IoT Network Communication Security Using Steganography And Cryptography System

MSc Research Project
Cybersecurity

Anugraha Sureshkumar
Student ID: x21154325

School of Computing
National College of Ireland

Supervisor: Muhammad Salahuddin Jawad

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Anugraha Sureshkumar……………………………………………………………………………………

Student ID: ………x21154325……………………………………………………………………………………………………….……

Programme: …Cybersecurity……………………………………………………… **Year:** ……………2022……………..

Module: Research Project…………………………………………………………………………………………….………

Supervisor: …… Muhammad Salahuddin Jawad ………………………………………………………………………………………………….………

Submission Due Date: ………………………15/12/2022………………………………………………………………………………….………

Project Title: IoT Network Communication Security Using Steganography And Cryptography System ………………………………………………………………………………………….………

Word Count: ………………6321………… **Page Count**…17……………………………………………….…….

I hereby certify that the information contained in this (my submission) is research information I conducted for this project. All information other than my contribution will be fully referenced and listed in the relevant bibliography section at the project's rear. <u>ALL</u> internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other authors' written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ………………Anugraha…………………………………………………………………………………………………

Date: …14/12/2022……………………………………………………………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online submission to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on the computer.	□

Assignments that are submitted to the Programme Coordinator's Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	

Date:	
Penalty Applied (if applicable):	

IoT Network Communication Security Using Steganography And Cryptography System

Anugraha Sureshkumar(x21154325)
[1] National College of Ireland, Mayor Street, IFSC, Dublin 1, Ireland.

Abstract.
Sensors, autonomous processing, software, and other technology integrated into everyday things are what the Internet of Things refers to. Typically, these things will be able to communicate with one another and exchange data with other online devices and systems. Businesses can freely access IoT devices. As a result, it's crucial to assess the potential dangers of digitization and data security concerns. Cryptography is used to transform the data into a hash key before it is stored in the image. In this procedure, data is first converted from plain text into a key before transmission and then decrypted using the received key. As a result, the encoded image will look slightly different from the original. The work described demonstrates the use of cryptography to secure the network's data, steganography to conceal a message within a picture, and image compression to increase storage capacity. It has been found that the amount of data that can be communicated through a picture steadily increases when deep learning is properly applied to steganography. The technique is then modified to increase the payload size while still allowing for the secure decoding of the original message.

Keywords – IoT Network Security, Deep Learning, Steganography, bpp, cryptography

1 Introduction

IoT links devices to the internet. The internet of things is a network of linked devices and people that collect and share data. IoT devices comprise sensors and small processors that use sensor data. IoT gadgets accomplish much work without human intervention, therefore unemployment may rise. Despite becoming acclimated to IoT gadgets. The Internet of things makes life, work, and administration easier. It's also important from a business perspective. IoT connects internet-connected devices. An ongoing investigation of how their structures work, providing information on device administration, supply chain, and logistics. The Internet of things automates manual real-times to reduce job expenses and time. IoT devices are small, internet-connected computers vulnerable to malware and attack. IoT improves worker productivity. IoT devices improve operation management. Clarify marketing and business growth. The IoT's wireless transmission of information raises privacy concerns daily. A stable, safe system is needed to protect sensor data and wirelessly controlled industrial equipment from tampering or hacking. Internet-connected devices are widely available. Thus it's important to be aware of security concerns and digital assault risks. Despite IoT's benefits, wirelessly transferred data between network nodes remains insecure. It's hackable.

To address this issue, we developed a strategy that combines Steganography with cryptography to ensure the most secure data transmission across wireless networks.

Although widespread knowledge of steganography exists, its practical application is limited due to its drawbacks. It's purpose is to cloak sensitive information within an image. This concept, however, has a major flaw in that it is processed using antiquated, inflexible algorithms that prevent it from being integrated with modern methods. Because of this, it can't be used in real-time situations. The fact that it is the most secure method of data protection does not change the fact that it is cumbersome to employ due to this issue. To circumvent this issue, we will examine how to implement convolutional neural networks into the steganography process. Steganography and cryptography are the most secure methods for preserving information.

Steganography refers to the practice of concealing data through a visual medium. A hash key, or encrypted data, is a form of cryptographic transmission that transforms plain text. After deciphering the encrypted information, the message is revealed. Using steganography, a method for concealing data as a secret hash key, the encoded secret data can be hidden within an image. Therefore, the photograph that is sent cannot be hacked.

Research Question. The above research problem motivates the following research questions:

1. Does the data of IoT networks are secure?
2. What type of data can be stored in this?

The organisation of the Thesis. The following thesis is organised in a manner such that the coming up chapter will discuss about the literature review done with various papers, chapter 3 will speak about the proposed methodology and design followed, chapter 4 will discuss the results obtained and chapter 5 will conclude the project as required.

2 Related Work

Using diamond-encoded prediction unit (PU) partition modes, Jindou Liu et al. (2015) developed a novel multilevel steganography algorithm. Guidelines such as diamond coding were developed to increase the information capacity of partition modes. Convolutional neural networks have since surpassed the efficacy of this antiquated technique [15]. Convolutional neural networks outperformed previous approaches in terms of the volume of data that could be stored and the quality of the images produced. As a result, neural networks are now a more viable option for use in steganography. The analysis relied solely on the PSNR values of the photos. There is currently no accepted method for estimating how much information can be kept. Our proposed strategy incorporates both PSNR and payload estimations for reliable outcomes.

The authors R'emi Cogranne et al. (2021) demonstrated a practical application of the DCT coefficients in steganography. Their primary objective is to weaken the detection capabilities of this "omniscient detector," which is the worst-case scenario in terms of security. A safe and difficult-to-decode steganography method was developed for this study. As an example, when a data-hidden image is accessible, and when an uncompressed image is available, these are the two primary use cases considered. Instead of aiming to boost the performance of the current model to store more data or make the image clearer, the method employed in this study investigates how much data can be contained in an image [5]. They also discuss the issues with the independent coefficients assumption used in determining the DCT coefficients. Moreover, its efficacy is demonstrated by contrasting it with more conventional algorithms like MiPOD, UERD, and J-MiPOD. No effort is taken to improve steganography-based data storage, and the method isn't particularly excellent at predicting where data is stored, therefore it's not very useful for finding it.

Lightweight certificates key agreement is proposed for protecting communications between IoT devices by Pietro Tedeschi et al. (2020). The latest Zigbee 3.1 convention stack and IoT devices are supported by this lightweight matching free certificates key understanding convention. It has been proven that normal devices may be used to demonstrate the security qualities of certificate-free key agreement, making it an ideal solution for securing communications across the Internet of Things [32]. We have also successfully deployed and tested it on Internet of Things gadgets in multi-hub networks with as many as thirteen nodes. The code's original creators have released it to the public. The results are surprising: lightweight for secure Internet of Things connections only needs 4.659 seconds and 0.268% of the total battery capacity to set up keys on each device on the open-mote-b equipment stage. Finally, the arrangement correlations demonstrate the proposal's primary quality and viability. Protecting data kept on the domain authority and preventing man-in-the-middle attacks are both made easier with lightweight certificates key agreement for secure Internet of Things communications. Without the user's public key, it is impossible to deduce the private one.

Manju Khari et al. (2020) discuss the risks associated with the daily sharing of data over the internet. In this paper, we will talk about how we plan to use cryptography and steganography to fix our security flaws. It entails vital processes while protecting sensitive client data [10]. Consideration of cryptography's applications has been made. It is used to disentangle data from several sources so that unique keys may be generated for each.

Depending on the outcomes, the current procedure takes a broader view of the possible bounds. At this point, we just need to decode the image data. The Matlab test system used at the EGCI conference successfully hid 92% of the data. As a result, it allows for larger messages to be encoded than RSA1 encryption.

Daily developments in IoT and how remote network innovation is increasing data transmission are discussed by Yumei Li et al. (2020). An important development in IoT businesses, appropriate coding enhances data transmission rates [11]. It is important for businesses that transmit data from remote sensors to understand network coding in order to protect their customers' personal information. This approach provides defence against subspace mistakes and methods of fraud that rely on dynamic data. The new approach is more efficient and generates a healthy amount of data flow because it uses Java-based cryptography and its library. However, the construction is flawed since it relies on an inefficient direct linear homomorphic signature plot.

Xianyi Chen et al. (2020) discuss how concealing information in an image affects its overall quality. As a result, they presented a coverless steganography-based solution to maintain the integrity of the cover image used to conceal the data. For this purpose, we employ the Star Generative Adversarial Network, which restores the image to its original state. Star GAN is used to reconstruct the image with high quality without damaging the base image, after the secret information has been saved in an image and the other half has been linked via image mapping depending on the attributes of the image. The performance is quite good, and the method has improved both the quality and the quantity of concealed capacity. The method may not perform well in terms of quality and data storage limit, and it is limited to face photos, even if it is capable of reconstructing images in general.

