JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES
www.journalimcms.org

ISSN (Online) : 2454 -7190 Vol.-19, No.-8, August (2024) pp 27-40 ISSN (Print) 0973-8975

DATA HIDING ON DIGITAL IMAGES USING DEEP NEURAL NETWORK (DNN)

Bikash Chandra Bag[1], Hirak Kumar Maity[2], Chaitali Koley[3]

1,2 Department of Electronics and Communication Engineering, College of Engineering and Management, Kolaghat, West Bengal, India

3 Department of Electronics and Communication Engineering, National Institute of Technology Mizoram, Chaltlang, Aizawl-796012, Mizoram, India

Email: mebikash1986@cemk.ac.in, hmaity@cemk.ac.in, chaitali.ece@nitmz.ac.in

Corresponding Author: **Bikash Chandra Bag**

https://doi.org/10.26782/jmcms.2024.08.00003

(Received: April 15, 2024; Revised: July 16, 2024; Accepted: July 30, 2024)

Abstract

In this paper, the framework of Data Hiding on Digital Images Using Deep Neural Network (DNN). Here DeepSteg architecture is considered to evaluate the performance of the multiple secret images that can be concatenated to the single cover image, the image data will be hidden on the single cover image using the Tiny ImageNet dataset. The proposed model outperformed earlier results. To compare our results two parameters, normally Secret loss(λ_s) and cover loss(λ_c) are considered. Our plan is to use deep neural networks for the encoding and decoding of multiple secret images inside a single cover image of a similar goal.

Keywords: Deep Neural network, Deep Steganography. Multiple Secret Images, Single Cover Image, Tiny ImageNet dataset.

I. Introduction

Steganography alludes to the procedure of concealing secret messages inside a non-secret message to stay away from discovery during message transmission. The restricted information is then extricated from the encoded non-secret message at its objective. The utilization of steganography can be joined with encryption as an additional step for stowing away or safeguarding information. Generally, steganography is performed to implant low-goal pictures onto a high-goal picture utilizing gullible techniques like LSB control.

Inspiration for the task comes from the new works, similar to that of [I], [IV], and [VIII]. These papers recommend the utilization of profound brain organizations to demonstrate the information-concealing pipeline. These techniques have essentially worked on productivity as far as keeping up with the mystery and nature of the encoded messages. As of late, comparative work as far as sound sign steganography, similar to

[VI], has demonstrated the way that profound brain organization can be utilized to encode numerous sound messages onto a solitary cover message.

Our mean to try in a comparable course, by using the thoughts from the previously mentioned papers to encode different pictures into a single cover picture. Dissimilar to conventional strategies, we utilize a similar goal cover and mystery pictures and expect to hold the progressions to the encoded cover picture unnoticeable to human discernment and factual investigation, while simultaneously keeping the decoded pictures profoundly comprehensible.

II. Related Work

Out of the several implementations, the below two are the most aligned and important to our goal.

II.i. Hiding Images in Plain Sight:

Deep Steganography [I] endeavors to put a full-sized color image inside one more image of a similar size. Deep neural networks are at the same time trained to create the hiding and revealing processes and are designed to specifically work as a pair. The framework is trained on images drawn haphazardly from the ImageNet data set and functions admirably on regular images from a wide assortment of sources. Not at all like numerous famous steganographic techniques that encode the secret message inside the most un-critical pieces of the cover image, their methodology packs and conveys the secret image's portrayal across all of the available bits.

The three components involved in the system include-

1. Preparation Network - readies the mystery picture to be covered up. If the mystery picture (size M×M) is more modest than the cover picture (N×N), the arrangement network logically builds the size of the mystery picture to the size of the cover, accordingly dispersing the mystery picture's pieces across the whole N × N pixels.

2. Hiding Network - takes as info the result of the arrangement organization and the cover picture, and makes the Compartment picture. The contribution to this organization is an N × N pixel field, with profundity linked RGB channels of the cover picture and the changed channels of the mystery picture.

