2024 10th International Conference on Control, Decision and Information Technologies
CoDIT 2024 | Valletta, Malta / July 01-04, 2024 Technically co-sponsored by IEEE & IFAC

2024 10th International Conference on Control, Decision and Information Technologies (CoDIT) | 979-8-3503-7397-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/CoDIT62066.2024.10708275

A 3D Deep CNN Network-based Data Hiding Scheme for Images

1st Mounir Telli
National Engineering School of Sfax.
University of Sfax, Tunisia
Advanced Technologies for the Environment and Smart Cities
ATES
Faculty of Sciences of Sfax
University of Sfax, Tunisia
ORCID=0000-0002-3183-5487

2nd Mohamed Othmani
Faculty of Sciences of Gafsa
University of Gafsa, Tunisia
Advanced Technologies for the Environment and Smart Cities
ATES
Faculty of Sciences of Sfax
University of Sfax, Tunisia
ORCID=0000-0001-5617-6062

3rd Hela Ltifi
Faculty of Sciences and Techniques of Sidi Bouzid.
University of Kairouan,Tunisia
Research Groups in Intelligent Machines
REGIM
National Engineering School of Sfax.
University of Sfax, Tunisia
ORCID=0000-0003-3953-1135

***Abstract*—The fundamental concept behind image steganography is to conceal one image within another. To advance steganography, it may be wise to conceal multiple images within other multiple images. The major goal of our suggested strategy is to conceal a collection of related images while taking into account size equality. With the aid of a 3D-DeepCNN grounded autoencoder, we introduce a novel multi-image steganography approach in this study. Within four cover images, we attempt to encode and decode four secret images. The quantitative findings show that the suggested model shares the embedded hidden image information over every component of the cover image, without compromising image quality. The results of our model using the peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) are (28.56, 0.928) for global secret input and (32.516, 0.973) for global cover input. The qualitative results were evaluated and give an effective performance in comparison to the current models.**

***Index Terms*—3D Deep CNN, Steganography, Image, Auto-Encoder**

I. Introduction

Maintaining security for the information on the Internet is crucial since the volume of information there is growing quickly. [1], [2], [4], [5]. Steganography is a primary technique in the realm of information [2], [6]–[8], it is a technique for embedding secret information into a publicly available item (such an image, voice file, or text signal), which is referred to as a cover object. [9]. A steganographic scheme is composed of two algorithms: the embedding one and the extraction one [11]. Using image steganography, we can conceal sensitive data in seemingly unimportant pictures to prevent its destruction or alteration [14], [15]. Most image steganography work has been done to conceal certain material in a cover image [5], and because of its limited data-capacity, images can only contain a minimal amount of information. Our goal is to pass from sender and receiver the maximum of hidden data so we present an improvement steganography technique for hiding four images in four photographs in this research. Based on U-Net, we created a deep learning network [16], [23] to accomplish this. It automatically picks the best attributes from both the cover and secret images. The most significant benefit of our method is that it is general and can be applied to any sort of picture. By incorporating the concepts from previous studies, we want to make an effort to encode four images into four cover images. Convolutional neural networks were the deep learning model we ultimately decided to use because It is widely employed in numerous fields, such as security [2], [5], tracking and identification of objects [18], [19] and treatment for neurological disorders [16]. Also, the use of 3D here means that we will consider the four images of input as a short video, thus we can use The 3D convolutions to better extract the spatiotemporal features from these images [10], [22], [23]. The following are the major contributions:

- Proposed an improved model related to multi-image steganography using 3D-CNN;
- Quantitative performance evaluation using standard performance metrics;
- Qualitative performance evaluation to compare with the modern and the latest model.

The remaining sections are arranged as follows: Section 2 showcases related work. Section 3, focuses on the proposed model. The results and performance of our model, and a comparison with the previous model has been discussed in Section 4. Finally, Section 5 concludes our method.

II. Related work

In this section, we provide a study of the literature on some of the numerous digital steganography techniques for images based on deep CNN.

