IMAGE STEGANOGRAPHY USING GAN

A PROJECT REPORT

Submitted By

VARSHA S. 195001121

YUVEGA S. 195001311

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Department of Computer Science and Engineering

Sri Sivasubramaniya Nadar College of Engineering
(An Autonomous Institution, Affiliated to Anna University)
Kalavakkam - 603110

May 2023

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project report titled **"IMAGE STEGANOGRAPHY USING GAN"** is the *bonafide* work of "**VARSHA. S (195001121)**, and **YUVEGA. S (195001311)**" who carried out the project work under my supervision.
Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Dr. T.T. Mirnalinee
Head of the Department
Professor,
Department of CSE,
SSN College of Engineering,
Kalavakkam - 603 110

Dr. A. Chamundeswari
Supervisor
Professor,
Department of CSE,
SSN College of Engineering,
Kalavakkam - 603 110

Place:
Date:

Submitted for the examination held on. . . . . .

Internal Examiner **External Examiner**

ACKNOWLEDGEMENTS

I thank GOD, the almighty for giving me strength and knowledge to do this project.

I would like to thank my guide **Dr.A.CHAMUNDESWARI**, Professor, Department of Computer Science and Engineering, for her valuable advice and suggestions as well as her continued guidance, patience and support that helped me to shape and refine my work.

My sincere thanks to **Dr.T.T.MIRNALINEE**, Professor and Head of the Department of Computer Science and Engineering, for her words of advice and encouragement and I would like to thank our project Coordinator **Dr.B. BHARATHI**, Associate Professor, Department of Computer Science and Engineering for her valuable suggestions throughout this project.

I express my deep respect to the founder **Dr. SHIV NADAR**, Chairman, SSN Institutions. I also express my appreciation to our **Dr. V. E. ANNAMALAI**, Principal, for all the help he has rendered during this course of study.

I would like to extend my sincere thanks to all the teaching and non-teaching staffs of our department who have contributed directly and indirectly during the course of my project work. Finally, I would like to thank my parents and friends for their patience, cooperation and moral support throughout my life.

Varsha S. **Yuvega S.**

ABSTRACT

Steganography is a way of concealing sensitive information by embedding it in an audio, video, image, or text file.One method is to conceal data in bits that represent the same colour pixels in a row of an image file.By inconspicuously applying the data to this redundant data, the outcome will be an image file that appears identical to the original image but contains noise patterns of regular, encoded data.Our study proposes to encode data by hiding it inside an image file and decoding it safely from the encoded picture file using a Generative Adversarial Neural Network (GAN).The use of Convolutional Neural Networks in the standard picture data concealing approach greatly expands the payload capacity that can be hidden within an image.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Steganography is an approach of concealing sensitive information by embedding it in an audio, video, image, or text file.It is one of the approaches used to protect exclusive or private information from nefarious attempts.In modern digital steganography, data is first encrypted or obfuscated in some other way before being injected into data that is part of a certain file type, such as a PNG picture, using an encoding algorithm.Many diverse methods exist for embedding the secret message into conventional data files.While cryptography and steganography are linked, they are not the same.Cryptography is used to transform messages such that they are unintelligible.It does not conceal the existence or presence of the message.Steganography, on the other hand, conceals the existence of the real message by concealing the real message in another.

1.1 MOTIVATION

The primary aim of data transmission is to provide security against the intruder.There are various Cryptography and Steganography algorithms with the aim to provide secure data transmission where some algorithms stand strong and some are vulnerable to certain attacks and all algorithms do have weak places and they should be utilised wisely, where necessary and where they would work efficiently without getting vulnerable.Steganography is a less researched area while compared to Cryptography due to its overhead and less data embedding

capacity.But used at the right places can ensure security.Cryptography, encrypts the secret data and the data can be seen in encrypted form and hence unreadable.With Steganography, the data is hidden inside an other non-suspicious, non-secret data such that it conceals the presence of secret data itself.A strong Steganography algorithm is that which hides the secret data perfectly without raising suspicion and that which is not easily vulnerable to Steganalysis, an attack used on suspicious files detecting the presence or absence of secret data within the non-secret data being transmitted.

1.2 BACKGROUND

1.2.1 Steganography

Steganography is an information hiding technique used since the periodic times.With the birth of digital world, Steganography has also been digitalized and is used to protect the digital data.Since then, many different Steganography techniques have been developed.The digital cover media used to conceal the secret message is not only limited to text, but techniques to conceal the message in images, audio and video have been developed.There are various kinds of steganography techniques based on different criteria.

1.2.2 Image Steganography

Our project revolves around encoding the secret data within an image.The availability of images on internet and computational ease of encoding inside an

image favour in making this technique a widely used technique.The technique of embedding hidden text into an image file is known as image steganography.The encoding can occur in the spatial domain or in the transform domain of the images.Further Image Steganography can be done over a 2D image or a 3D image, reversible or irreversible or in an adaptive manner by enabling selection of less detection areas, or using Machine learning and Artificial Intelligence to encode.

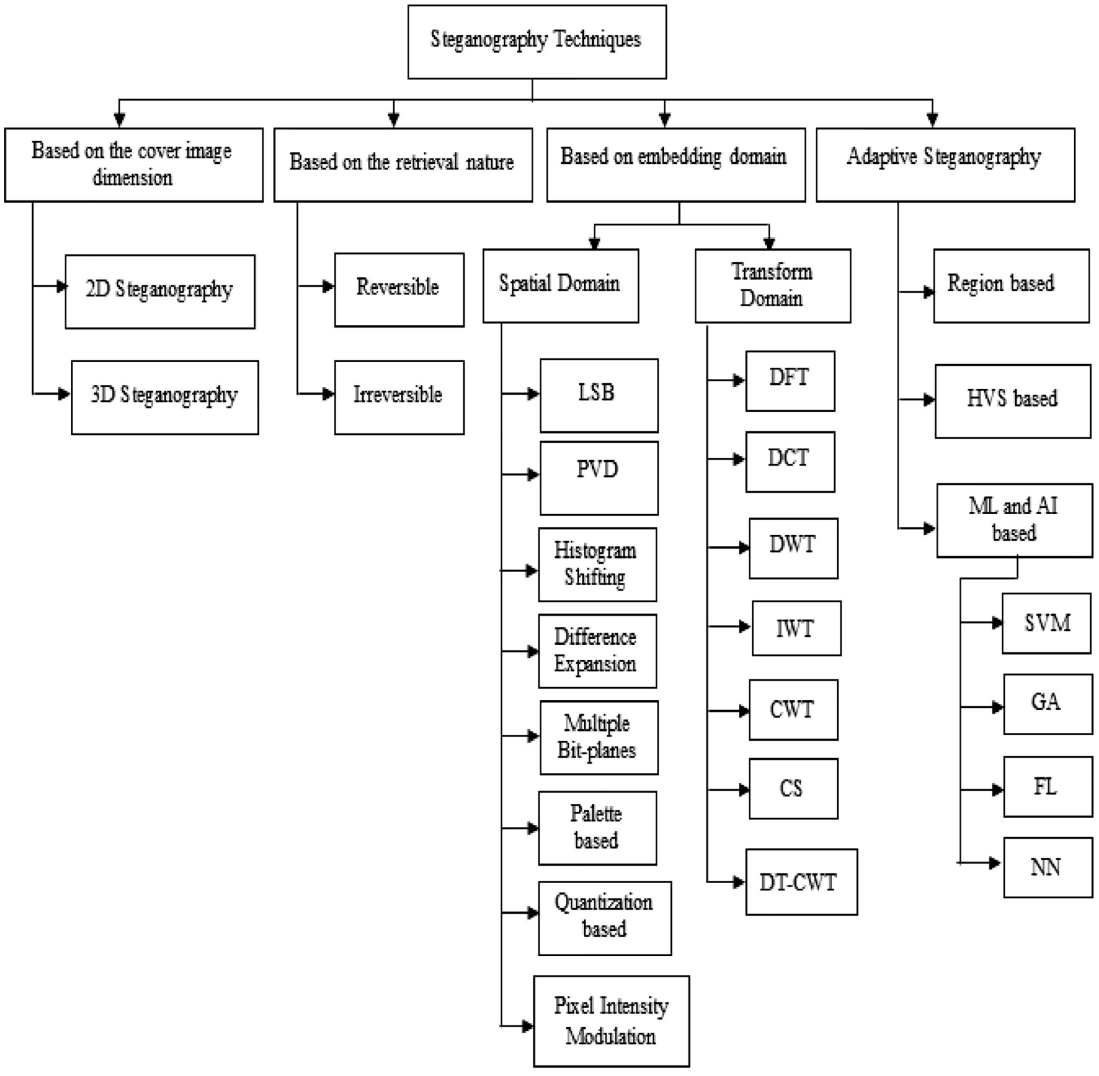

FIGURE 1.1: Classification of Image Steganography techniques

Cryptography	Steganography
The structure of data is affected during cryptography.	The structure of data is not frequently affected with steganography.
Cryptography conceals only the secret message.	Steganography masks the reality that a secret communication is occurring.
Information is altered in cryptography.	Steganography conceals information.
Information that has been transformed is visible.	The information that is hidden is not visible.

TABLE 1.1: Cryptography Vs Steganography

Both Cryptography and Steganography aim to maintain the data security. But both achieve it in different ways.Steganography algorithm maintains three properties - Imperceptibility, Embedding Capacity and Security.The trade-off between these three properties must be optimal for the Steganography algorithm to be efficient.

1.2.3 Deep Learning

Deep learning is a component of Machine Learning, which is a specialisation of AI in and of itself.Artificial Intelligence tries to mimic the intelligence of human brain in order to process the data and to learn information from the data.Deep learning works for the same motive but using Artificial Neural Networks.Artificial Neural Network models mimic the neural networks in our brain to mimic our performance and characteristics of remembering data, predicting outcomes based on experience and arriving at conclusions.Each deep

learning algorithm is an ANN.Artificial Neural Network is a model of interconnected nodes called perceptrons resembling the neurons connected in our brain, organised in several layers.