3. Reveal Network - utilized by the recipient of the picture; it is the decoder. It gets just the Compartment picture (neither the cover nor the mystery picture). The decoder network eliminates the cover picture to uncover the mystery picture.

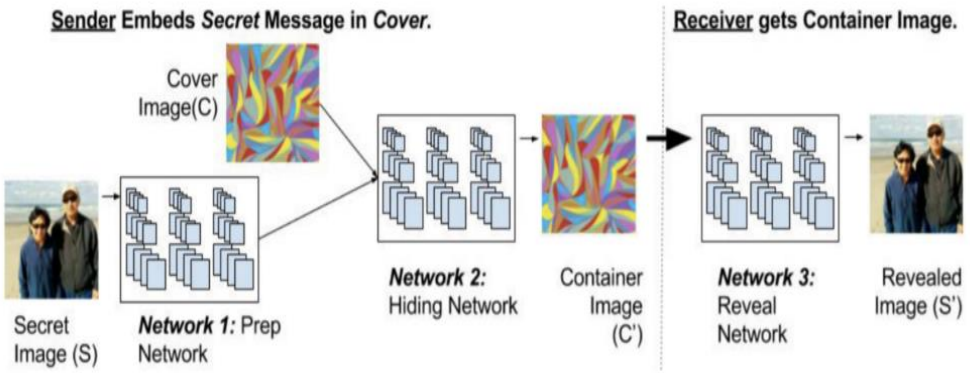

Fig. 1. The three components of the full system. Left: Secret-Image preparation. Centre: Hiding the image in the cover image. Right: Uncovering the hidden image with the reveal network; this is trained simultaneously, but is used by the receiver.

The paper by [I] discusses how a prepared framework should figure out how to pack the data from the mystery picture into the most unobservable bits of the cover picture. In any case, no unequivocal endeavor has been made to conceal the presence of that data from machine recognition effectively. They prepared the steganalysis networks as parallel classifiers, utilizing the unperturbed ImageNet pictures as bad examples, and their compartments as sure models. The paper serves as a standard for single mystery picture encoding. Be that as it may, it doesn't discuss multi-picture steganography.

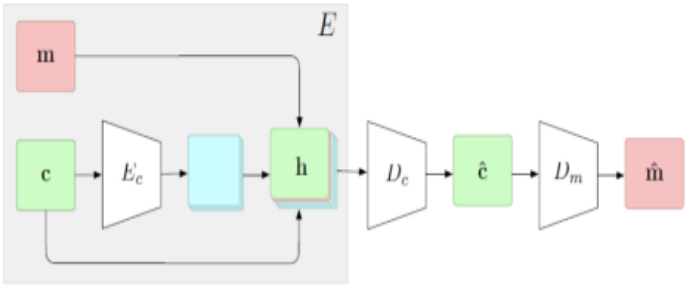

Fig. 2. Model overview: the encoder E gets as input the carrier c and the message m, it encodes c using the carrier encoder E_c and concatenates $E_c(c)$ to c and m to generate h. Then, the decoder carrier Dc generates the new encoded carrier, from which the decoder message D_m decodes the message $\hat{m}$. During training the reconstruction loss is applied between c and m to $\hat{c}$ and $\hat{m}$, respectively

II.ii. Hide and Speak: Profound Brain Organizations for Discourse Steganography [VI] executes steganography for discourse information utilizing profound brain organizations. In view of a design containing 3 shoddy

Decoder network, and a Message Decoder network. They use thoughts from [VIII] to stretch out the encoder network to sound signs. The design of the model involves 3 sub-parts:

1. An Encoder Network (E_c)
2. A Carrier Decoder Network (D_c)
3. A Message Decoder Network (D_m)

In the carrier/cover encoder network, the encoded transporter (Ec(c)) is attached to the transporter (c) and the mystery message (m), shaping, [Ec(c);c,m]. This result is taken care of by the carrier Decoder (Dc) which yields the transporter inserted with a secret message. At last, this is taken care of by the Message Decoder (Dm) which recreates the secret message.