A. Hiding Images in Plain Sight: Deep Steganography

Baluja introduced a deep CNN-based steganography model for images in order to make an image completely encloses another image [17]. This task is carried out using a deep learning algorithm using autoencoders [7]. The weighted average of the losses, due to reconstruction between the cover and container photos as well as between the concealed and exposed concealed images, are used for validation. Their model has implemented global neural networks to perform image

steganography. Three subnetworks comprise this deep model: Pre-Network, Concealing Network, and Revealed Network. One of these, called Pre-Network, sets up the hidden picture. Preparation-Network first distributes to container NxN (cover picture size) pixels the bits of the initial MxM (hidden image size) since the cover image's size might be larger than the secret image's size. Secondly, the method is modifying the original image's points of pixels based on color into more advantageous data, that makes the image simpler to encode. The Concealing network's inputs are the cover image and the product from Pre-Network. This network generates a container image (sometimes referred to as a stego image). NxN represents the network's input size, and its depth is equal to the sum of the feature channels that are transformed from the RGB 3 channels and the initially found hidden image. Then, utilizing the Reveal Network, which serves as a decoder, the image receiver inputs the stego picture from the concealing Network and outputs a hidden picture that has been decoded. This technique has demonstrated its security against several kinds of attacks, even in instances where an attacker has access to the original cover and also to the steganographic images. Baluja has done extensive research, consulted reputable specialists, and examined the applicability of his steganography technique in the actual world. His approach sets a benchmark for concealing a single secret image, but it ignores the issue of multiple images. The model used just two images, with the first one acting as a cover and the second as a secret.

B. End-to-End Trained CNN Encoder-Decoder Networks For Image Steganography

A generalized encoder-decoder architecture for image steganography based on deep learning was proposed by Rehman et al, [4]. For the purpose of concealing one image from another, it uses automatic steganography. Encoder-decoder networks must be trained cooperatively from beginning to end thus the researchers created a new loss function. The suggested architecture was also thoroughly empirically evaluated using several challenging publically available datasets, and results were evaluated in terms of the Peak ratio of signal to noise (PSNR) and the structural similarity index (SSIM) values [3]. The evaluation of image quality typically uses these criteria, particularly in steganography images where the quality of imperceptibility is measured using these two measuring instruments. More specifically, the encoder network combines the cover picture named "host" and the payload image named "guest" into a single output image. As a result, the encoder network's goal is to produce a hybrid image that incorporates the visual similarity of the two images (cover and container). The decoder network receives the hybrid picture that the encoder has produced and reveals the guest image from this hybrid image. The decoder network aims to retrieve the input hybrid's guest image while preserving visual resemblance to the encoder's input guest image. This model using the ImageNet dataset [13], allowing for the cover picture to have an average PSNR and SSIM (32.9/0.96) and for the hidden image to have an average PSNR and SSIM (36.6/0.96). It has outstanding experimental outcomes on several datasets, including ImageNet, in the other side, it is unable to display the outcomes of the model's input when more than one picture is used as a cover and a secret.

C. A Novel Grayscale Image Steganography Scheme Based on Chaos Encryption and Generative Adversarial Networks

To solve the issue of color distortion in the cover image, Li et al [1] suggest a novel steganography technique in which the cover picture is in grayscale. Prior to combining the secret and cover images, the chaotic encryption technique is employed. In other words, the details of a concealed image are obscured using the method of chaotic encryption, then the hidden image is combined with an image named cover to pass the trouble in transmission-related leaking of sensitive image information. Except for the hidden network and extraction network, a discriminatory network is added using this steganography method. To mix low-level and high-level components, the hidden network uses skip connections, which should help to stow the characteristics of the hidden image. The training of hiding-extracting networks is also guaranteed by the use of a new weight distribution technique. The findings of the experiment indicate that the container image and the cover picture can be seen as identical objects, and the steganography model described by Li Model can legitimately embed the hidden picture using every bit of the cover image. The payload, PSNR, and SSIM were compared to show the distinctions between the suggested steganography method and existing steganography systems. The cover image's PSNR and SSIM average, using the ImageNet dataset, reached (42.3 / 0.987), and the secret image's PSNR and SSIM average, with the same dataset, reached (38.45 / 0.953). As same as the previous architecture, this scheme reveal the potential of image steganography, but does not show the results of multi-images.