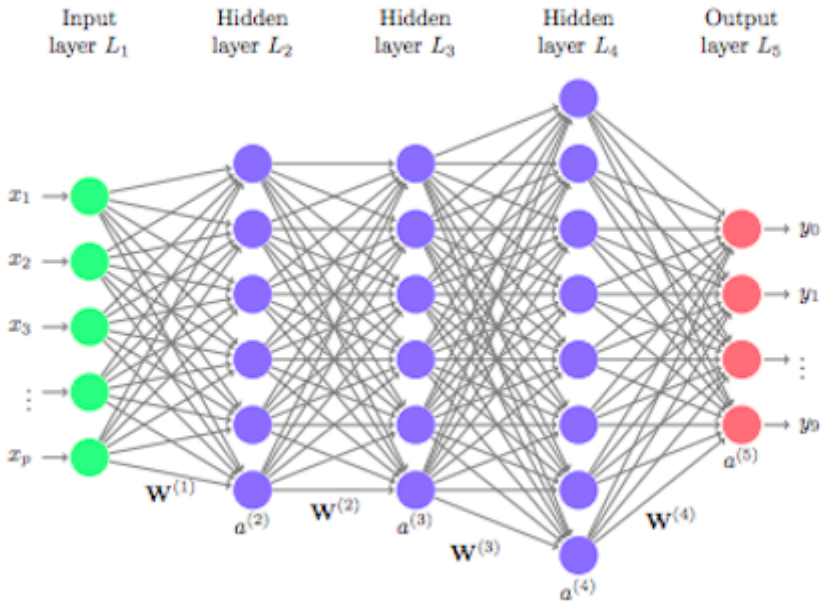

FIGURE 1.2: Deep Neural Network layers

1. The input layer that takes the initial input the is initial layer.
2. The output layer that gives the final output of the task is the final layer.
3. Hidden layers are all the layers in between.

A single perceptron in a network consists of :

1. **Set of weighted inputs:** corresponds to synapses of the connections.
2. **An adder:** sums the incoming input signals (similar to how a cell membrane accumulates electrical charge).

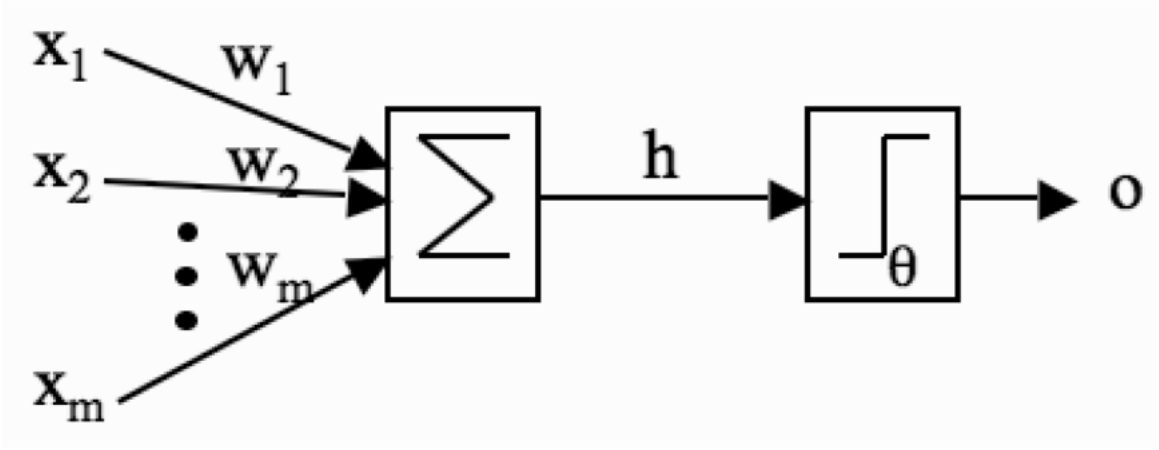

FIGURE 1.3: Perceptron Model

3. **An activation function:** A threshold function that determines whether or not a neuron spikes in response to current input.

Each perceptron is a binary classifier which produces 1 or 0 as output, where 1 indicates activation of the neuron.

Inputs arrive from other neurons through their synapses which are connections in our perceptron model.Those connection have certain strengths which are called as weights.Each input is fed to the perceptron to constitute the input vector through connections associated with weights.Hence the adder function sums the total electrical charge by multiplying each input with the corresponding weight of its connection and summing up the results.

The activation function, suggested by its name is used for deciding the activation of the perceptron.The perceptron outputs 1, if the h value is equal or higher than the threshold.Else, outputs 0, if the h value is lesser.

If for a following input the perceptron outputs 0, but the training dataset states that the output is 1, the difference between expected and actual output is then

calculated, which gives us the error and the weights of the connections are optimised using an optimization function in order to obtain the desired, predicted correct output.This process continues for all input vectors, for a certain number of times.At the end, the single perceptron model gives the expected output. This is supervised learning.

In unsupervised learning, the learning data set doesn't contain the output labels.The model learns by observing the oatterns, forming groups of similar samples and learns to give an output.For Similar samples, it produces same output.The choice of learning depends on the nature of the problem.

As a result, the network consumes massive amounts of input information and interprets it across numerous layers, allowing it to gain knowledge of highly complicated data properties at each layer.

Deep learning is a strong technique for turning predictions into practical outcomes.Deep learning is particularly good at pattern recognition (unsupervised learning) and knowledge-based prediction.Deep learning is powered by big data.When these two factors are coupled, an organisation can achieve extraordinary levels of production, sales, management, and creativity.

Deep learning has the potential to surpass standard methods.Deep learning algorithms outperform machine learning algorithms in image classification by 41%, facial recognition by 27%, and voice recognition by 25%.

1.3 Deep Learning Process

The process of ANN's learning is divided into two stages:

1. A nonlinear transformation is applied to transform the input and generate a statistical model as the outcome in the initial stage.

2. The produced model is optimized using a mathematical process called as differentiation in the final stage.

The above steps are repeated a multitude number of times until the ANN achieves a reasonable level of accuracy.Each repetition is an iteration, also known as an epoch of training or learning.

1.4 CLASSIFICATION OF NEURAL NETWORKS

1. **Shallow Neural Network:** Between the input and output of the Shallow neural network, there is just one hidden layer. with the same successors.

2. **Deep Neural Network:** Deep neural networks consist of multiple layers.For example, the GoogleNet model for image recognition has 22 layers.This category includes all networks other than shallow networks.

3. **Feed Forward Neural Networks:** The most basic kind of Artificial Neural Network.Information travels exclusively forward in this type of design.It means that information flows from the input layer to the output layer, passing through all of the hidden layers one by one. There is no loop in the network.The information is terminated at the output layers.

4. **Recurrent Neural Networks (RNN):** RNN is a multi-layered Neural Network that can learn data sequences and output a number or another

sequence by storing information in context nodes.It is an artificial neural network with looping connections between neurons.RNNs are ideally suited to processing input sequences.The following describes the RNN process:

(a) The RNN neurons will receive a signal pointing to the beginning of the sentence.

(b) The network takes the word ”Hello” as input and outputs a vector of numbers.This vector is returned to the neuron in order to supply memory to the network.This stage aids the network in remembering that it received ”Hello” in the first position.

(c) The network will proceed with the next words in the same manner.

(d) The final stage begins after obtaining the letter ”a”.For each English word that can be used to finish the sentence, the neural network will return a likelihood.A well-trained RNN will most likely assign a high probability.

Common uses of RNN:

(a) Assist in the creation of analytical reports for stock traders.

(b) Identify irregularities in the financial statement contract

(c) Create captions for the pictures.

(d) Detect fraudulent credit-card transaction

(e) Power Chat bots

The standard uses of RNN occur when the practitioners are working with time-series data or sequences.

5. **Convolutional Neural Networks (CNN):** CNN is a multi-layered neural network with a distinct architecture that extracts more complicated data features at each layer to decide the output.CNNs are highly adapted for perceptual tasks.

   When practitioners need to extract information from an unstructured data set (like photographs), CNN is frequently used.

   For an instance, if the job is to forecast the caption for an image:

   (a) The CNN gets an image, say of a cat, which is a collection of pixels. A gray-scale image typically comprises one layer, but a colour image has three layers.

   (b) The network will recognise distinctive features throughout the feature learning (i.e., hidden layers), such as the cat's tail, ear, etc.

   (c) The network may offer a probability for each image it knows once it has properly mastered the art of image recognition.The network's forecast will be the label with the highest probability.

CHAPTER 2

CONVOLUTIONAL NEURAL NETWORKS

A Convolutional Neural Network (CNN, or ConvNet) are a special kind of multi-layer neural networks, designed to recognize visual patterns directly from pixel images with minimal Pre-processing.A sizable visual database created for use in research on visual object recognition software is called the ImageNet project.The ImageNet Large Scale Visual Recognition Challenge (ILSVRC), an annual software competition hosted by the ImageNet project, pits computer programs against one another to identify and classify objects and scenes accurately.

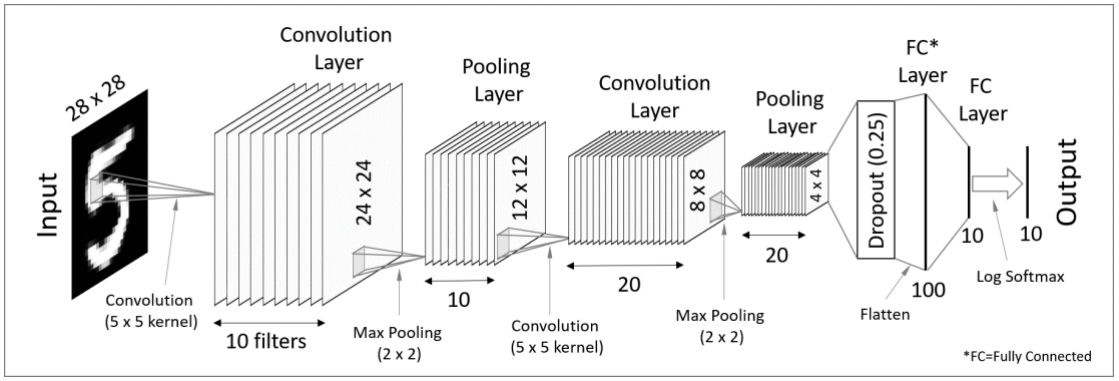

FIGURE 2.1: Convolutional Neural Network

Convolutional Neural Networks are used for features extraction.The Features extracted can be fed to a multilayer perceptron that classifies the image.But the use of CNN is not only restricted to image classification but also for other more image processing tasks are made possible by using CNN.The Features extracted can be fed to various complex networks for other tasks which involves usage of features of the image.

2.1 GENERATIVE ADVERSARIAL NEURAL NETWORK

Generative modelling is a way to produce the outcome by training the ANN using a discriminator that judges the outcome and decides whether that ANN is to be trained or it itself should be trained enough.The discriminator is also an ANN used for classifying the outcome obtained.Generative as the word indicates, the outcome obtained is entirely a new or only a transformation of the input given to the ANN.