The initial segment figures out how to remove a guide of expected redundancies from the carrier signal. The subsequent part uses the guide to best "stuff" a mystery message into the carrier to such an extent that the transporter is insignificantly impacted. The part third figure out how to separate the stowed-away message from the steganographically altered carrier. Every one of the parts in these organizations are Gated Convs. Ec is made out of 3 blocks of Gated Convs, Dc 4, and Dm 6 blocks of Gated Convs. Each block contains 64 portions of size 3×3.

This paper shows the capacity to conceal various mystery messages in a solitary carrier, which lines up with our objectives. In the paper, five free discourse messages have been concealed in a solitary discourse recording. This is accomplished by 2 distinct methodologies. One methodology uses various decoders, with every decoder prepared to disentangle an alternate message. The other methodology uses a restrictive decoder that likewise takes in as info a code showing the message file to be encoded.

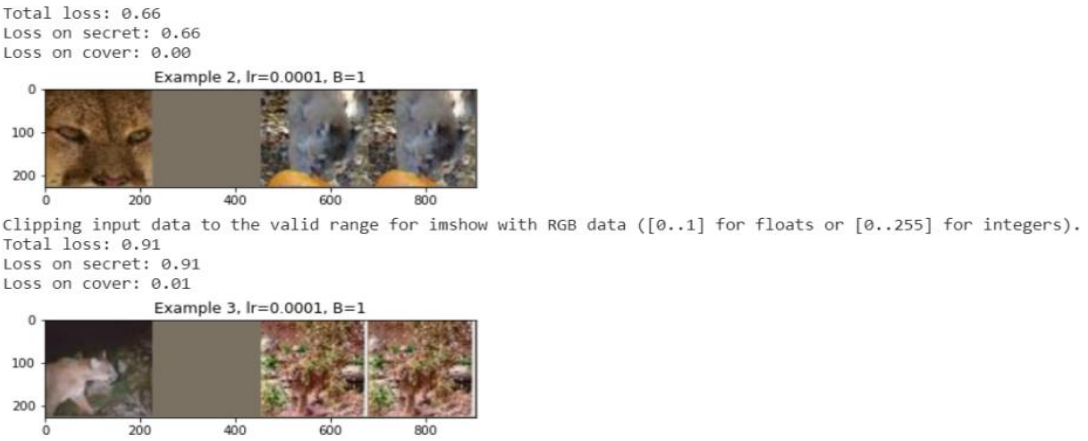

Fig. 3. The following results shows the cover and hidden images before and after running the model for the first 2 images after 20 epochs.

We acquired the idea of various decoders from this paper and utilized it to bring different secret images from the cover image which seems to be the cover picture yet comprises of the secret images being concealed inside the cover image using going through independent prep organizations and afterward linked together. For this, we use the misfortune as characterized by this paper for the different decoders. For our case, we broaden the misfortune capability characterized in this paper for our utilization case. We assume the uncovered misfortune for every decoder as

$$Reveal\ loss = \lambda_s * \sum \|s - s'\|^2$$

and for the entire system as a summation of reveal losses for each decoder added with the loss calculated for cover image as well.

$$L(c, M) = \lambda_c * \|C - D_c(E(c, M))\|^2 + \lambda_m * \sum \|m_i - D_m(D_c(E(c, M))\|^2 \ \ Where\ M = m_k^i$$

We used this approach to introduce multi-image steganography using the above idea from this paper and extending it to images to add multiple images in one cover image and then retrieving it.