D. Encoder-Decoder Architecture for Image Steganography using Skip Connections.

Using CNN to analyse and extract features, this research [6] suggests a method for hiding one image inside another. The auto-encoder architecture's basic principles are the foundation of the suggested model and consist of two modules, one of which is the Hiding network/Encoder. Its job is to process the secret image, extract "n" features from it, and then incorporate those features into the cover image while maintaining the cover image's original aesthetic qualities. The other component of this network is the Reveal network/Decoder, which is in charge of extracting the concealed secret image and restoring it to its original form. The proposed technique's entire architecture is built using an end-to-end methodology. The cover image's PSNR and SSIM average, using the dataset ImageNet, reached (36.754 / 0.99174), and the secret image's PSNR and SSIM average, reached (36.805 / 0.99183). This technique of image steganography appears to function well in comparison to other approaches, but one issue still exists: it cannot be applied to the steganography of several images. The literature suggests that retrieving hidden images remains a difficult task

[11]. Additionally, extracting a group of hidden images from content photographs is still an area that has not been fully researched. To recover several secret photos from the same number of cover images with less loss of image quality, we have suggested an Image Steganography Model in this study that is according to 3D Convolutional Neural Networks with Auto-Encoder.

III. Proposed Model

We describe our steganography model in depth in this part, which makes use of a general encoder-decoder architecture constructed using 3D deep learning models [7], [10], [21] for image steganography. Convolutional neural networks (CNNs or ConvNets) were first offered as a way to address computer vision issues. Because of the enormous developments in computational technology, the CNN method has recently been revived and now includes common graphics processing machines (GPUs). With the help of the spatiotemporal relationship between images, 3D convolutional neural networks will gain power. To calculate the low-level feature representations, 3D convolutions apply a three-dimensional filter to the dataset that travels in three directions (x, y, and z) [10]. The proposed technique's entire architecture was created using an end-to-end methodology. This shows that the model was not specifically taught how to obscure an image within another image, prior to extraction training. As an alternative, the end-to-end strategy trains the model in a way that makes it possible for it to simultaneously hide and extract the image, in a single cycle, from the stego image. Numerous well-known steganographic techniques to encrypt secret communications distinct from the proposed steganography architecture. Our technique makes use of compression to distribute hidden images across every accessible bit on the cover images. The model is made up of two components: the Reveal Network and the Hiding Network (encoder and decoder, respectively) [20]. The thorough model is shown in Figure 1. The proposed encoder network's total

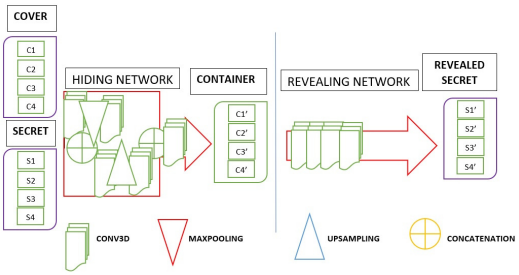

Fig. 1. Global model

workflow is depicted in Figure 2. Four cover images C1, C2, C3, and C4 of size 64x64 (block 1), and four secret images S1, S2, S3, and S4 of size 64x64 (block 2), concatenated chronologically, are sent as input to the concealing network. There are five 3D convolutions of 3x3x3 shapes applied to the feature maps, with 32 filters on the two blocks, during the first phase (feature extraction). These feature maps are subsampled with the use of max-pooling on the two blocks following

