Image Steganography (SSIS) method which was more secure. The SSIS technique was more secure because the keywords from this method were known only to the sender and the receiver. From this research work it was observed that if larger the image size (pixels), larger is the number of characters or messages that can be inserted. But the number of characters that can be inserted in the RGB digital image was more than the Grayscale digital image since RGB has 3 color channels and a greater number of characters can be inserted into it. So, for grayscale images the system seems to be comparatively low in efficiency than the RGB digital images.

Islam et al. [16] focused on the implementation of LSB Steganography on Bitmap along with Advanced Encryption Standard (AES) cryptography technique for better security. The main focus was on the usage of Bitmap images since Bitmap images are uncompressed and more convenient than any other image format. Their research work involved a new Steganography technique to hide large data in Bitmap image using filtering-based algorithm, which used Most Significant Bit bits for filtering purpose. This method used the concept of status checking for insertion and retrieval of messages which was more efficient than LSB method. But it is found to be more powerful for grayscale images only.

The Literature review analysis has shown that many solutions have been proposed to solve the problem of data security but these existing solutions still have certain drawbacks. Based on this information, it can be said that it is important to develop a system with high data security which is the main focus of this paper.

3 System Flow

The flow of the system is shown in Fig. 1. The input images are preprocessed in which the Cover and Secret images are resized and normalised. Encryption of the secret text by the user is done after which the text is transformed into an image which is considered as the Secret image. From the sender's side, the Cover and Secret images are given as input to the Stacked Autoencoder model which outputs the Container image. The sender then sends this container image to the receiver. The receiver extracts the Secret image from the Container image and the text is decrypted. Hashing concept is incorporated for checking the integrity of the file. It is used to verify if the container image sent by the sender is same as the one that is received by the receiver.

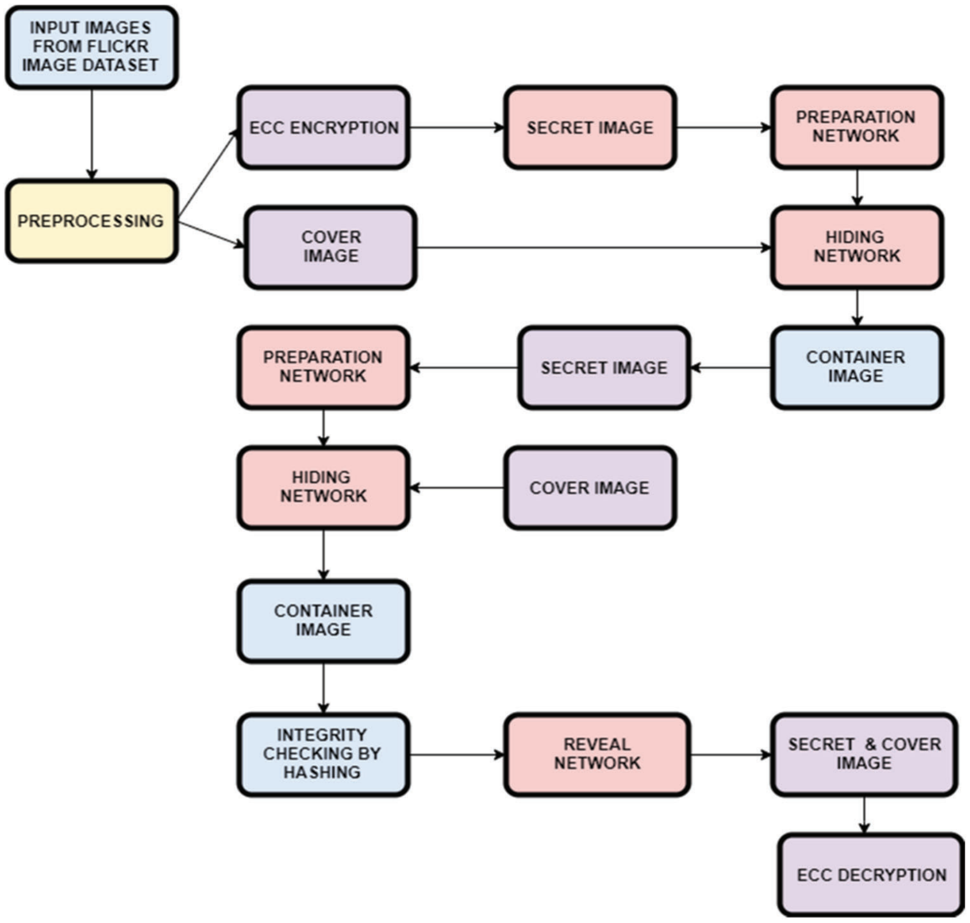

Figure 1: Flow of the system

3.1 Dataset and Preprocessing

The dataset used is the Flickr image dataset [17] from Kaggle which is a public dataset that can be accessed by anyone. It consists of 30000 random images like a person riding a cycle or two men working or a little girl with pink dress etc. The images are of different dimensions, which can be resized to a uniform size for feeding as input to the model. Around 2000 random images were drawn from Flickr image dataset and were pre-processed. All the images were uniformly resized to 64×64 and were normalized by dividing the pixel values by 255.