III. Baseline Implementations

Our intended to carry out a standard single picture steganography model over which we could perform out expansions. Since (I) has a few executions, we carried out two of the most famous executions and examined them to be reasonable for the expansions of our model. The subtleties of these executions are as per the following:

III.i. Ingham's Implementation

(IV) is a PyTorch-based execution which follows the design displayed in Figure 1. The design incorporates a Prep Organization, a Secret Organization, and an Uncover Organization and inserts a solitary mystery picture onto a solitary cover picture. The model architecture is defined as follows:

1. Prep Network - Two arrangements of three successive layers comprising of (Conv2d - Relu) mixes, linked and taken care of into the following set.

2. Hidden Network - Like the prep network above yet incorporates an extra Conv2D successive layer for adding Gaussian noise to the cover. This permits the secret data to be encoded in bits other than the LSB of the cover image.

3. Reveal Network - Like the above organizations, with an extra Conv2d toward the end.

III.i.a. Implementation Details

The model is explained below:

1. Transformations: Scaling, random crop, and normalization

2. Optimizer: Adam, with learning rate 0.001

3. Customized Loss, as suggested by (I). Cover loss + β×hidden loss)

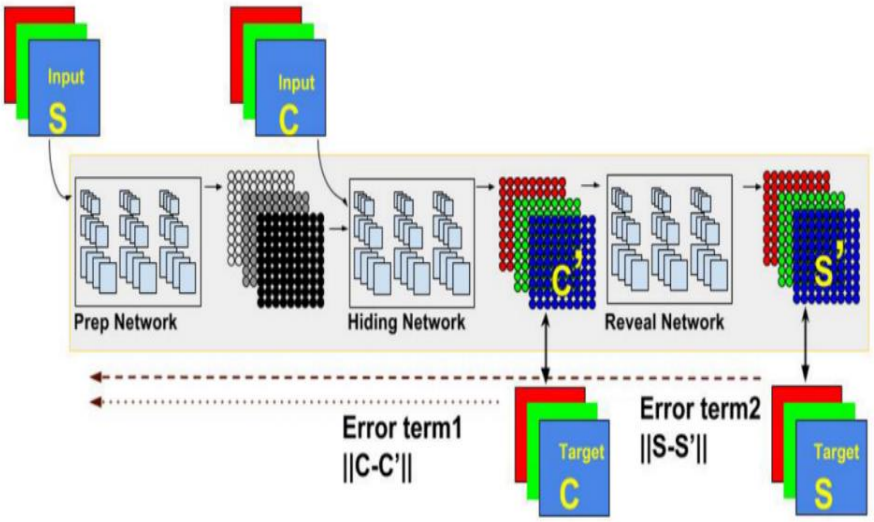

Fig. 4. Baluja's implementation by Alexandre

III.i.b. Results

The author of the execution had shown positive outcomes for higher-resolution images. Since we involved resolution images with an equivalent resolution for both secret and cover, we didn't see comparable outcomes in our executions. See Figure 3. The execution produced lossy secret images, [left-side images in results] while holding the cover image totally [right-side images in results]. Since other pattern models performed altogether better, we chose not to continue with this execution.

III.ii. Alexandre's Implementation

Alexandre's is another execution [VII] of Baluja's paper, in light of Keras. There are new highlights presented in this model that others have not executed like commotion expansion and mean adjustment. We have attempted different models to see which arrangement gives the best outcomes and is quickest. The model engineering has three sections:

1. Preparation Network: Transforms secret image to be concatenated with cover.
2. Hiding Network: Converts the concatenated image into an encoded cover.
3. Reveal Network: Extracts the secret image from the encoded cover.

Hiding and reveal networks use aggregated Conv2D layers: 5 layers of 65 filters [50 3x3 filters, 10 4x4 filters, and 5 5x5 filters]. The prep network uses 2 layers of similar structure. All Conv 2D layers are followed by ReLU activation.

III.ii.a. Implementation Details

The model is explained below:

1. Adam optimizer, with a learning rate of 0.001 and a custom scheduler.

2. The model has been trained for 800 epochs with a batch size of 256 and an additional 100 epochs with a batch size of 32.