Self-comparability [25] is the basis of the deep model fractal-net suggested by Brijesh Singh et al. (2021). The fractal-net model, which employs steganalysis recognition, uses the embedded figure as input. In a positive situation, as the breadth and depth of the network increase, so does the number of test figures that can be found by steganography. To maintain a consistent depth-to-width ratio, a central fractal block is reused during the deep learning process. The proposed model has been shown to be superior to others through an extensive battery of testing. The network's fractal model allows it to expand both in depth and breadth, leading to improved accuracy in detection. Since the payload fractal-net receives is just 0.5bpp, it is unable to get optimal results when processing large amounts of data with various algorithms.

In order to conceal a stego message and trick a convolutional neural network-based step analyzer, Weixuan Tang et al. (2019) present a steganographic approach with a clever activity dubbed ADVI-EMB. The proposed method is based on the principle of minimising data. ADVI-EMB, in particular, adapts the costs of modifying image components in response to the goals of the convolutional network steganalyzer [18]. The steganographic layout provides more protection from the steganalyzer, as evidenced by the results of tests, because the number of times it fails to find something increases. Additionally, it deconstructs the presentation of competing forms of enemy-aware steganography, clearing the way for a new category of steganographic strategies that can compete with and even outperform those based on powerful convolutional networks. ADVI-EMB has a greater rate of incorrect location estimates compared to the state-of-the-art benchmark steganographic system for both on-target and off-target steganalysis. The ADVI-EMB technique may produce subpar results due to its focus on gradient indications alone.

An intermittent neural network is the basis of the code word correlation model discussed by Zinan Lin et al. (2018). Cover and stego picture feature classifications are also highlighted using a feature classification approach. Based on the intermittent neural organisation, the steganography model as a whole is implemented within a managed learning framework [31]. Full testing demonstrate that the intermittent neural organization-steganalysis model has high recognition accuracy, maintaining over 90% even when the test is as brief as 0.2s and generally outperforming other best-in-class methodologies. Surprisingly, given how challenging it is to apply steganography for low-rate installation tests, the intermittent neural organization-steganalysis model is also quite accurate. Less than 0.25% of the total test time is spent on testing each example. These hints verify that the online voice-over-IP protocol steganalysis model based on intermittent neural networks satisfies the brief example identification interest. This technique is superior to others at detecting brief steganographic instances; in fact, it can achieve an accuracy of 95% or higher even when the sample is only 0.2 seconds long. The use of enactments makes it more challenging to set up recurrent neural networks on a lengthy timeline, which is a disadvantage.

As deep convolutional neural networks have established a strong footing for themselves as superior to previous recognition viewpoint classifiers in light of rich media models, Mehdi Boroumand et al. (2019) detail how the steganography IDs functioned. Hand-planned components can still be found in currently used organisation structures such as fixed or forced convolutional bits, the edge direct unit that copies in rich models, quantization of element guides, and awareness of the image stage. Because it provides precise position data for both spatial area and picture steganography [25], deep leftover engineering is frequently used. This is due to the fact that it restricts the usage of algorithms and components that can be run from afar. The prolonged forward section of the finder, which analyses pooled data to conceal the stego signal, is the most crucial aspect of the design. Extensive studies have shown an overall improvement in the public's perception of this company. More aid is seen with execution when the determination channel is made the next channel. For payloads between 0.5 and 0.7 bits per pixel, the first network layer increase is around 2%, and for the smallest payload tried, 0.2 bits per pixel, it is always around 4%. Accuracy is poor for complicated information, and the way individuals interact is both perplexing and dull. The system managed to double the profit (from 1% to 4%) however the process is very slow.

We've determined that the present practices are founded on antiquated, outmoded practices that are useless in the emerging technology arena. The time it takes to put them to use or make a decision is lengthened because of this. Adjusting for halftone images and correcting for distortion are primary concerns for older algorithms. It is difficult to merge these methodologies with real-time systems due to the outdated nature of the tools and procedures used. The main drawback of the CNN-based approach is that they have only managed to achieve a payload (or the quantity of data that can be stored) of roughly 0.5bpp. The widespread recommendation of fractal networks, which leads to a compact layout of the algorithms but reduces their efficiency, is a common criticism of fractal methods. Even with data augmentation, when additional data is added to the training set, existing methods still don't achieve the required accuracy.

3 Methodology

The methodology involved is developing a steganography model to encode and decode the data successfully which will be developed using convolutional neural networks. The proposed research provides a methodology to encode the information transmitted via the internet of things network after converting it to a hash key using the cryptography technique and hiding it inside the image using steganography methodology. This proposed work also aims to improve the data storage capacity of the images measured in terms of bits per pixel. The data that is going to be stored inside the image can be termed in the units of bits per pixel and this tells how many bits can be stored in each pixel. This method will overcome the existing systems storage capacity which is only 0.5 bits per pixel. The increase in the payload or storage is achieved with the help of deep learning in the steganography technique aiming to increase the data storage capacity of an image. Then the neural networks algorithm is enhanced and updated than the normal algorithm to increase the data storage, by which we can securely encode as well as decode the data back in its original form. For training the steganography to a model using deep learning we basically need data and here we use Div2K dataset which is a high-resolution image dataset containing 1000 images with two categories namely train and validation. Train data will be used during the training process and validation during the evaluation process. After successfully obtaining the model, one can perform encoding of the text data and decoding of text information into the image. The tool that is used for implementing the work is Google Colab which is an online platform providing us with the required specifications to work on python based development.

The design of the overall algorithm which we aim to develop can be seen in the below figure

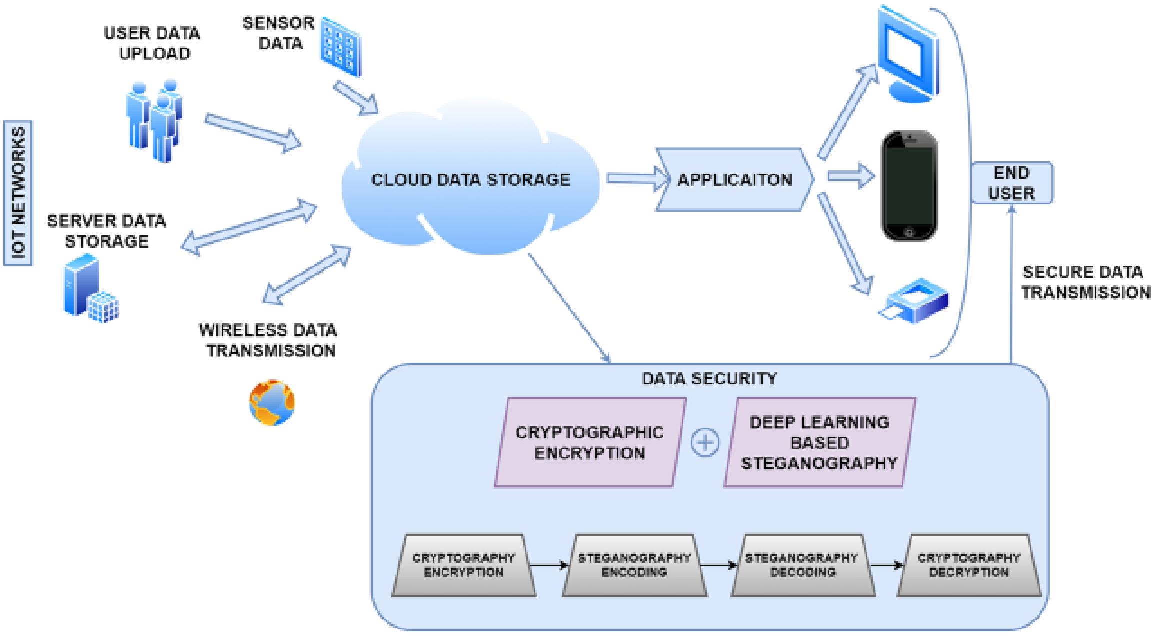


Fig 1. Research Methodology

4 Design Specification

The way things work is shown in the picture above, which is figure 1. To do this, we will buy some modules from which we expect step-by-step implementation and results before moving on to the next step. These modules are described below.

i. Dataset Collection

In the proposed work, the collected dataset will be used to train deep learning algorithms. Deep learning has become the go-to method for a lot of testing problems. Deep learning can figure out what's important about each pixel in a picture. These machines that have been doing a great job of deep learning need fuel, and that fuel is information. Google has even looked at the possibility that more information leads to better performance with a huge dataset of 400 million pictures. When sending a deep learning model in a real application, information should always be taken care of so that the model can keep showing off its skills. Also, information seems to be the most valuable thing in deep learning analysis. Usually, the information used is split into information for getting ready and information for the test. The preparation dataset has a proven result, and in the future, the network that uses this data will need to include a wide range of information. It uses the Keras strategy and the training dataset to train the network. It does this by calling library functions in Python. In this work, the steganography process was trained on a well-known dataset called Div2k. It has a mix of train and validation images that are used to teach the training algorithm how to use the data and learn things like what an image is, how to hide data in it, and how to get the hidden data out of a steganography image. gather information in the form of learned features such as what is an image, how to encode data inside it and how to decode data from a steganography image.