3.2 Autoencoder

An autoencoder [18] is a type of artificial neural network that is used to learn unsupervised data encodings. The goal of an autoencoder is to train the network to capture the most essential bits of the input picture in order to learn a lower-dimensional representation (encoding) for a higher-dimensional data [19], often for dimensionality reduction. The usage of a neural network [20] to reproduce an input may look straightforward, however the size of the input is decreased throughout the replication process, resulting in a smaller representation. As it copies the data from the input to the output in an unsupervised manner, the autoencoder is also known as a replicator neural network. Autoencoders are made up of the following three parts:

Encoder: A module that compresses the input data into an encoded form.

Bottleneck: The most significant element of the network since it holds the compressed knowledge representations.

Decoder: A module that aids the network in decompressing knowledge representations and reassembling data from its encoded state. The output is then compared to the original data.

The basic architecture of the Autoencoder is shown in Fig. 2.

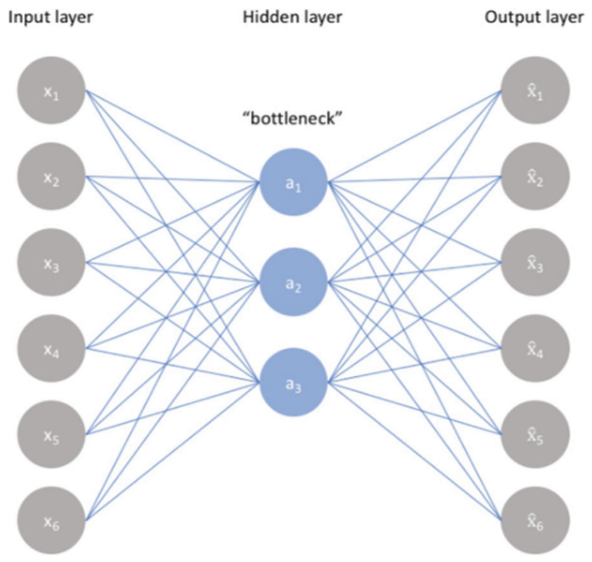

Figure 2: Basic architecture of autoencoder

3.3 Stacked Autoencoder

Datasets with random images have complex relationships among the features. As a result, utilizing just one Autoencoder is insufficient since it is possible that a single Autoencoder won't be able to minimize the input features' dimensionality. As a result, stacked autoencoders [21] can be employed in these situations. Multiple encoders are placed on top of one another in stacked autoencoders as shown in Fig. 3. This work combines two Autoencoders to form the Stacked Autoencoder model.

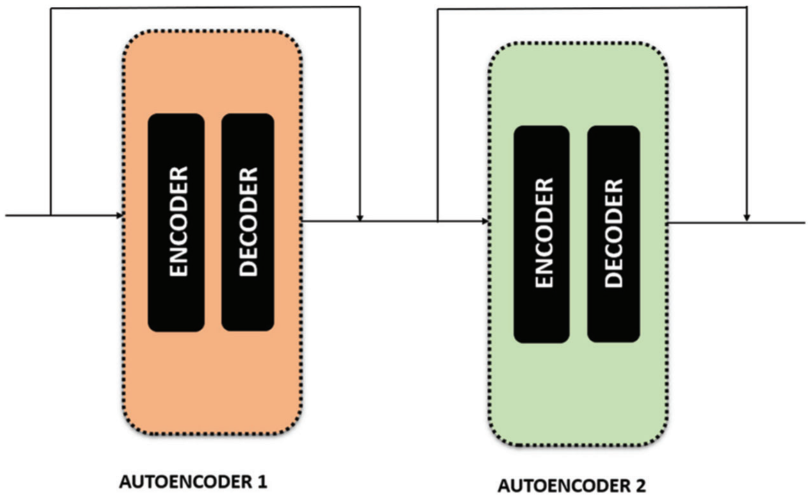

Figure 3: Stacked autoencoder

3.3.1 Preparation Network, Hiding Network and Reveal Network

The system consists mainly of three networks. The first network is the preparation network which prepares the secret image. This network transforms the pixels of the secret image into features which is given as input to hiding network.

The secret image is hidden into the cover image by the Hiding Network. It takes the cover images and the output of preparation network as input and displays the container image which holds the secret image inside it. The container image should be precise and should also recreate the secret image successfully.

The last network is the Reveal Network which is responsible for extracting the secret image from the container image. The Reveal Network is used by the receiver whereas the other two networks are used by the sender.

