Enhanced and Secured Hybrid Steganography Model for Hiding Large Data

by

Srabony Majumder
19101326
Mohon Dash Tanu
19301184
S. M. Fahim Faisal
19301186
Mumtahinah Rahman Sristy
19301169
Rigan Paul
19301129

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
May 2023

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Srabony Majumder 19101326	Mohon Dash Tanu 19301184
S. M. Fahim Faisal 19301186	Mumtahinah Rahman Sristy 19301169

Rigan Paul
19301129

Approval

The thesis/project titled "Increasing the Efficiency of steganography by using Convolutional Neural Network or CNN" submitted by

1. Srabony Majumder (19101326)
2. Mohon Dash Tanu (19301184)
3. S. M. Fahim Faisal (19301186)
4. Mumtahinah Rahman Sristy (19301169)
5. Rigan Paul (19301129)

Of Spring, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on May 25, 2023.

Examining Committee:

Supervisor:
(Member)

Dr. Muhammad Iqbal Hossain
Associate Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

Recent works on steganography basically are focused on various network layers and multiple data-hiding techniques. These researches lead to image manipulation, contortion and small-scale payload. This paper proposes a new dimension of the steganography model by merging the techniques of hiding texts and images. Instead of hiding only one type of data this proposed model is focusing on veiling two types of data (first text and then image) by masking one data (text) into another stego data (image) and then covering them both into a cover image. This model ensures a minimal peak-signal-to-noise ratio (PSNR) and no noise. This will be a model of suppressing more data in a stego container for one cover image using AES, MLSB and LSB respectively. This also should increase the payload capacity of embedding images through the steganographic system architecture with recognition. Here, we have used a pair of encoders and decoders for encryption and decryption to take two inputs and generate one hybrid output. The reverse of the encryption process will do the work for decrypting and decoding particular cryptic data. The AES (Advanced Encryption Standard) algorithm and LSB (Least Significant Bit) have been used to ensure our proposed model's accuracy and effectiveness. Moreover, using a modified 16-bit key is ensuring the safety of any confidential data of the user. It also accomplishes better execution by using ImageNet datasets. This model will expand and escalate the safety mechanism of steganography for concealing a larger amount of hybrid data. Furthermore, it can decide how many bits need to be changed in LSB depending on the length of the text that has to be encrypted in the host image. It is a hierarchy of two steganographic and one encrypt model. Last but not least, it makes certain of a distortion-free process where retrieving data after successfully concealing it is fruitful depending on higher accuracy, SSIM, PSNR and lower MSE.

Keywords: Steganography; Text; Image; AES; MLSB LSB; Stego; PSNR; SSIM; MSE; Encoder-Decoder; Encryption-Decryption Hybrid; ;

Acknowledgement

To begin with, I would want to give thanks to the Almighty Allah for allowing us to finish our thesis without significant disruptions. Secondly, we would like to thank our Supervisor, Dr. Muhammad Iqbal Hossain, for his assistance and counsel with our work. When we needed assistance, he was there for us. Thirdly, Mr. Md. Sabbir Ahmed Sir, a lecturer in the CSE department at BRAC University, helped us during the entire thesis process. Lastly, to our parents, without their unwavering support, it may not be feasible. We are currently on the verge of graduation thanks to their wonderful assistance and prayers.

Table of Contents

Declaration **i**

Approval **ii**

Abstract **iii**

Acknowledgment **iv**

Table of Contents **v**

List of Figures **vii**

List of Tables **1**

1 Introduction **2**

1.1 Motivation 2

1.2 Problem Statement 2

1.3 Objective and contributions 3

1.3.1 Why We Are Not Using CNN 4

1.3.2 Why LSB Over CNN 4

1.4 Thesis Structure 5

2 Background **6**

2.1 Literature Review 6

2.2 Algorithms 8

2.2.1 AES 8

2.2.2 LSB Text Steganography 9

2.2.3 LSB Image Steganography 10

2.3 Existing techniques 11

2.3.1 GAN–BASED methods 11

2.3.2 Techniques based on traditional methods 12

2.3.3 Techniques that are based on CNN 13

3 Dataset **16**

3.1 Overview of the Dataset: 16

3.1.1 ImageNet: 16

3.1.2 Cifar10: 17

4 Methodology 18
4.1 Working Plan 18
4.2 Proposed Model 21
4.3 Model description 22

5 Experimentation 28
5.1 CNN based Approach: 28
5.1.1 Encoder Architecture: 29
5.1.2 Decoder Architecture: 29
5.2 Alpha Channel-based Approach: 29
5.3 Experimentation on our hybrid model 32
5.4 Effects of Distortion : 37

6 Results 39

7 Conclusion and Future Work 43

Bibliography 45

List of Figures

2.1 Workflow of AES Algorithm 8
2.2 Workflow of LSB Text Steganography 9
2.3 Workflow of LSB Image Steganography 10

3.1 Validation images, extracted the 4096-dimensional fc7 CNN 17

4.1 Flowchart of Encryption Algorithm 19
4.2 Flowchart of Decryption Algorithm 20
4.3 Model Architecture 21
4.4 Byte to Hex conversion 22
4.5 LSB Stenography 23
4.6 Hiding text into image 23
4.7 Modified LSB 24
4.8 Image hiding process 25
4.9 Image into Image LSB 26
4.10 Image after LSB 26
4.11 Encryption process 27
4.12 Decryption Process 27

5.1 Cover Image and Payload Image and Encoded Image and Decoded payload image 28
5.2 Original images 30
5.3 Encrypted image 30
5.4 Decrypted image 31
5.5 Actual text and retrieved text 31
5.6 AES Encryption 32
5.7 MLSB (Hide data in image) 32
5.8 LSB (Hide image into image) 33
5.9 LSB (Reveal image from image) 34
5.10 MLSB (Retrieve data from image) 34
5.11 AES Decryption 35
5.12 Text encrypted image, Hidden image, Recovered image 1 35
5.13 Text encrypted image, Hidden image, Recovered image 2 36
5.14 Text encrypted image, Hidden image, Recovered image 3 36
5.15 Text encrypted image, Hidden image, Recovered image 4 36
5.16 Distorted image 1 38
5.17 Distorted image 2 38
5.18 Distorted image 3 38

6.1 Using same cover image 42

List of Tables

2.1 Techniques based on traditional methods 13
2.2 Techniques that are based on CNN 15

6.1 Comparing original image with the recovered image 40
6.2 Comparing Cover image with the stego-image 40
6.3 Analysis of changes in the encrypted image for the length of the hidden text data 41
6.4 Training and testing results of the model trained on the CIFAR10 dataset with a varying number of epochs. 41
6.5 COVERING BY SAME IMAGE 42

Chapter 1

Introduction

Steganography is a term of Greek origin that signifies “Hidden Written” or “Cover Writing”. It is a process of hiding information in an audio, video or image so that the information cannot be detected yet can be decrypted when necessary. Moreover, it is a process of hiding data for protection or for later use. Numerous forms of steganography exist. For example, text steganography, image steganography, audio steganography, video steganography, etc. Text steganography is where we can hide data into text files similarly image Steganography is a technique used to conceal data within an image. Steganography can be used for both good and bad purposes. Military, corporate, defense, or any official agencies use steganography for sending personal information and confidential messages. On the other hand, hackers use steganography as their hacking tool by sending encrypted data or malware to the victim’s computer for taking control of that device. Without enhancing features and proper security, this process cannot be fertile so it is big-time to look into these for upgrading the system of its working methods.

1.1 Motivation

Steganography has a long history and can be traced back to ancient Greece. This is important to hide data from enemies or from anyone or anything from whom it can be a threat. The digitalizing world is using it in almost every aspect such as cyber security, data security, etc. for its excellent resistance capacity. However, it also has some challenges that one can face during the process and our paper will discuss some tentative solutions for this with the fusion of some other techniques. The existing techniques are not able to hide larger data. They cannot give a distortion-free output or enough security. It is high time we know the values and use them accordingly.

1.2 Problem Statement

Although there are few existing working models of steganography, it is still not very popular for the drawbacks it has. The primary purpose of this technique is to hide secret data in a visually different one which is a promising thing to imply for confidential data that cannot be shared to all without proper verification. However, if it becomes faulty, distorted, insecure, unretrievable, unsafe, or insufficient that will not be much beneficial to use. We already know some of the existing models such as GNCNN, GAN, Adaptive, Spatial domain, Spread spectrum, and Frequency domain or Transform steganography, etc. which frequently use the framework of CNN, R-CNN using deep learning or machine learning to some extent. Those models claim to reduce the bit error rate, increase Peak-signal-to-noise-ratio (PSNR), decrease Mean-square-error (MSE), etc. yet these are not too safe to apply and also not reliable for many official and unofficial secret classified data like images or text messages. Again, it cannot hide hybrid data (for example, text + image) within a unified network nor can hide much more data without noise and distortion. On the contrary, those used models or techniques are able to hide less data not without deformation that causes more MSE rate. These are also prone to viruses and cyber-attacks, so, there are criteria for improvement in the steganographic methods using existing algorithms but with a little bit of change in some parts of the existing model and merging the processes.

Now, we want to propose a model that minimizes the noise ratio and gives a higher rate of accuracy and security. while a large amount of data is hidden and improves the existing model's architecture. Additionally, we are focusing on concealing hybrid data that may give the users the pleasure of hiding two types of data together if needed where one data may follow another or may not. This paper also talks about the security issues of steganography so we are applying a set of encryption-decryption algorithms with a specific security key or passcode to ensure the safety of private data. Hopefully, it will increase the use and popularity of steganography and help people with their confidential document sharing. For our architectural model, we will be applying AES, MLSB, and LSB for veiling secret data (text + image) in a cover data (image) involving a pair of encoders and decoders for a hybrid carrier image output.

1.3 Objective and contributions

The objective and contribution of our paper is to explore and develop new methods for combining text and image steganography techniques in order to enhance the security and efficiency of covert communication and evaluate the performance of these methods in terms of robustness, clarity, and capacity.

We are working on improving the steganographic methods so that it is not only more secure but also more usable. Today, steganography is not frequently used for a variety of reasons, including the fact that we can hide only a small amount of data and one type of data alone at a time. Additionally, we are unable to convert the hybrid image after it has been encrypted and the hidden data can be easily recovered without consulting with the owner meaning there is less security than intended. We are making an effort to conceal two types of data information more thoroughly and securely through AES and LSB. The obstacles that we are up against at this moment, the majority of the time we want to try to eliminate some of them. By utilizing the mentioned techniques, we will be able to conceal a greater quantity of dynamic text data and one image data together in one carrier data and do so in a manner that is more secure and collaborative.

Our purpose is to investigate and propose novel techniques for text and image hybrid steganography, aiming to achieve both high security and imperceptibility while minimizing any potential distortions in the cover data. The goal is to develop methods that can effectively embed both hidden text and image into cover image domains simultaneously, using complementary and synergistic approaches, and enhance their presentation based on security and visual quality To showcase the efficiency of the suggested method.

1.3.1 Why We Are Not Using CNN

For Text: Text steganography involves embedding hidden messages in a text while preserving the semantic and syntactic structure of the original text. This is a more complicated task than manipulating pixels in an image. CNNs are not well suited for this task because they fail to capture language-specific properties and nuances that are important for maintaining textual integrity when embedding hidden messages.
For Image: Although CNNs have achieved great success in various image processing tasks, they are not suitable for image steganography due to their limited ability to preserve cover image integrity while embedding hidden messages because CNNs introduce distortions into the cover image during the embedding process. This is because CNN's convolutional layers are designed to extract high-level features and patterns from images and may not be optimal for preserving the visual quality and integrity of cover images.

1.3.2 Why LSB Over CNN

Invisibility: Image LSB steganography can embed hidden messages in ways imperceptible to the human eye, without introducing significant distortions or artifacts into the cover image. In contrast, CNNs can distort the cover image during the embedding process, which can affect the visual quality and unrecognizability of hidden messages.
Security: Image LSB steganography is a highly secure method for embedding hidden messages, as it requires significant effort and computational resources to detect the presence of hidden messages in cover images. On the other hand, CNN-based approaches can be vulnerable to attacks that can compromise the security and reliability of steganography applications.
Capacity: Image LSB steganography can embed relatively large amounts of data into a single image with minimal distortion. Conversely, CNN-based approaches may limit the amount of data that can be embedded due to model complexity and the need to preserve the visual quality and integrity of cover images.
Overall, This method is optimized to minimize distortion of the cover image and render the hidden message to remain unseen to the human eye while maintaining a higher level of security and reliability in steganography applications.

1.4 Thesis Structure

Chapter 1 has our Introduction including motivation, problem statement, objective, contribution, and thesis structure.
Chapter 2 contains background research, a literature review, algorithms, and existing techniques.
Chapter 3 is the overview of our used datasets.
Chapter 4 discusses the methodology of our entire work along with the working plan, the proposed model, and the model description.
Chapter 5 highlights the whole experimentation throughout our thesis.
Chapter 6 focuses on the results, analytical objects, and comparison among the experimentation parameters.
Chapter 7 draws the conclusion of our paper and showcases some of the aspects of our future work.

Chapter 2

Background

2.1 Literature Review

This literature review aimed to demonstrate the relationship with other works on this topic Steganography using various sectors of it and identify the need for additional research and further study.