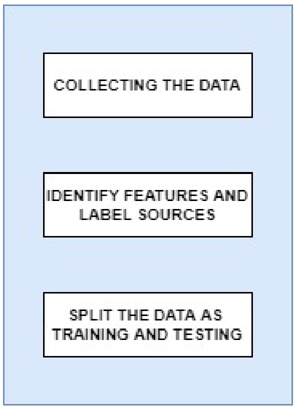

COLLECTING THE DATA
IDENTIFY FEATURES AND LABEL SOURCES
SPLIT THE DATA AS TRAINING AND TESTING

Fig 2. Dataset Collection

ii. Image Reading And Writing

In this unit, we get to put in place the fundamentals of reading and writing images to and from a folder, respectively. OpenCV is one such way used directly as a library; it includes pre-installed libraries for reading and writing images automatically. Many images in OpenCV() have a blue-green-red colour scheme. When working with images, reading and writing may be done quickly and easily with the help of OpenCV's library. Data can be read from a folder, written to a folder, etc. using specific methods. That requires using the rule's syntax in the language. These OpenCV libraries will be used to read in photos for the training process.

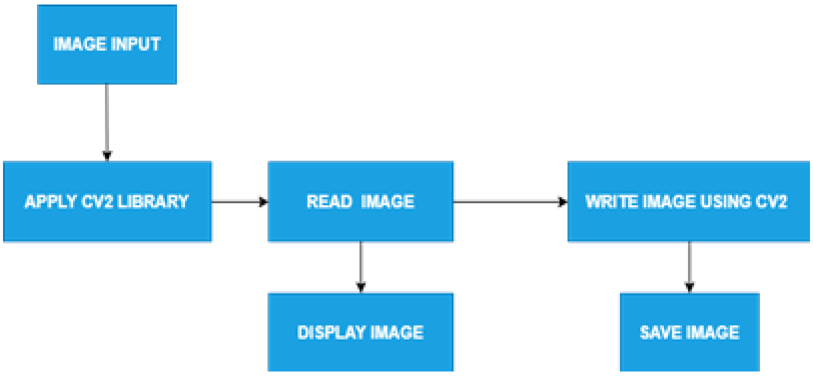


Fig 3. Image Read and Write

iii. Text To Bits Conversion

Text to bits conversion module is mainly used for converting the text to bits before storing it inside the image. Text data cannot be directly fed to an image, hence it is converted to bits using various mathematical techniques involved in python language so that it can be easily stored inside the image in terms of zeroes and ones. Data which is stored inside the pixel will have to fit between the value of that particular pixel. Hence this should be given care as the data is stored in between the pixel values. For example, 1101 is processed and links to the data such as a letter or the number of a specific pixel. These things are determined using the deep learning-based Keras environment which includes a lot of algorithms inbuilt for this kind of conversion. This assists in generating appropriate data. One such technique is one hot function utilized for these purposes. This function helps in returning an integer-based output for any record of data. After this when we get the original data in terms of bits we can process it for storing it further by following the next steps.

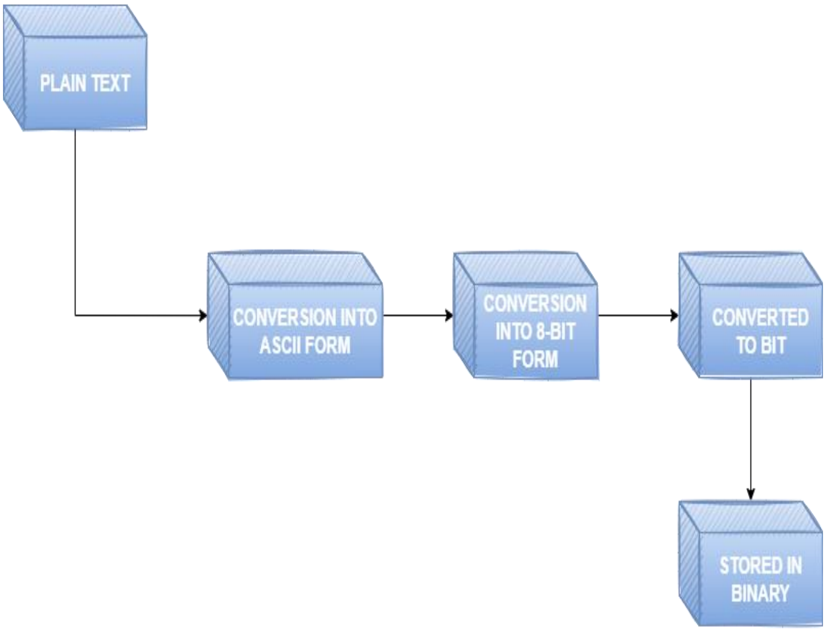


Fig 4. Bit Conversion Process

iv. Deep Learning Algorithm

Convolutional neural networks, the focus of deep learning, are a type of neural network whose shape and function have inspired techniques in the field of machine learning. Making up a deep learning algorithm on the spot is a highly rewarding learning process. It's best to start with the basics while developing algorithms, as there are few that are more complicated than others. The supplementary 6-step cycle for composing untutored mathematical compositions is

1. Get an essential comprehension of the calculation
2. Find some unique learning sources
3. Break the calculation into pieces
4. Start with a straightforward model
5. Validate with a confided-in execution
6. Write up the cycle

When it comes to model creation, Deep Learning Steganography is a type of convolutional neural network that forgoes the use of residual connections. Among these techniques is the recycling of an application of a direct extension rule to generate a deep communication network, the designs of which unmistakably truncate the model. Sub-paths of varying lengths cooperate within these networks of communication; nevertheless, neither pass-through nor residual connections are present, and each internal signal is modified by a channel and non-linearity before being observed by successive layers. The networks achieve parity with the spectacular performance of a normal neural network communication network trained on the Div2k dataset, demonstrating the significance of the algorithm structure to the development of deep convolutional neural network design. Convolutional layers, a pooling layer, a batch normalisation layer, and an activation layer are all part of the proposed algorithm's structure. These are examples of the layers that can be found in a deep neural network. In any algorithm, the accuracy and other metrics will change depending on how these layers are arranged algorithmically. The below diagram depicts the proposed framework for the arrangement.