Preparation Network contains two Convolutional layers with 65 filters and both the Hiding and Reveal Network contains five Convolutional layers with the same 65 filters. All the layers use ReLu as activation function which is defined as in Eq. (1) and the optimiser used is Adam as defined in Eq. (2).

$$y = \max(0, x) \tag{1}$$

$$w_{t+1} = w_t - \hat{m}_t \left(\frac{\alpha}{\sqrt{\hat{v}_t} + \varepsilon} \right) \tag{2}$$

where w_t is the weight at time t, w_{t+1} is the weight at time $t+1$, α is the learning rate and ε is a small positive constant.

Preparation and Hiding networks are a part of the encoder and the Reveal network is a part of the decoder shown in Fig. 4. Collectively, these networks form Stacked Autoencoder. The loss of the decoder is calculated by finding difference between the original secret image pixels and the reconstructed secret image pixels.

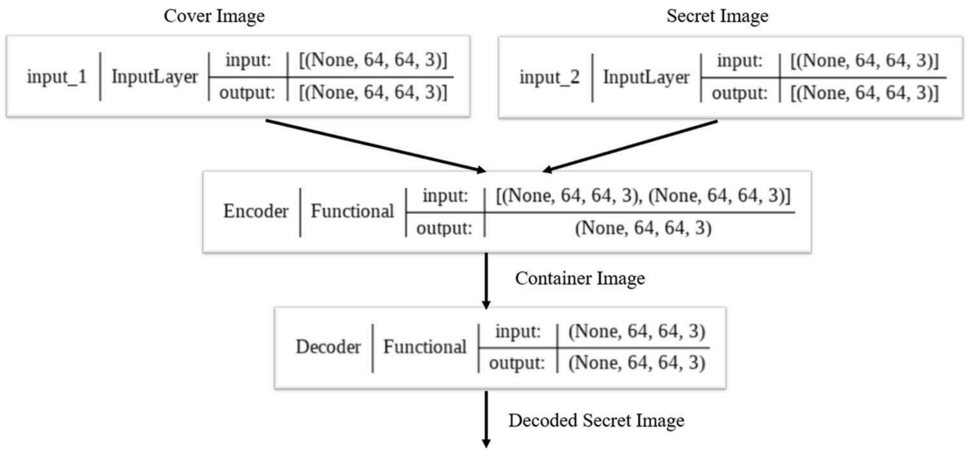

Figure 4: Basic structure of encoder and decoder

The model is trained by reducing the error with the Eq. (3).

$$L(C,\ C',\ S,\ S') = ||C - C'|| + \beta||S - S'|| \tag{3}$$

where C is the Cover image, C' is the Container image, S is the Secret Image, S' is the Reconstructed Secret image. The value of β is taken as 1.

3.4 Elliptic Curve Cryptography

ECC encryption [22] is not a primitive operation since it does not give an encryption mechanism directly. Instead, a hybrid encryption scheme is used by generating a shared secret key for symmetric data encryption and decryption using the Elliptic Curve Diffie–Hellman (ECDH) key exchange technique. This is called Elliptic Curve Integrated Encryption Scheme (ECIES). Inputs are the plaintext message, an ECC public key of the recipient. A shared ECC key is generated afterwards. The public key is generated from the private key. The ECDH shared secret is the ephemeral private key combined with the recipient's public key. The public key by ECC is used for encryption and the private key is used for decryption.

3.4.1 Encryption

For ECC algorithm, the curve that is used is "brainpoolP256r1" which is a 256-bit prime field Weierstrass curve that is imported using registry .get_curve function. The equation corresponding to the curve

$$y^2 = x^3 + ax + b(\text{mod } p) \tag{4}$$

This elliptic curve over the finite field contains a group of integer coordinates {x, y}, where $0 \le x, y < p$, where the value of the constants is given in Tab. 1.

All these values are represented in hexadecimal form which in binary form will be represented as 256 bits.

Table 1: Elliptic curve equation values

Constant	Value
a	0x7d5a0975fc2c3057eef67530417affe7fb8055c126dc5c6ce94a4b44f330b5d9
b	0x26dc5c6ce94a4b44f330b5d9bbd77cbf958416295cf7e1ce6bccdc18ff8c07b6
G	(0x8bd2aeb9cb7e57cb2c4b482ffc81b7afb9de27e1e3bd23c23a4453bd9ace3262, 0x547ef835c3dac4fd97f8461a14611dc9c27745132ded8e545c1d54c72f046997)
n	0xa9fb57dba1eea9bc3e660a909d838d718c397aa3b561a6f7901e0e82974856a7
p	0xa9fb57dba1eea9bc3e660a909d838d726e3bf623d52620282013481d1f6e5377

Initially, the sender will have to generate a private key. After the private key is generated, public key is calculated by multiplying the value of private key and G. The product of public key and private key results in a shared key (which will be shared both by sender and receiver but calculated with public and private keys).

The shared key will be in the form of Eq. (4) in which the value of x, y is required to perform encryption. The values (x, y) which forms the Elliptic Curve point is converted into a 256 bit secret key by hashing and concatenating the x and y coordinates. The message is then encrypted using these 256 bits (32 bytes) secret key.