3. To make sure weights are updated only once, reveal network weights are frozen before adding them to the full model.

4. Gaussian noise with 0.01 standard deviation is added to the encoder's output before passing it through the decoder.

5. The mean sum of squared error has been used for calculating the decoder's loss.

III.ii.b. Results

As it very well may be found in Figure 5, the model creates fair outcomes in the Minuscule ImageNet dataset. The produced pictures showed an insignificant misfortune for both the cover and the mystery pictures. Inferable from the presentation of the model with a single image steganography we chose to involve this execution for our work going for it.

IV. Datasets

Since our model does not have specific requirements pertaining to the classes of the images, we used the Tiny ImageNet(tin) dataset to obtain the secret and cover images. The dataset is the collection of 64×64×3 images, used by the Stanford CS231 class. Further extensions of the final model can also be applied to larger images from datasets like ImageNet (II). We have also used Tiny ImageNet for faster training.

Our training set is made of a random subset of images from all 200 classes. 2000 images are randomly sampled. The image vectors are normalized across RGB values. We split the entire training data into four halves, one for the cover image and the other three halves for three secret images.

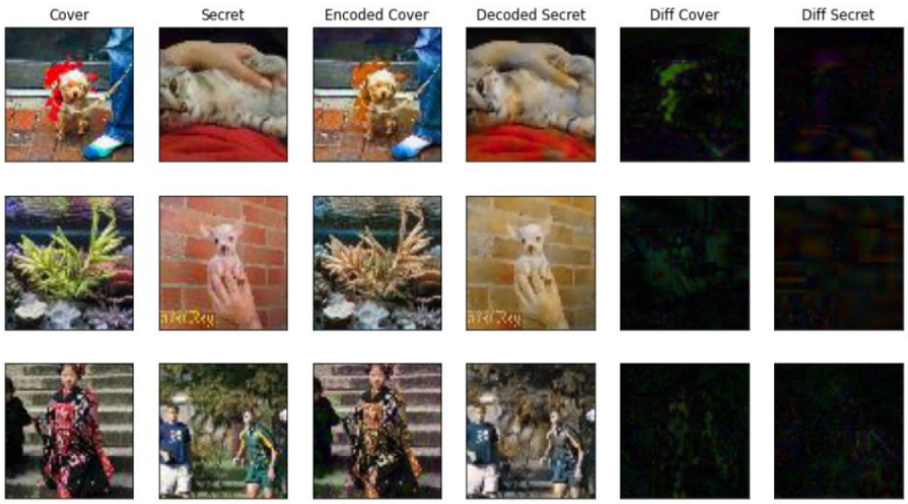

Fig. 5. The results show the cover and hidden images before and after running the model for 900 epochs. Left to Right Columns are: Cover Image, Secret Image, Encoded Cover Image, Decoded Secret Image, Diff Cover Image, and Diff Secret Image. We can notice that the differences between the original cover and the encoded cover are almost null. Same with the original secret and the decoded secret image.

V. Proposed Methodology

Our mean to perform multi-image steganography, concealing at least three images in a single cover image. The embedded secret images should be retrievable with minimum loss. The encoded cover image should seem to be the original cover image. To play out this, we consolidate the possibility of (I) and (V). We take the organization execution thought of having a prep and concealing organization as an encoder and an uncover network as a decoder from (I). To broaden this for various pictures, we pass numerous mystery pictures using the prep organization and afterward connect this subsequent information with the transporter picture and afterward at long last send this through the Concealing organization. We then take numerous decoders, one for each secret image, from (V) to recover every one of the secret images from the compartment picture. To work on the security of our image recovery model, we expand the thought introduced by (I) of placing secret pictures with commotion in the original cover image as opposed to putting them at the LSBs of the first cover image.