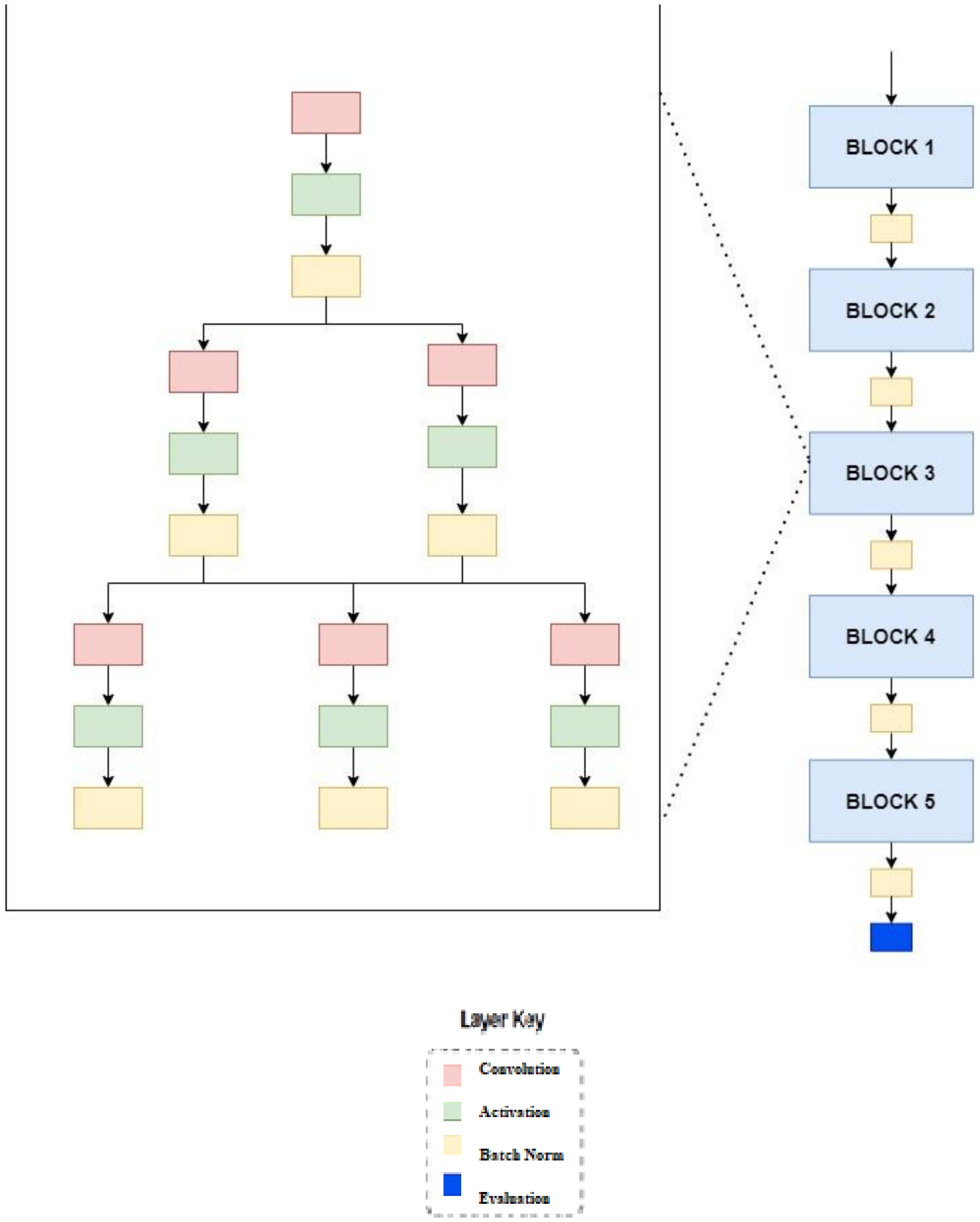


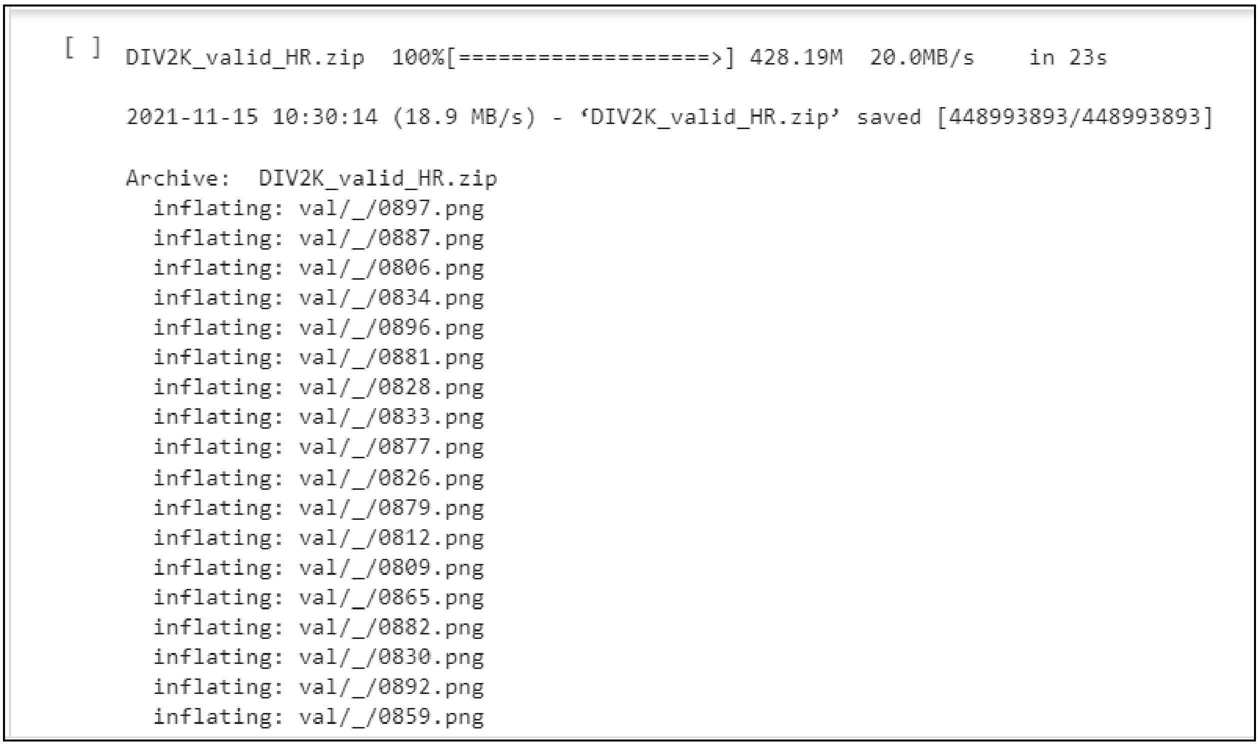
Fig 5. Deep Learning Steganography Architecture

In the above figure, the blocks are arranged in a manner and each block is considered a network which involves specific layer arrangement. The arrangement of the layers is given on the left side of the diagram. In this we can see that initially convolutional layers are used for training, then comes the activation layer to remove unwanted noises, and the arrangement goes on. After this, the output of each block is given to a batch normalization layer which distributes the dataset to be learned. Then comes the evaluation layer which predicts the trained features so that it can give the outputs in terms of accuracy and loss.

5 Implementation

We can divide the solution into several module types to initiate the implementation process.

The collection of datasets is based on the process of downloading datasets from relevant links. The following image can be seen as follows.

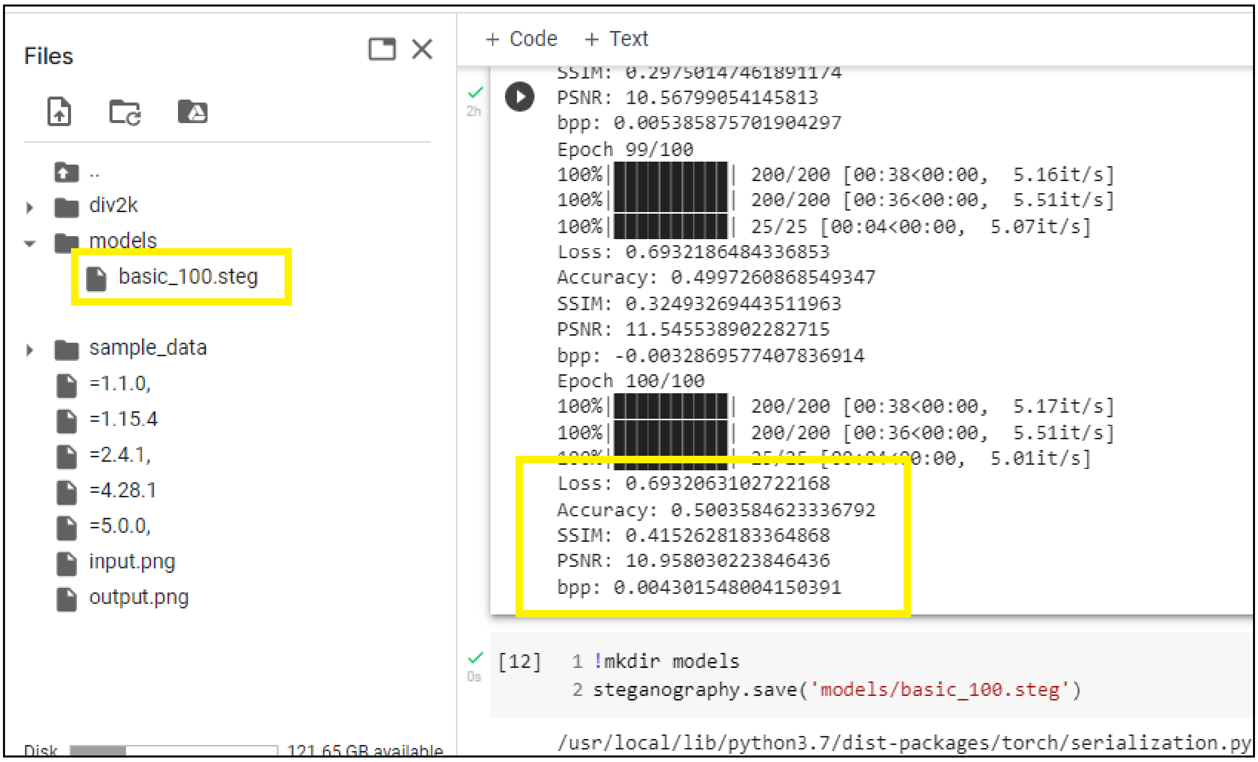


```
[ ]  DIV2K_valid_HR.zip  100%[===================>] 428.19M  20.0MB/s    in 23s

     2021-11-15 10:30:14 (18.9 MB/s) - ‘DIV2K_valid_HR.zip’ saved [448993893/448993893]

     Archive:  DIV2K_valid_HR.zip
       inflating: val/_/0897.png
       inflating: val/_/0887.png
       inflating: val/_/0806.png
       inflating: val/_/0834.png
       inflating: val/_/0896.png
       inflating: val/_/0881.png
       inflating: val/_/0828.png
       inflating: val/_/0833.png
       inflating: val/_/0877.png
       inflating: val/_/0826.png
       inflating: val/_/0879.png
       inflating: val/_/0812.png
       inflating: val/_/0809.png
       inflating: val/_/0865.png
       inflating: val/_/0882.png
       inflating: val/_/0830.png
       inflating: val/_/0892.png
       inflating: val/_/0859.png
```

Fig 6 Dataset Download

After downloading the dataset, the fundamental deep learning steganography algorithm development is used for training. The training method includes 100 epochs of encoding, decoding, and picture verification. The calculation of several metrics such as accuracy, SSIM, loss, and PSNR is depicted in the graph below.

After training with the fundamental algorithm is done, a file was prepared for encoding and decoding the output.