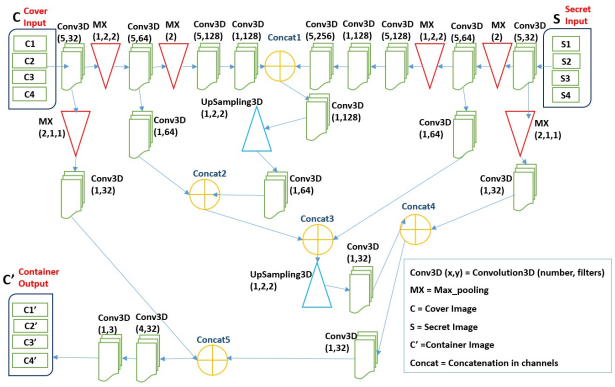

Fig. 2. Pipeline of the Encoder

the initial convolution, and this will produce two different outcomes. A total of five 3D convolutions of 3x3x3 shapes will be applied to the initial outcome of this max-pooling, together with 64 filters, for the two blocks. The second outcome of this max-pooling will undergo a 3D convolution using a single set of 32 filters and a single set of 3x3x3 shapes. The feature maps are subsampled using max-pooling on the two blocks following the second convolution phase. Six 3D convolutions of 3x3x3 shapes will be performed to the results of this max-pooling, and 128 filters will be applied to the two blocks. The feature maps are subjected to five 3D convolutions of 3x3x3 shapes, applied with 256 filters on the two blocks, in the same third phase. A spatial concatenation between the results of the two blocks can be performed at the reconstructed phase, which is almost at the four-phase, followed by a 3D convolution of the shapes 3x3x3. The output will then undergo upsampling and a 3D convolution of three three-dimensional shapes using 64 filters. A few 3D convolutions, upsampling, and concatenation with earlier results (from earlier phases) will come after this phase. The concealing network produces four "64x64-pixel" container pictures C1', C2', C3', and C4' as the final product. Table 1 contains comprehensive illustrations of the hidden network parameters. The revealing network requires four concatenated "64x64" container pictures, C1', C2', C3', and C4', as input. In Figure 3, the suggested decoder network's whole pipeline is depicted. The feature maps are subjected to five 3D convolutions of 3x3x3 shapes, applied with 32 filters, during the first step (feature extraction). After the initial convolution, there are two sides. The first side subsamples the feature maps using max-pooling, and the second side convolves the feature maps using a single 3D convolution of 3x3x3 forms using 32 filters after subsamples. Five 3D convolutions of 3x3x3 shapes will be applied to the first result, and 64 filters will be applied to the two blocks. Utilizing max-pooling, the feature maps are subsampled following the second convolution phase. Six 3D convolutions of 3x3x3 shapes will be applied to the results of this max-pooling, followed by 128 filters. The feature maps are subsampled using max-pooling on the first side of the convolution process and are then convolved with a single 3D convolution of 3x3x3 forms using 256 filters on the

TABLE I
CNN HIDING NETWORK PARAMETERS

Layer	Block	Index	Type	Step	Filters	Kernel	Stride	Padding	Input	Output	Concatenation
01	1	01	Conv3D	5	32	$3 \times 3 \times 3$	3	1	$4 \times 64 \times 64 \times 3$	$4 \times 64 \times 64 \times 32$	N/A
		02	MaxPooling	1				$2 \times 1 \times 1$	$4 \times 64 \times 64 \times 32$	$2 \times 64 \times 64 \times 32$	N/A
		03	MaxPooling	1				2	$4 \times 64 \times 64 \times 32$	$2 \times 32 \times 32 \times 32$	N/A
		04	Conv3D	1	32	$3 \times 3 \times 3$	3	1	$2 \times 64 \times 64 \times 32$	$2 \times 64 \times 64 \times 32$	N/A
	2	05	Conv3D	5	32	$3 \times 3 \times 3$	3	1	$4 \times 64 \times 64 \times 3$	$4 \times 64 \times 64 \times 32$	N/A
		06	MaxPooling	1				$2 \times 1 \times 1$	$4 \times 64 \times 64 \times 32$	$2 \times 64 \times 64 \times 32$	N/A
		07	MaxPooling	1				2	$4 \times 64 \times 64 \times 32$	$2 \times 32 \times 32 \times 32$	N/A
		08	Conv3D	1	32	$3 \times 3 \times 3$	3	1	$2 \times 64 \times 64 \times 32$	$2 \times 64 \times 64 \times 32$	N/A
02	1	09	Conv3D	5	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 32$	$2 \times 32 \times 32 \times 64$	N/A
		10	Conv3D	1	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 32$	$2 \times 32 \times 32 \times 64$	N/A
		11	MaxPooling	1				$1 \times 2 \times 2$	$2 \times 32 \times 32 \times 64$	$2 \times 16 \times 16 \times 64$	N/A
	2	12	Conv3D	5	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 32$	$2 \times 32 \times 32 \times 64$	N/A
		13	Conv3D	1	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 32$	$2 \times 32 \times 32 \times 64$	N/A
		14	MaxPooling	1				$1 \times 2 \times 2$	$2 \times 32 \times 32 \times 64$	$2 \times 16 \times 16 \times 64$	N/A
03	1	15	Conv3D	5	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 64$	$2 \times 16 \times 16 \times 128$	N/A
		16	Conv3D	1	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 128$	$2 \times 16 \times 16 \times 128$	N/A
	2	17	Conv3D	5	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 64$	$2 \times 16 \times 16 \times 128$	N/A
		18	Conv3D	1	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 128$	$2 \times 16 \times 16 \times 128$	N/A
04	2	19	Conv3D	5	256	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 128$	$2 \times 16 \times 16 \times 256$	N/A
	1	20	Concat	1					$2 \times 16 \times 16 \times 256$	$2 \times 16 \times 16 \times 384$	with Index 19
05		21	Conv3D	1	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 384$	$2 \times 16 \times 16 \times 128$	N/A
		22	Up-Sample	1				$1 \times 2 \times 2$	$2 \times 16 \times 16 \times 128$	$2 \times 32 \times 32 \times 128$	N/A
		23	Conv3D	1	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 128$	$2 \times 32 \times 32 \times 64$	N/A
		24	Concat	1					$2 \times 32 \times 32 \times 64$	$2 \times 32 \times 32 \times 128$	with Index 10
		25	Concat	1					$2 \times 32 \times 32 \times 128$	$2 \times 32 \times 32 \times 192$	with Index 13
06		26	Up-Sample	1				$1 \times 2 \times 2$	$2 \times 32 \times 32 \times 192$	$2 \times 64 \times 64 \times 192$	N/A
		27	Conv3D	1	32	$3 \times 3 \times 3$	3	1	$2 \times 64 \times 64 \times 192$	$2 \times 64 \times 64 \times 32$	N/A
		28	Concat	1					$2 \times 64 \times 64 \times 32$	$2 \times 64 \times 64 \times 64$	with Index 08
07		29	Conv3D	1	32	$3 \times 3 \times 3$	3	1	$2 \times 64 \times 64 \times 64$	$2 \times 64 \times 64 \times 32$	N/A
		30	Concat	1					$2 \times 64 \times 64 \times 32$	$4 \times 64 \times 64 \times 32$	with Index 04
08		39	Conv3D	4	32	$3 \times 3 \times 3$	3	1	$4 \times 64 \times 64 \times 32$	$4 \times 64 \times 64 \times 32$	N/A
		40	Conv3D	3	1	$3 \times 3 \times 3$	3	1	$4 \times 64 \times 64 \times 32$	$4 \times 64 \times 64 \times 3$	N/A