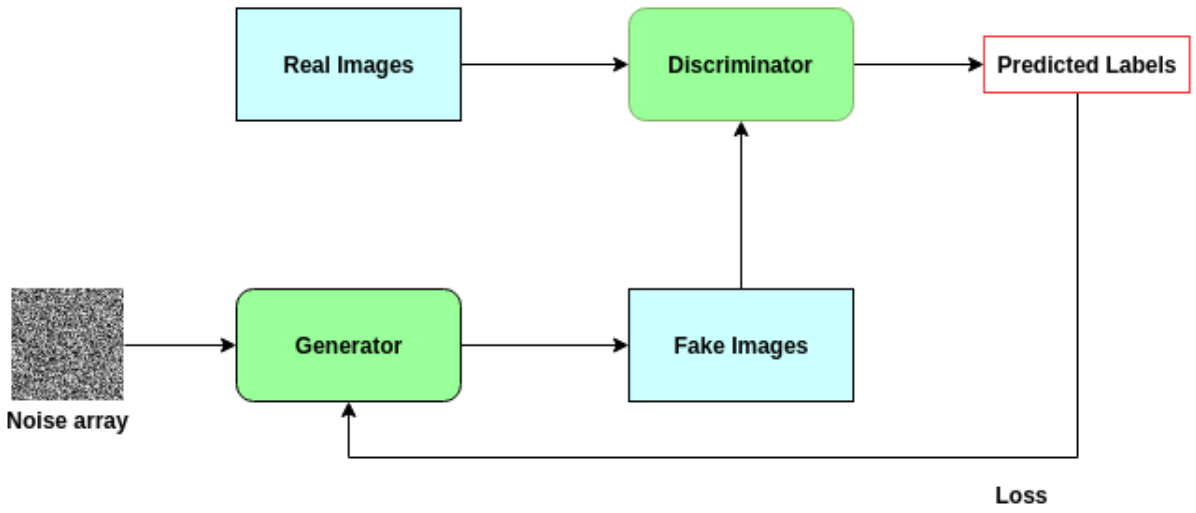

FIGURE 2.2: GAN Architecture

The purpose of this approach is to produce outcomes, fresh or changed from input, that are realistic in both appearance and characteristics.As a result, the goal is attained through training using a discriminator model.Along with the discriminator submodel, the model has a generator submodel.One is used to generate the result, while the other is utilised to determine whether it is true or

phoney.Following that, both models are trained using that information.The discriminator is optimised for output as real.The generator is optimised for output as fake.GANs are an ingenious method for training a generative model.The two models are trained together in an adversarial, zero-sum game, the discriminator model is deceived around half of the time, showing that the generator model is creating convincing outcomes.

GANS are an intriguing and quickly evolving field that serves the potential of generative models by generating realistic outcomes across a wide range of problem domains, most notably in image to image translation tasks such as translating photos of winter to summer or night time to day time, and in generating photorealistic photos of objects, scenes, and people that are so realistic that even the human eye cannot tell the difference that they are unreal and were generated using GAN.

So, the GAN can be easily used to translate an image from a normal image to an image that holds some secret information without bringing distortion to the image, maintaining the realistic nature of the image and moreover not bringing any noticeable change to the image.

2.2 FRACTAL NEURAL NETWORK

The Fractal Network is a form of Convolutional Neural Network that does not use residual connections in favour of a fractal construction.To create deep networks

with exactly trimmed fractal structure patterns, they repeatedly apply a straightforward expansion rule.These networks have interacting subpaths of various lengths, but they don't have any pass-through or residual connections.Instead, every internal signal is altered by a filter and nonlinearity before being picked up by higher layers.

On both the CIFAR and ImageNet classification tests, Fractal Networks outperform standard residual networks, demonstrating that residual representations may not be essential for the accuracy of extremely deep Convolutional Neural Networks.Rather, the capacity to move from effectively shallow to deep throughout training may be the key.Strong proof that path length is crucial for training ultra Deep Neural Networks is provided by FractalNet networks.

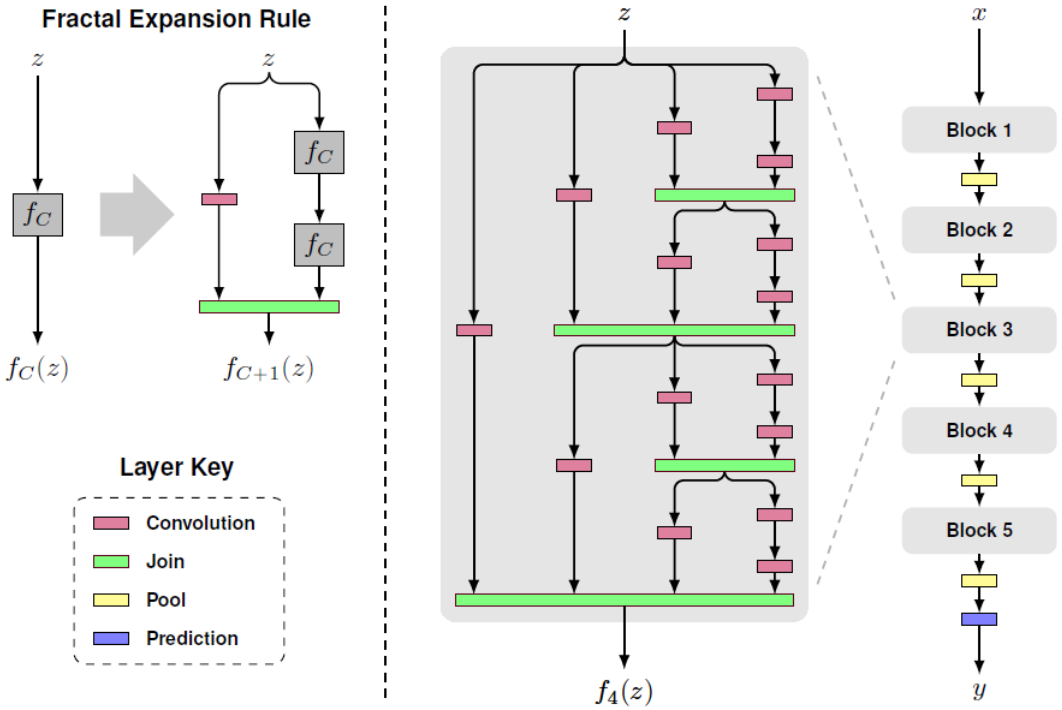

FIGURE 2.3: FractalNet Architecture

2.3 ORGANIZATION OF THE REPORT

1. CHAPTER 1 : INTRODUCTION
2. CHAPTER 2 : CONVOLUTIONAL NEURAL NETWORKS
3. CHAPTER 3 : LITERATURE SURVEY
4. CHAPTER 4 : PROBLEM DEFINITION
5. CHAPTER 5 : PROPOSED SYSTEM
6. CHAPTER 6 : EXPERIMENTAL RESULTS
7. CHAPTER 7 : PERFORMANCE ANALYSIS
8. CHAPTER 8 : CONCLUSIONS AND FUTURE WORK

CHAPTER 3

LITERATURE SURVEY

M. Hassaballah, M. A. Hameed, A. I. Awad, and K. Muhammad present HHOIWT, a method based on digital image steganography, for covert communication and secure data in the IIoT context.The method embeds secret text in cover images by employing the Harris Hawks Optimisation (HHO) metaheuristic optimisation algorithm to select image pixels that can be utilised for concealing bits of secret text within integer wavelet transforms.Objective function evaluation is incorporated by the HHO based pixel selection in two ways: Exploitation and Exploration.The objective function is used to find the appropriate encoding vector to utilise in order to convert secret text into an encoded form created by the HHO method.

Pros and Cons: Based on several experiments to validate the proposed method's performance in terms of visual quality, payload capacity, and attack resistance, it is revealed that the HHO-IWT method achieves greater degrees of security than state-of-the-art methods and exhibits higher resistance against various forms of Steganalysis.Except for the slower convergence rate compared to other algorithms, HHO-IWT enables greater recovery of the secret text than prior Steganography approaches.

R. Vishalchandar and P. Madhavan present an IoT medical data encoding system based on Cryptography and Steganography.To begin, a cryptography technique is utilised to encrypt confidential data obtained from various healthcare providers.The encrypted data is subsequently incorporated into a low

dimensional image using a Steganography approach that uses Matrix XOR embedding.The suggested work additionally employs an optimisation technique known as Adaptive Firefly to optimise the choice of cover units inside the image, as well as data that is hidden within the image.Finally, the concealed data in the image is recovered and decoded.

Pros and Cons: The EGC technique was tested in a MATLAB simulator and found to have an 86 percent steganography embedding efficiency.The downside is that, as compared to RSA encryption, it greatly increases the overall size of the encrypted message.In addition, the EGC method is more sophisticated and difficult to implement.

W. Lu, J. Chen, J. Zhang, J. Huang, J. Weng, and Y.Zhou proposed a feature space and layer embedding based secure halftone picture steganographic technique.First, a feature space is created using a characterization approach based on the statistical results of 4 x 4 pixel blocks in halftone photographs.A generalised steganalyzer with strong classification capacity is proposed on the feature space, and it is utilised to measure the encoding distortion.As a result, a distortion model built on a mixed feature space outperforms several 'state of the art' models.After establishing the distortion model on the statistical results of local regions, a layer encoding technique is provided to minimise MIEM.The host image is divided into several layers based on their relative placements in 4 x 4 blocks, and the embedding procedure is carried out by layers, one by one.Any two pixels in each layer are positioned in different 4 x 4 blocks in the input image, and the distortion model ensures that the distortions of pixels are calculated independently.Between layers, the current layer's pixel distortions are adjusted based on the prior embedding adjustments, lowering the total embedding

distortion.When compared to prior systems, the suggested steganographic procedure achieves good statistical security while defying 'state of the art' Steganalysis.

Pros and Cons: STCs are often used in picture Steganographic techniques to minimise encoding distortion, yet most distortion models do not reflect the Mutual Interaction of Embedding Modifications (MIEMs).

Including secret text on the cover image directly affects MIEM and reduces undetectability performance.

Recent breakthroughs in adaptive steganography reveal that adopting non additive models that represent the dependencies between nearby pixels can increase the performance of picture steganographic communication.W. Su, J. Ni, X. Hu, and J. Fridrich proposed a Gaussian Markov Random Field model (GMRF) with four-element cross neighbourhood to characterise the connections among local elements of cover images, and the problem of robust image steganography is put forward as one of the reductions of KL-divergence in the context of a sequence of low dimensional clique structures linked to the GMRF by exploiting GMRF's conditional independence.When the suggested GMRF is used, the cover image is rasterized into two sub-images which are disjoint, and an alternating optimisation approach that is iterative is designed to efficiently encode the supplied payload on the parallel, minimising the total KL-divergence between the Original and the Steganography image.Experiment results show that the presented GMRF outperforms previous art model based approaches such as MiPOD and competes with the 'state of the art' HiLL for commercial steganography where steganalyzers do not have access to the selection channel understandings.