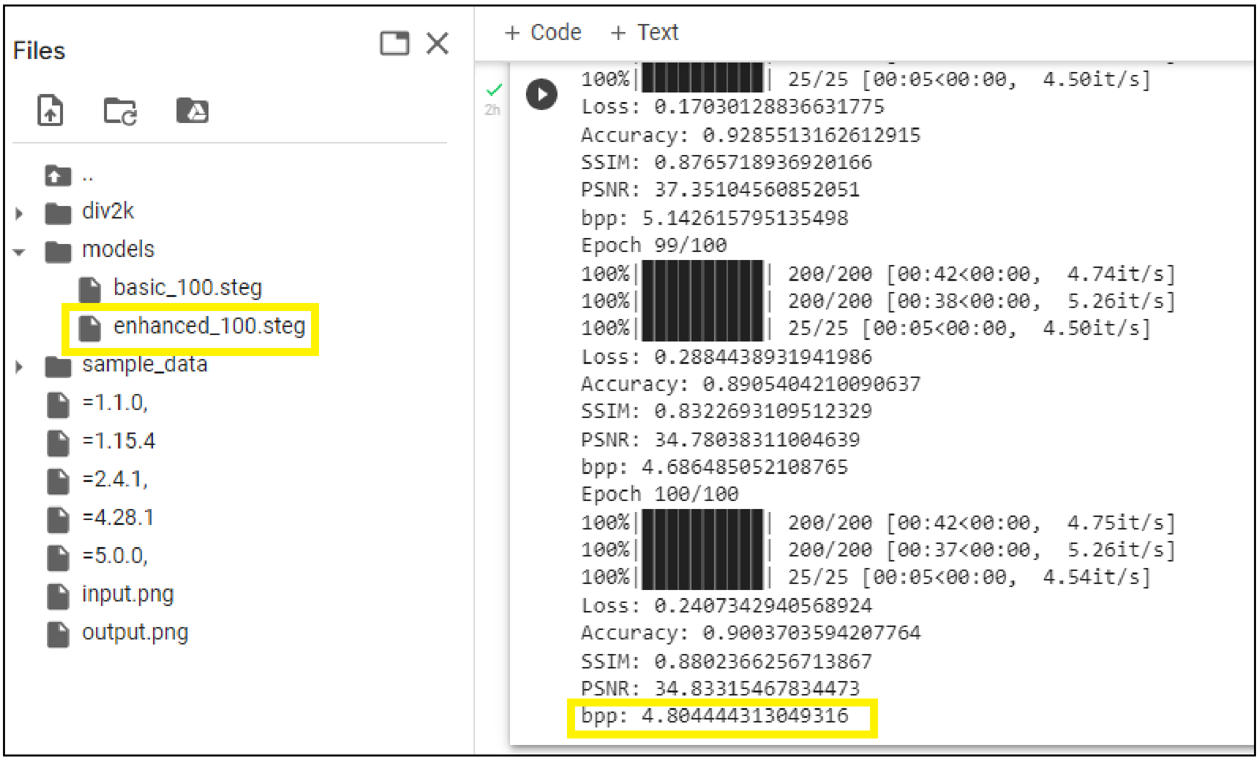
```
SSIM: 0.29750147461891174
PSNR: 10.56799054145813
bpp: 0.005385875701904297
Epoch 99/100
100%|██████████| 200/200 [00:38<00:00,  5.16it/s]
100%|██████████| 200/200 [00:36<00:00,  5.51it/s]
100%|██████████| 25/25 [00:04<00:00,  5.07it/s]
Loss: 0.6932186484336853
Accuracy: 0.4997260868549347
SSIM: 0.32493269443511963
PSNR: 11.545538902282715
bpp: -0.0032869577407836914
Epoch 100/100
100%|██████████| 200/200 [00:38<00:00,  5.17it/s]
100%|██████████| 200/200 [00:36<00:00,  5.51it/s]
100%|██████████| 25/25 [00:04<00:00,  5.01it/s]
Loss: 0.693206310272168
Accuracy: 0.5003584623336792
SSIM: 0.4152628183364868
PSNR: 10.958030223846436
bpp: 0.004301548004150391
```

```
[12]
!mkdir models
steganography.save('models/basic_100.steg')

/usr/local/lib/python3.7/dist-packages/torch/serialization.py
```

Fig 7 Basic Algorithm Output

This method development appears to attain a maximum of 0.004 bits per pixel, which is insufficient for application.

A better algorithm model is created and trained, with the training outcomes for this model displayed in the figure below.



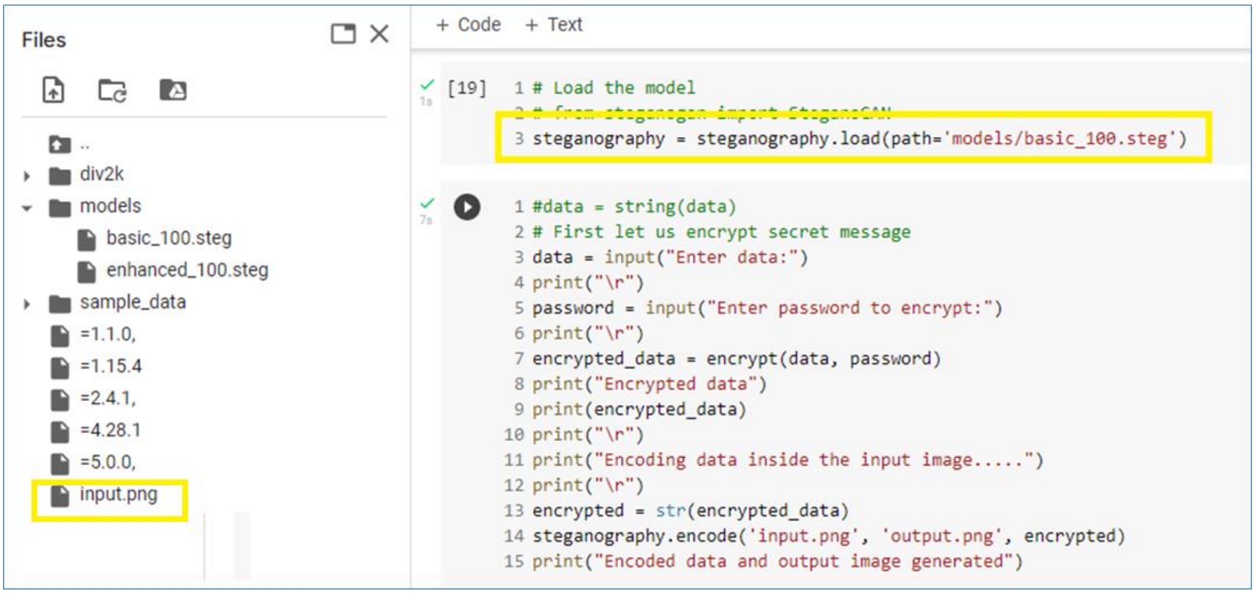
```
100%|██████████| 25/25 [00:05<00:00,  4.50it/s]
Loss: 0.17030128836631775
Accuracy: 0.9285513162612915
SSIM: 0.8765718936920166
PSNR: 37.35104560852051
bpp: 5.142615795135498
Epoch 99/100
100%|██████████| 200/200 [00:42<00:00,  4.74it/s]
100%|██████████| 200/200 [00:38<00:00,  5.26it/s]
100%|██████████| 25/25 [00:05<00:00,  4.50it/s]
Loss: 0.2884438931941986
Accuracy: 0.8905404210090637
SSIM: 0.8322693109512329
PSNR: 34.78038311004639
bpp: 4.686485052108765
Epoch 100/100
100%|██████████| 200/200 [00:42<00:00,  4.75it/s]
100%|██████████| 200/200 [00:37<00:00,  5.26it/s]
100%|██████████| 25/25 [00:05<00:00,  4.54it/s]
Loss: 0.2407342940568924
Accuracy: 0.9003703594207764
SSIM: 0.8802366256713867
PSNR: 34.83315467834473
bpp: 4.804444313049316
```

Fig 8 Enhanced Algorithm Output

As seen in the diagram above, the model file is generated following the training, as well as parameters are also obtained. The project's objective of obtaining over 3 bits per pixel has been fully achieved, and the findings show that we obtained 4.884 bits per pixel, feasible in real-time.

Now that training is complete and the files have been developed, the model files are evaluated for their ability to encode and decode picture data.

First, the model file is loaded, and an image file is given to conceal the text input, as shown below.