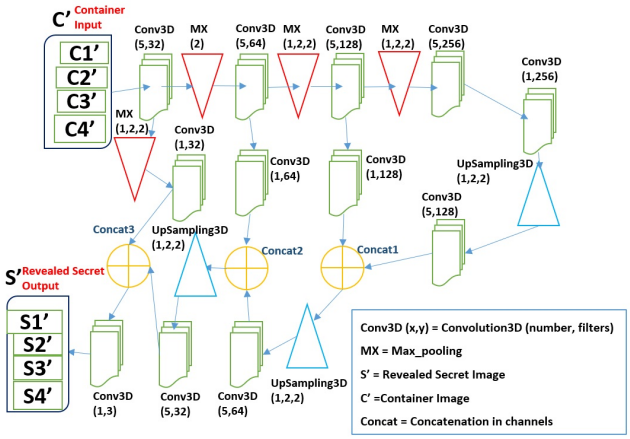

Fig. 3. Pipeline of the Decoder

second side. A spatial concatenation can be performed during the reconstruction phase, which is almost the fourth phase and is then followed by some 3D convolutions, upsampling, and concatenation with earlier results (from previous phases). The hidden network's ultimate output consists of four secret images. S1' for the first, S2' for the second, S3' for the third, and S4' for the fourth, each measuring 64x64. Table 2 includes comprehensive illustrations of the revealing network parameters. The output column indicates the obtained map's feature size.

IV. EXPERIMENTAL RESULTS, ANALYSIS AND DISCUSSION

The collection of images we utilized was Tiny ImageNet Visual Recognition Challenge. A haphazard collection of images from each of the 200 classes composes our training data. The dataset is made up of 10,000 photos for testing and 500 images per class for the train, for a total of 100,000 images. Our experimental components and results were obtained on a computer with a GPU card from Nvidia called the Geforce 1650TI. The Python programming (Anaconda toolkit) architecture of the model depends on the TensorFlow package and the CUDA technology. The Adam optimizer was employed. The learning rate is always 0.001 and is constant. The model was trained over a period of 2000 epochs with 125 steps using a batch size of 4. Below is an illustration of the function used to calculate the decoder's failure:

$$\textbf{Loss} = \sum_{i=1}^{4} \|C - C'\|^2 + \beta * \sum_{i=1}^{4} \|S - S'\|^2 \qquad (1)$$

where C is the cover image, C' is the container image, S secret image, S' is the revealed secret image and β is introduced to weigh the reconciliation errors between cover-container images and hidden images that have been disclosed in secret. The training has been conducted with a value of β equal to 1.00 at first and then with values equal to 0.75. The sum of bits