Pros and Cons: The suggested GMRF characterises the high-dimensional joint distribution of the original image elements and the related KL-divergence efficiently in terms of a succession of clique structures of low dimensions.

The most often used approach for adaptive picture steganography is based on the framework of least distortion embedding, which includes the cumulative encoding cost for each element of the original image and the embedding method, which is usually Syndrome Trellis Codes (STC).

W. Tang, B. Li, S. Tan, M. Barni, and J. Huang present an adversarial embedding based on CNN for image steganography.Steganographic techniques are frequently developed in such a way that image statistics or steganalytic properties are preserved.Because the majority of cutting-edge steganalytic approaches use a machine learning based classifier, it is appropriate to investigate combating steganalysis attacks by attempting to mislead the Machine Learning classifiers.Nevertheless, only applying deviations to steganography images to obtain adversarial examples might result in data extraction failure and the introduction of unplanned artefacts visible to other classifiers.In this work, a steganography solution with a novel function termed Adversarial Embedding (ADV-EMB) was presented, with the purpose of concealing a discreet message while tricking a steganalyzer that was built based on the Convolutional neural network.The approach suggested here is based on the traditional paradigm of distortion reduction.ADV-EMB, in particular, adjusts the costs of image element adjustments based on gradients returned through back propagation from the target CNN steganalyzer.As a result, the modification direction is more likely to be equivalent to the gradient's inverse sign.Therefore, this results in what are known as adversarial steganography images. Investigations show that the

proposed steganography technique surpasses the target adversary-unaware steganalyzer in terms of security by boosting its rate of missed detection.Furthermore, it lowers the effectiveness of other steganalyzers that are adversarially aware paving the door for a fresh class of contemporary steganographic systems that has the ability to beat the sophisticated convolutional steganalysis models.

Pros and Cons: When weighed against the 'state of the art' baseline steganographic method, the implied ADV-EMB results in a greater detection error rate irrespective of target or non target steganalyzers.

The suggested technique solely uses the gradient signs alone, which may result in a higher alteration rate and lesser quality of the image.

3.1 DISADVANTAGES OF THE EXISTING SYSTEMS

1. Encoding secret messages in the cover image directly causes MIEM and reduces undetectability performance and the encoded image easily falls prey to steganalysis.

2. It is a challenging and a tedious task to train RNNs on exceptionally long sequences, particularly when utilising Rectified linear activation function or hyperbolic tangent activation functions.

3.2 RESEARCH OBJECTIVES

1. To produce Steganography images which are identical to that of the cover image such that the image distortion due to embedding is ignorable.

2. To enhance the embedding capacity upto 4 bits per pixel.

3. To produce Steganography images that are resistant to Steganalysis.

CHAPTER 4

PROBLEM DEFINITION

Data is being generated on a large scale and transmitted through unsecured channels.Data security has become the soul of data transmission.There are various algorithms developed to ensure secure data transmission.One such technique is Steganography.Steganography can be applied to the data to hide its presence in a cover media.In this way, it doesn't even arise any suspicion to the intruder about the presence of secret data.A strong Steganography algorithm is that which hides the secret data perfectly without raising suspicion and that which is not easily vulnerable to steganalysis, an attack used on suspicious files detecting the presence or absence of secret data within the non-secret data being transmitted.Traditional Steganography methods encode data directly in the pixels of the image which cause huge change in the pixel values which becomes easy for Steganalysts to easily discover the presence of secret data.Modern ways to encode and decode safely has to be developed and improved in terms of its performance.Hence, our project proposes a Steganography technique to conceal the secret information within an image file and to safely decode the secret message from the encoded image file using Generative Adversarial Neural Networks in deep learning.Our project also aims to increase the bits per pixel capacity of the image to 4 bits per pixel without compromising the security and imperceptibility properties of Image Steganography.The proposed model generates Steganography images that are realistic, identical to the cover image and not easily vulnerable to Steganalysis.This Steganography technique can stand

alone or be used along with Cryptographic algorithms as an additional layer of security.

CHAPTER 5

PROPOSED METHODOLOGY

5.1 BRIEF DESIGN

The brief design of our proposed system is presented in the Figure 5.1.Cover image and secret message in text are the input to our whole steganography model.The text is converted into bit vector.The cover image and bit vector are the input to the encoder network. The encoder network encodes the bit vector inside the desired cover image in a fractal network architecture.This increases the bits per pixel that can be encoded within the image to 4.The encoded image thus is realistic and the secret data within, not easily detectable by steganalysis.The decoding process is the reverse of encoding process.The decoder network takes the steganography image as the input and decodes the bit vector from the steganography image.The bit vector is converted to text and given as output.Thus, the secret message would be decoded successfully.

5.2 DESIGN MODULES

5.2.1 Data set Collection

The Div2k data sets are collected from the following sources: http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_valid_HR.zip

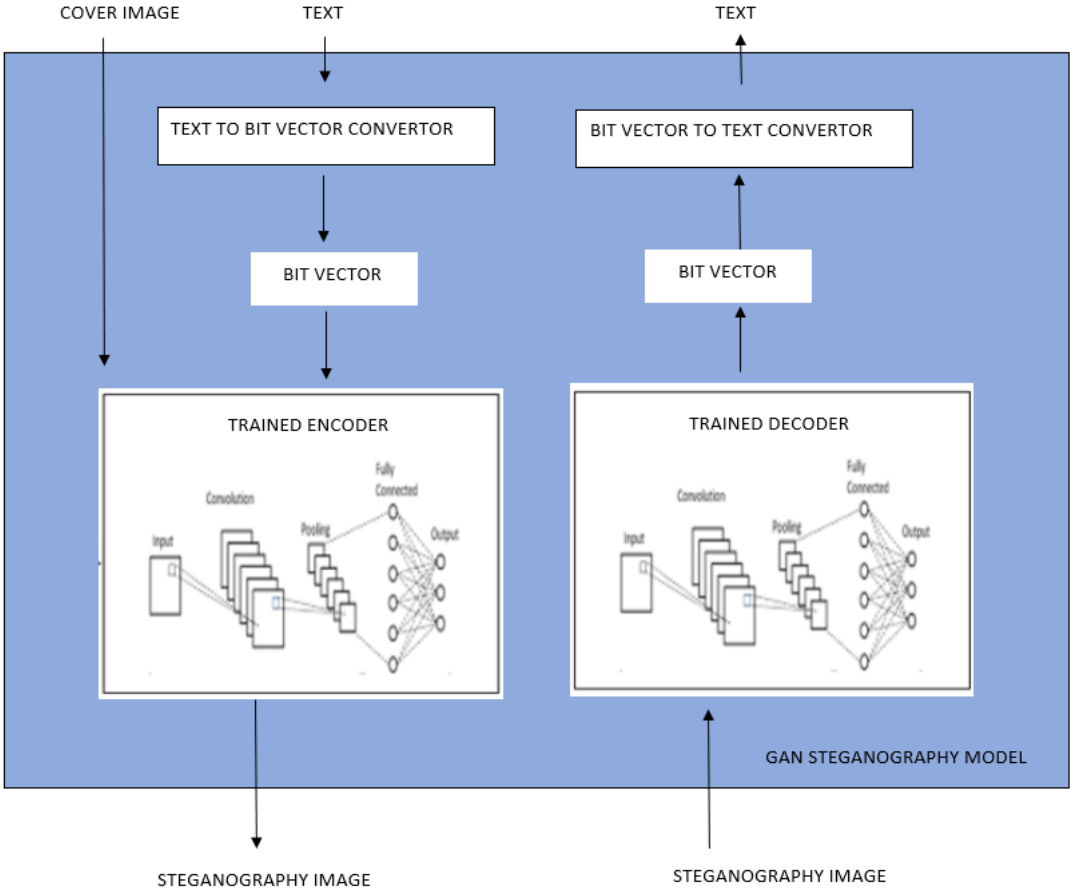

FIGURE 5.1: Design of the proposed system

http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_train_HR.zip

The div2k dataset of 900 images is split into 800 images for training and 100 images for testing.

A collection of high resolution PNG format images are utilised for training and validation of the model.The model can be further trained over any other format images for it's effective functioning with those images as well.

5.2.2 Text to Bits Conversion

This function takes the secret message as a string input.For the data to be fed inside a Convolutional Neural Network, it is necessary that the text should be in binary format.Hence the text is translated to it's corresponding binary digits of 0s and 1s.

5.2.3 Bits to Text Conversion

This function takes a list of bits as input and returns the corresponding string.

5.2.4 Make Payload

The function takes the height, width and data depth of the image and returns a bit vector from a list of bits.The generated bit vector is a tensor, a data structure commonly used for representation of image data in PyTorch framework.

5.2.5 Build the Generative Adversarial Neural Network

Our proposed model consists of three convolutional neural networks which are Encoder, Decoder, Critic.

1. **Encoder network:** The encoder network generates the encoded image by using an image and the secret bit vector as input.

The encoder network is a fractal type convolutional neural network consisting of four sequential convolutional units.Each unit is a stack of a convolution layer, which performs convolution of the image tensor, an activation layer, to decide the activation and a batch normalization layer for normalizing the values.

The encoding process happens in a fractal network architecture.The first unit takes the image tensor as input and extracts the features and produces a feature map tensor which is appended to the list of tensors.The bit vector tensor is concatenated with the output tensor of the first unit and this concatenated tensor is passed as the input for the next unit.The output tensor from this next unit is added to the list of tensors.The bit vector tensor is concatenated with the output tensors of previous units and this concatenated tensor is the input for the following unit and the output from this unit is added to the list of tensors.The process continues for how many convolution units present.Each unit takes the previous output tensors along with bit vector tensor as the input tensor except the first unit which takes the image.This forms the fractal architecture.Using Fractal architecture makes the embedding process more efficient and makes it difficult for Steganalysis.

2. **Decoder network:** The decoder network takes the steganography image as the input and decodes the bit vector from the image.The data can be decoded only in the reverse process of encoding.Hence decoder also is an fractal type network which extracts the bit vector in a fractal

procedure.Thus, the decoder network also has a structure similar to that of the encoder network.

3. **Critic network:** The critic is a Convolutional Neural Network used at the training phase of the model alone for classification purpose.It is used for training the encoder and decoder.It takes an image as its input and classifies whether its a real image or a steganography image.This network is used so that the encoder produces realistic images identical to the original image and the decoder accurately decodes the message.