AES and LSB based steganography:

Singh, A., Singh, H. 2015 introduced an LSB technique for color images through the process of embedding information into the three planes of an RGB image, the quality of the image is improved while simultaneously achieving a substantial embedding capacity.[4] While hiding the image in the cover images, the technique puts focus on the color, as the stego-image has a high noise ratio, it is easier to detect. Replace the bits of the cover image in a specific order of 2:2:4, focusing on the least significant bits (LSB) of the three planes (Red, Green, and Blue). These LSBs are substituted with the corresponding bits from the message image. Furthermore, they compare the images based on PSNR And MSE counts, which show better results than traditional LSB techniques.

Sofyane Ladgham Chikouche and Noureddine Chikouche 2017 The study presented three approaches that are based on LSB techniques, where the bits of the message are inserted into the least significant bits of each pixel in the image.[8] In this survey they also bring two algorithms "The LZ77 algorithm" and" the Huffman algorithm" for the purpose of reducing the length of the data. In terms of encryption, they used the AES algorithm (Advanced Encryption Standard) to secure the message. Then they applied the given algorithm to reduce loss and create more memory. Then they hide the stego image in the cover image. The process gives a distortion-free image but the disadvantage of this technique is the algorithms are easy so detection is easy and the information storage requires a large image.

Priya Paresh Bandekar and Suguna G C 2018 A model was proposed to conceal confidential data within a cover image using the LSB technique for both encoding and decoding purposes.[12] Additionally, the Advanced Encryption Standard (AES) Algorithm was employed to ensure the security and protection of the hidden message or image. The survey conducted a comparison among various image formats, considering different text lengths or image sizes, in order to assess the performance.

CNN-Based Steganography: CNNs, developed by Yann LeCun and Yoshua Bengio in 1998, are specialized deep learning models designed specifically for processing two-dimensional grid data. From that time many proposed models came forward with different models.

Yinlong Qian, Jing Dong, Wei Wang, Tieniu Tan [3]in 2015 proposed a customized CNN model called Gaussian-Neuron CNN (GNCNN) which has three stages of architecture: local, shared weight, and pooling. Firstly, it takes raw images or pixels as input and processes the image in the processing layer. Secondly, in feature extraction, these pixels go through multiple convolutional layers. For the Gaussian function, the expression is: $f(x) = e^{-\frac{x^2}{\sigma^2}}$. Lastly, classification, it goes into fully connected layers and softmax layers rather than SVM or ensemble classifiers. Moreover, they conducted experiments on three advanced spatial domain steganographic algorithms: HUGO, WOW, and S-UNIWARD. After the experiment, comparing with the result of SVM and GNCNN (SRM) on BOSSbase, GNCNN gives lesser error for 3 spatial domains. But the probability of error is not less than 20%.

Ye, J., Ni, J., Yi, Y In 2017 came forward with a new function truncated Linear Unit (TLU) in addition with ReLU. The effective CNN can achieve better performance with TanH than only ReLU.[10] TLU works for embedding operations in steganography. It's not taking raw pixels rather than model noise residuals. Then the 10 layers of CNN give an attainable performance to achieve image steganalysis.

Ziegler, Couchot, J. F., Couturier, R., Guyeux, C., Salomon 2016 proposed a more efficient model with a higher payload. The first database they utilized is the widely recognized BOSS (Break Our Steganographic System) database,[5] comprising grayscale images with dimensions of 512 × 512. The second database employed is the Raise database. In their experiments, they specifically tested three steganographic tools: WOW, HUGO, and J-UNIWARD. The former two operate in the spatial domain, while the latter operates in the frequency (JPEG) domain. Using this algorithmWOW 0.4, HUGO 0.4, and J-UNIWARD 0.4 got over 95% of accuracyWOW, HUGO, and J-UNIWARD .but in some versions of this tools got around 75% accuracy.

Li et al. proposed a CNN architecture that incorporates diverse activation modules (DAMs) and three parallel subnets to improve the detection of embedding artifacts.[13] Their model builds upon Xu's existing architecture but introduces modifications to include DAMs. The subnets within the architecture are independently pre-trained using a fully connected layer and a Softmax function to establish connections between the container and stego. These subnets are structured based on the DAM framework, which consists of a convolution layer followed by parallel activations of ReLU, Sigmoid, and TanH.[7] As a result, three output feature maps are generated and concatenated for further processing. Two DAM modules are present in the architecture, located in group 2 and group 3 respectively, to process information in a diverse manner. The experiments were conducted using the BOSSbasev1.01 database, and the evaluation involved the S-UNIWARD, HILL, and

CMD-HILL steganographic algorithms.

2.2 Algorithms

The algorithms we have used for the experiment purpose are AES(Advanced Encryption Standard), LSB Text Steganography, and LSB Image Steganography.

2.2.1 AES

The Advanced Encryption Standard (AES), also known as Rijndael, is a specification for encrypting electronic data that was established in 2001. AES is based on the substitution-permutation network design principle and is highly efficient in both software and hardware implementations. It is a symmetric block cipher that operates on 128-bit blocks and supports key lengths of 128 bits, 192 bits, and 256 bits. The algorithm encrypts these individual blocks using the selected key length, and the resulting encrypted segments are combined to form the ciphertext. During the initial phase, a single key is required; later, multiple keys are required for individual cycles. This algorithm performs operations on bytes rather than bits. Consequently, In the AES encryption procedure, the 128-bit block size is considered 16 bytes. The number of encryption rounds to be performed depends on the length of the encryption key. For a 128-bit key, ten rounds are executed. For a 192-bit key, twelve rounds are performed, and for a 256-bit key, fourteen rounds are carried out. Encrypting the information that is transmitted in the image and encrypting the image containing the data with the AES algorithm provides security. This method provides a double layer of data security. The workflow of the AES Algorithm is represented in Figure 2.1 below.

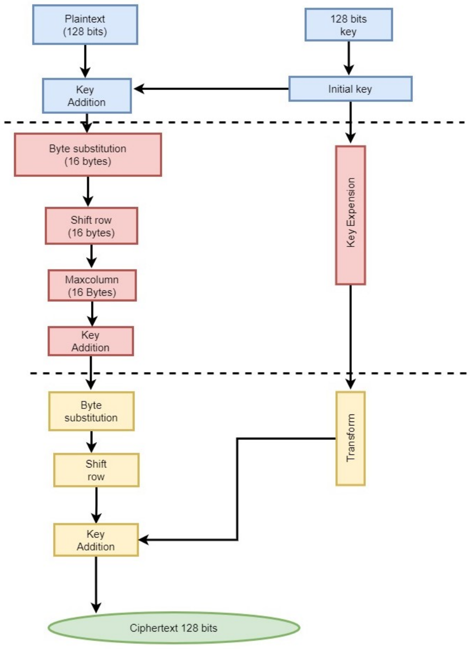


Figure 2.1: Workflow of AES Algorithm

2.2.2 LSB Text Steganography

LSB Text Steganography is a method for concealing secret communications in text files. Similar to LSB Image Steganography, this method replaces the least significant bit of each character in a text file with a bit of hidden information. Characters are represented in digital text files by a series of bits that correspond to their ASCII code or Unicode code point. The ASCII binary code for the letter "A", for example, is 01000001. Each of these bits' least significant bit (LSB) can be replaced with a piece of confidential information. To conceal a message within a text file using LSB Text Steganography, each character's least significant bit is replaced with a portion of the message. The message is typically encoded in binary, and the embedding procedure may involve additional security-enhancing techniques such as encryption or compression. To extract the concealed message, the recipient must know the precise location of the modified LSBs and apply a decoding algorithm to the modified text file in order to extract the message. The Workflow of LSB Text Steganography is represented in Figure 2.2 below.

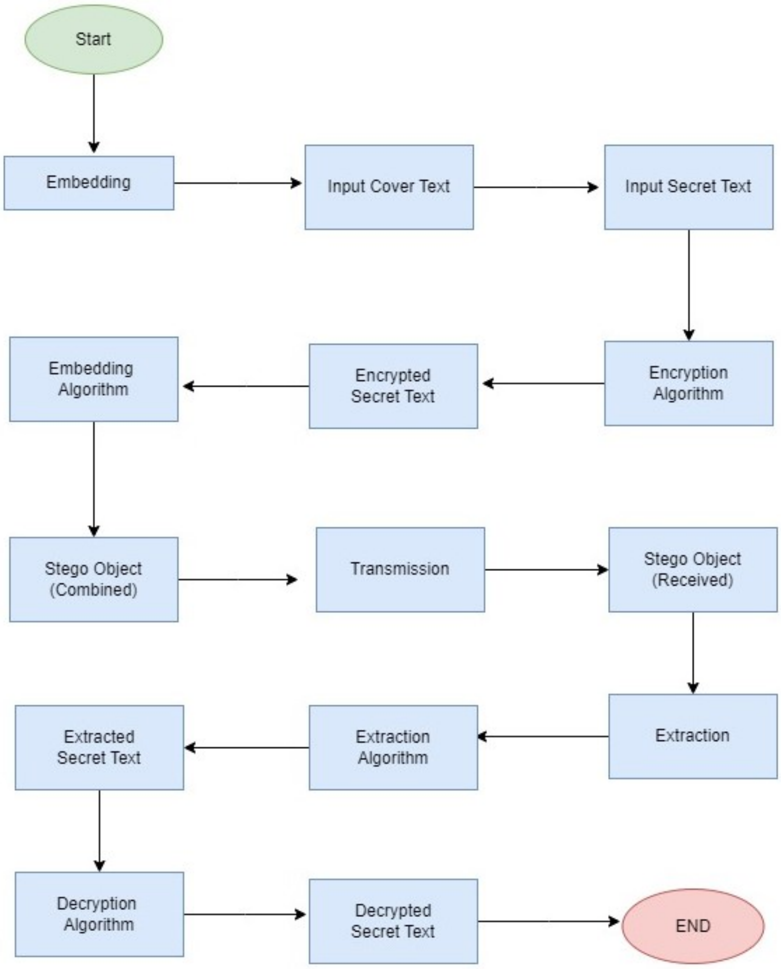


Figure 2.2: Workflow of LSB Text Steganography

2.2.3 LSB Image Steganography

The steganographic technique of LSB (Least Significant Bit) Image Steganography can be used to conceal information in digital photographs. With this technique, secret information is substituted for the image's LSB (least significant bit). The color information in a digital image is stored as a binary code for each pixel. For example, in a 24-bit color image, 8 bits are devoted to each of the pixel's red, green, and blue values. The bit with the smallest numerical value in each of them is called the LSB. The least significant bit (LSB) of each pixel value can be substituted with a secret identifier to hide information within an image without significantly altering its aesthetic aspect. LSB steganography can be used with digital photographs saved as JPEG, BMP, or PNG files. Compressed images don't fare as well with the method due to the fact that compression methods may alter the LSBs, resulting in lost information. Both the image quality and the sophistication of the detecting methods affect how successfully and safely it functions. The Workflow of LSB Image Steganography is represented in Figure 2.3 below.

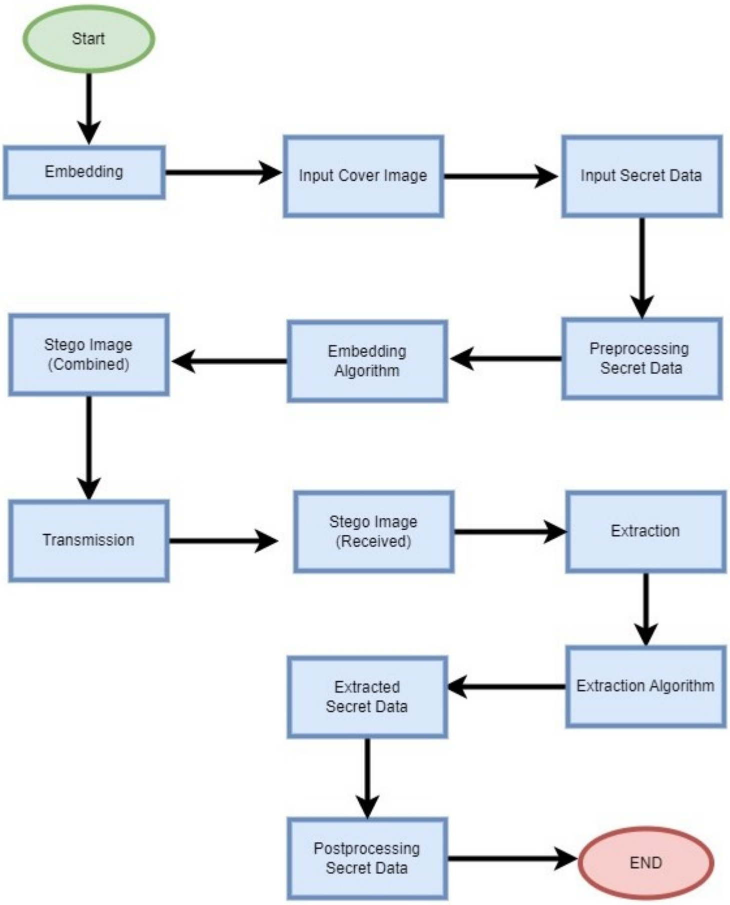


Figure 2.3: Workflow of LSB Image Steganography

2.3 Existing techniques

2.3.1 GAN–BASED methods

Deep convolutional neural networks, also known as CNNs, are a subclass of adversarial networks in general (GAN). A GAN will use game theory to train a generative model utilizing an adversarial learning strategy in order to complete image-generating tasks.[15] Game theory will be used to train the model. The generating networks and the discriminator networks of a GAN architecture compete with one another to produce the best possible image. The data are entered into the generator model, and the result is an image that is very similar to the one that was used as the input.[17] The results of the analysis carried out by the discriminator networks are used to assign a status of either false or true to the images that have been generated. The two networks are trained using a framework where the generator model aims to replicate the input data as accurately as possible. While simultaneously generating as little noise as is practically achievable.[16] This is accomplished through the training process. In order to achieve the highest possible degree of precision, this step is taken. Existing techniques for image steganography that make use of a GAN architecture can be categorized into the following five categories: three network-based GAN models, cycle-GAN-based architectures, sender-receiver GAN architectures and two other categories yet to be specified. coverless models; and models in which the cover image is generated randomly rather than being provided as input.[15] Every one of them can be further broken down into a plethora of other categories. The generator and the discriminator are the two primary components that make up a GAN model.