Using of Multiple Prep and Reveal Networks:

In their execution for numerous sound sign steganography, (V), proposed the utilization of different decoders to get the decoded secret from a solitary encoded cover. This method is an augmentation of a similar thought in the picture space. It doesn't need downsizing the size of the picture or forfeiting on the variety channels of the mystery pictures. There was one more procedure of utilizing restrictive decoders rather than

various decoders yet in this execution, we have just dealt with carrying out numerous prep/uncover networks. Subsequently, we chose to construct our augmentation in light of this method. A short depiction of the encoder/decoder system according to this strategy is as per the following:

• ENCODER: Consists of multiple prep networks, each corresponding to separate secret image input. Prep network outputs are concatenated together with the cover image and then fed through the Hiding network.

• DECODER: The decoder network comprises multiple reveal networks, each of which is trained separately to decode its corresponding message.

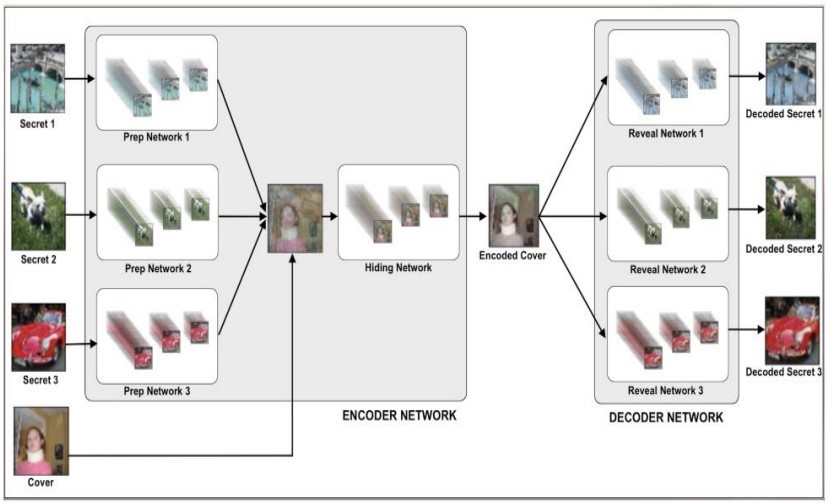

Fig. 6. The image shows the DeepSteg architecture with multiple CNN-based sub-networks. Inside the encoder, the prep networks convert the input secret images into images that can be concatenated to the cover. The concatenation is then passed through the hiding network to generate the encode cover. In the decoder network, separate reveal networks are deployed to generate the decoded secrets out of the encoded cover.

V.i. Model Architecture

The model architecture is described in Figure 6. The underlying architecture of each of the sub-networks is as follows:

1. Prep Networks: Every one of the prep networks comprises a total of 2 layers. With every one of the layers comprised of 3 separate Conv2D layers. The quantity of channels for these 3 Conv2D layers is 50, 10, and 5 separately with the bit sizes like 3, 4, and 5 for each layer. The step length continually stays 1 along both the pivot. Proper cushioning is added to each Conv2D layer to keep the yield picture in similar aspects. Each Conv2d layer is followed with a ReLU initiation.

2. Hiding Network: The hiding network is an aggregation of 5 layers. Each of these layers is made up of the 3 separate Conv2D layers. The underlying structure of the Conv2D layers in the hiding network is similar to the Conv2D layers in the Prep Network.

3. Reveal Network: Each of the reveal networks shares a similar underlying architecture with the hiding network, using 5 layers of similarly formed Conv2D layers.

V.ii. Implementation Details

The training details are explained below:

1. Adam optimizer has been used with a custom LR scheduler.

2. The learning rate remains constant at 0.001 till the first 200 epochs, decreasing to 0.0003 from 200 epochs to 400 epochs and further decreasing it 0.00003 for the remaining iterations.

3. The model has been trained for 750 epochs with a batch size of 256 and an additional 400 epochs with a batch size of 32.

4. Tiny Image Dataset has been used, where images are 64x64. The dataset is created by taking 10 images per class for train and 2000 images in total for train and test.