TABLE II
CNN REVEALING NETWORK PARAMETERS

Layer	Index	Type	Step	Filters	Kernel	Stride	Padding	Input	Output	Concatenation
01	01	Conv3D	5	32	$3 \times 3 \times 3$	3	1	$4 \times 64 \times 64 \times 3$	$4 \times 64 \times 64 \times 32$	N/A
	02	MaxPooling	1				$2 \times 1 \times 1$	$4 \times 64 \times 64 \times 32$	$2 \times 64 \times 64 \times 32$	N/A
	03	MaxPooling	1				2	$4 \times 64 \times 64 \times 32$	$2 \times 32 \times 32 \times 32$	N/A
	04	Conv3D	1	32	$3 \times 3 \times 3$	3	1	$2 \times 64 \times 64 \times 32$	$2 \times 64 \times 64 \times 32$	N/A
02	05	Conv3D	5	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 32$	$2 \times 32 \times 32 \times 64$	N/A
	06	Conv3D	1	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 32$	$2 \times 32 \times 32 \times 64$	N/A
	07	MaxPooling	1				$1 \times 2 \times 2$	$2 \times 32 \times 32 \times 64$	$2 \times 16 \times 16 \times 64$	N/A
03	08	Conv3D	5	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 64$	$2 \times 16 \times 16 \times 128$	N/A
	09	Conv3D	1	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 128$	$2 \times 16 \times 16 \times 128$	N/A
	10	MaxPooling	1				$1 \times 2 \times 2$	$2 \times 16 \times 16 \times 64$	$2 \times 8 \times 8 \times 128$	N/A
04	11	Conv3D	5	256	$3 \times 3 \times 3$	3	1	$2 \times 8 \times 8 \times 128$	$2 \times 8 \times 8 \times 256$	N/A
	12	Conv3D	1	256	$3 \times 3 \times 3$	3	1	$2 \times 8 \times 8 \times 256$	$2 \times 8 \times 8 \times 256$	N/A
	13	Up-Sample	1				$1 \times 2 \times 2$	$2 \times 8 \times 8 \times 256$	$2 \times 16 \times 16 \times 256$	N/A
05	14	Conv3D	5	128	$3 \times 3 \times 3$	3	1	$2 \times 16 \times 16 \times 256$	$2 \times 16 \times 16 \times 128$	N/A
	15	Concat	1					$2 \times 16 \times 16 \times 128$	$2 \times 16 \times 16 \times 256$	with Index 09
	16	Up-Sample	1				$1 \times 2 \times 2$	$2 \times 16 \times 16 \times 256$	$2 \times 32 \times 32 \times 256$	N/A
06	17	Conv3D	5	64	$3 \times 3 \times 3$	3	1	$2 \times 32 \times 32 \times 256$	$2 \times 32 \times 32 \times 64$	N/A
	18	Concat	1					$2 \times 32 \times 32 \times 64$	$2 \times 32 \times 32 \times 128$	with Index 06
	19	Up-Sample	1				$1 \times 2 \times 2$	$2 \times 32 \times 32 \times 128$	$2 \times 64 \times 64 \times 128$	N/A
07	20	Conv3D	5	32	$3 \times 3 \times 3$	3	1	$2 \times 64 \times 64 \times 128$	$2 \times 64 \times 64 \times 32$	N/A
	21	Concat	1					$2 \times 64 \times 64 \times 32$	$4 \times 64 \times 64 \times 32$	with Index 04
08	22	Conv3D	1	3	$3 \times 3 \times 3$	3	1	$4 \times 64 \times 64 \times 32$	$4 \times 64 \times 64 \times 3$	N/A

of data that each pixel can hold is referred to as the payload capacity. It is calculated as [12]:

$$\textbf{Payload} = \textbf{Revealed Rate} * 8 * 3 \tag{2}$$

The possibility that a hidden image can be successfully extracted is known as the revealed rate. It is calculated as [12]:

$$\textbf{Revealed Rate} = 1 - \sum_{i=1}^{N} \sum_{j=1}^{4} \|C - C'\|^2 / N * 4 \tag{3}$$

Figure 4 shows the result of covering a single cover picture with one hidden images. The model input is shown on the left side, while the output (container) picture is shown on the middle and the differences in the left.