These two aspects are the model's most fundamental building blocks. The steganalyzer is a new network that is added by some approaches for the process of image steganography. In the following section, the primary purposes of each of these three components will be discussed:

- The model, represented by the letter G, is designed for generating stego images by utilizing the cover image and a random message as inputs.

- A discriminator model denoted by the letter D, which can determine if the image that is produced by the generator is genuine or not.

- A steganalyzer, denoted by the letter S, to assess whether or not the image being analyzed contains confidential or secret data.

Architecture	Dataset	Advantages	Disadvantages
Alice, Bob, Eve	BOSSbase, and celebs	Embedding don't require you to know anything about the domain.	Images are not used; instead, messages are sent. Grid search for choosing an embedding scheme takes time.
DCGAN	celebA and BOSSbase and MSE	Game theory is used to make decisions.	Steganalyzer is used to figure out the probability by itself, which adds to the cost and time of computing.
GAN	CelebA	There is no limit to the number of cover images that can be made.	Both the generator and the distinguisher are shared. The pictures that have been made are not real.
DCGAN	CelebA	Making images that are more real Extremely safe	It delivers probabilistic rather than confidential information. There is no information regarding how to acquire the confidential data.

2.3.2 Techniques based on traditional methods

The Least Significant Bits (LSB) substitution method is the one that is most commonly used while carrying out image steganography.[11] Images typically have a greater pixel quality, but not all of the pixels are utilized in the final product. The least significant bit (LSB) technique is predicated on the idea that altering a small number of pixel values will not result in observable shifts in the image. The confidential information is then encoded using a binary system.[17] Scanning the cover image makes it possible to discover the bits in the noisy area that is the least significant. The cover image's least significant bits (LSBs) are then swapped out from the secret image for binary ones and zeros. Caution is required while utilizing the alternative method, as producing an excessive amount of visible adjustments to the cover photo may occur from doing so.[2] These alterations may betray the existence of confidential information.

Dataset	Metrics	Advantages	Disadvantages
Lena	PSNR	Reduced calculation time. The image is coded.	It's not safe.
Lena and Baboon	PSNR and MSE	The computation time are lowered. The artwork itself has a hidden message. Acceptable image formats include any	Security is an issue when compared to CNN technologies.
RGB Image	PSNR and Time	The computation time is lowered. It embeds data easily. Steganography and steganography analysis are independent of one another.	It is unsafe. Text is used to transmit confidential information.

Table 2.1: Techniques based on traditional methods

2.3.3 Techniques that are based on CNN

When steganography starts using CNN models, the encoder-decoder architecture goes through a significant transformation. The encoder takes in two images, the one that serves as the cover and the one that serves as the secret, and then it mixes them to produce the stego image.[1] The steganographic image is input into the decoder, which then produces the hidden image that the steganographic image conceals. Different strategies with a variety of structures have been implemented, but the fundamental idea has remained the same. Alterations in the convolutional layer and the pooling layer, in addition to shifts in the manner in which the input cover picture and the hidden image are linked through a variety of processes, are to be anticipated.

The specific parameters such as the number of filters, stages, filter size, activation function, and loss function can vary across different techniques and methods. These variations depend on the specific requirements and goals of each approach.[9] It is necessary for each pixel of the hidden image to be dispersed uniformly across the cover image. The convolution operation is a sort of linear operation that represents the degree to which two shifted functions overlap with one another.

Convolutional networks are simplified versions of neural networks that have at least one layer and instead of doing matrix multiplication, they perform matrix convolution. Utilizing a CNN-based architecture for encoding and decoding gives a number of benefits, including the following:

•CNN automatically pulls out visual details.

•CNN ”downsamples” an image by using information from nearby pixels, first through convolution and then through a prediction layer at the end.

•CNN is more accurate and works well.

Using a deep neural network, in this example a CNN, one can gain a reasonable notion of the patterns of natural photos. The network will be able to recognize whether regions are redundant, enabling the concealment of additional pixels in particular regions.[6] It is possible to increase the amount of hidden data by conserving space in unnecessary regions. Because the network's structure and weights are randomizable, it conceals data that is unavailable to anyone who does not possess the weights.

Architecture	Dataset	Advantages	Disadvantages
Encoder decoder with SCR	ImageNet	Highly secure and dependable	The used loss is not optimal. In black or white areas, noise may be visible.
Encoder decoder with VGGbase	COCO and wikiart.org	The used loss is not optimal. In black or white areas, noise may be visible.	A large quantity of pictures is necessary for computational purposes.
CNN	ImageNet and Holiday	The artwork itself has a hidden message. The most fundamental architecture is selected. A novel error backpropagation function is implemented to accelerate training.	However, the image size is only 64 by 64 pixels, which is rather small. The input photos are simply concatenated together.
Encoder decoder	ImageNet	The artwork itself has a hidden message.	However, the image size is only 64 by 64 pixels, which is rather small.

Table 2.2: Techniques that are based on CNN

Chapter 3

Dataset

3.1 Overview of the Dataset:

We conducted experiments on the following datasets:

3.1.1 ImageNet:

ImageNet is a picture dataset that is structured in a hierarchy similar to that of WordNet. ImageNet is known as "ImageNet." A "synonym set" or "synset" is the name given to each significant concept that may be articulated by WordNet using several words or word combinations. WordNet has more than one hundred thousand synsets, with the vast majority being nouns (over 80,000 to be exact). The ImageNet collection includes 14,197,122 images that have been tagged in accordance with the WordNet hierarchy. This competition serves as a baseline for image classification and object recognition. The dataset that has been made available to the public includes a collection of manually annotated training images. In addition, a collection of test photographs has been made available, excluding any manual annotations. Annotations in the ILSVRC can be divided into two classes: There are two types of image annotation: (1) image-level annotation, which provides binary status denoting either "present" or "absent" of an object class in the image, such as "there are cats in this image" or "there is no lion in this image," and (2) object-level annotation, which pinpointed class label and image-bounding box around a single object instance. Validation images are shown in Figure 3.1.

- 21841 total WordNet synsets are not empty.
- Total number of photographs: 14197122
- There are 1,034,908 photos annotated with the bounding box
- 1000 synsets contain SIFT characteristics.
- 1.2 million photographs include SIFT characteristics.

Figure 3.1: Validation images, extracted the 4096-dimensional fc7 CNN

3.1.2 Cifar10:

The CIFAR-10 dataset is a well-known collection of images used in the field of machine learning. It comprises 60,000 images, each with a size of 32x32 pixels and in RGB color format. The images are sorted into ten different classes, each with 6,000 images: Birds, Trucks, Dogs, Horses, Airplanes, Ships, Cats, Deer, Frogs, and Automobiles.

This dataset is commonly used as a benchmark for evaluating image classification tasks. It has been extensively employed to train and evaluate various machine learning models, particularly deep convolutional neural networks. Although the images in the dataset are relatively small and low-resolution, this makes training quicker than on larger datasets, while also posing a challenge for image classification.

In the CIFAR-10 dataset, each image is composed of three color channels: green, red, and blue. These channels are represented by 8-bit values ranging from 0 to 255. The dataset is divided into a training set, which contains 50,000 images, and a test set, which contains 10,000 images. The images are evenly distributed across the ten categories to ensure a balanced representation and prevent bias during the training process.

While the CIFAR-10 dataset has been preprocessed to maintain pixel values between 0 and 1, the dataset has not been normalized or standardized, so different pixel values may have different ranges. Researchers use the dataset for transfer learning, domain adaptation, and active learning studies, among other things.

Chapter 4

Methodology

4.1 Working Plan

The thesis work plan consists of several important stages. In the methodology design stage, the modified least significant bit (LSB) technique will be selected for text and image hiding, along with the Advanced Encryption Standard (AES) algorithm for data encryption. The encoding and decoding steps, as well as the parameters for LSB steganography, will be determined. The proposed model will then be implemented and tested through software development, including the AES encryption algorithm and modified LSB technique. The model's performance will be evaluated by testing and verification, assessing its data hiding capacity, security level, and impact on image distortion. Flowchart of Encryption Algorithm Decryption Algorithm is represented in figure 4.1 and figure 4.2 below.

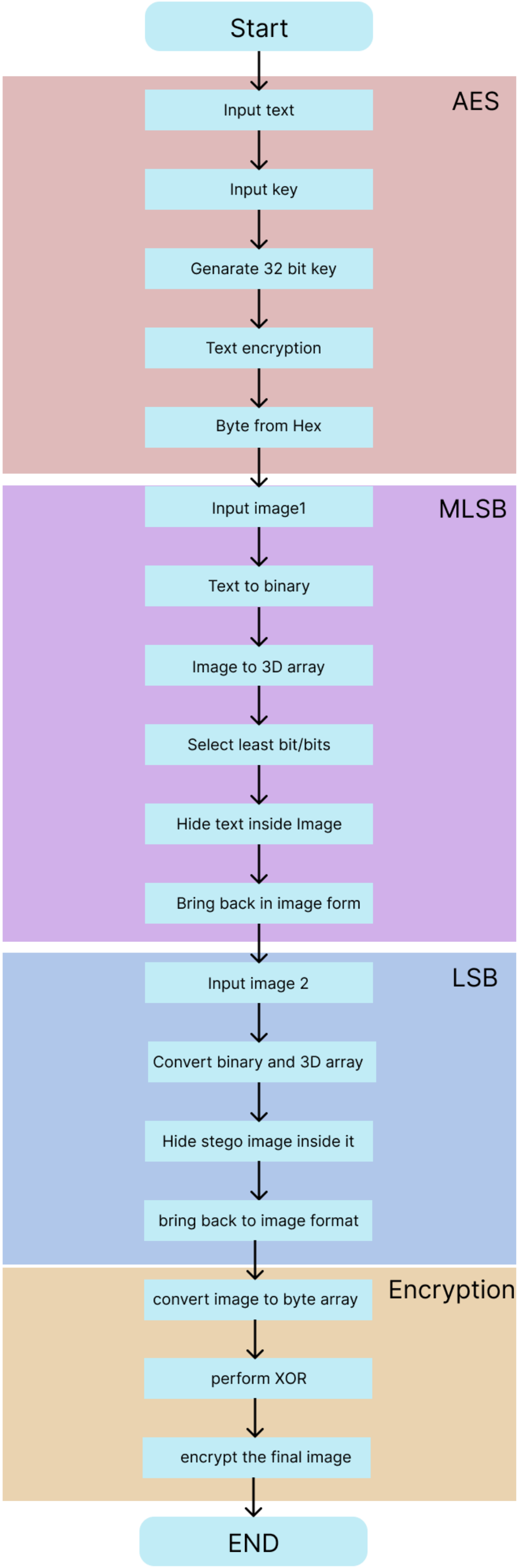


Figure 4.1: Flowchart of Encryption Algorithm

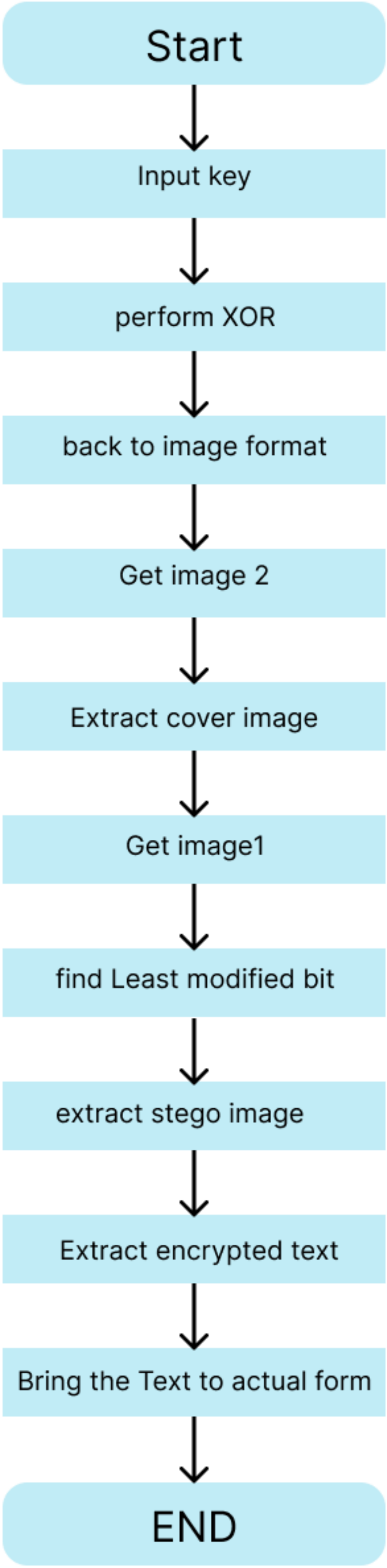


Figure 4.2: Flowchart of Decryption Algorithm

4.2 Proposed Model

Our proposed technique has focused on implementing the modified least significant bit (LSB) with text and the least significant bit (LSB) with the image. Besides that, to enhance the security level of data hiding we used an encryption algorithm advanced encryption standard (AES). Our model builds up in two major manners, we encode the data and then decode it by passing some layers. Imagine it as two layers that make it easy to understand as well as easy to implement The encoder model has 4 sub-steps. In the first step, we will encode the text by using AES encryption and make the text unreadable without a key. The unreadable text or in hex formatted text will be hidden inside the image called stego image which is the second step. For the third step, the image will be hidden in another image. Therefore the stego images distortion will be untraceable. and lastly, the image will be encrypted by using the previous key. For decoding the process is quite similar to the encoder in reverse, firstly we will decrypt the image and make it an image format. secondly, we will retrieve the stego image hidden in it previously. It is very important to retrieve the exact same image we hide inside the image otherwise it will be impossible to retrieve the exact text from it. Thirdly the image we retrieved, we will extract the encoded text from it.lastly, we will the data will be changed into a human-readable format.In figure 4.3 the encryption and decryption process is demonstrated in step by step process.