5.2.6 Train the GAN model

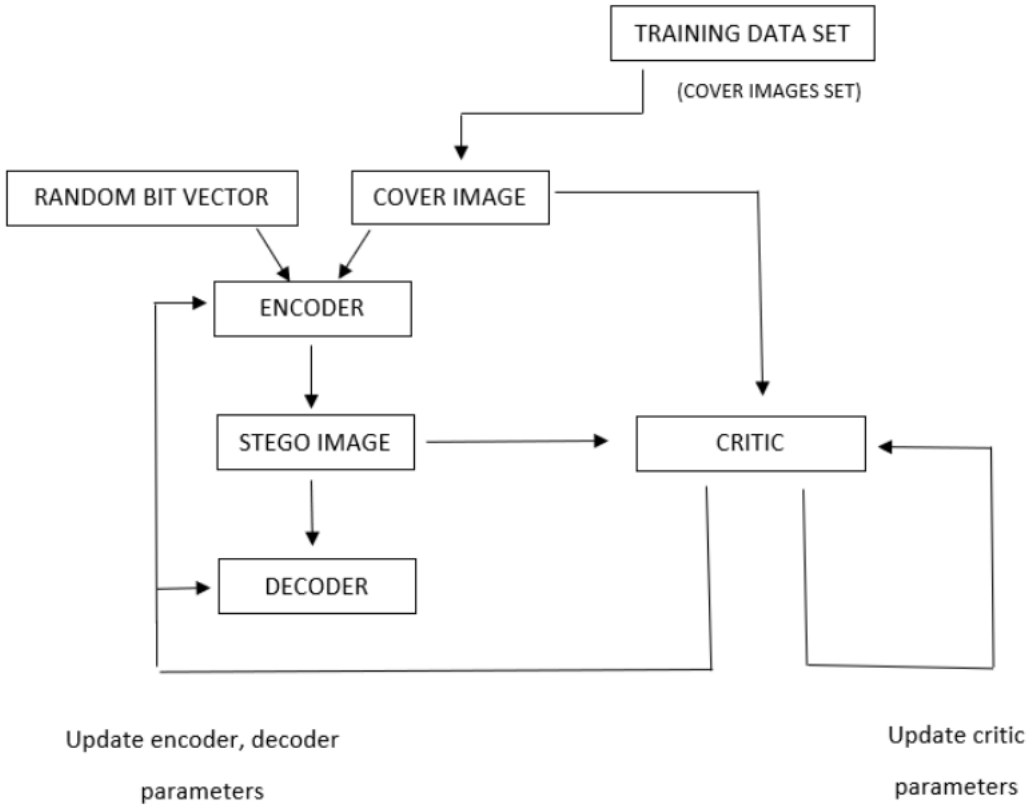

FIGURE 5.2: Flowchart of GAN training

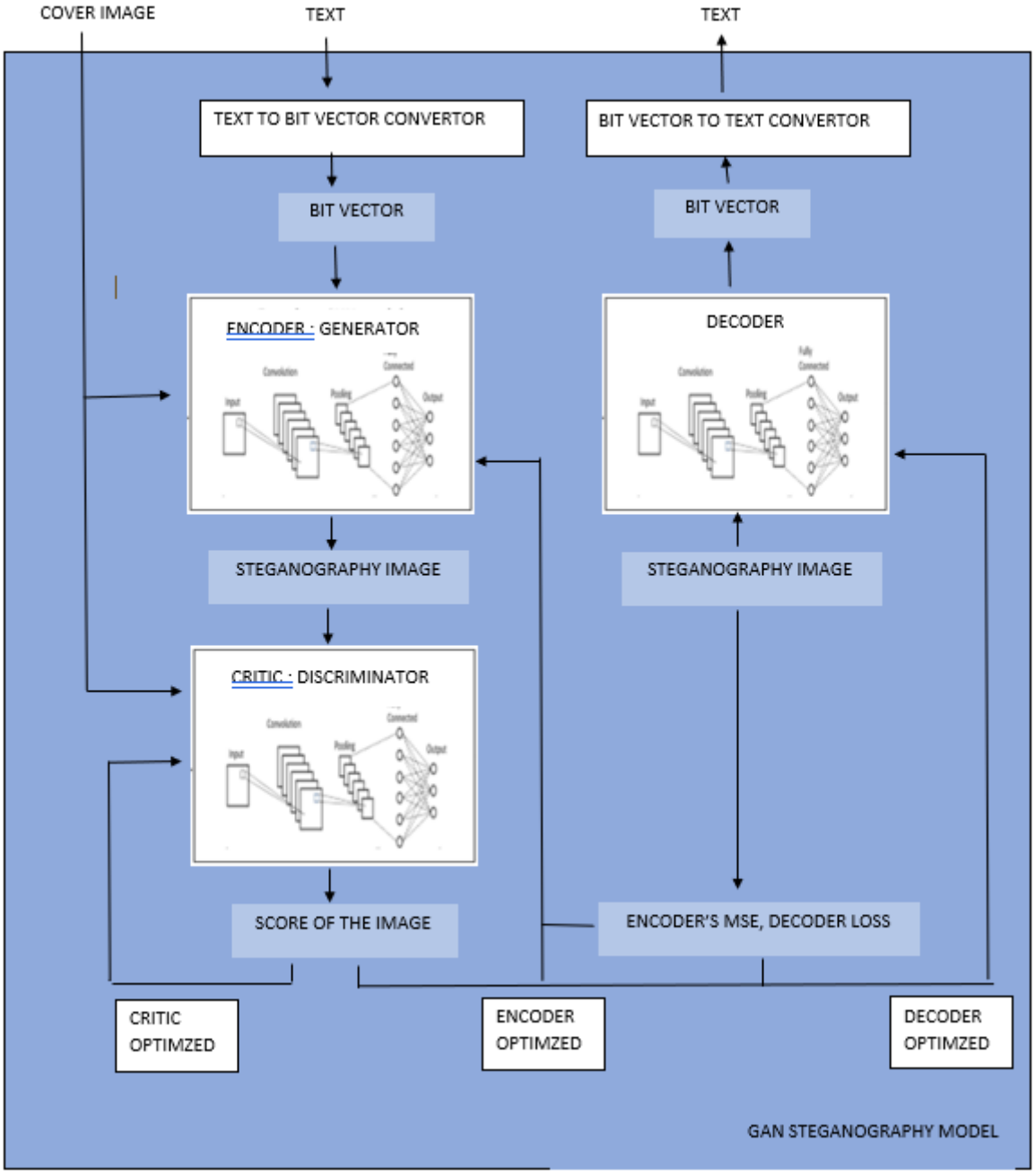

FIGURE 5.3: Architecture of the model for GAN training

The model is trained for 100 epochs.The critic's ouput decides whether the critic should be updated or the encoder - decoder duo.

When the critic classifies correctly the Steganography image as the same, this

indicates that the encoder couldn't trick the critic, and the encoder and decoder's weights are updated for their improved performance.

When the critic classifies incorrectly, this indicates that the encoder successfully fooled the critic and critic's performance should be enhanced.Therefore, it's weights are updated.

Adam Optimization algorithm is utilised for Optimization of weights.

This forms the GANS architecture making the encoder strong enough to produce images that are identical to the cover images in their appearance as well as their characteristics so that it's nearly not possible to perform Steganalysis easily.The decoding process is difficult in Steganography than encoding.Often, images are encoded strongly but the decoder not being compatible, could not decode the message and hence the message is lost.Here, the decoder trained along with encoder makes it compatible with the encoder such that the decoding is possible correctly.

5.2.7 Test the Model

The model is checked for its effciency on the Div2k validation or called the testing data set to generate samples and rate the model's performance.

5.2.8 Save the Model

The model after training and testing is stored at the desired location of the Colab space.

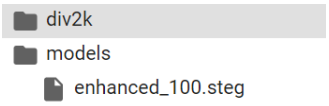

FIGURE 5.4: Model saved in Colab space

5.2.9 Load the Model

The model is loaded whenever the task of encoding and decoding is to be performed.

5.2.10 Download the Model

Since the Colab environment gets refreshed, after regular intervals of time, the trained model was saved on our device by downloading the model as a python file.

5.2.11 Encode Function

This function reads the cover image and creates a tensor representation of the image.The tensor is permuted and unsqueezed to attain the desired dimensions for it's easy manipulation.The device to load the tensor is set as the model's device.The model and the data should be on the same device.It passes the

dimensions of the image to make payload function and obtains the payload tensor.It then passes the image tensor and the payload tensor to the encoder network for encoding.The returned tensor is permuted again to get back the original dimension and the tensor is converted to numpy array and is written and stored as an image in the specified location.This is the generated Steganography image.

5.2.12 Decode Function

This function reads the generated image, creates a tensor, permutes and unsqueezes the tensor and allots the model's device as in encode function.Then, passes the tensor to the decoder network.The decoder network returns the extracted bit vector tensor.The message is then extracted from the bit vector tensor using bits to text conversion function.

CHAPTER 6

EXPERIMENTAL RESULTS

6.1 DATA SET DESCRIPTION

The experiment was conducted using Div2k PNG images data sets.

The Div2k data sets are collected from the following sources:
http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_valid_HR.zip
http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_train_HR.zip

It is a very random datset of PNG images of resolution 2040 x 1404.The images do not belong to any particular class to help the model in generalise to any image.It contains images of cars, nature, animals, travelling, people, etc.Our model can further be trained over other format images for its effective functioning with those images as well.

The div2k dataset of 900 images is split into 800 images for training and 100 images for testing.

Hence, a collection of high resolution PNG format images are utilised for training and for the validation of the model.The model can be further trained over any other format images for its effective functioning with those images as well.