3.4.2 Decryption

To decrypt, private key and shared key are required. The shared key is generated by multiplying the receiver's private key and sender's public key. The shared key will be an Elliptic Curve point which will be converted to 256 bits and then used for decryption.

3.5 Hashing

The internet has become an integral component of everyday life in the era of cyber warfare. Due to the prevalence of rapid and sophisticated cyber threats, downloading a file from the internet cannot be guaranteed to be completely safe. Security flaws, data breaches, viruses, and malware have all become increasingly prevalent, allowing anybody to abuse the originality, integrity, and validity of any file downloaded from the internet. Hashing could be used to check the integrity of a file, i.e., it can be used to see if a file is authentic or if a file has been tampered or altered by unreliable outsiders.

Every single file on the internet is distinct in its own manner, and even if a single byte or bit of information is altered, the hash value of the original and modified files will differ. Hashing shown in Fig. 5 is incorporated to see if the image transmitted by the sender and the image received by the receiver are the same.

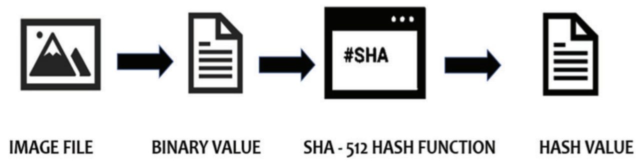

Figure 5: Hashing

Various hash algorithms were analysed and it is observed that Secure Hash Algorithm-512 (SHA-512) [23] is one of the strongest and secure hash functions. So, first, this algorithm takes the Image file which is the

Container Image and the binary form [24] of the image file is given as input to the hash function which outputs a hash value which is a 512-bit number.

3.5.1 Working of SHA-512

It involves a total of 4 steps as shown in Fig. 6.

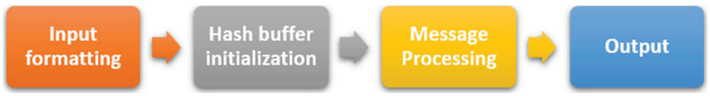

Figure 6: Working of SHA-512

Input Formatting

SHA-512 is confined in its ability to hash messages of any size, i.e., it has an input size limit. The message's length should be a multiple of 1024 bits. Because the formatted message will be processed in 1024-bit blocks, each block should have a total of 1024 bits to deal with. If the message's length is not a multiple of 1024, padding bits are added.

Hash Buffer Initialization

The algorithm works in such a way that it processes each 1024-bit block of the message using the preceding block's result. Each intermediate result must be used in the processing of the following block; thus, it must be saved someplace. This is accomplished via the hash buffer, which will also store the final hash digest of the whole SHA-512 processing phase as the last of these 'intermediate' results.

Message Processing

Message processing is done by extracting one block of 1024 bits at a time from the formatted input. The 1024-bit block, as well as the outcome of the preceding operation, are used in the actual processing. There are 80 rounds and an addition operation in this phase of the SHA-512 algorithm.

Output

The final 512-bit Hash value of the original message is outputted after every block of 1024 bit has gone through the message processing phase, i.e., the last iteration of the phase.

4 Results

This work utilises Python and its related tools for image preprocessing, model building and training stages. Graphics Processing Unit (GPU) is utilised in training the model.

4.1 Performance Metrics

The metric used to assess the performance of the developed Stacked Autoencoder model is "Loss"–loss of Container image after encoding and the loss of Secret image after decoding. The Loss value of the images are calculated using the Eq. (3).

4.2 Evaluation Results

To compare and analyse the results of Stacked Autoencoder model, Autoencoder model consisting of multiple networks was trained with Flickr Image Dataset (1000 cover images and 1000 secret images) after 500 epochs which took around 4 h resulted in Secret Image Loss of 6% and Container image Loss of 25%.

The Graph in Fig. 7 shows the Training Loss of Autoencoder model with Number of Epochs in x-axis and Loss in y-axis. From the plot, it can be inferred that initially the value of loss gets reduced but after a few iterations the loss increases to a peak value and then gradually reduces.

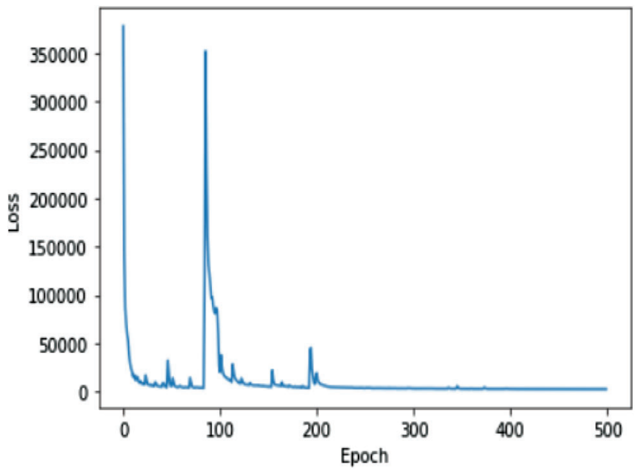

Figure 7: Autoencoder model loss

The Stacked Autoencoder model which consists of multiple networks was trained with Flickr Image Dataset (1000 cover images and 1000 secret images) for 500 epochs took around 6 h which resulted in Secret Image Loss of 4% and Container Image Loss of 14%. The developed model was able to successfully hide a secret image into cover image of same size. The reconstructed secret image clearly resembles the original secret image as the visual loss is low.