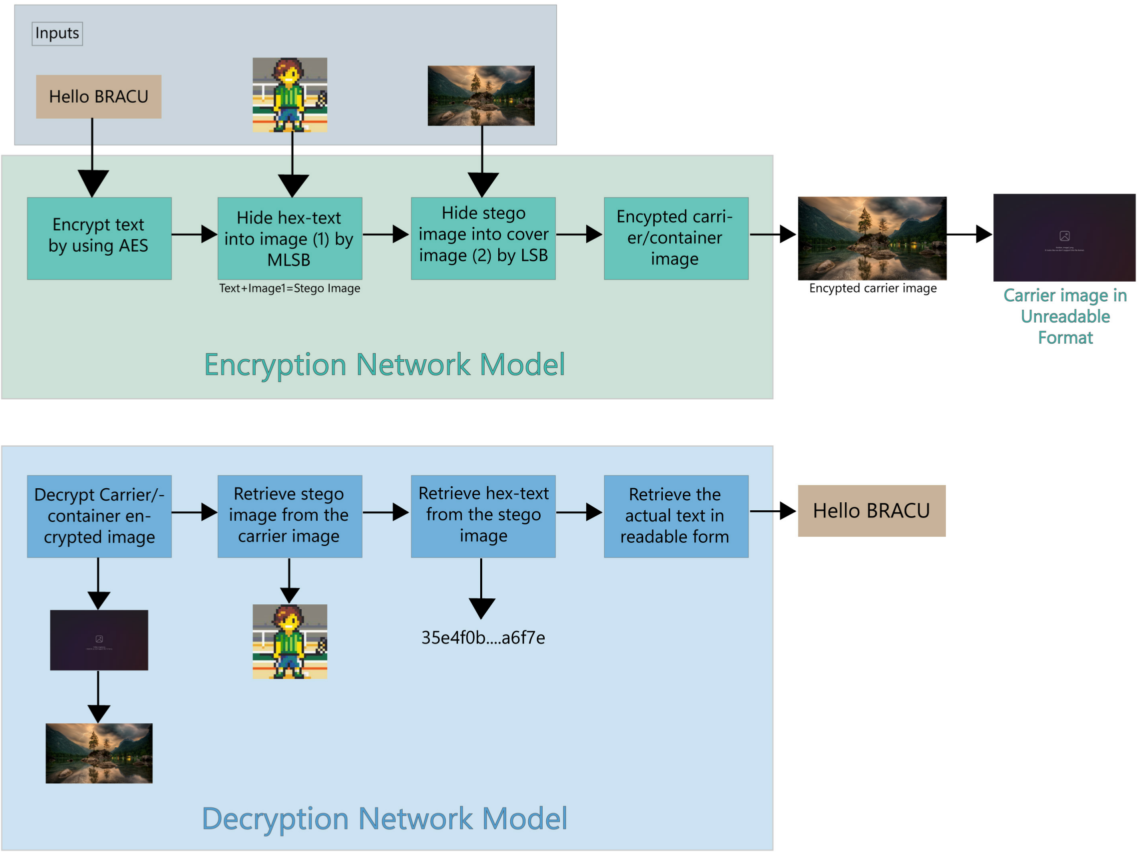
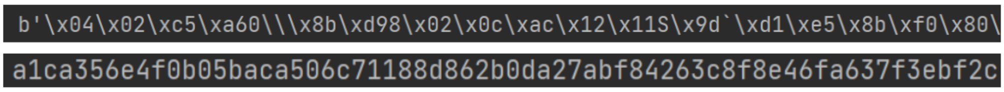

Inputs
Hello BRACU
Encrypt text by using AES
Hide hex-text into image (1) by MLSB
Text+Image1=Stego Image
Hide stego image into cover image (2) by LSB
Encypted carri-er/container image
Encypted carrier image
Carrier image in Unreadable Format
Encryption Network Model
Decrypt Carrier/-container en-crypted image
Retrieve stego image from the carrier image
Retrieve hex-text from the stego image
Retrieve the actual text in readable form
Hello BRACU
35e4f0b....a6f7e
Decryption Network Model

Figure 4.3: Model Architecture

4.3 Model description

In the beginning, we will encode the text by using the AES algorithm, AES stands for advanced encryption standard which is the widely used most secure process. Firstly it takes text from the user, the AES takes usually 16-32 character input as key, we modify it as it can take any number of inputs and later make the key 32-bit. After that, the encryption takes place and the output came which is in byte format. We converted the output to hex format. Otherwise in byte format, it creates an error while we try to retrieve the text from the stego image. Figure 4.4 shows how the text looks in byte format and we have converted it into hex format. Then we hide that hex text inside the image.



```
b'\x04\x02\xc5\xa60\\\x8b\xd98\x02\x0c\xac\x12\x11S\x9d`\xd1\xe5\x8b\xf0\x80\
a1ca356e4f0b05baca506c71188d862b0da27abf84263c8f8e46fa637f3ebf2c
```

Figure 4.4: Byte to Hex conversion

Secondly, we hide the encrypted text into an image. It is important to get the text as it is, here LSB is one of the efficient techniques that can retrieve the data accurately. LSB stands for the least significant bit. Where the text is first converted into binary data character by character and the image is converted into a 3D array format. Later hide the binary bits into the least significant bit of the 3D array, this can be shown in Figure 4.5. The dimension decides how much data can be hidden in the image. For example, the dimension of an image is 220*220, hence the bits that can be hidden by using LSB steganography are 220*220=48400, the maximum number of massage bits that can be hidden in the entire image is 48400*3=145200.since 8 bits make up 1 byte so the character can be hidden is 145200/8=18150. But we modify the LSB and made it dynamic. For example, inside a 220*220-dimensional image, the maximum byte that can be hidden by using 1 bit LSB is 18150, if we want to hide more data than that the mode changes automatically and it alters at least 2 significant bits where it was using 1 significant bit for hiding data previously. If the data is bigger than 18150 characters it automatically shifts to 2 or 3 significant bits which can be seen in Figure 4.7.The most difficult problem we was facing while implementing is figure out length of the text because while retrieving we need to know the length of the text to understand how many least significant bits are altered,we had to store or take input the length of the key somehow.Asking the length of the hidden text from user was a discomfort,so we hide the length of the text in first 18 bit of the image,and while retrieving these fixed bit just contains the length of the text and rest of the bits contains data,therefore the problem of hiding and retrieving large data was solved But the image gets distorted if we hide a large amount of data. Our next step cover up the distortion. Figure 4.6 demonstrates the text hiding in the image process, we have used an image size of 8.41Kb after hiding the text the size of the images became 51.1Kb.

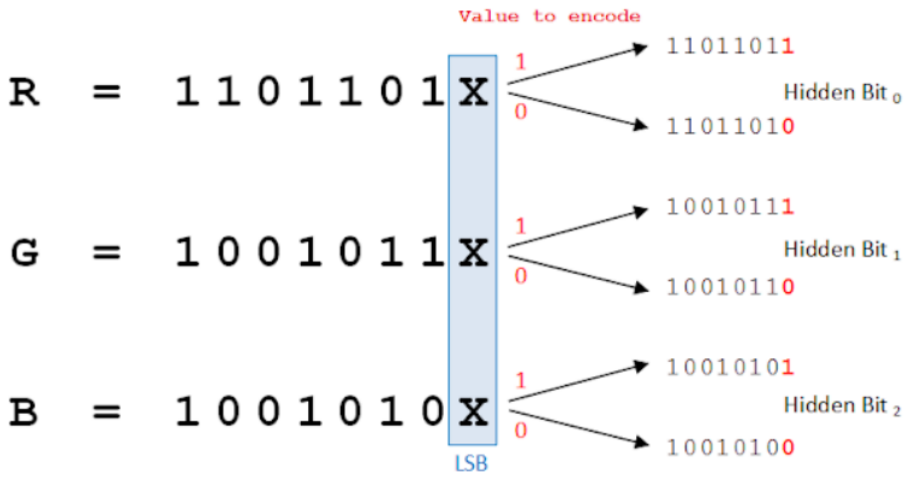


Figure 4.5: LSB Stenography

Figure 4.6: Hiding text into image

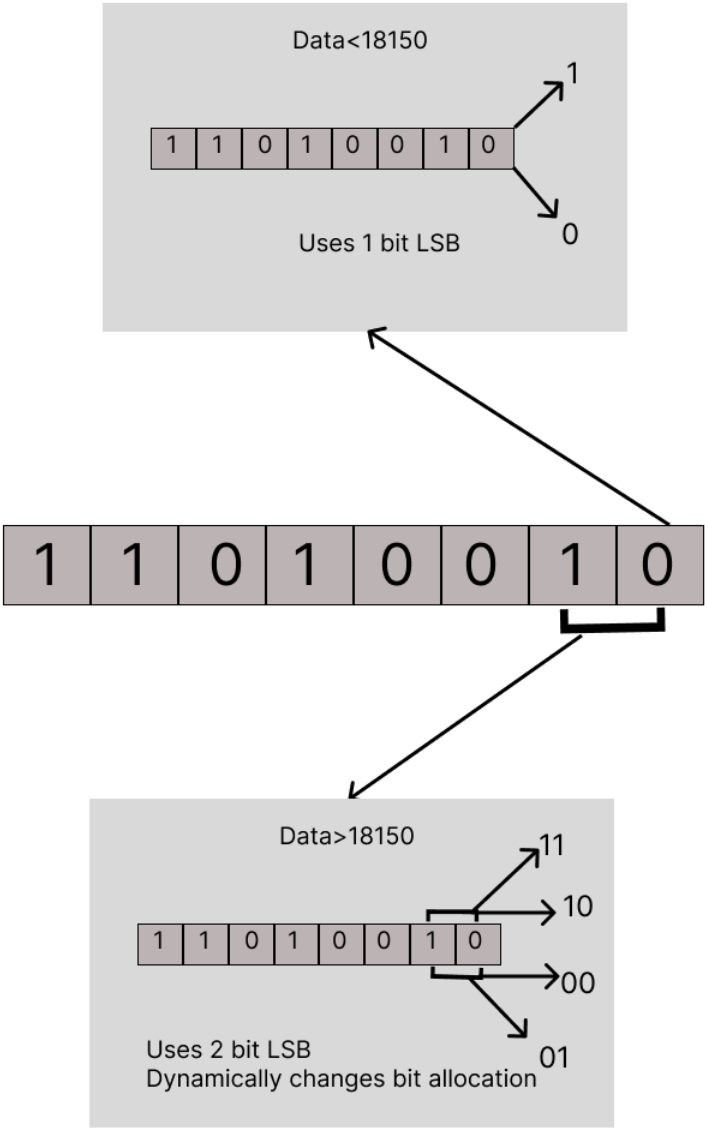
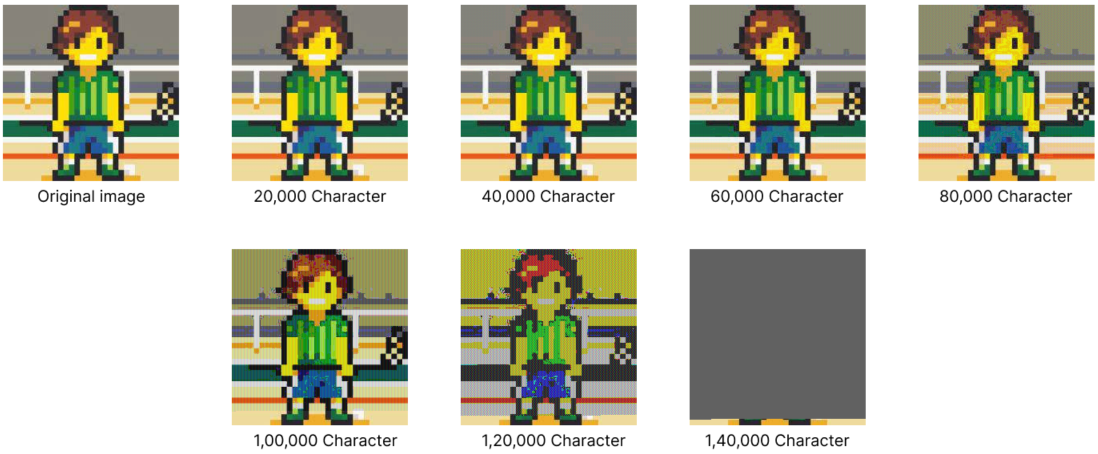


Figure 4.7: Modified LSB

Figure 4.8 shows how modified LSB also deals with a large number of characters. We have hidden up to 1,20,000 characters after the image gets totally distorted but the text is still retrievable. The figure also shows the number of characters and the distortion of the image side by side. The more data we want to hide it do not reject the data but gets distorted. Our next step is to hide the distortion of the stego image.