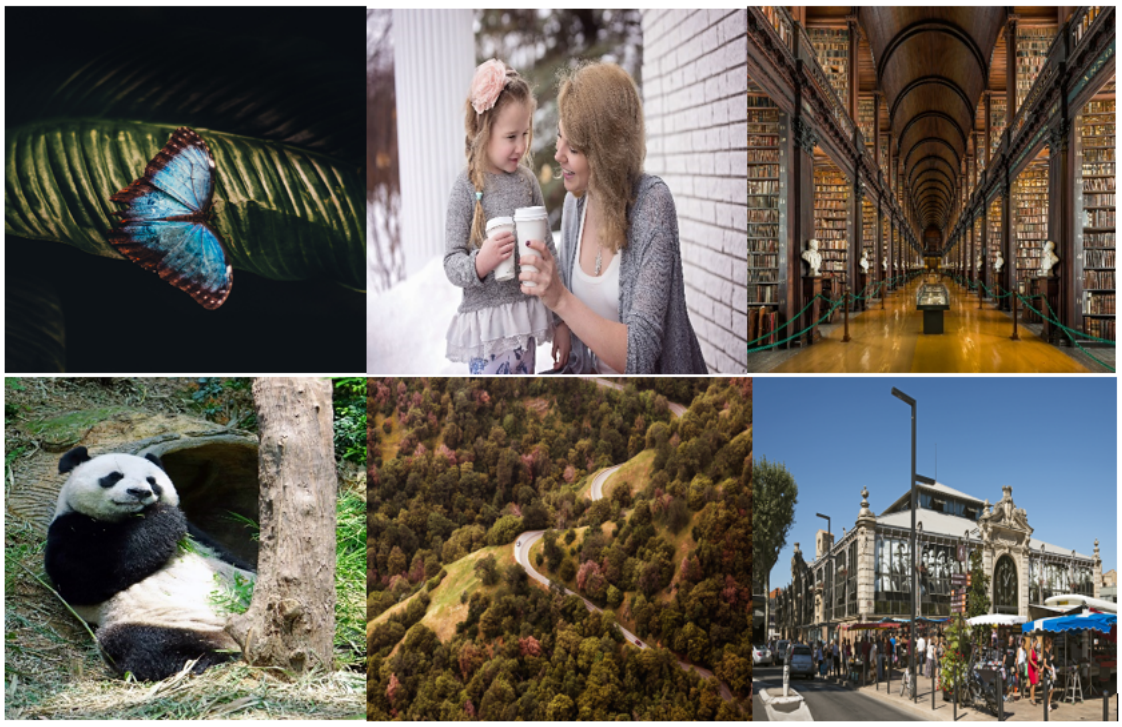

FIGURE 6.1: Sample images from Training Data Set

FIGURE 6.2: Sample images from Testing Data Set

6.2 SOFTWARE SPECIFICATIONS

6.2.1 Python

The model is programmed in Python.

It is a high-level object-oriented programming language with unified fluid semantics designed largely for the development of web and other applications. Python is quite appealing in the area of development of rapid applications since it supports dynamic type and fluid binding.It is simple, has high readability and is effcient. Compared to other languages, its syntax is easy and it provides a diverse variety of in-built functions.As a consequence, development and maintenance of the program is easier and of lesser costs. Python also permits the creation and use of modules and packages, allowing huge programs to be built in an efficient modular fashion also enabling the reuse of same code at various places reducing the development time.The modules can be easily scaled for its use in other programs and can be easily imported or exported.One of Python's most enticing qualities is that its default library and interpreter are both publicly available in the form of source and binary.No exclusivity is present either, because Python and other necessary tools are accessible on all significant platforms.As a result, it becomes an enticing choice for developers that do not want to incur high development expenditures.

6.2.2 PyTorch Framework

PyTorch is a free and open source deep learning framework built on Python and the Torch library.Its ease of usage and the ease of creating artificial neural networks on PyTorch made PyTorch popular.Its diverse built-in functions and a variety of mathematical operatons it supports are contributing factors for its popularity.Data scientists primarily utilise PyTorch for research and artificial intelligence applications.Its compatibility with Python makes it famous among Python programmers.PyTorch is notable for its outstanding GPU support and

usage of reverse mode auto differentiation, which allows computation graphs to be changed on the fly.As a result, it is a popular alternative for rapid experimentation and prototyping.

Hence, PyTorch Framework was used as the deep learning framework for the development of our model.

6.2.3 Google Colab Pro

Google Colaboratory is a free cloud-based Jupyter notebook environment that enables machine learning and deep learning models to be trained on CPUs, GPUs for parallel processing, and ML accelerators - TPUs.It provides us with a good GPU for free, which can be used constantly for 12 hours.This was sufficient for the model's requirements of computational power.Colab provides us with a continuous execution time of 12 hours before clearing the entire virtual computer and restarting it.Several instances of CPU, GPU, and TPU can run concurrently, but by sharing of resources.

Hence, Google colab was utilised for building, training and testing the model.

6.2.4 Visual Studio Code

The trained model was downloaded to be executed on Visual Studio after installing Python extension.The model downloaded can be used anytime to encode and decode the data.

6.3 EXPERIMENTS CONDUCTED

6.3.1 Text to Bits Conversion and Vice-Versa

The text converted into a list of bits. The reverse process where a list of bits into the same text.

```
Enter text : Hello World
Text to bits :  [0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0,
Bits to text : Hello World
```

FIGURE 6.3: Text and bits conversion output

6.3.2 Payload Creation

The make_payload() function outputs the payload which is a tensor of 0s and 1s with the same dimensions as the input image.

6.3.3 Image Tensor Creation

The encode() function reads the cover image and generates, permutes and converts the image tensor to the following dimensions [1,data depth, height, width].

```
payload after view change: tensor([[[[0., 1., 1.,  ..., 1., 1., 0.],
          [0., 0., 0.,  ..., 1., 0., 1.],
          [1., 1., 0.,  ..., 0., 0., 1.],
          ...,
          [0., 0., 0.,  ..., 1., 1., 0.],
          [1., 0., 1.,  ..., 1., 1., 0.],
          [0., 0., 1.,  ..., 0., 0., 0.]],

         [[0., 1., 1.,  ..., 1., 1., 0.],
          [0., 0., 0.,  ..., 1., 0., 1.],
          [1., 1., 0.,  ..., 0., 0., 1.],
          ...,
          [0., 0., 0.,  ..., 1., 1., 0.],
          [1., 0., 1.,  ..., 1., 1., 0.],
          [0., 0., 1.,  ..., 0., 0., 0.]],

         [[0., 1., 1.,  ..., 1., 1., 0.],
          [0., 0., 0.,  ..., 1., 0., 1.],
          [1., 1., 0.,  ..., 0., 0., 1.],
          ...,
          [0., 0., 0.,  ..., 1., 1., 0.],
          [1., 0., 1.,  ..., 1., 1., 0.],
          [0., 0., 1.,  ..., 0., 0., 0.]],

         [[0., 1., 1.,  ..., 1., 1., 0.],
          [0., 0., 0.,  ..., 1., 0., 1.],
          [1., 1., 0.,  ..., 0., 0., 1.],
```

FIGURE 6.4: Payload tensor

```
image at encoder: tensor([[[[-0.6549, -0.6549, -0.6784,  ...,  0.2471,  0.4745,  0.4039],
          [-0.6471, -0.7255, -0.7725,  ...,  0.3647,  0.5451,  0.4980],
          [-0.6549, -0.7176, -0.7490,  ...,  0.5373,  0.5294,  0.5529],
          ...,
          [-0.6627, -0.7020, -0.7569,  ..., -0.5451, -0.5137, -0.6471],
          [-0.6863, -0.7647, -0.7569,  ..., -0.4039, -0.4196, -0.6000],
          [-0.7333, -0.7882, -0.8039,  ..., -0.4980, -0.3098, -0.3569]],

         [[-0.3333, -0.3098, -0.3412,  ...,  0.1294,  0.4039,  0.3255],
          [-0.3255, -0.3804, -0.4196,  ...,  0.2784,  0.4588,  0.4431],
          [-0.3333, -0.3647, -0.4039,  ...,  0.4745,  0.4745,  0.4745],
          ...,
          [-0.7333, -0.7804, -0.8588,  ..., -0.4824, -0.4510, -0.6078],
          [-0.7725, -0.8588, -0.8667,  ..., -0.3020, -0.3412, -0.5529],
          [-0.8118, -0.8667, -0.8824,  ..., -0.4275, -0.2392, -0.3020]],

         [[-0.0196,  0.0118, -0.0118,  ..., -0.0118,  0.2392,  0.1608],
          [-0.0118, -0.0588, -0.0902,  ...,  0.1216,  0.3020,  0.2706],
          [-0.0039, -0.0431, -0.0667,  ...,  0.3098,  0.3020,  0.3176],
          ...,
          [-0.7882, -0.8118, -0.8824,  ..., -0.4275, -0.3725, -0.5137],
          [-0.8275, -0.8824, -0.8824,  ..., -0.2863, -0.2941, -0.4667],
          [-0.8588, -0.9059, -0.9137,  ..., -0.3804, -0.2078, -0.2471]]]])
image at encoder size: torch.Size([1, 3, 2040, 1356])
```

FIGURE 6.5: Image tensor

6.3.4 Model Training for 100 Epochs

```
[ ]  2 steganography.fit(train, validation, epochs=100)

Epoch 1/100
100%|██████████| 200/200 [00:38<00:00,  5.19it/s]
100%|██████████| 200/200 [00:38<00:00,  5.14it/s]
100%|██████████| 25/25 [00:05<00:00,  5.00it/s]
Epoch 2/100
100%|██████████| 200/200 [00:38<00:00,  5.16it/s]
100%|██████████| 200/200 [00:38<00:00,  5.14it/s]
100%|██████████| 25/25 [00:04<00:00,  5.13it/s]
Epoch 3/100
100%|██████████| 200/200 [00:38<00:00,  5.14it/s]
100%|██████████| 200/200 [00:38<00:00,  5.16it/s]
100%|██████████| 25/25 [00:04<00:00,  5.14it/s]
Epoch 4/100
100%|██████████| 200/200 [00:38<00:00,  5.21it/s]
100%|██████████| 200/200 [00:38<00:00,  5.20it/s]
100%|██████████| 25/25 [00:04<00:00,  5.06it/s]
Epoch 5/100
100%|██████████| 200/200 [00:38<00:00,  5.21it/s]
100%|██████████| 200/200 [00:38<00:00,  5.18it/s]
100%|██████████| 25/25 [00:04<00:00,  5.08it/s]
Epoch 6/100
```

FIGURE 6.6: Training of 100 epochs

6.4 ACTUAL RESULTS

Encoding:

```
Enter data:fhhgjhkj

Encoding data
```

FIGURE 6.7: Input message

The encoder result in Figure 6.10 proves that the input image fed to the encoder network and the output steganography image are identical.