The Graph in Fig. 8 shows the Training Loss of Stacked Autoencoder model with Number of Epochs in x-axis and Loss in y-axis. From the plot, it can be inferred that the loss gets reduced in each epoch.

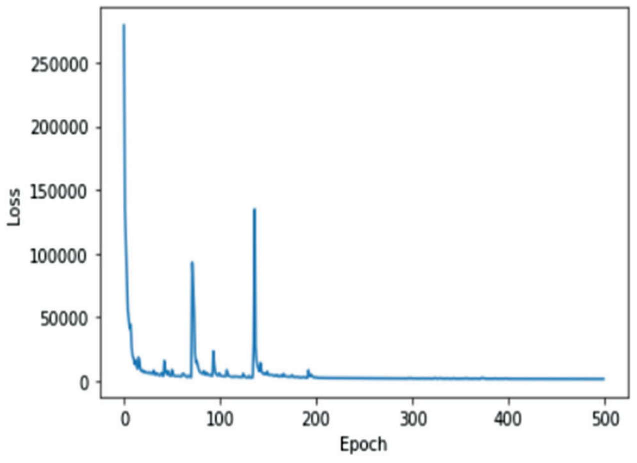

Figure 8: Stacked autoencoder model loss

Figs. 9 and 10 show the distribution of pixel errors in the Container image and distribution of pixel errors in Secret image respectively with the number of pixel errors in x-axis and the frequency of those pixel errors in y-axis. The error distribution is comparatively lesser than the Autoencoder model.

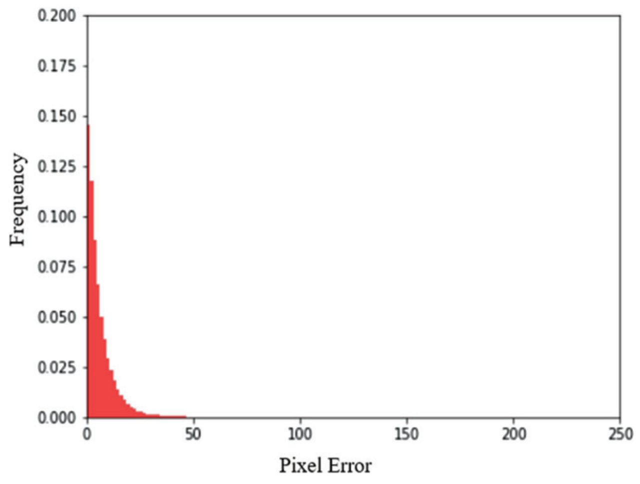

Figure 9: Distribution of pixel errors in container image

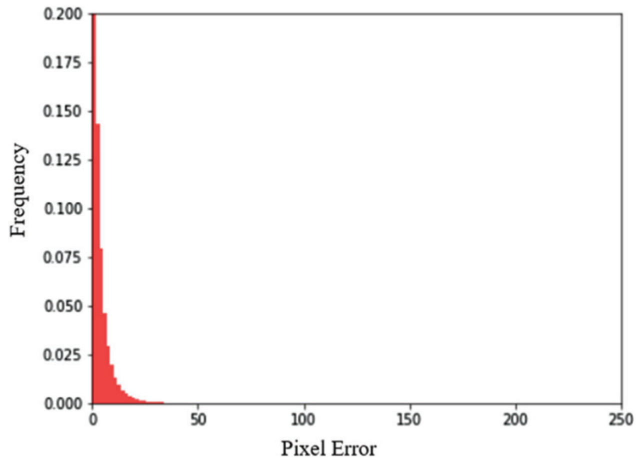

Figure 10: Distribution of pixel errors in secret image

Comparing the following figures: Figs. 11–14, it is evident that the Stacked Autoencoder model has performed the decoding better than the Autoencoder model, LSB and DCT technique [25]. In the Decoded secret image of the Autoencoder model, though the image appears to be clear, it seems to be distorted whereas the Stacked Autoencoder model has given good visible results with the same 500 epochs. The Decoded secret images of LSB and DCT techniques are also visibly good but this technique can easily be detected and the Secret image can be extracted by any intruder who knows about the existence of the Secret data inside the Container image. Another disadvantage of DCT is that it is applicable only to gray-scale images.