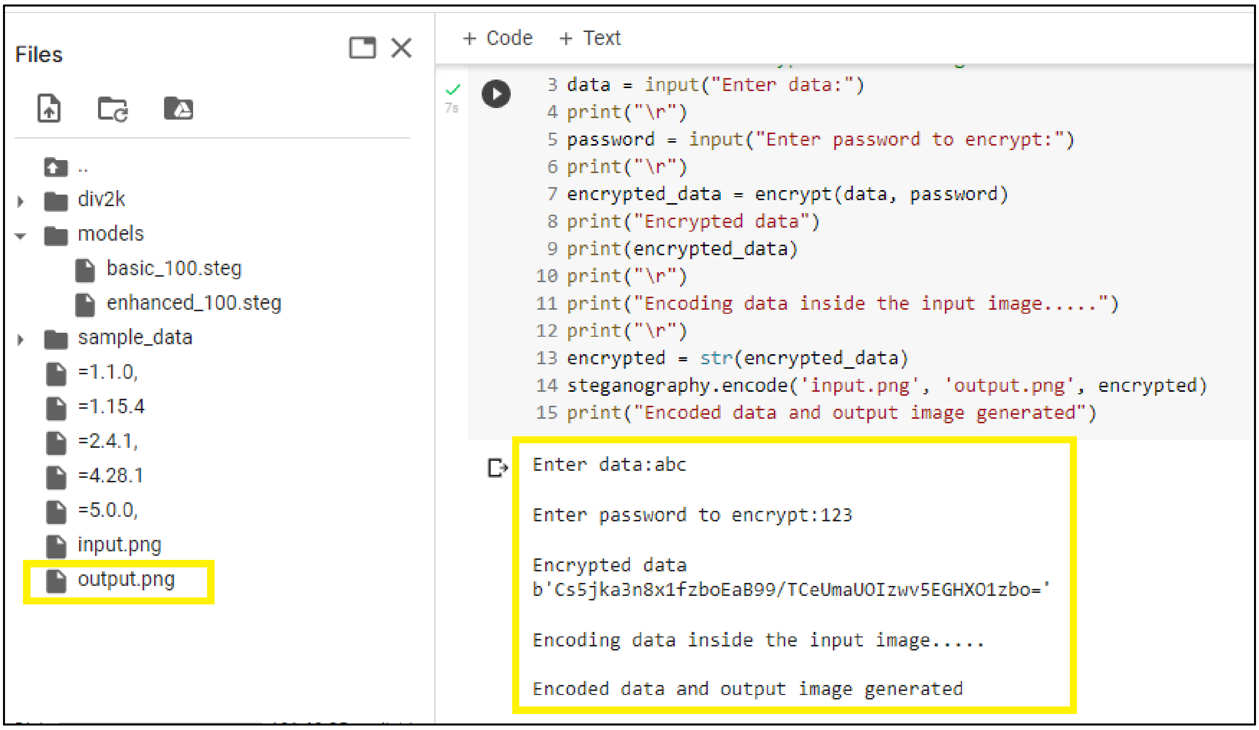


```
# Load the model
# from steganogan import SteganoGAN
steganography = steganography.load(path='models/basic_100.steg')
```

```
#data = string(data)
# First let us encrypt secret message
data = input("Enter data:")
print("\r")
password = input("Enter password to encrypt:")
print("\r")
encrypted_data = encrypt(data, password)
print("Encrypted data")
print(encrypted_data)
print("\r")
print("Encoding data inside the input image.....")
print("\r")
encrypted = str(encrypted_data)
steganography.encode('input.png', 'output.png', encrypted)
print("Encoded data and output image generated")
```

Fig 9 Loading of Model

Following the conversion of the data into the ciphertext, it is concealed somewhere inside the source image so that the output can be seen down
below.

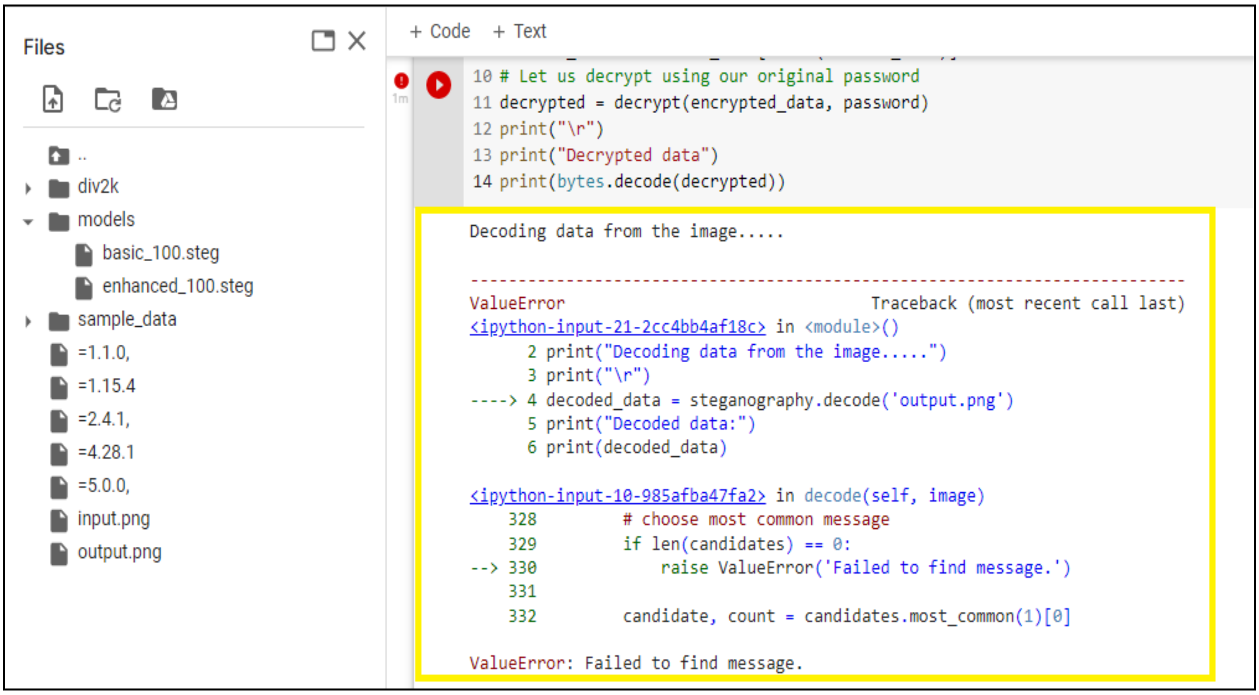


```
data = input("Enter data:")
print("\r")
password = input("Enter password to encrypt:")
print("\r")
encrypted_data = encrypt(data, password)
print("Encrypted data")
print(encrypted_data)
print("\r")
print("Encoding data inside the input image.....")
print("\r")
encrypted = str(encrypted_data)
steganography.encode('input.png', 'output.png', encrypted)
print("Encoded data and output image generated")
```

```
Enter data:abc

Enter password to encrypt:123

Encrypted data
b'Cs5jka3n8x1fzboEaB99/TCeUmaUOIzwv5EGHXO1zbo='

Encoding data inside the input image.....

Encoded data and output image generated
```

Fig 10 Encrypting And Encoding Data

The figure below shows the decoded output image file.



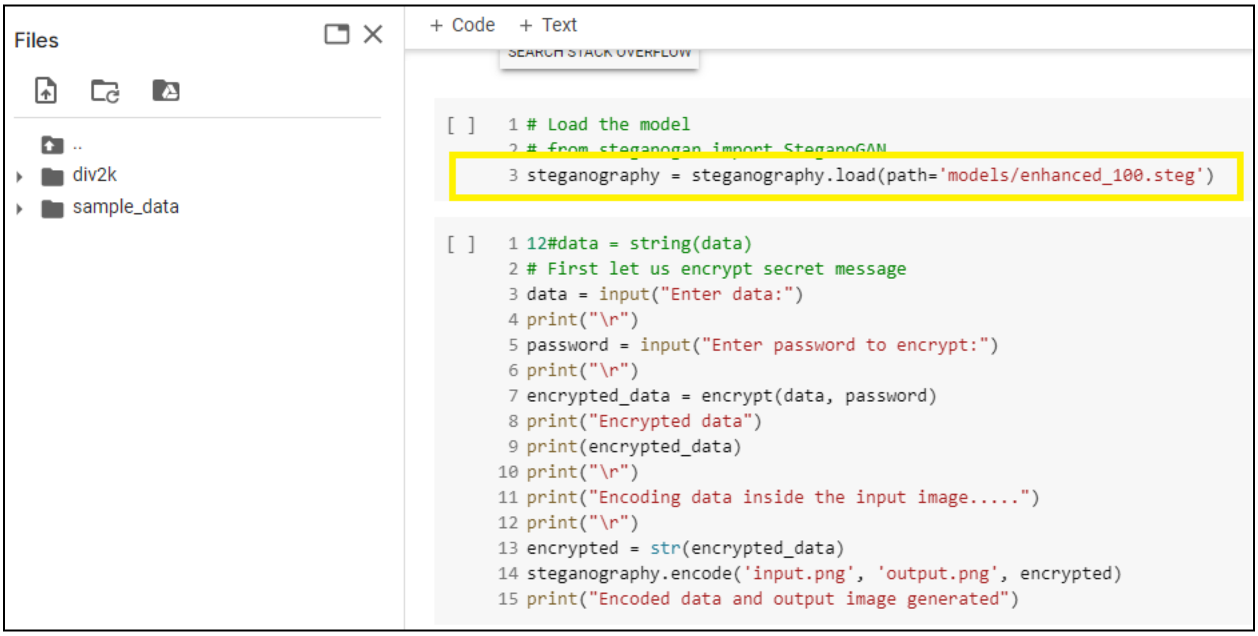
```
# Let us decrypt using our original password
decrypted = decrypt(encrypted_data, password)
print("\r")
print("Decrypted data")
print(bytes.decode(decrypted))
```

```
Decoding data from the image.....

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-21-2cc4bb4af18c> in <module>()
      2 print("Decoding data from the image.....")
      3 print("\r")
----> 4 decoded_data = steganography.decode('output.png')
      5 print("Decoded data:")
      6 print(decoded_data)

<ipython-input-10-985afba47fa2> in decode(self, image)
    328         # choose most common message
    329         if len(candidates) == 0:
--> 330             raise ValueError('Failed to find message.')
    331
    332         candidate, count = candidates.most_common(1)[0]

ValueError: Failed to find message.
```

Fig 11 Failed Decoding of Data

No data was identified when an image file was inspected to hide data. The low number of bits per pixel prevented proper data encoding.