Original image

20,000 Character

40,000 Character

60,000 Character

80,000 Character

1,00,000 Character

1,20,000 Character

1,40,000 Character

Figure 4.8: Image hiding process

Thirdly, We hide an image into another image by using LSB steganography. For this step, we took the stego image and hide the image into another image. The process is to take the stego image and convert it into 8-bit binary format and convert the carrier image into a 3D array after that binary format and lastly hide the stego image into the carrier image. Figure 4.9 shows the hiding process in short. Hence the Stego image is distorted or not cannot be understood by seeing the final output and we can hide as much data as we want. Figure 4.10 is how we hide the image into another image. The size of the carrier image was 559kb after hiding the stego image the size became 2.82MB.

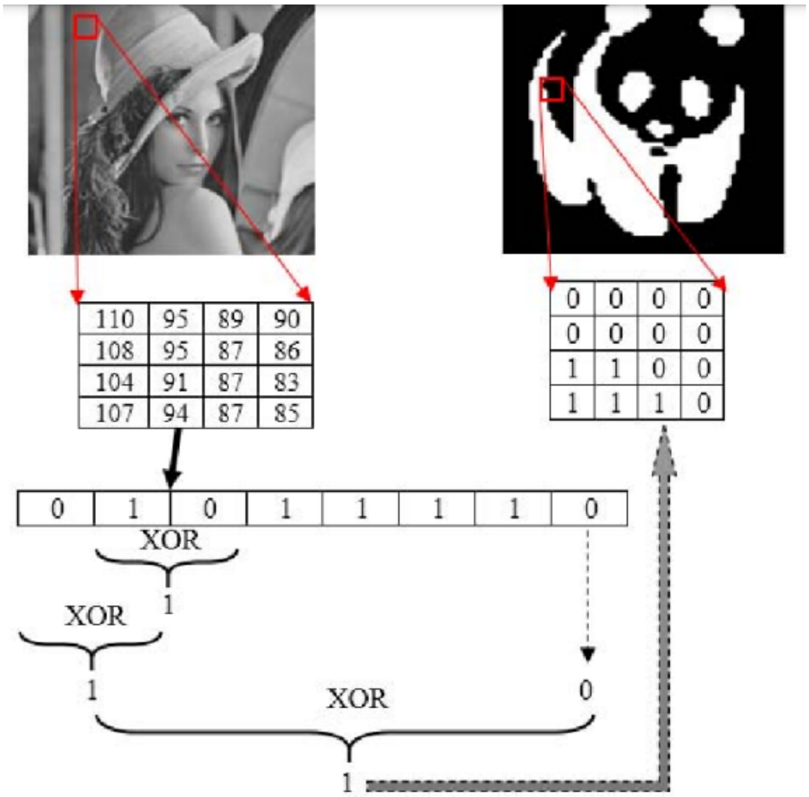


Figure 4.9: Image into Image LSB

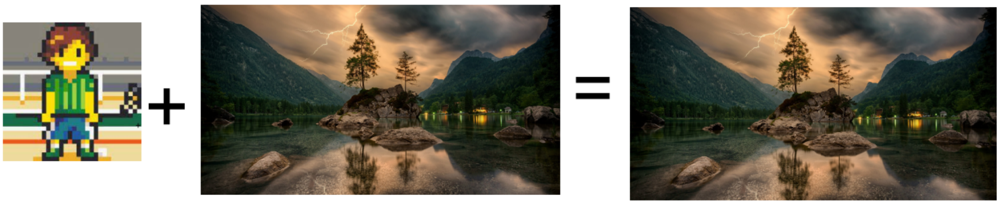
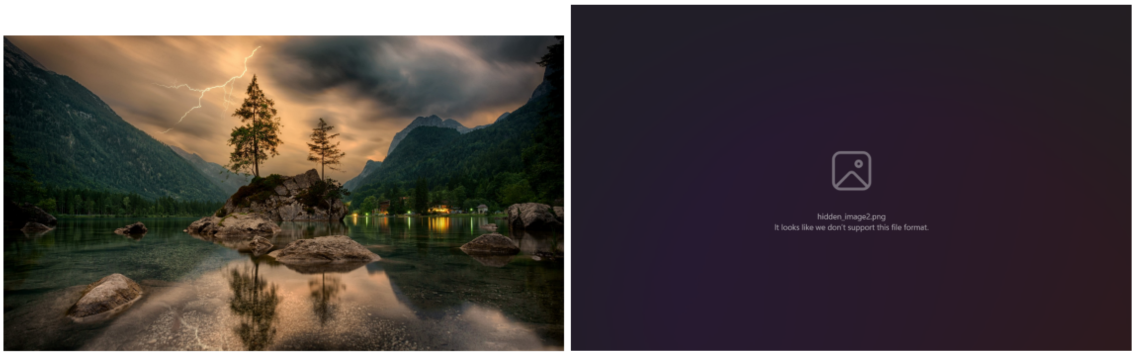


Figure 4.10: Image after LSB

The last step is quite simple, here firstly select the final image and convert it into a byte array hence the image will be totally converted into numeric form, Then we perform an XOR operation with the key, and the data inside the array be changed and unable to access. Figure 4.11(a) shows how the image looks like before encryption and 4.11(b) shows the image has been converted into an unreadable format



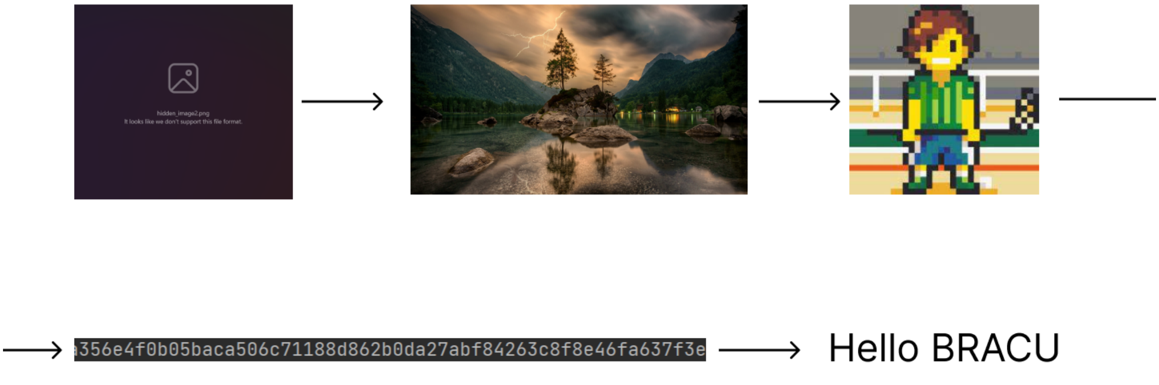
hidden_image2.png
It looks like we don't support this file format.

(a) Before encryption

(b) After encryption

Figure 4.11: Encryption process

For decryption we reversed the whole process, firstly we decrypt the unreadable image to an readable image by using the key. Hence the key is taken and no other input is needed for decryption. After retrieving the image from our next step is to retrieve the image from the carrier image. We used the LSB technique to hide the image hence the data of the stego image is hidden inside it. We reversed the process to retrieve the image. Later extract text from the image we hide inside it. And lastly, we reverse the AES encryption to extract the actual data from an encrypted text. Figure 4.12 shows the whole decryption process step by step.



hidden_image2.png
It looks like we don't support this file format.

```
a356e4f0b05baca506c71188d862b0da27abf84263c8f8e46fa637f3e
```

Hello BRACU

Figure 4.12: Decryption Process

Chapter 5

Experimentation

Experimental results are given in this section to highlight the performance of our proposed model and other approaches applied to our model before reaching the expected results.

5.1 CNN based Approach:

By using Convolutional Neural Networks(CNN) we will train a pair of encoders and decoders so that they can create a hybrid image from the input. We will take two inputs one is the host and another one is the image we want to hide. In our approach, we leverage the observation that CNN layers have the ability to learn a hierarchy of image features, starting from low-level generic features and progressing to high-level domain-specific features. Based on this insight, our encoder is designed to identify and capture specific features from the cover image, concealing the details from the payload images. On the other hand, the decoder is trained to extract and separate these hidden features from the combined "hybrid" image.[14] This process allows us to effectively hide and retrieve the desired information while preserving the overall integrity of the image. The encoder will take two images: the first one is the host and the second one is the guest by using CNN layers creates a new image that is a hybrid of both images but the guest image is hidden under the host image. The main purpose of the encoder is to create a hybrid image from both and later on the decoder image will take the hybrid image as input and finds out the guest image from it. Figure 5.1 shows the cover image before and after encryption and before and after retrieving the stego image

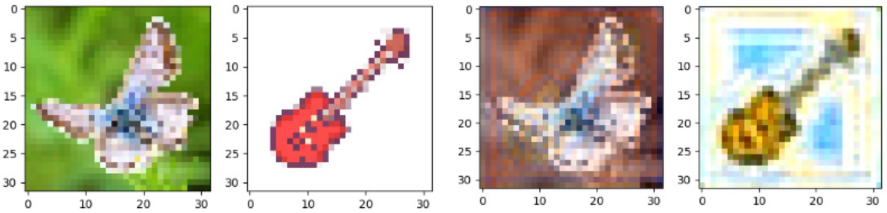


Figure 5.1: Cover Image and Payload Image and Encoded Image and Decoded payload image

5.1.1 Encoder Architecture:

The encoder has two branches for two inputs. For host a host branch and for guest a guest branch. After receiving input the input goes through of a couple of ReLU and convolution layers. Suppose the host image is Ih and the guest image is Ig. To hybrid, both images or merge images the encoder finds the layer and concatenates the extracted feature map lastly alternates the corresponding feature maps. This procedure is repeated several times and using ReLU layers we got the desired image. For encoding we used a 32x32 images container and 32x32 payload. The architecture contains 10 convolution layers with 3x3-sized kernels for payload. Besides for container, the model goes through 9 convolution layers. after every two layers, a concatenation happens with the payload layers to hide the input image more efficiently. We think ReLu is a better activation function than softmax for getting better performance.

5.1.2 Decoder Architecture:

The decoder will input the hybrid image and go through a couple of sequences of ReLU and convolution layers and recover the image from it. For decoding we decoded 32x32-sized image using 10 convolution layers. Unlike encoding, here also used the ReLu activation function rather than softmax. Another technique that we have given a try is Multi-image steganography using deep learning. It is a cutting-edge technique that utilizes complex neural networks to hide confidential data within multiple cover images. This method employs a sophisticated network that is trained to create stego images that comprise the hidden data. The network takes several cover images and the secret data as input and produces a set of stego images as output. During the training process, the network adapts to the characteristics of diverse cover images and various types of secret data. This method adds an extra layer of security because it makes the process of hiding data much more complicated. This technique did not work out properly because this technique necessitates a massive amount of training data to learn how to embed the secret data effectively. Obtaining such a vast amount of training data had become a challenging task for us. Furthermore, some of the cover images are of low quality or have poor resolution, so the steganography technique may not be effective in hiding the secret data.

5.2 Alpha Channel-based Approach:

The Alpha channel, an ingenious technique employed in digital imaging and computer graphics, renders transparent and merges images or elements to forge seamless compositions. An image file typically comprises three color components - Red, Green, and Blue (RGB). However, the Alpha channel transcends these limitations by adding a fourth component to each pixel, which embodies the pixel's opacity or transparency. The Alpha channel operates on a scale that assigns a numerical value between 0 and 255 to each pixel. A value of 0 designates that the pixel is entirely transparent, whereas a value of 255 signifies that the pixel is wholly opaque. Values between 0 and 255 offer a broad spectrum of transparency levels. This technique was a generous approach for us to hide one image from another, but we face problems with it. The image we tried to hide after retrieving was black and white and the carrier image was distorted so much that it was visible to the human eye. Figure 5.2 are the actual images before using the Alpha channel and 5.3 images after encryption, there is a visible distortion in the image.