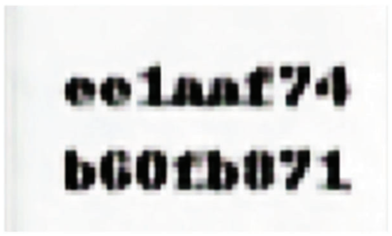

Figure 11: Decoded secret image of autoencoder model

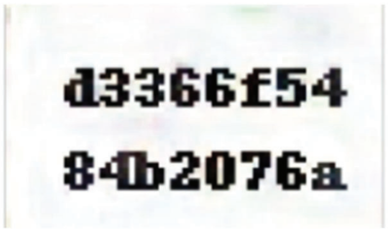

Figure 12: Decoded secret image of stacked autoencoder model

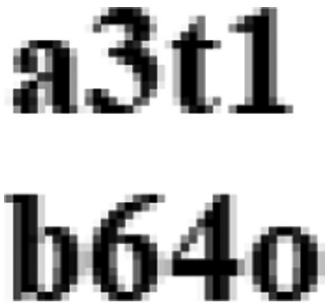

Figure 13: Decoded secret image of LSB technique

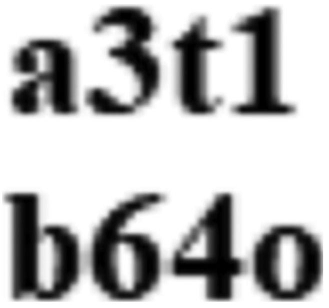

Figure 14: Decoded secret image of DCT technique

Fig. 15 shows the Container images (originally 64 × 64, enlarged for visualization) of LSB technique, Autoencoder, Stacked Autoencoder, DCT technique and their corresponding original image. The LSB technique is an approach which modifies and replaces the last bit of each pixel with the secret message's data bit. As a result, series of pixels are modified directly which in turn modifies the color of the image. The Container image of the Autoencoder model looks similar to the original image but is of low quality. The Container image of DCT is a plain image which doesn't resemble the original Cover image. All the Container images have lost few pixels due to encoding but the Container image of Stacked Autoencoder model looks almost similar to the Original Cover image with minimal loss.

Fig. 16 shows the Container images (originally 64 × 64, enlarged for visualization) of LSB technique, Autoencoder, Stacked Autoencoder, DCT technique and their corresponding original image. Similar to the previous example, in the Container image of LSB technique, series of LSB of the pixels are modified which results in the color change of the Container image and the Container image of Autoencoder model has higher loss than the Container image of the Stacked Autoencoder model which resembles its original image with minimal loss. This proves that implementing Image steganography using Neural Networks produces better results than implementing it through LSB and DCT steganography techniques. The LSB technique is implemented using an online tool called StegOnline: A New GUI Steganography Tool.

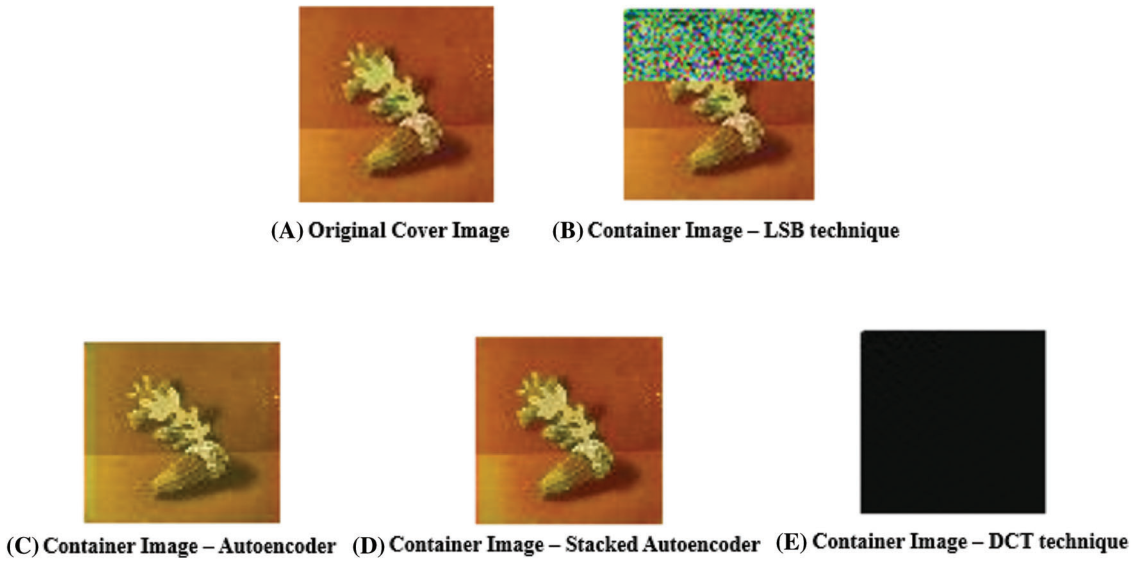
(A) Original Cover Image
(B) Container Image – LSB technique
(C) Container Image – Autoencoder
(D) Container Image – Stacked Autoencoder
(E) Container Image – DCT technique