The model file which is generated is loaded for encoding the data and it can be seen below.

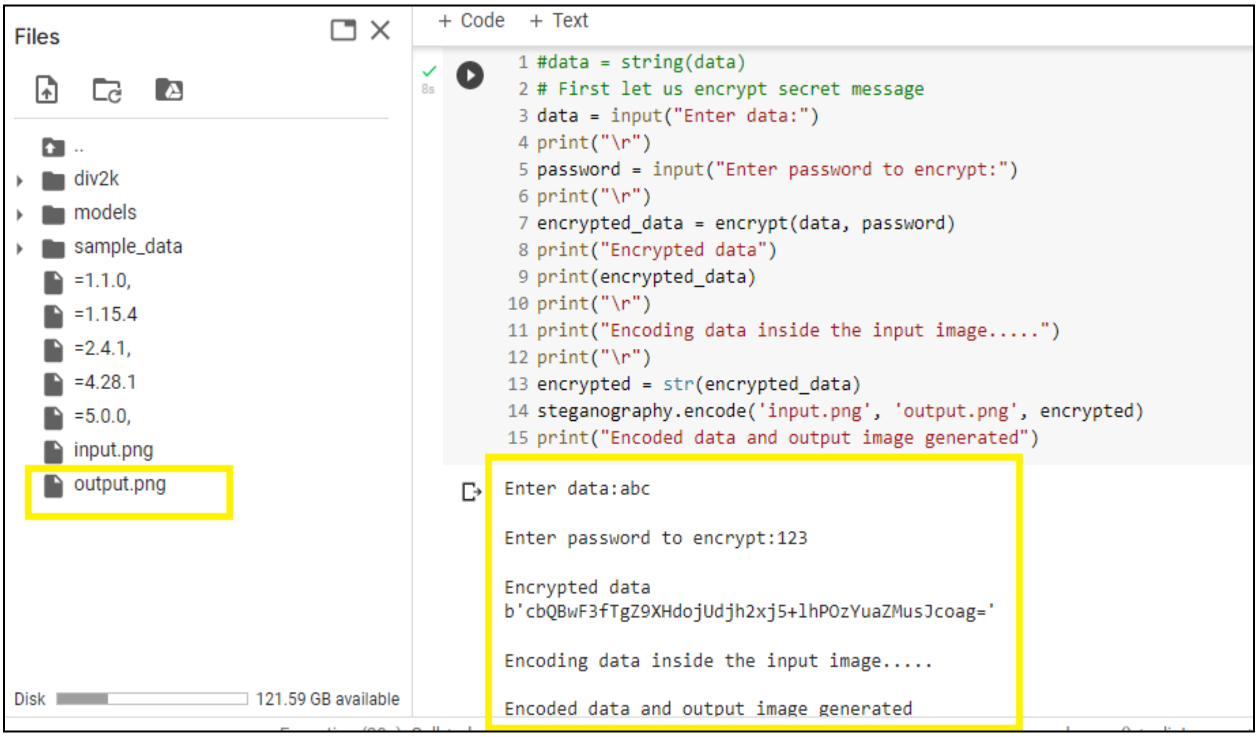


```
# Load the model
# from steganogan import SteganoGAN
steganography = steganography.load(path='models/enhanced_100.steg')
```

```
12#data = string(data)
# First let us encrypt secret message
data = input("Enter data:")
print("\r")
password = input("Enter password to encrypt:")
print("\r")
encrypted_data = encrypt(data, password)
print("Encrypted data")
print(encrypted_data)
print("\r")
print("Encoding data inside the input image.....")
print("\r")
encrypted = str(encrypted_data)
steganography.encode('input.png', 'output.png', encrypted)
print("Encoded data and output image generated")
```

Fig 12 Loading of the Enhanced Model File

After the data encoding procedure, the information is turned to cypher text, as illustrated below.

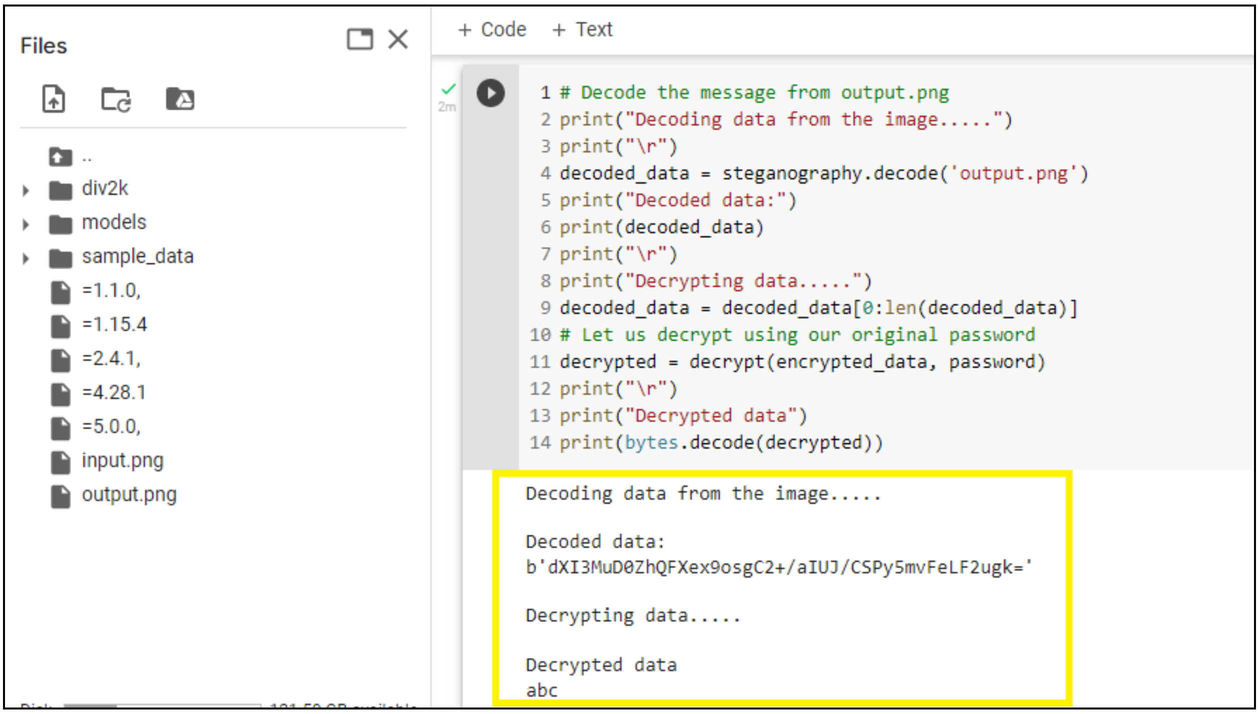


```
#data = string(data)
# First let us encrypt secret message
data = input("Enter data:")
print("\r")
password = input("Enter password to encrypt:")
print("\r")
encrypted_data = encrypt(data, password)
print("Encrypted data")
print(encrypted_data)
print("\r")
print("Encoding data inside the input image.....")
print("\r")
encrypted = str(encrypted_data)
steganography.encode('input.png', 'output.png', encrypted)
print("Encoded data and output image generated")
```

```
Enter data:abc

Enter password to encrypt:123

Encrypted data
b'cbQBwF3fTgZ9XHdojUdjh2xj5+1hPOzYuaZMusJcoag='

Encoding data inside the input image.....

Encoded data and output image generated
```

Fig 13 Encrypting And Encoding

Information Decoding recovers data from an image file, as shown below.



```
# Decode the message from output.png
print("Decoding data from the image.....")
print("\r")
decoded_data = steganography.decode('output.png')
print("Decoded data:")
print(decoded_data)
print("\r")
print("Decrypting data.....")
decoded_data = decoded_data[0:len(decoded_data)]
# Let us decrypt using our original password
decrypted = decrypt(encrypted_data, password)
print("\r")
print("Decrypted data")
print(bytes.decode(decrypted))
```

```
Decoding data from the image.....

Decoded data:
b'dXI3MuD0ZhQFXex9osgC2+/aIUJ/CSPy5mvFeLF2ugk='

Decrypting data.....

Decrypted data
abc
```

Fig 14 Successful Decrypting And Decoding

6 Evaluation

The accuracy, SSIM, payload and PSNR of the Deep learning steganography algorithm built for the steganography model have been estimated. The accuracy of the fundamental method was 50.04%, the PSNR was around 10.958, the SSIM was approximately 0.4152, and the bits per pixel were approximately 0.00430, which is insufficient for deployment. Adding layers to the method enhanced its accuracy to 90.04 per cent, the PSNR to 34.833, the SSIM to 0.8802, and the payload to 4.8844, which can be highly adapted for real-world applications. Also accomplished are encoding and decoding in real-time. This allows the user to apply the information in real-time.