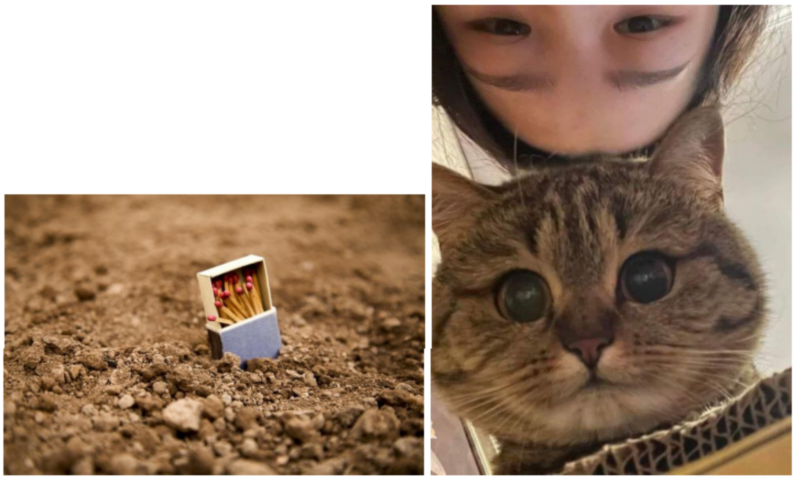


Figure 5.2: Original images

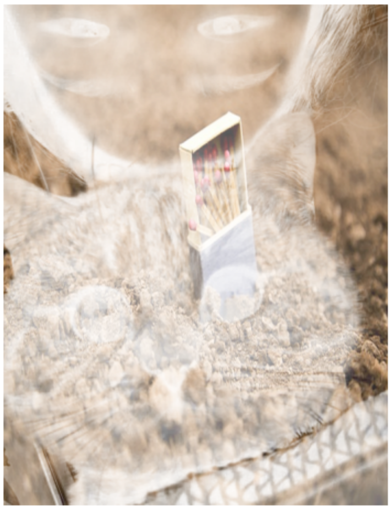


Figure 5.3: Encrypted image

It can be easily traced that there is another image hidden in it. Therefore we didn't use that method. Figure 5.4, the image retrieved is in black and white which is not suitable for our model because we cannot get back the data hidden within it.

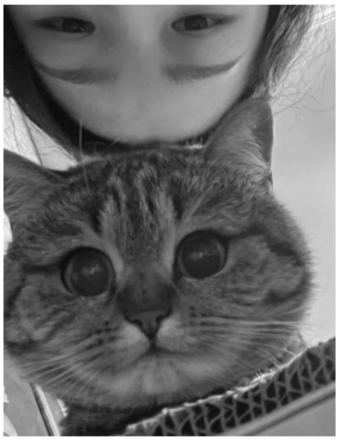
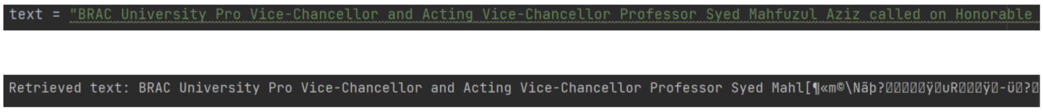


Figure 5.4: Decrypted image

Another renowned technique that we have used is Spread Spectrum. It is well known for hiding text inside an image as well as for hiding a large data. This technique is ingeniously used to conceal data throughout the entire signal. This renders the extraction and detection of hidden data an onerous task, and success requires appropriate expertise and tools. In steganography, the spread spectrum technique bears an uncanny resemblance to Direct Sequence Spread Spectrum (DSSS). A pseudo-random noise sequence is deployed to encode the hidden data, and this noise sequence is added to the original signal. The resultant signal is then dispatched or saved as a file. To extract the hidden data, the receiver must subtract the original signal from the received signal and then correlate the outcome with the same pseudo-random noise sequence that was used to encode the data. Spread spectrum techniques in steganography bestow an unparalleled level of security and resistance to detection. The hidden data is diffused across the entire signal, making it arduous to pinpoint and extract. Moreover, the use of pseudo-random noise sequences makes the hidden data appear as random noise, thereby providing an added layer of camouflage. Keeping these in mind we hide a text inside a 220*220 dimensional image, but when we try to hide the stego image inside another image and later retrieve it we are unable to get the full text we hide. Sometimes the image also gets distorted too much which is visible because the size of the hidden data must be modest to circumvent compromising the quality of the original signal. Additionally, the use of spread spectrum techniques can augment the size of the signal, rendering it more conspicuous and thereby easier to detect. As a result, this technique does not fit into our model and provides lower accuracy than LSB. Figure 5.5 shows the actual text and we are unable to retrieve the actual text from the image, some character gets distorted.



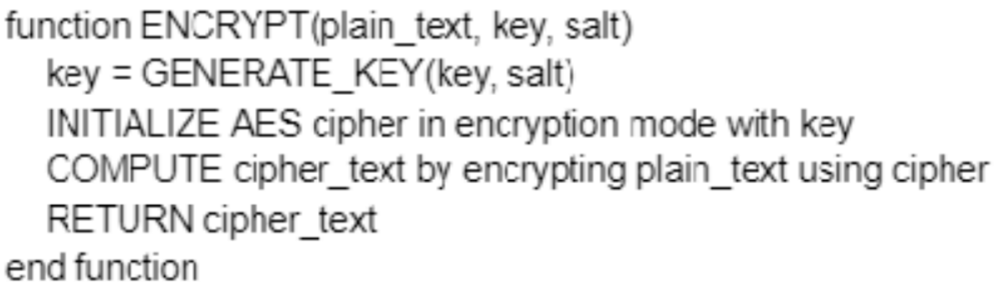
```
text = "BRAC University Pro Vice-Chancellor and Acting Vice-Chancellor Professor Syed Mahfuzul Aziz called on Honorable
```

```
Retrieved text: BRAC University Pro Vice-Chancellor and Acting Vice-Chancellor Professor Syed Mahl[¶«m©\Nãþ?▯▯▯▯▯ÿ▯υR▯▯▯ÿ▯-ü▯?▯
```

Figure 5.5: Actual text and retrieved text

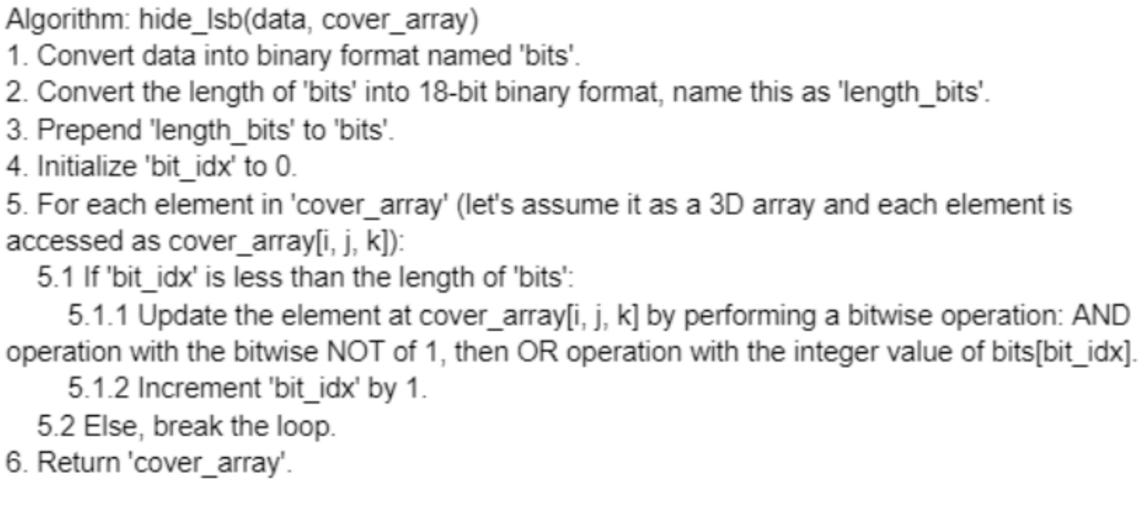
5.3 Experimentation on our hybrid model

For this approach we at first experimented on a dataset “Tiny ImageNet ” which contains over 100000, 64x64 color images as the cover images for LSB layer 1 and cover image and hidden images both for LSB layer 2. Furthermore, to hide large data we used various-sized images to verify our model. We did the encoding part in 3 steps. Initially, we did AES step encryption shown in [5.6] where we encrypted one text data of various lengths. In the next step, we hide encrypted data in an RGB color image using MLSB shown in [5.7] . Lastly, shown in [5.8] we hide the color in another RGB color image using LSB-based steganography.



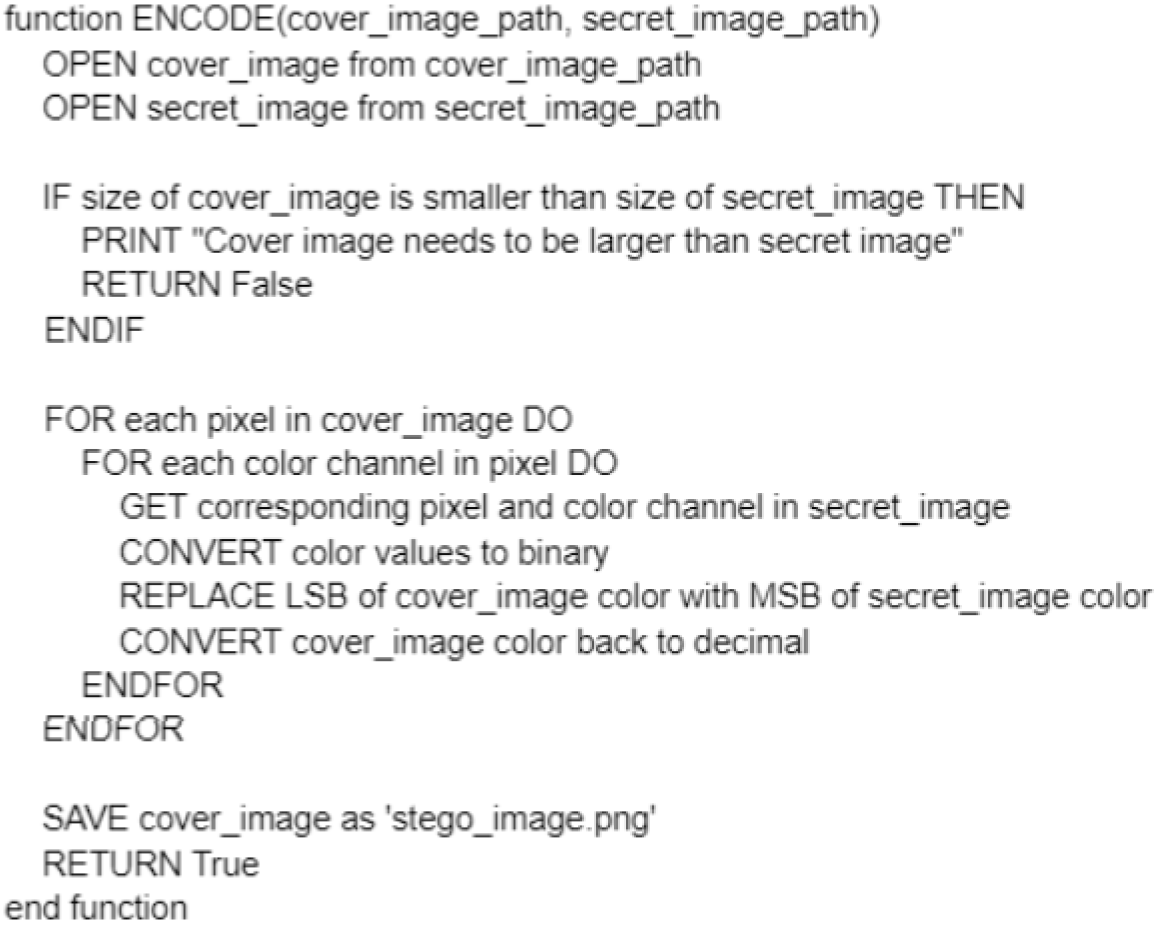
```
function ENCRYPT(plain_text, key, salt)
   key = GENERATE_KEY(key, salt)
   INITIALIZE AES cipher in encryption mode with key
   COMPUTE cipher_text by encrypting plain_text using cipher
   RETURN cipher_text
end function
```

Figure 5.6: AES Encryption



```
Algorithm: hide_lsb(data, cover_array)
1. Convert data into binary format named 'bits'.
2. Convert the length of 'bits' into 18-bit binary format, name this as 'length_bits'.
3. Prepend 'length_bits' to 'bits'.
4. Initialize 'bit_idx' to 0.
5. For each element in 'cover_array' (let's assume it as a 3D array and each element is
accessed as cover_array[i, j, k]):
   5.1 If 'bit_idx' is less than the length of 'bits':
      5.1.1 Update the element at cover_array[i, j, k] by performing a bitwise operation: AND
operation with the bitwise NOT of 1, then OR operation with the integer value of bits[bit_idx].
      5.1.2 Increment 'bit_idx' by 1.
   5.2 Else, break the loop.
6. Return 'cover_array'.
```