Fig. 4. Result of testing 1 image with our model. Left-Right: Cover Image, Container Image, Differences

The performance evaluation, which is averaged over all 10000 test samples, is summarized in Table 3 (with β = 1.00) and Table 4 (with β = 0.75).

In this experiment, samples from the Tiny-ImageNet training batch were taken randomly and incongruously for the cover 4x(64x64x3) and payload images 4x(64x64x3). We were successful in concealing a payload of $97,3\%$ (i.e. 23,352 bpp) of our cover images with an average PNSR of 32.516 db with β = 1. Also, we succeeded in concealing a payload equal to $97,8\%$ (i.e. 23,472 bpp) in the cover pictures with a value of an average PNSR of 36.218 db with β = 0.75. With regard to

TABLE III
MODEL PERFORMANCE QUANTITATIVE ANALYSIS (β = 1.00)

β = 1.00	PSNR		SSIM	
	(C, C')	(S, S')	(C, C')	(S, S')
Good	41.408	36.629	0.998	0.993
Medium	32.116	34.516	0.990	0.936
Bad	15.717	16.136	0.168	0.399
Global Meduim	32.516	28.56	0.973	0.928

the quantitative analysis, results of the model was confirmed using metrics such as peak signal-to-noise ratio (PSNR), and structural similarity index (SSIM) [3]. The different resultant values of SSIM and PSNR show effectiveness of our model. In Table 5, we compare the performance indicators SSIM and

TABLE IV
MODEL PERFORMANCE QUANTITATIVE ANALYSIS (β = 0.75)

β = 0.75	PSNR		SSIM	
	(C, C')	(S, S')	(C, C')	(S, S')
Good	45.088	34.679	0.998	0.982
Medium	35.409	28.226	0.992	0.94
Bad	26.146	15.227	0.192	0.322
Global Meduim	36.218	25.449	0.978	0.85

PSNR of our proposed model (just the good results) with those of the Baluja Image model [17], the Rehman model [4], the QiLi model [1] and the Kumar model [6]. Our dataset was used to train the Baluja Image network since only an error per pixel is displayed by Baluja. As shown in Table 5, we compared the good values obtained from the Tiny-ImageNet dataset execution result of our proposed model to

TABLE V
MODEL PERFORMANCE QUALITATIVE ANALYSIS (β = 1.00)

	Baluja		Rehman		QiLi		Kumar		Ours	
	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
$C - C'$	0.984	36.153	0.96	32.9	0.987	42.3	0.991	36.754	0.998	41.408
$S - S'$	0.983	35.439	0.96	36.6	0.953	38.45	0.991	36.805	0.993	36.629

those obtained from different modern models. These values of PSNR and SSIM for our suggested model are shown alongside those of several other models already in use that also employ the CNN auto-encoder architecture. Our model excels at hiding information within images while maintaining high quality. After embedding, the cover image appears nearly identical to the original thanks to the model's high PSNR. Even after complete data extraction, the secret image retains most of its visual features, as shown by the extracted image's PSNR. The cover image preserves the bulk of its structural characteristics, such as gradient changes, brightness, edges, etc., even when new data (an image) is concealed in the cover image. Due to the improved SSIM, there is relatively low data loss during image extraction, and the structural properties of the returned image are similar to those of the original hidden image. Extensions may be the use of this model as a video steganography model.

V. CONCLUSIONS

In this study, a method for multi-image steganography that uses a new deep 3D convolutional neural network model with an auto-encoder approach to cover up one batch of four images with another batch of four images has been developed, and the major objective is to improve various features of steganography, especially visibility. The specific contributions are the use of 3D deep CNN which allows us to use more images from the dataset as input in our model. Experimental findings demonstrate the advantages of our model both qualitatively and quantitatively, over the existing model for steganography. Our model beat the imperceptibility results of various current approaches without losing the accuracy of the retrieved images. The focus of the effort going forward is on the enhancement of our model to boost capacity through certain experiments utilizing various sets of data and more steganalysis approaches.