Figure 15: Container images obtained through different techniques

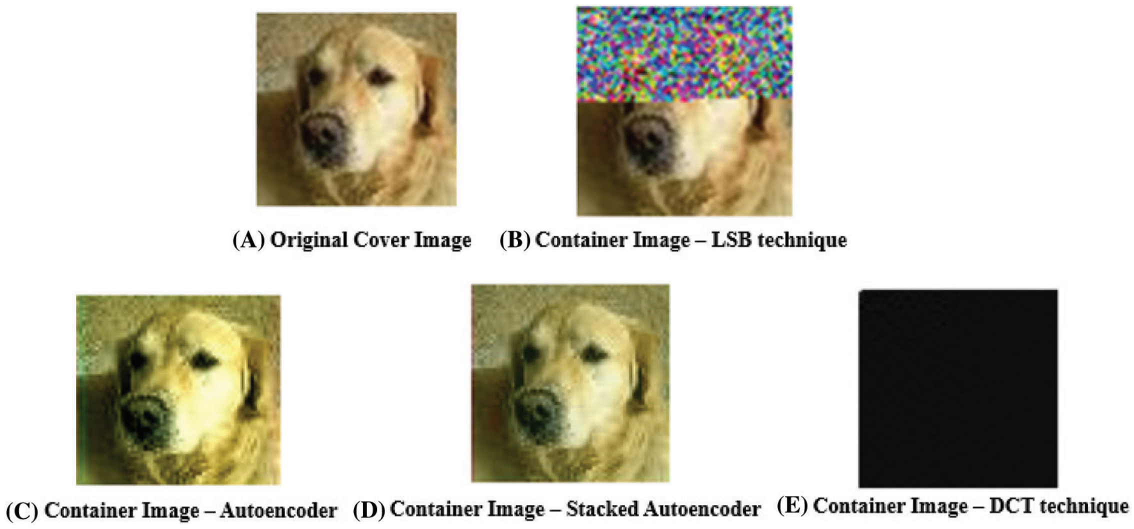
(A) Original Cover Image
(B) Container Image – LSB technique
(C) Container Image – Autoencoder
(D) Container Image – Stacked Autoencoder
(E) Container Image – DCT technique

Figure 16: Container images obtained through different techniques

There are tools that may be used to look for hidden information in images. Steganalysis [26], or the study of discovering messages concealed via Steganography, is the term referring to this method. There are currently only a few web tools that may be used to execute this method. One such Steganalysis detection tool is StegExpose [27], a freely available steganalysis toolset. "The StegExpose rating system is created from an innovative and extensively tested mix of pre-existing pixel-based steganalysis approaches such as Dumitrescu's Sample Pairs, Fridrich's RS Analysis, Westfeld's Chi Square Attack, and Dumitrescu's Primary Sets," according to the tool's description. The tool can analyse several photos in the background without requiring human intervention, delivering a thorough steganalytic report after the tool has completed its task.

For experimental purposes, a set of Container images of the Stacked Autoencoder model were exposed to Steganalysis with StegExpose tool which failed to detect the hidden messages present inside the Container images but the Container images of LSB technique is prone to Steganalysis which can easily detect the existence of some hidden content within the image. This proves the reliability of the model that has been developed.

5 Conclusion

This paper has implemented the Image Steganography technique using Deep Neural Networks which showed promising results. The developed model is capable of hiding any RGB image inside another RGB image. The security of the system was also improved by using cryptography and hashing techniques. The major disadvantage of cryptography is that it catches attention and if the key value is revealed, the data is compromised. The major disadvantage of steganography is that if the intruder figures out the hidden image, then secret data is revealed. Hence this work combines Cryptography and Steganography which provides double layer of security. In future, works related to reducing the loss of the Container image i.e., improving the Container image in terms of its quality can be carried out.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Cheddad, J. Condell, K. Curran and P. Mc Kevitt, “Digital image steganography: Survey and analysis of current methods,” *Signal Processing*, vol. 90, no. 3, pp. 727–752, 2010.

[2] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio and P. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” *The Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[3] I. Cox, M. Miller, J. Bloom, J. Fridrich and T. Kalker, “Steganography,” in *Digital Watermarking and Steganography*, 2nd ed., vol. 24, San Mateo, CA, USA: Morgan Kaufmann, 12, pp. 425–469, 2007.

[4] X. R. Zhang, W. F. Zhang, W. Sun, X. M. Sun and S. K. Jha, “A robust 3-D medical watermarking based on wavelet transform for data protection,” *Computer Systems Science & Engineering*, vol. 41, no. 3, pp. 1043–1056, 2022.

[5] X. R. Zhang, X. Sun, X. M. Sun, W. Sun and S. K. Jha, “Robust reversible audio watermarking scheme for telemedicine and privacy protection,” *Computers, Materials & Continua*, vol. 71, no. 2, pp. 3035–3050, 2022.

[6] M. Juneja and P. Sandhu, “Information hiding using improved LSB steganography and feature detection technique,” *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 2, no. 4, pp. 275–279, 2013.