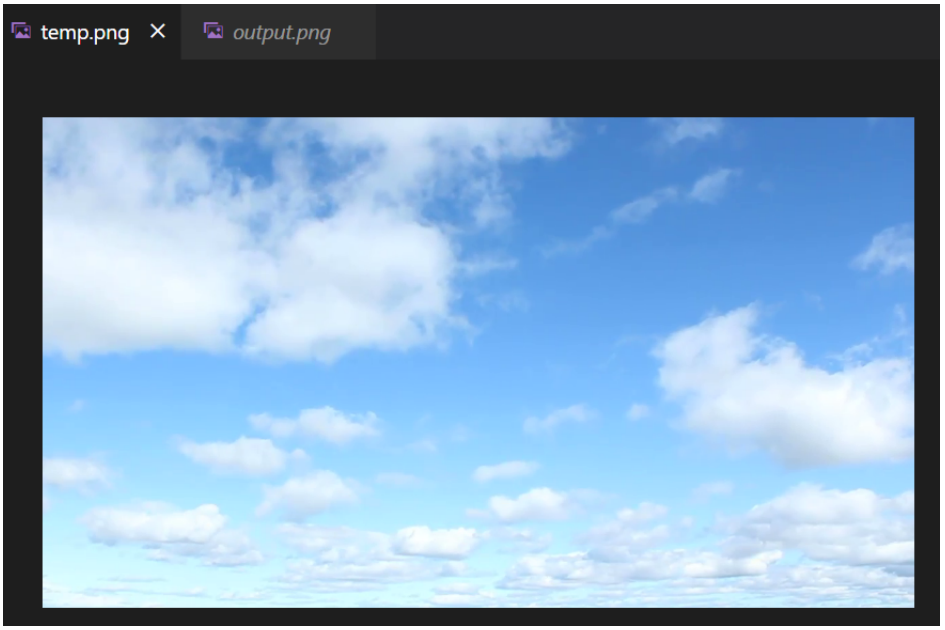

FIGURE 6.8: Input Cover image

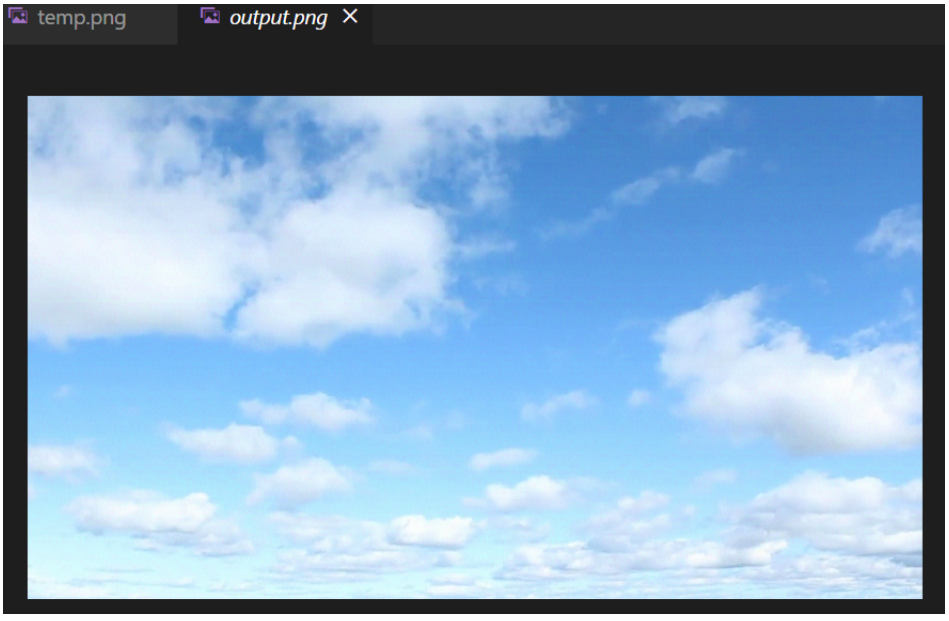

FIGURE 6.9: Output Steganography image

Decoding:

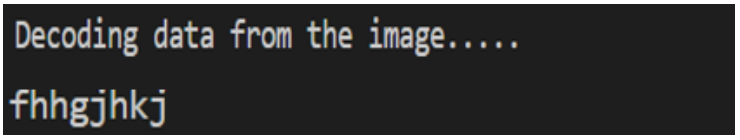

FIGURE 6.10: Output message from decoded from the steganography image

The decoder result proves that it is possible for our model to safely decode the input secret message from the steganography image.

6.5 DIMENSIONAL OUTPUT FROM THE CONVOLUTION UNITS

Encoding:

Decoding:

CONVO 2D	INPUT OUTPUT	(1,3,2040,1404) (1,32,2040,1404)
LEAKY RELU	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)
BATCH NORM	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)
CONVO 2D	INPUT OUTPUT	(1,38,2040,1404) (1,32,2040,1404)
LEAKY RELU	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)
BATCH NORM	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)
CONVO 2D	INPUT OUTPUT	(1,70,2040,1404) (1,32,2040,1404)
LEAKY RELU	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)
BATCH NORM	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)
CONVO 2D	INPUT OUTPUT	(1,102,2040,1404) (1,32,2040,1404)
LEAKY RELU	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)
BATCH NORM	INPUT OUTPUT	(1,32,2040,1404) (1,32,2040,1404)

TABLE 6.1: Encoding

CONVO 2D	INPUT	(1,3,2040,1404)
	OUTPUT	(1,32,2040,1404)
LEAKY RELU	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
BATCH NORM	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
CONVO 2D	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
LEAKY RELU	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
BATCH NORM	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
CONVO 2D	INPUT	(1,64,2040,1404)
	OUTPUT	(1,32,2040,1404)
LEAKY RELU	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
BATCH NORM	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
CONVO 2D	INPUT	(1,96,2040,1404)
	OUTPUT	(1,32,2040,1404)
LEAKY RELU	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)
BATCH NORM	INPUT	(1,32,2040,1404)
	OUTPUT	(1,32,2040,1404)

TABLE 6.2: Decoding

6.6 LIMITATIONS

- The data to be hidden should be compact.
- Higher overhead than Cryptography.
- 8 GB RAM would result in faster working and hence faster results.

CHAPTER 7

PERFORMANCE ANALYSIS

7.1 METRICS USED FOR THE EVALUATION OF THE MODEL

For comparing steganography image with cover image the following metrics are commonly used :

1. Payload
2. Loss
3. Accuracy
4. Peak Signal to Noise Ratio (PSNR)
5. Structural Similarity Index (SSIM)

1. **PAYLOAD CAPACITY:** The maximum size of a message that can be inserted in a cover image is defined as payload capacity.Bits per pixel (bpp) measure is commonly used to determine payload capacity.

   BPP = No.of secret bits embedded in an image element / Total pixels in the cover image

2. **LOSS:** There are different types of loss functions based on the problem statement.Loss functions compute the distance between an expected value

and its actual value.A loss function connects decisions to their costs. Loss functions change depending on the problem under study and the outcomes that are expected.

The loss function in each epoch is analysed to understand how the gradient is descending.

The loss functions used in our model:

(a) Binary Cross entropy with Logits

(b) Mean Squared error

These functions were provided in-built with the PyTorch Framework.

3. **ACCURACY:** The percentage of correct predictions for the test data is known as accuracy.It is simple to calculate by dividing the number of right guesses by the total number of forecasts.A deep learning model's accuracy is the metric used to determine which model is best at spotting correlations and trends between features in a dataset according to the input or training data.The better a model generalises to new data, the more precise forecasts and insights it may create, resulting in more commercial value.Errors might be costly, but improving model accuracy lowers that cost.

4. **PEAK SIGNAL TO NOISE RATIO (PSNR):** Peak signal to noise ratio (PSNR) is a measurement for the ratio of a signal's maximum possible value which is its power to the strength of warping noise that influences the quality of its depiction. According to the idea, the greater the PSNR, the better the degraded image has been recreated to match the original image and the superior the reconstructive method.This would be the case since it

was designed for reducing the MSE between images with respect to the image's maximum signal value.

PSNR is calculated as :

$$10 * logbase10(MAXI^2/MSE)$$

where MAXI is the image's maximum potential pixel value.

5. **STRUCTURAL SIMILARITY INDEX MEASURE (SSIM):** The Structural Similarity Index (SSIM) is a subjective metric that estimates image quality degradation due to compression of data or transmission losses.It is an all-encompassing metric that necessitates the capture of two images from the same image capture, namely a original image and an image that has underwent some alterations.In our proposed method, bits encoded.Typically, the altered image is compressed.It can be obtained, for example, by storing a original image at any quality level and then reading it back in. SSIM is well recognised in the video business, but it has significant implications in still photography as well.Unlike PSNR, SSIM is based on visible picture structures.

   The SSIM index is obtained on several image windows.The distance between two windows x and y of the same size N x N is:

$$(2\mu_x\mu_y + c_1)(2\sigma_xy + c2)/(\mu_x^2 + \mu_y^2 + c1)(\sigma_x^2 + \sigma_y^2 + c2)$$

   where,

μ_x	The pixel sample mean of x.
μ_y	The pixel sample mean of y.
σ_x^2	Variance of x.
σ_y^2	Variance of y.
σ_{xy}	Co variance of x and y.
$c1 = (k_1 L^2)$	Variable to stabilize the division with weak denominator.
$c2 = (k_2 L^2)$	Variable to stabilize the division with weak denominator.
L	is the dynamic range of the pixel values.
$k_1 = 0.01$	Default
$k_2 = 0.03$	Default

TABLE 7.1: SSIM Variables

7.2 OBTAINED RESULTS

After training for 100 epochs, the performance evaluation is as below:

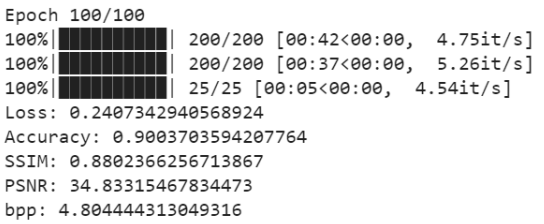

FIGURE 7.1: Performance of the model on 100th epoch

Epochs	Loss	Accuracy	SSIM	PSNR	bpp
Epoch 1	0.56	0.70	0.90	36.97	2.37
Epoch 25	0.30	0.83	0.88	37.22	4.00
Epoch 50	0.25	0.86	0.84	36.37	4.32
Epoch 75	0.25	0.90	0.78	36.51	4.80
Epoch 100	0.24	0.90	0.88	34.83	4.82

TABLE 7.2: The efficiency of the model calculated at respective epochs

1. **Payload Capacity:** The payload capacity of the model was increased to 4 bits per pixel.

2. **Loss:** The loss of the model was reduced to 24%.

3. **Accuracy:** The obtained accuracy of the model is upwards of as 90%.

4. **PSNR:** PSNR value was attempted to maintain an optimum of 35%.