REFERENCES

[1] Q. Li, X. Wang, X. Wang, B. Ma, C. Wang, Y. Xian, Y. Shi, A novel grayscale image steganography scheme based on chaos encryption and generative adversarial networks, IEEE Access 8 (2020) 168166–168176.
[2] L. Singh, A. K. Singh, P. K. Singh, Secure data hiding techniques: a survey, Multimedia Tools and Applications (2020).
[3] A. Horé, D. Ziou, Image quality metrics: Psnr vs. ssim, in: 2010 20th International Conference on Pattern Recognition, 2010, pp. 2366–2369.
[4] R. Rahim, S. Nadeem, et al., End-to-end trained cnn encoder-decoder networks for image steganography, in: Proceedings of the European Conference on Computer Vision (ECCV) Workshops, 2018.
[5] N. Subramanian, O. Elharrouss, S. Al-Maadeed, A. Bouridane, Image steganography: A review of the recent advances, IEEE Access 9 (2021) 23409–23423.
[6] A. Kumar, R. Rani, S. Singh, Encoder-decoder architecture for image steganography using skip connections., Procedia Computer Science 218 (2023) 1122–1131. International Conference on Machine Learning and Data Engineering.
[7] P. Baldi, Autoencoders, unsupervised learning, and deep architectures, in: I. Guyon, G. Dror, V. Lemaire, G. Taylor, D. Silver (Eds.), Proceedings of ICML Workshop on Unsupervised and Transfer Learning, volume 27 of Proceedings of Machine Learning Research, PMLR, Bellevue, Washington, USA, 2012, pp. 37–49.
[8] P. C. Mandal, I. Mukherjee, G. Paul, B. Chatterji, Digital image steganography: A literature survey, Information Sciences 609 (2022) 1451–1488.
[9] Ghoul, S., Sulaiman, R., Shukur, Z. (2023). A review on security techniques in image steganography. International Journal of Advanced Computer Science and Applications, 14(6).
[10] Elouni, Jihed, et al. "Intelligent health monitoring system modeling based on machine learning and agent technology." Multiagent and Grid Systems 16.2 (2020): 207-226.
[11] X. Duan, N. Liu, M. Gou, W. Wang, C. Qin, Steganocnn: Image steganography with generalization ability based on convolutional neural network, Entropy 22 (2020).
[12] X. Duan, D. Guo, C. Qin, Image information hiding method based on image compression and deep neural network, "Computer Modeling in Engineering and Sciences" 124 (2020) 721–745.
[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.
[14] A. Yang, Y. Bai, T. Xue, Y. Li, J. Li, A novel image steganography algorithm based on hybrid machine leaning and its application in cyberspace security, Future Generation Computer Systems 145 (2023) 293–302.
[15] Telli, M., Othmani, M., Ltifi, H. (2022, December). An Improved Multi-image Steganography Model Based on Deep Convolutional Neural Networks. In International Conference on Intelligent Systems Design and Applications (pp. 250-262). Cham: Springer Nature Switzerland.
[16] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical image computing and computer-assisted intervention, Springer, 2015, pp. 234–241.
[17] S. Baluja, Hiding images in plain sight: Deep steganography, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017.
[18] Huang, C. H., Wu, J. L. (2022). Image data hiding with multi-scale autoencoder network. arXiv preprint arXiv:2201.06038.
[19] Bouazizi, S., Ltifi, H. (2024). Novel diversified echo state network for improved accuracy and explainability of EEG-based stroke prediction. Information Systems, 120, 102317.
[20] A. Jaiswal, S. Kumar, A. Nigam, En-vstegnet: Video steganography using spatio-temporal feature enhancement with 3d-cnn and hourglass, 2020 International Joint Conference on Neural Networks, 2020, pp. 1–8.
[21] Telli, M., Othmani, M., Ltifi, H. (2023). A new approach to video steganography models with 3D deep CNN autoencoders. Multimedia Tools and Applications, 1-17.
[22] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri, Learning spatiotemporal features with 3d convolutional networks, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 4489–4497.
[23] Khlifi, A., Othmani, M., Kherallah, M. (2023, September). Novel Approach For Scene Semantic Segmentation Using The Recurrent-Based UNET. In 2023 International Conference on Innovations in Intelligent Systems and Applications (INISTA) (pp. 1-6). IEEE.