[7] R. Halder, S. Sengupta, S. Ghosh and D. Kundu, “A secure image steganography based on RSA algorithm and hash-LSB technique,” *IOSR Journal of Computer Engineering (IOSR-JCE)*, vol. 18, no. 1, pp. 39–43, 2016.

[8] A. Rehman, R. Rahim, M. Nadeem and S. Hussain, “End-to-end trained CNN encode-decoder networks for image steganography,” in *ECCV Workshops*, Munich, Germany, pp. 723–729, 2017.

[9] A. K. Pal, K. Naik and R. Agrawal, “A steganography scheme on JPEG compressed cover image with high embedding capacity,” *The International Arab Journal of Information Technology*, vol. 16, no. 1, pp. 116–124, 2019.

[10] R. J. Rasras, Z. A. AlQadi and M. R. A. Sara, “A methodology based on steganography and cryptography to protect highly secure messages,” *Engineering, Technology & Applied Science Research*, vol. 9, no. 1, pp. 3681–3684, 2019.

[11] R. Zhang, S. Dong and J. Liu, “Invisible steganography via generative adversarial networks,” *Multimedia Tools and Applications*, vol. 78, no. 7, pp. 8559–8575, 2019.

[12] S. Baluja, “Hiding images within images,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 7, pp. 1685–1697, 2020.

[13] X. Duan, D. Guo, N. Liu, B. Li, M. Gou *et al.,* “A new high capacity image steganography method combined with image elliptic curve cryptography and deep neural network,” in *IEEE Access*, vol. 8, pp. 25777–25788, 2020.

[14] A. Das, J. S. Wahi, M. Anand and Y. Rana, “Multi-image steganography using deep neural networks,” arXiv, vol. 2101.00350, 2021.

[15] B. Oktavianto, T. W. Purboyo and R. E. Saputra, “A proposed method for secure steganography on PNG image using spread spectrum method and modified encryption,” *International Journal of Applied Engineering Research*, vol. 12, no. 21, pp. 10570–10576, 2017.

[16] M. R. Islam, A. Siddiqa, Md. P. Uddin, A. K. Mandal and M. D. Hossain, “An efficient filtering based approach improving LSB image steganography using status bit along with AES cryptography,” *International Conference on Informatics, Electronics & Vision (ICIEV)*, vol. 3, pp. 1–6, 2014. https://doi.org/10.1109/ICIEV.2014.6850714.

[17] G. Wang, D. Hoiem and D. Forsyth, “Learning image similarity from flickr groups using fast kernel machines,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2177–2188, 2012.

[18] J. Zhai, S. Zhang, J. Chen and Q. He, “Autoencoder and its various variants,” in *IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, Miyazaki, Japan, 2018, pp. 415–419. https://doi.org/10.1109/SMC.2018.00080.

[19] A. Gionis, P. Indyk and R. Motwani, “Similarity search in high dimensions via hashing,” in *Proc. of the 25th Int. Conf. on Very Large Data Bases*, Edinburgh, Scotland, 1999, pp. 518–529.

[20] M. Yedroudj, F. Comby and M. Chaumont, “Yedrouj-net: An efficient cnn for spatial steganalysis,” in *IEEE Int. Conf. on Acoustics, Speech, & Signal Processing*, Calgary, Alberta, Canada, pp. 2092–2096, 2018.

[21] T. Silhan, S. Oehmcke and O. Kramer, “Evolution of stacked autoencoders,” in *IEEE Congress on Evolutionary Computation (CEC)*, Wellington, New Zealand, 2019, pp. 823–830. https://doi.org/10.1109/CEC.2019.8790182.

[22] M. Amara and A. Siad, “Elliptic curve cryptography and its applications,” in *Int. Workshop on Systems, Signal Processing and their Applications, WOSSPA*, Tipaza, Algeria, 2011, pp. 247–250. https://doi.org/10.1109/WOSSPA.2011.5931464.

[23] H. N. Bhonge, M. K. Ambat and B. R. Chandavarkar, “An experimental evaluation of SHA-512 for different modes of operation,” in *11th Int. Conf. on Computing, Communication and Networking Technologies (ICCCNT)*, Kharagpur, West Bengal, India, pp. 1–6, 2020. https://doi.org/10.1109/ICCCNT49239.2020.9225559.

[24] Y. Gong, S. Lazebnik, A. Gordo and F. Perronnin, “Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013.

[25] J. Fridrich, M. Goljan and R. Du, “Detecting lsb steganography in color and gray-scale images,” *IEEE Multimedia*, vol. 8, no. 4, pp. 22–28, 2001.

[26] J. Fridrich and M. Goljan, “Practical steganalysis of digital images: State of the art,” *Electronic Imaging*, vol. 4675, pp. 1–13, 2002.

[27] B. Boehm, “Stegexpose-A tool for detecting LSB steganography,” ArXiv, vol. abs/1410.6656, 2014.