5. **SSIM:** The similarity measure of the Cover and Steganography images is of 88%

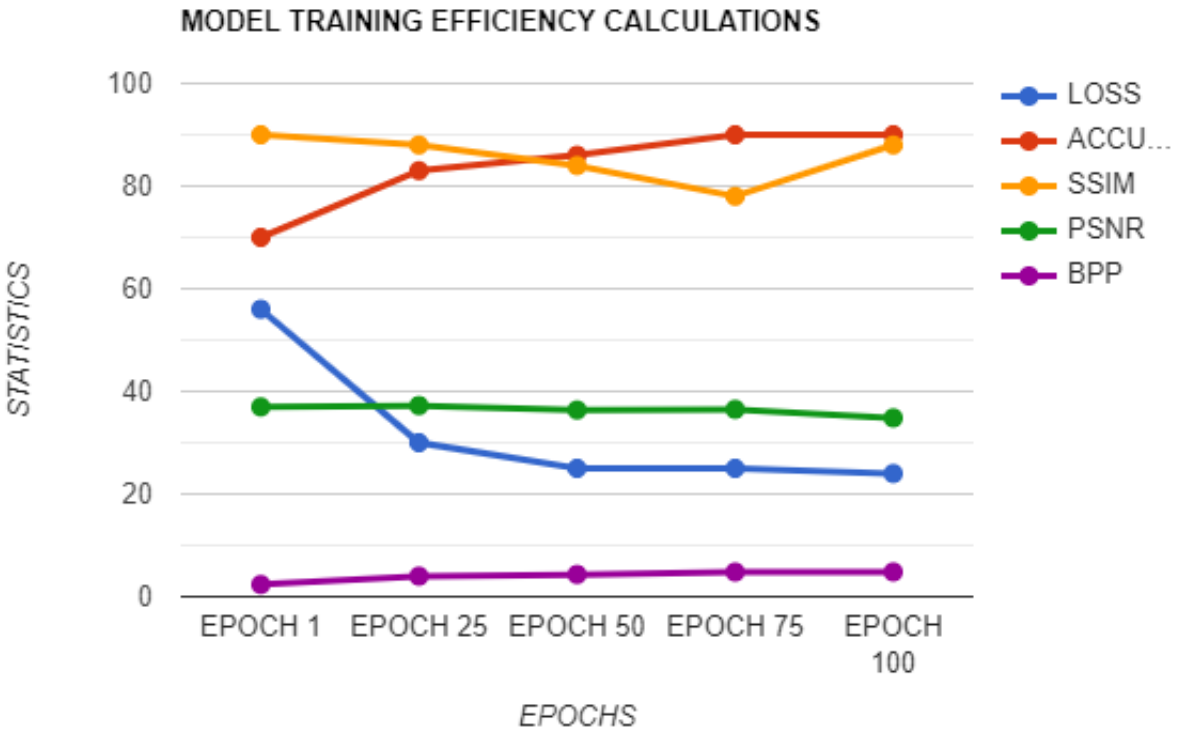

FIGURE 7.2: Line graph of model's evaluation statistics

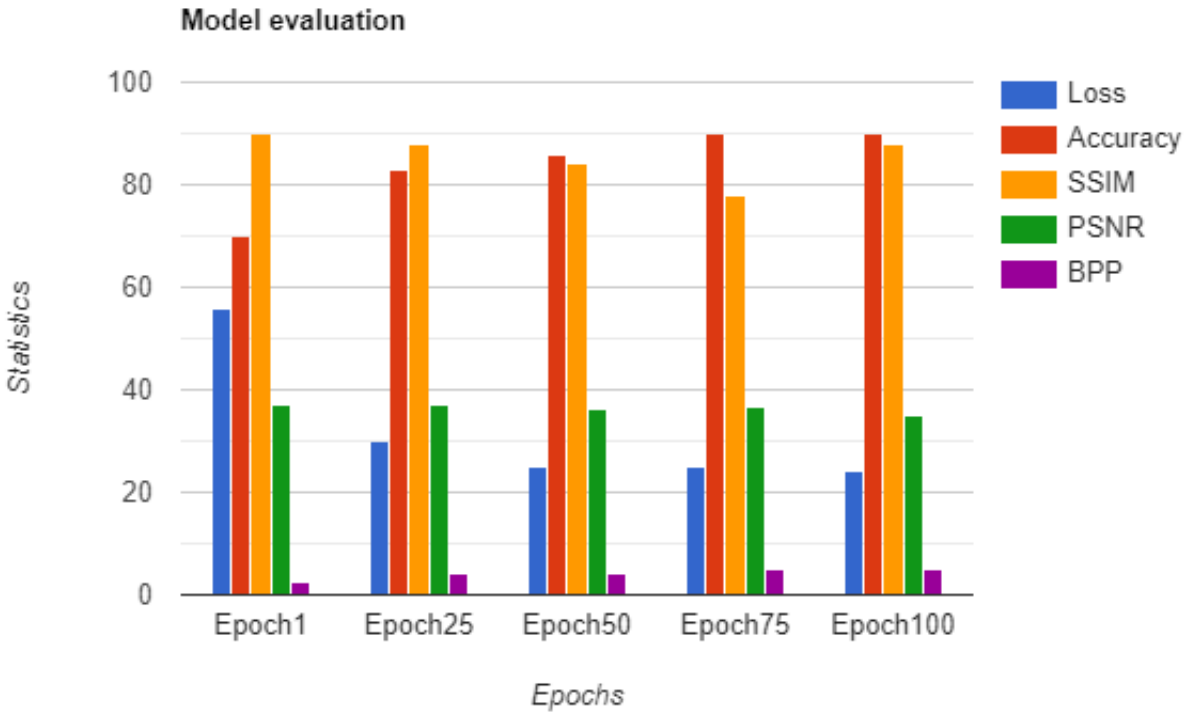

FIGURE 7.3: Bar graph of model's evaluation statistics

From the above graphs, we infer the following:

1. Payload Capacity is increasing.
2. Loss is decreasing.
3. Accuracy is increasing.
4. PSNR is minutely decreasing.
5. SSIM is variating and is maintained at an optimum.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

Our proposed work has demonstrated that using Generative Adversarial Neural Networks for Image Steganography with efficient training produces Steganography images that are realistic and that are identical to the cover images without any significant changes in the characteristics of the cover image.Hence, making the Steganalysis process indeed difficult.

Also, our project enhanced the amount of data that could be encoded into the image to 4 bits per pixel by using FractalNet Architecture.The proposed Steganography process using GAN can be used efficiently where the data is compact and is to be discreet.

By stacking of more layers and by efficient grouping of such layers, the model can be furthermore enhanced to produce better results.

REFERENCES

1. B. Sultan and M. A. Wani (2022) ”Multi-data Image Steganography using Generative Adversarial Networks”, 9th International Conference on Computing for Sustainable Global Development, pp.454-459.

2. S. D. Desai, N. Patil, S. R. Nirmala, S. Kulkarni, P. D. Desai and D. Shinde (2022) ”Deep Neural Network based Medical Image Steganography”, International Conference on Smart Technologies and Systems for Next Generation Computing, pp.1-5.

3. M. K. Linga Murthy, P. B. Madhavi, S. A. Ahamad, K. Vamsi and Y. Mallikarjuna Rao (2022) ”Implementation and Analysis of Image Steganography using Convolution Neural Networks”, International Interdisciplinary Humanitarian Conference for Sustainability, pp.108-113.

4. A. G. Devi, A. Thota, G. Nithya, S. Majji, A. Gopatoti and L. Dhavamani (2022) ”Advancement of Digital Image Steganography using Deep Convolutional Neural Networks”, International Interdisciplinary Humanitarian Conference for Sustainability, pp.250-254.

5. B. Wei, X. Duan and H. Nam (2022) ”Image Steganography with Deep Learning Networks”, 13th International Conference on Information and Communication Technology Convergence (ICTC), pp.1371-1374.

6. Kuri J. L. (2020) ”Securing Data in Internet of Things (IoT) using Cryptography and Steganography Techniques”, International Journal for Research in Applied Science and Engineering Technology, pp.1933-1939.

7. M. Boroumand, Chen M. and Fridrich J. (2019) ”Deep Residual Network for Steganalysis of Digital Images”, IEEE Transactions on Information Forensics and Security, pp.1181–1193.

8. Hassaballah M., Hameed M. A., Awad A. I. and Muhammad K. (2021) ”A Novel Image Steganography Method for Industrial Internet of Things Security”, IEEE Transactions on Industrial Informatics, pp.1–1.

9. Karati A., Islam S.H. and Karuppiah M. (2018) ”Provably Secure and Lightweight Certificateless Signature Scheme for IIoT Environments”, IEEE Transactions on Industrial Informatics, pp.3701-3711.

10. Li Y., Zhang F. and Liu X. (2020) ”Secure Data Delivery with Identity-based Linearly Homomorphic Network Coding Signature Scheme in IoT”, IEEE Transactions on Services Computing, pp.1–1.

11. Liao X., Yin J., Chen M. and Qin Z. (2020) ”Adaptive Payload Distribution in Multiple Images Steganography Based on Image Texture Features”, IEEE Transactions on Dependable and Secure Computing, pp.1–1.

12. Lin Z., Huang Y. and Wang J. (2018) ”RNN-SM: Fast Steganalysis of VoIP Streams Using Recurrent Neural Network”, IEEE Transactions on Information Forensics and Security, pp.1854–1868.

13. Lu W., Chen J., Zhang J., Huang J., Weng J. and Zhou Y. (2022) ”Secure Halftone Image Steganography Based on Feature Space and Layer Embedding”, IEEE Transactions on Cybernetics, pp.5001–5014.

14. Singh B., Sur A. and Mitra P. (2021) ”Steganalysis of Digital Images Using Deep Fractal Network”, IEEE Transactions on Computational Social Systems, pp.599–606.

15. Su W., Ni J., Hu X. and Fridrich J. (2021) ”Image Steganography With Symmetric Embedding Using Gaussian Markov Random Field Model”, IEEE Transactions on Circuits and Systems for Video Technology, pp.1001–1015.

16. Tang W., Li B., Tan S., Barni M. and Huang J. (2019) ”CNN-Based Adversarial Embedding for Image Steganography”, IEEE Transactions on Information Forensics and Security, pp.2074–2087.

17. Tedeschi P., Sciancalepore S., Eliyan A. and Di Pietro R. (2020) ”LiKe: Lightweight Certificateless Key Agreement for Secure IoT Communications. IEEE Internet of Things Journal”, pp.621–638.

18. Wang W., Xu P., Liu D., Yang L.T. and Yan Z. (2020) ”Lightweighted Secure Searching Over Public-Key Ciphertexts for Edge-Cloud-Assisted Industrial IoT Devices”, IEEE Transactions on Industrial Informatics, pp.4221–4230.

19. Wu, S., Zhong, S. and Liu, Y. (2020). A Novel Convolutional Neural Network for Image Steganalysis With Shared Normalization. IEEE Transactions on Multimedia, 22(1), pp.256–270. doi:10.1109/tmm.2019.2920605.

20. Zhang, R., Zhu, F., Liu, J. and Liu, G. (2020). Depth-Wise Separable Convolutions and Multi-Level Pooling for an Efficient Spatial CNN-Based Steganalysis. IEEE Transactions on Information Forensics and Security, 15, pp.1138–1150. doi:10.1109/tifs.2019.2936913.