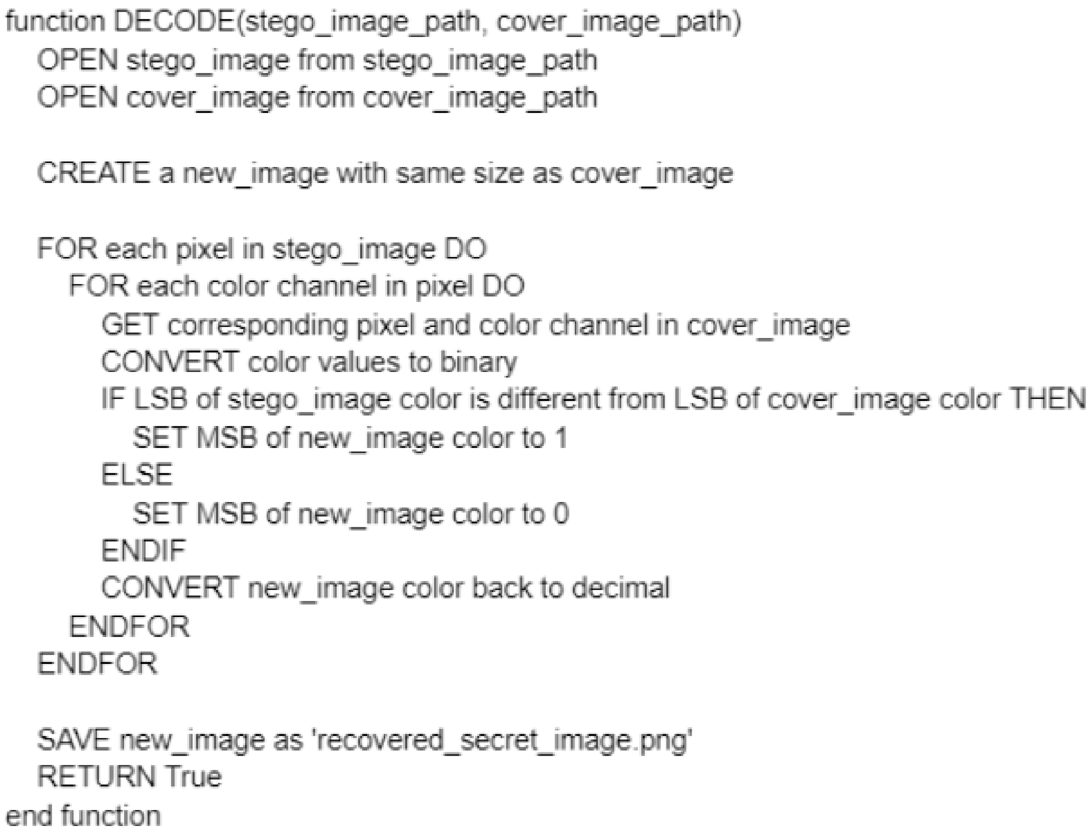
Figure 5.7: MLSB (Hide data in image)



```
function ENCODE(cover_image_path, secret_image_path)
   OPEN cover_image from cover_image_path
   OPEN secret_image from secret_image_path

   IF size of cover_image is smaller than size of secret_image THEN
      PRINT "Cover image needs to be larger than secret image"
      RETURN False
   ENDIF

   FOR each pixel in cover_image DO
      FOR each color channel in pixel DO
         GET corresponding pixel and color channel in secret_image
         CONVERT color values to binary
         REPLACE LSB of cover_image color with MSB of secret_image color
         CONVERT cover_image color back to decimal
      ENDFOR
   ENDFOR

   SAVE cover_image as 'stego_image.png'
   RETURN True
end function
```

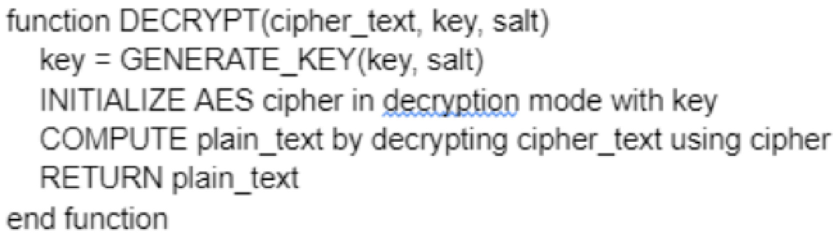
Figure 5.8: LSB (Hide image into image)

To decode the encrypted image and retrieve the text data, we repeated all the steps in reverse. Firstly we decoded the stego-image to retrieve the image encrypted with text data using LSB shown in [5.9]. Then we pass the encrypted image into the reserved MLSB to find the encrypted data[5.10]. Lastly, we applied revered AES [5.11] and finally decrypted the text data



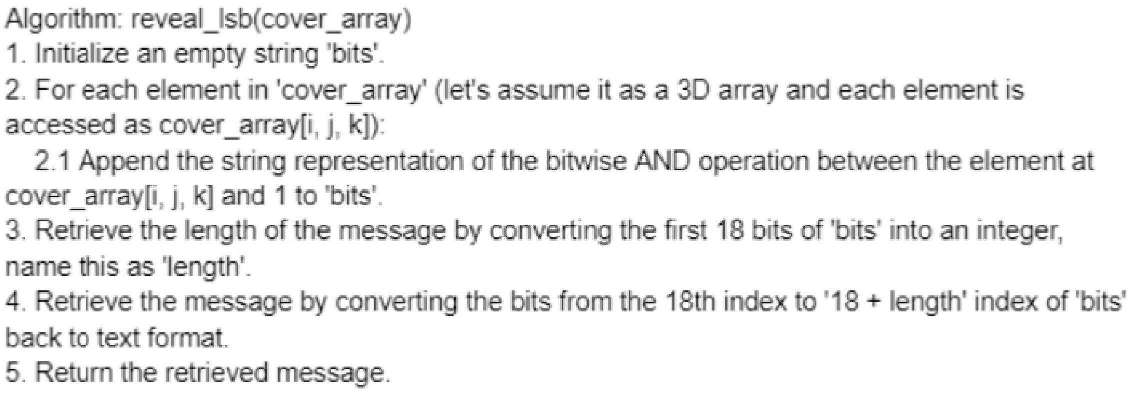
```
function DECODE(stego_image_path, cover_image_path)
   OPEN stego_image from stego_image_path
   OPEN cover_image from cover_image_path

   CREATE a new_image with same size as cover_image

   FOR each pixel in stego_image DO
      FOR each color channel in pixel DO
         GET corresponding pixel and color channel in cover_image
         CONVERT color values to binary
         IF LSB of stego_image color is different from LSB of cover_image color THEN
            SET MSB of new_image color to 1
         ELSE
            SET MSB of new_image color to 0
         ENDIF
         CONVERT new_image color back to decimal
      ENDFOR
   ENDFOR

   SAVE new_image as 'recovered_secret_image.png'
   RETURN True
end function
```

Figure 5.9: LSB (Reveal image from image)



```
function DECRYPT(cipher_text, key, salt)
   key = GENERATE_KEY(key, salt)
   INITIALIZE AES cipher in decryption mode with key
   COMPUTE plain_text by decrypting cipher_text using cipher
   RETURN plain_text
end function
```

Figure 5.10: MLSB (Retrieve data from image)



```
Algorithm: reveal_lsb(cover_array)
1. Initialize an empty string 'bits'.
2. For each element in 'cover_array' (let's assume it as a 3D array and each element is accessed as cover_array[i, j, k]):
    2.1 Append the string representation of the bitwise AND operation between the element at cover_array[i, j, k] and 1 to 'bits'.
3. Retrieve the length of the message by converting the first 18 bits of 'bits' into an integer, name this as 'length'.
4. Retrieve the message by converting the bits from the 18th index to '18 + length' index of 'bits' back to text format.
5. Return the retrieved message.
```

Figure 5.11: AES Decryption

Here, in [5.12], [5.13], [5.14] and [5.15] we are showing the images we used, got after encryption and the images we got After decryption :

Image 1(64x64):

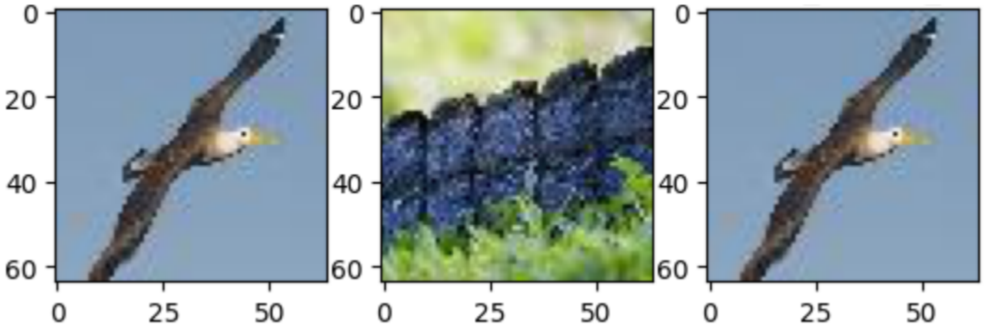


Figure 5.12: Text encrypted image, Hidden image, Recovered image 1

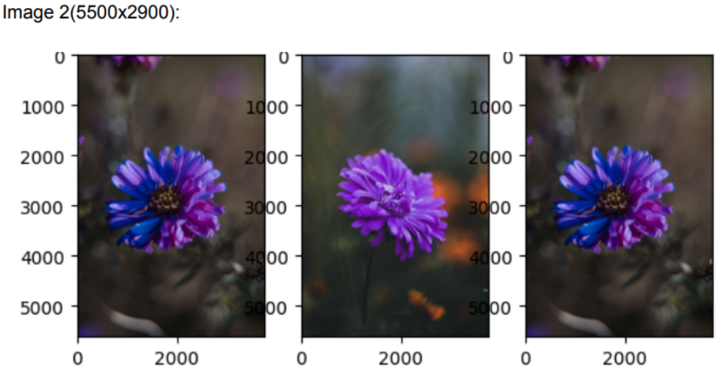


Figure 5.13: Text encrypted image, Hidden image, Recovered image 2

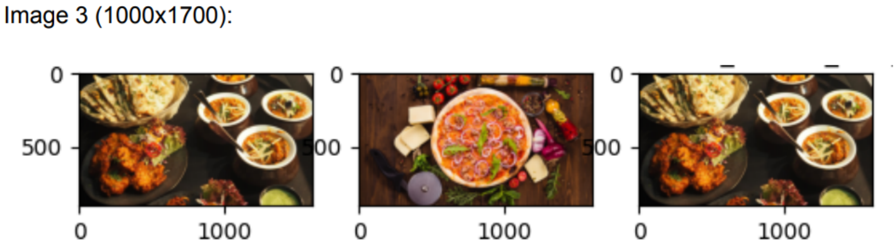


Figure 5.14: Text encrypted image, Hidden image, Recovered image 3

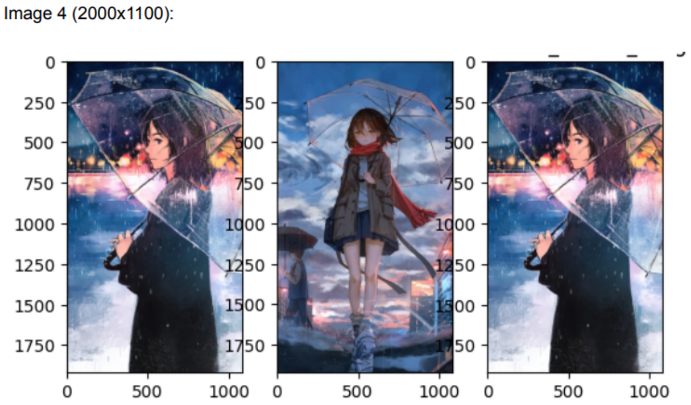


Figure 5.15: Text encrypted image, Hidden image, Recovered image 4

5.4 Effects of Distortion :

Recovering the hidden data from a distorted image in steganography can pose several difficulties. Here are some problems that can arise when attempting to retrieve hidden data:

1.Distortion Detection: If an image is distorted, it is very challenging to retrieve the hidden data from it. In an image noise, artifacts or alterations are basically the effects of distortions. To reveal data from a distorted image can change the original data by corrupting the data with false positives or false negatives in the phase of detection. So, it becomes difficult to detect the distortion and solve it.

2.Data Extraction: Detecting the original data with accuracy from a distorted image is difficult. The algorithm may fail to solve the complexity of the distortion. As a result the algorithm can present inaccurate data.

3.Distortion Impact on Data Integrity: The reliability and accuracy of the retrieved data can be ruined by the effect of distortion. If the distortion can not be handled properly, it can cause errors in data, data loss, and corrupted data, which will give a negative impact on the data integrity.

As our encrypted data after 3 layers of encryption remains distortion-free, we could easily retrieve the hidden data from it. Though we modified our LSB algorithm, it is still simpler than any neural network-based steganography. But the other techniques somehow bring distortion or data loss after encryption, and also after decryption. We couldn't find any work or model which is completely lossless. Many papers claim to have less error but any kind of data loss can corrupt our final decryption of text data. Our model is a hybrid model with the combination of 3 layers, so if in any layer, any change in pixel will harm the final output.

To simplify the harmful effect of data loss or distortion, we experimented by distorting the encrypted image which consisted of hidden text and image as well. We manually corrupted the encrypted image just before the decryption with a randomized distortion function, The distortion effect in this code is a simple addition of a random noise map to the image which can be seen in Figures 5.16, 5.17, and 5.18.