Expect further investigation to expand data storage. In the foreseeable future, algorithms can be improved to be more efficient and hold more info within an image.

Once the data have been acquired, an evaluation of those outcomes can be conducted using the most important criteria. The focus was placed on accuracy, SSIM, PSNR and payload as the primary measures.

The comparison of metrics between the basic method and the expanded algorithm is shown in the table below.

Algorithm Used	Basic Steganography	Enhanced Steganography
PSNR	10.958	34.833
Accuracy	50.04	90.04
Bpp	0.00430	4.8844
SSIM	0.4152	0.8802

Table 1 Table of Metrics

Based on the results obtained, an evaluation is made to answer our research questions with respect to the results.

The notion we employ determines whether or not the data transmitted over the IoT network are secure. Using convolutional neural networks in real time to execute the secure notion of steganography ensures that data transported over the internet is kept safe.

Since we cannot afford to retain more data while steganography is still in its early stages, sensor data is selected as the type of data that can be stored. However, interpretation of sensor data is essential. Sensor data refers to information collected in real time from a wide variety of sensors, including but not limited to those used to measure temperature, humidity, vibration, and light. Each day, countless networks convey this kind of data. With this, data security has risen to the level of an essential requirement. Due to their compact nature, the sensor data we collect fit comfortably within the image's data storage. Therefore, it is of great use in protecting and storing data collected by sensors.

And the solution to the question of how to improve steganography is that the generated algorithm is evaluated according to the results it provides after each training procedure. As a first step, a simple algorithm is created to compute all of the metrics we've covered so far. All of these measurements are computed at the end of each epoch, and the algorithm is tweaked accordingly to achieve the desired outcomes.

The following discussion above helps in answering the research questions focused on this project work.

7 Conclusion and Future Work

The use of steganography offers an innovative technique of protection and lessens the burden of the communication being made public. Within the scope of this project, steganography is being utilised to ensure the confidentiality of all communications. This study proposed expanding the number of bits that may be contained in a picture and securing IoT network data with cryptography and steganography to hide translated data in image communication. Both of these recommendations were made in light of the findings of this study. which we have a complex algorithm for, which is used effectively to encrypt and protect information

We will be able to use bit storage in more applications in the future as we progress. We can examine the facts of the information security software in greater depth, and it can support the rebuttal and improvement of the facts more precisely. There are additional threats in this area, each of which has the potential to amplify or transform this difficulty differently. Utilizing innovative approaches and algorithms that result in improvising the precision of the prediction. Because of this, in the future, the payload of data that may be encrypted securely and then decrypted to display a one-of-a-kind message may be able to be enhanced using this technology.

References

[1] A. D. Ker, “Steganalysis of LSB matching in grayscale images,” *IEEE Signal Process. Lett.*, vol. 12, no. 6, pp. 441–444, Jun. 2005.

[2] B. Li, M. Wang, J. Huang, and X. Li, “A new cost function for spatial image steganography,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 4206–4210.

[3] Boroumand, M., Chen, M. and Fridrich, J. (2019). Deep Residual Network for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security*, 14(5), pp.1181–1193. doi:10.1109/tifs.2018.2871749.

[4] Chen, X., Zhang, Z., Qiu, A., Xia, Z. and Xiong, N.N. (2022). Novel Coverless Steganography Method Based on Image Selection and StarGAN. *IEEE Transactions on Network Science and Engineering*, 9(1), pp.219–230. doi:10.1109/tnse.2020.3041529.

[5] Cogranne, R., Giboulot, Q. and Bas, P. (2022). Efficient Steganography in JPEG Images by Minimizing Performance of Optimal Detector. *IEEE Transactions on Information Forensics and Security*, 17, pp.1328–1343. doi:10.1109/tifs.2021.3111713.

[6] G. Larsson, M. Maire, and G. Shakhnarovich, “FractalNet: Ultra-deep neural networks without residuals,” in *Proc. 5th Int. Conf. Learn.*

[7] 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. doi:10.1109/tii.2021.3053595.

in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2014, pp. 48–53.

[8] J. Kodovský and J. Fridrich, “Steganalysis in high dimensions: Fusing classifiers built on random subspaces,” *Proc. SPIE*, vol. 7880, Feb. 2011, Art. no. 78800L.

[9] J. Mielikainen, “LSB matching revisited,” *IEEE Signal Process. Lett.*, vol. 13, no. 5, pp. 285–287, May 2006.

[10] Karati, A., Islam, S.H. and Karuppiah, M. (2018). Provably Secure and Lightweight Certificateless Signature Scheme for IIoT Environments. *IEEE Transactions on Industrial Informatics*, [online] 14(8), pp.3701–3711. doi:10.1109/TII.2018.2794991.

[10] 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*, 8(7), pp.1933--1939. doi:10.22214/ijraset.2020.30485.

[11] 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. doi:10.1109/tsc.2020.3039976.

[12] 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*, [online] pp.1–1. doi:10.1109/TDSC.2020.3004708.

[13] 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*, 13(7), pp.1854–1868. doi:10.1109/tifs.2018.2806741.

[14] 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*, [online] pp.1–1. doi:10.1109/TDSC.2020.3004708.

[15] Liu, J., Li, Z., Jiang, X. and Zhang, Z. (2022). A High-Performance CNN-Applied HEVC Steganography Based on Diamond-Coded PU Partition Modes. *IEEE Transactions on Multimedia*, 24, pp.2084–2097. doi:10.1109/tmm.2021.3075858.

[16] 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*, 52(6), pp.5001–5014. doi:10.1109/tcyb.2020.3026047.

[17] 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*, 52(6), pp.5001–5014. doi:10.1109/tcyb.2020.3026047.

[18] M. Boroumand, M. Chen, and J. Fridrich, “Deep residual network for steganalysis of digital images,” *IEEE Trans. Inf. Forensics Security*,

[19] M. Li and B. Yuan, “2D-LDA: A statistical linear discriminant analysis for image matrix,” *Pattern Recognit. Lett.*, vol. 26, no. 5, pp. 527–532, Apr. 2005.

[20] N. Provos and P. Honeyman, “Hide and seek: An introduction to steganography,” *IEEE Security Privacy*, vol. 1, no. 3, pp. 32–44, May 2003. *Representations (ICLR)*. Toulon, France: OpenReview.net, Apr. 2017. [Online]. Available: https://openreview.net/forum?id=S1VaB4cex

[21] Singh, B., Sur, A. and Mitra, P. (2021). Steganalysis of Digital Images Using Deep Fractal Network. *IEEE Transactions on Computational Social Systems*, 8(3), pp.599–606. doi:10.1109/tcss.2021.3052520.

[22] 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*, 31(3), pp.1001–1015. doi:10.1109/tcsvt.2020.3001122.

[23] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, “Selection-channel-aware rich model for steganalysis of digital images,”

[24] T. Pevn`y, T. Filler, and P. Bas, “Using high-dimensional image models to perform highly undetectable steganography,” in *Proc. Int. Workshop Inf. Hiding*. Berlin, Germany: Springer, 2010, pp. 161–177.

[25] T. Pevny, P. Bas, and J. Fridrich, “Steganalysis by subtractive pixel adjacency matrix,” *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 2, pp. 215–224, Jun. 2010.

[26] 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*, 14(8), pp.2074–2087. doi:10.1109/tifs.2019.2891237.

[27] 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*, 7(1), pp.621–638. doi:10.1109/jiot.2019.2953549.

[28] V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," in *Proc. IEEE Int. Workshop Inf. Forensics Secure (WIFS)*, Dec. 2012, pp. 234–239.

[29] V. Holub and J. Fridrich, "Random projections of residuals for digital image steganalysis," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 12, pp. 1996–2006, Dec. 2013.

[30] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 2, pp. 221–234, Feb. 2016.

[31] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer, 2013. vol. 14, no. 5, pp. 1181–1193, May 2019.

[32] 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*, 16(6), pp.4221–4230. doi:10.1109/tii.2019.2950295.

[33] 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.

[34] 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.

[35] P. Bas, T. Filler, and T. Pevn`y, "'Break our steganographic system': The ins and outs of organizing BOSS," in *Proc. Int. Workshop Inf.*
Hiding. Berlin, Germany: Springer, 2011, pp. 59–70.

[36] P. Bas and T. Furon. (Jul. 2017). *BOWS-2*. [Online]. Available: http://bows2.ec-lille.fr

[37] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP J. Inf. Secure.*, vol. 2014, no. 1, p. 1, Dec. 2014.

[38] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," 2019, *arXiv:1912.01703*. [Online]. Available: https://arxiv.org/abs/1912.01703

[39] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, Jun. 2009, pp. 41–48.

[40] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.