5. The train set is divided into 2 sections. The first 1000 images are used for training as secret images and the remaining 1000 for cover images.

6. Preparation and Hiding networks share the same stacked Keras model and loss. The reveal network has its stacked model and its loss function.

7. Currently, the learning rate is 0.001.

8. To make sure weights are updated only once, reveal network weights are frozen before adding them to the full model.

9. Gaussian noise with a 0.01 standard deviation is added to the encoder's output before passing it through the decoder.

10. The mean sum of squared error has been used for calculating the decoder's loss.

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

$$Loss = \lambda_c * \|C - C'\|^2 + \lambda_s * \|S_1 - S'_1\|^2 + \lambda_s * \|S_2 - S'_2\|^2 + \lambda_s * \|S_3 - S'_3\|^2$$

12. While training the reveal network we only consider the secret image component of the loss.

13. During the full model training, the loss for both cover and secret image is taken into consideration.

14. Currently we are taking both the λs and λc as 1.0.

Figure 7. Result of hiding two secret images. Left to Right Columns are: Cover Image, Secret Image1, Secret Image2, Encoded Cover Image, Decoded Secret Image1, Decoded Secret Image2.

VI. Results and Discussion

Figure 7 portrays the consequences of concealing two secret images over a single cover image. The information images are portrayed on the left side while the encoder/decoder yields are introduced on the left-hand side. The encoded cover image seems to be like the first cover by and large, and it doesn't uncover data about the secret images.

The results of hiding three secret images are shown in Figure 8. The encoded cover is more lossy as compared to the case when only two secret images are used. The secret images are retrieved successfully in both cases. The losses received for the below results after 750 epochs

were as below -

1. Loss of Entire Setup - 182053.70
2. Loss secret1 - 51495.24
3. Loss secret2 - 39911.16
4. Loss secret3 - 39337.07
5. Loss Cover - 51310.23

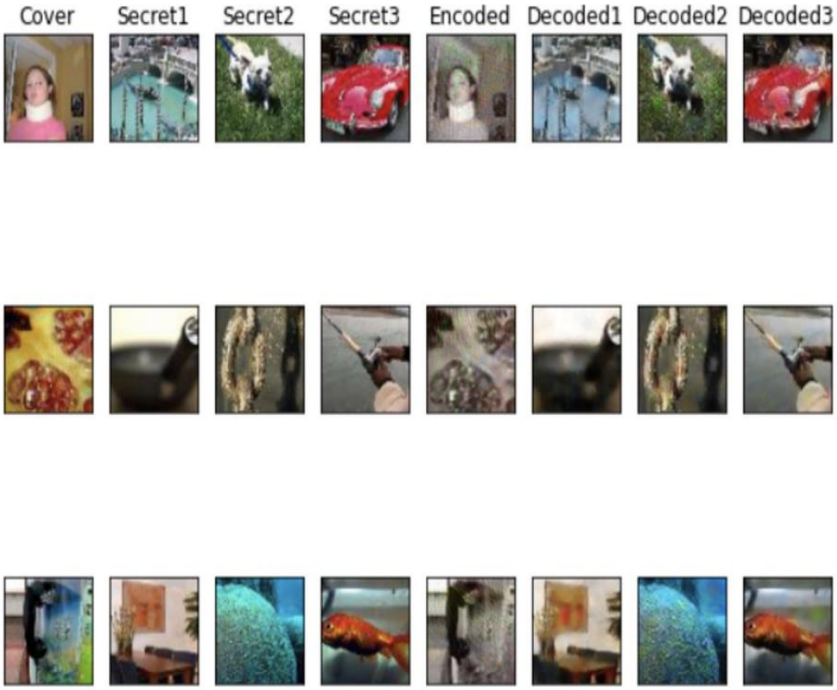

Fig. 8. Result of hiding three secret images. Left to Right Columns are: Cover Image, Secret Image1, Secret Image2, Secret Image2, Secret Image3, Encoded Cover Image, Decoded Secret Image1, Decoded Secret Image2, Decoded Secret Image3.