When we tried to decrypt the data from the corrupted encrypted image we got an unknown text, which clearly proves that any kind of data loss is disapproved of by our model for now.

Example 1:

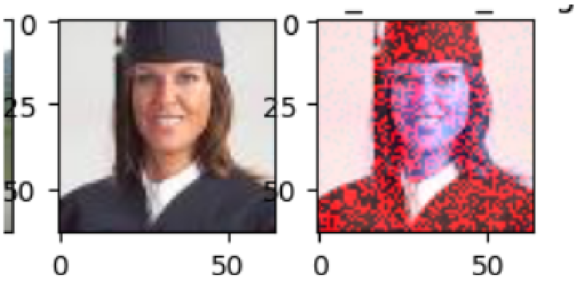


Figure 5.16: Distorted image 1

Plain Text: Steganograhy_done
Encrypted text: 4145c95b84b88cda4e0b8e7089ed383e2bbb62178e0f37812acc98ce 14f5f4f406e2e2dc4d74de84d6e3b1746de1696d
Decrypted Text: ÑÑé¡ôÓiˆh=íXýml?h‘/Ó‖|ý > Á + Cym.Ò =?Bë/y_i/Áã8R

Example 2:

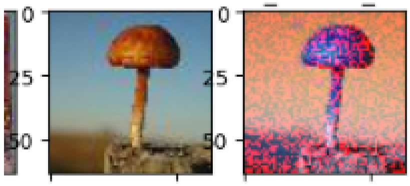


Figure 5.17: Distorted image 2

Plain Text: Hello world
Encrypted text: 820c3de69b571c503cf3d5ddde58546060de5e2273aaf76edda70642c fa37bca
Decrypted Text: iÃémQ%8 ì$_y$ináàmu − Ò = −yèu) = zíyoí)Àéaˆ ∗ Pz ∗ Ã
Example 3:

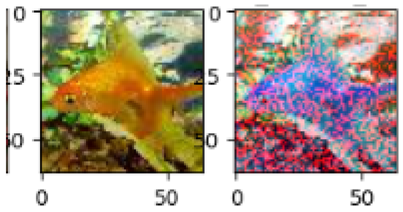


Figure 5.18: Distorted image 3

Plain Text: I Love Python
Encrypted text: b34d8a7bb891d5004893163ee40f1ed49606ce5c356f7181dd7a676f69 23a5d6
Decrypted Text: ÒéRk,¹ø|ø◯̀m—li—émÑm¡@ém m¡é?SýyÓáYô?+85Ã$´?i:Ó%M

Secondly, We also experimented with a CNN-based model to be sure about approached

Chapter 6

Results

In our experiment, we choose a hybrid model combined with AES, Text LSB steganography, and image Steganography. There are multiple steps in the model but we need the same image in every decoding which we got in the corresponding encoding. In this experimental result, the accuracy achieved by comparing the original text with the recovered text will be presented. Furthermore, the numbers obtained by comparing the original image with the recovered image and the cover image with the stego-image, based on metrics such as Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM), will be discussed in terms of image assessment. By analyzing these metrics, insights can be gained regarding the fidelity and quality of the recovered information and images.

Accuracy for Text data: To find an accuracy score to compare the original text and recovered text we used the accuracy score function from Scikitlearn. It counts the number of correct predictions by comparing the elements pairwise. The "accuracy score" function assumes the inputs have the same length and the lab. It compares two text data lengthwise and alphabetically. In our model, we used multiple lengthed texts to test the accuracy. The capacity of hiding text dynamically changes with the sizes of the carrier images.

In our model, In every attempt we successfully decrypted the encrypted text from the two steps of the steganographic layer. The accuracy score is always more than 99% every time..

MSE: The mean-square error (MSE) is a metric utilized to assess the quality of image compression. It quantifies the accumulated squared difference between the compressed image and the original image. A lower MSE value indicates a lower level of error in the compression process. The formula for calculating MSE is as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

PSNR: The peak signal-to-noise ratio (PSNR) calculates the ratio, measured in decibels, between the peak signal level and the noise level between two images. It is commonly employed as a quality metric for comparing the original image with a compressed or reconstructed image. A higher PSNR value indicates a higher quality of the compressed image. The formula for computing PSNR is as follows:

$$PSNR = 10 \cdot \log_{10}\left(\frac{\mathrm{MAX}^2}{\mathrm{MSE}}\right)$$

SSIM: The Structural Similarity Index Measure (SSIM) is employed to quantify the similarity between two images. SSIM serves as a full reference metric, meaning that it measures or predicts the quality of an image based on an initial uncompressed or distortion-free reference image. By comparing the structural information and visual content of the two images, the SSIM index provides a measure of their similarity.

Comparing the original image with the recovered image

Table 6.1: Comparing original image with the recovered image

Number of images	Sizes of image	MSE	PSNR(db)	SSIM
1	64x64(imageNet)	0.00	100	0.976
2	5500x2900	0.00	100	0.999
3	1000x1700	0.00	100	0.999
4	2000x1100	0.00	100	0.999

Table 6.2: Comparing Cover image with the stego-image

Number of images	Sizes of image	MSE	PSNR(db)	SSIM
1	64x64(imageNet)	2.0485	45.0164	0.94
2	5500x2900	0.1019	58.0463	0.999
3	1000x1700	1.169	47.449	0.999
4	2000x1100	0.4300	51.795	0.999

Table 6.3: Analysis of changes in the encrypted image for the length of the hidden text data

Number of images	Length of Hidden Text (char)	MSE	PSNR	SSIM
1	20,000	0.496	51.168	0.998
2	40,000	7.841	39.186	0.989
3	60,000	33.787	32.843	0.976
4	80,000	66.328	29.913	0.884
5	100,000	91.072	28.536	0.794
6	120,000	99.962	28.132	0.358
7	140,000	108.192	27.788	0.365

CNN-Based Steganography: We know CNN is more powerful than LSB in terms of security. So, our first attempt was with CNN-based Steganography. In this case, we used cifar10 which contains (32X32) small images. Our proposed approach shows how after using CNN we couldn't find the PSNR and SSIM values we need for our model compared to the LSB-based approach.

Table 6.4: Training and testing results of the model trained on the CIFAR10 dataset with a varying number of epochs.

Cover Image	Payload Image	No. of Epochs	Training Accuracy (%)	Testing Accuracy (%)	Training PSNR (dB)	Testing PSNR (dB)	Training SSIM	Testing SSIM
CIFAR10	CIFAR10	50	98.55	99.07	25.09	25.09	0.86	0.85
CIFAR10	CIFAR10	100	98.83	99.15	27.30	27.05	0.867	0.86
CIFAR10	CIFAR10	200	99.03	99.28	27.94	27.56	0.88	0.87
CIFAR10	CIFAR10	300	99.62	99.38	28.08	27.71	0.89	0.88

As we can see the PSNR values and SSIM values are lesser than our proposed model. In this CNN model, the images get distorted to some extent. So, if we would use CNN instead of LSB we can not get the same image in the image steganography step because of unavoidable changes in image pixels.

Here in Figure 6.1, we demonstrated what will happen if we use the same image for hiding text and pictures.We have used the same picture for hiding the distorted image and it can be seen that the distortion after putting a large data inside the image is visible by human eyes but if we hide the image inside the inputted image the distortion is lesser and untraceable by human eyes.

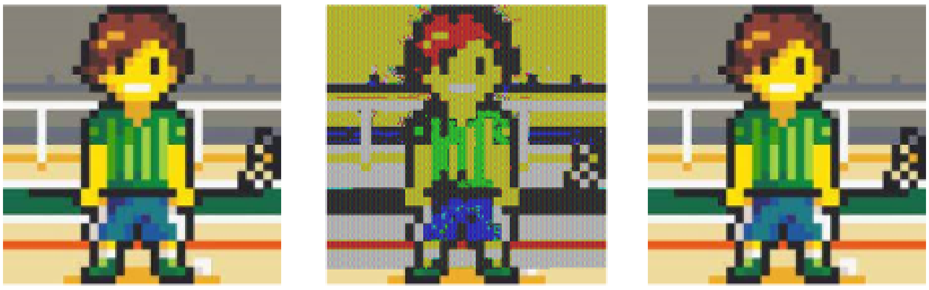


Figure 6.1: Using same cover image

Table 6.5: COVERING BY SAME IMAGE

Image Size	MSE	PSNR	SSIM
220x220	0.0005165	80.9998	0.9999

Chapter 7

Conclusion and Future Work

In this paper, we have tried to introduce a method to operate and handle steganographic algorithms by using an advanced encryption algorithm and a modified least significant bit for text and the least significant bit technique for the image which can increase small payload for larger file secreting. It also can successfully hide two types of data together and retrieve it without deformity. Again, this model decreases the generalized noise ratio for gaining upgrade performance. It is a hybrid model that includes an encoder and one decoder for two different types of incoming data and single outgoing hybrid data for masking secret documents. This represents a finer completion of production for hiding dynamic hybrid data. It can be ensured by employing and examining some datasets so here ImageNet and Cypher-10 have been used for figuring out the final comparative results which are positive, optimistic and propitious. We have tried to make a hybrid, malformation-free architecture of a steganography model that can fulfill our desired outcome so a better implementation with a standard presentation can be performed.

At present, we have worked on implementing the application of hiding dynamic text and a single image inside one cover image. In the future, we are looking forward to working with this same hybrid model with multiple texts and images where different texts will be hidden into multiple different carrier images and then those carrier (stego) images together will be encrypted into one single cover image. While decrypting the cover image, all the hidden texts should be retrieved from the different stego-images following the recovering or regaining of all of the encrypted stego-images from that cover image. Moreover, we will try to reduce the size of the encrypted cover image as it is higher than the actual carrier image. Finally, we will try to implement deep learning and a CNN-based model in our existing model if the accuracy parameters remain intact.

Bibliography

[1] A. Westfeld and A. Pfitzmann, "Attacks on steganographic systems," in *Information hiding*, Berlin, Heidelberg: Springer, 2000, pp. 61–76.

[2] J. Fridrich, M. Goljan, and R. Du, "Reliable detection of lsb steganography in color and grayscale images," in *Electronic Imaging, Security and Watermarking of Multimedia Contents III*, SPIE, vol. 4314, 2001, pp. 311–316.

[3] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in *Media Watermarking, Security, and Forensics 2015*, SPIE, vol. 9409, 2015, pp. 171–180.

[4] A. Singh and H. Singh, "An improved lsb based image steganography technique for rgb images," in *2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, IEEE, Mar. 2015, pp. 1–4.

[5] J.-F. Couchot, R. Couturier, C. Guyeux, and M. Salomon, "Steganalysis via a convolutional neural network using large convolution filters for embedding process with same stego key," *arXiv preprint arXiv:1605.07946*, 2016.

[6] D. Panchal and J. Patel, "A review on steganography and its different techniques," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 6, no. 11, pp. 206–209, 2016.

[7] G. Xu, Y. Zhang, Y.-Q. Shi, J. Huang, and Z. Zhang, "Structural design of convolutional neural networks for steganalysis," in *IEEE Signal Processing Letters*, vol. 23, 2016, pp. 708–712. DOI: 10.1109/LSP.2016.2533187.

[8] S. L. Chikouche and N. Chikouche, "An improved approach for lsb-based image steganography using aes algorithm," in *2017 5th International Conference on Electrical Engineering-Boumerdes (ICEE-B)*, IEEE, 2017, pp. 1–6. DOI: 10.1109/ICEE-B.2017.8258534.

[9] H. Al-Sarawi and O. Mahdi, "An improved image steganography technique using lsb substitution method," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 1, pp. 124–131, 2017.

[10] 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. DOI: 10.1109/TIFS.2017.2727123.

[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, 2017, pp. 2223–2232. DOI: 10.1109/ICCV.2017.244.

[12] P. P. Bandekar and G. C. Suguna, "Lsb based text and image steganography using aes algorithm," in *2018 3rd International Conference on Communication and Electronics Systems (ICCES)*, IEEE, 2018, pp. 782–788. DOI: 10.1109/ICCES.2018.8723864.

[13] B. Li, J. Huang, T. Lu, W. Wang, and W. Guo, "Rest-net: Diverse activation modules and parallel subnets-based cnn for spatial image steganalysis," *IEEE Signal Processing Letters*, vol. 25, no. 5, pp. 650–654, 2018. DOI: 10.1109/LSP.2018.2825638.

[14] R. Rahim and S. Nadeem, "End-to-end trained cnn encoder-decoder networks for image steganography," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[15] H. Wang, W. Li, and X. Luo, "Deep learning for steganography and steganalysis: A comprehensive review," *arXiv preprint arXiv:1905.12297*, 2019.

[16] W. Wang, X. Wang, X. Liu, and X. Qu, "Image steganography based on gans and deep convolutional neural networks," *International Journal of Distributed Sensor Networks*, vol. 15, no. 1, p. 1 550 147 718 817 417, 2019.

[17] J. Zhang, R. Ni, Y. Huang, and Y. Yang, "Deep learning-based image steganography: A comprehensive review," *IEEE Access*, vol. 9, pp. 44 016–44 033, 2021.