Currently, the above two results were taken for two and three secret images added to the cover image and then retrieved. As we increase the number of images the loss for all the values is expected to increase as more image features are being hidden in one single image. So, we need to find some threshold concerning how many images can be added to the cover image to get decent results. We also have not explored the λs value for the secret messages and the λc for the cover image. This parameter may help incorrectly define the loss equation and help in getting clearer results for the secret and encoded image. Currently for both the experiments we have taken the λs and λc as 1.

VII. Conclusion

We got to figure out Steganography in the image domain. The issue proclamation assumes a significant part in information security and we had the option to acquire experiences by perusing different papers. Our implementation extended the single-image steganography model proposed by (I) by implementing multiple reveal networks corresponding to each secret image, as suggested by (V). We had the option to encode and decode up to three different secret images over a single cover image of comparable size while keeping up with a fair minimum loss for secret images.

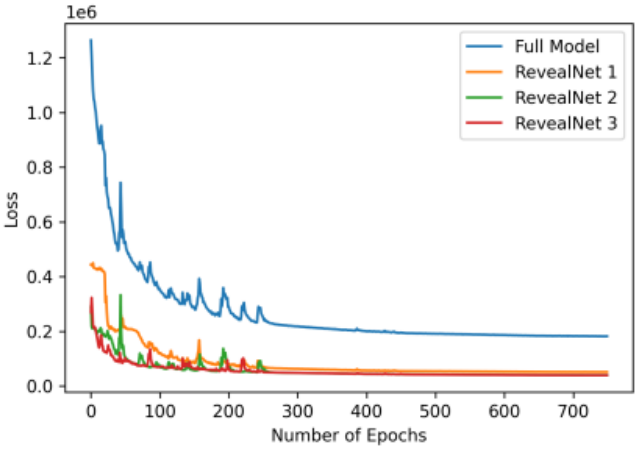

Fig. 9. Graph of Overall Training Loss vs Number of Epochs till 750 epochs

We depended intensely on visual insight for by and large misfortune and didn't explore different avenues regarding different kinds of misfortunes which might have been more qualified for our model. Another strategy to approve the strength of the methodology would assist with working on the outcomes in the correct course. We intend to additionally broaden the venture with additional secret images and work on different loss equations. We intend to change the model to recuperate the encoded picture with the least misfortune. Our commitment to a deep neural network model in the field of Multi-image Steganography can be broadened further with GANs or even deeper neural networks.

Conflict of Interest:

The authors declared that there are no conflicts of interest regarding this article

References:

I. Baluja, S. Hiding images in plain sight: Deep steganography. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), Advances in Neural Information Processing Systems 30, pp. 2069–2079. Curran Associates, Inc., 2017.

II. Bikash Chandra Bag, Hirak Kumar Maity , Chaitali Koley. : 'UNET MOBILENETV2: MEDICAL IMAGE SEGMENTATION USING DEEP NEURAL NETWORK (DNN)'. *J. Mech. Cont. & Math. Sci.,* Vol.-18(1), pp 21-29, 2023. 10.26782/jmcms.2023.01.00002

III. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09, 2009.

IV. Hayes, J. and Danezis, G. Generating steganographic images via adversarial training. In NIPS, 2017.

V. Ingham, F. Deepsteg: Implementation of hidding images in plain sight: Deep steganography in pytorch.

VI. Kreuk, F., Adi, Y., Raj, B., Singh, R., and Keshet, J. Hide and speak: Deep neural networks for speech steganography, 2019.

VII. Muzio, A. Deep-steg: Implementation of hidding images in plain sight: Deep steganography in keras.

VIII. Zhu, J., Kaplan, R., Johnson, J., and Fei-Fei, L. Hidden: Hiding data with deep networks. CoRR, abs/1807.09937, 2018.