Multimedia Tools and Applications (2021) 80:33475–33503
https://doi.org/10.1007/s11042-021-11177-4

# Image steganography using deep learning based edge detection

Biswarup Ray[1] · Souradeep Mukhopadhyay[1] · Sabbir Hossain[1] · Sudipta Kr Ghosal[2] · Ram Sarkar[1]

Received: 10 August 2020 / Revised: 23 May 2021 / Accepted: 24 June 2021 /
Published online: 19 August 2021


**Abstract**
This paper introduces a deep learning-based Steganography method for hiding secret information within the cover image. For this, we use a convolutional neural network (CNN) with *Deep Supervision* based edge detector, which can retain more edge pixels over conventional edge detection algorithms. Initially, the cover image is pre-processed by masking the last 5-bits of each pixel. The said edge detector model is then applied to obtain a gray-scale edge map. To get the prominent edge information, the gray-scale edge map is converted into a binary version using both global and adaptive binarization schemes. The purpose of using different binarization techniques is to prove the less sensitive nature of the edge detection method to the thresholding approaches. Our rule for embedding secret bits within the cover image is as follows: more bits into the edge pixels while fewer bits into the non-edge pixels. Experimental outcomes on various standard images confirm that compared to state-of-the-art methods, the proposed method achieves a higher payload.

**Keywords** Steganography · Edge detection · LSB · CNN · Deep learning

## 1 Introduction

Steganography is the art of hiding secret data within an audio, video, image or text file either in spatial or in the transform domain. It is also regarded as an effective solution employed to protect secret or sensitive data from malicious attacks. Steganography is different from Cryptography and Watermarking in various aspects. Cryptography is the practice and study

✉ Sudipta Kr Ghosal
sudipta.ghosal@gmail.com

[1] Department of Computer Science & Engineering, Jadavpur University, Kolkata 700032, India

[2] Department of Computer Science & Technology, Nalhati Government Polytechnic, Nalhati, Birbhum, Pin 731243, India

of techniques for secure communication in the presence of third parties called adversaries. More generally, cryptography is about constructing and analysing protocols that prevent third parties or the public from reading private messages. On the other hand, Steganography is the practice of concealing a file, message, image, or video within another file, message, image, or video. Steganography requires two files: one is the message which is to be hidden and the other is the cover file which is used to hide the data/message. In Watermarking, the message which is inserted into the object (image, audio or video) is related to the object, such as the ownership status of the object, Copyright-issues and fingerprinting etc. While in Steganography, the message is usually not related to the object. In Steganography, the message which is inserted into the object is hidden (covert), while in Watermarking it is optional (can be covert or overt). The effectiveness of a Steganography technique is primarily analysed utilizing the following four criteria: payload, embedding efficiency, stego-image quality and robustness against attacks [1]. It is a measure of how many binary bits can be fabricated into the cover media which is commonly measured in bits per pixel (bpp) unit. The term embedding efficiency denotes the number of secret data bits fabricated per one change as a result of embedding. Therefore, if embedding efficiency is high then one can say that the number of necessary changes in the cover pixels has been reduced for a given payload. Stego-image quality is the parameter by which one can find the visual difference between the cover and the stego image. Peak Signal to Noise Ratio (PSNR), Structural Similarity (SSIM) index etc. is some standard metrics used to analyse the stego-image quality. On the other hand, the ability of the stego-image to resist the external attacks (both visual and geometrical) is known as robustness. Several tests are conducted to summarize the performance of robustness against attacks such as primary set, Chi-square, RS analysis etc. There are many popular Steganography techniques based on Hashing [38], Least Significant Bit (LSB) [4], Pixel Value Differencing (PVD) [22] and Exploiting Modification Direction (EMD) [48] etc. For the convenience of the readers, the evolution of the Steganography methods has been shown in the Literature Survey section.

The organization of the paper is as follows: section 2 deals with the literature survey. Section 3 highlights the motivation of this work, whereas section 4 gives an overview of the edge detection based on CNN with the Deep Supervision method. The proposed method is elaborated in section 5. Results and discussion are reported in section 6, where the security analysis is provided in section 7. Finally, the paper is concluded in section 8.

## 2 Literature survey

To date, many Steganography methods have been developed as reported in the literature which uses the concept of Hashing [38]. The hash function was used to enhance the security but the problem with it lied in the fact that the concept was easily breakable by most of the attacks. Another popular embedding method that is widely used is the LSB [4] method. This method embeds the secret data bits into the LSB of the pixels of the cover images. The cover image pixels in which the secret bits are to be embedded are selected either sequentially or randomly depending upon the technique used. LSB technique may involve LSB matching or LSB replacement. However, simple LSB was an old concept and could be easily detected through any Steganalysis techniques like RS analysis. Image distortion was more in the case of a large payload, which was another major drawback of the LSB method. In 2013, Abdulla et al. [2] suggested a secure steganographic algorithm based on bit-planes index manipulation. The

index manipulation was obtained for the LSB-0 and LSB-1 of the cover image. It was secure from steganalysis attack but the distortion was higher for greater payload. In the same year, Abdulla et al. [3] extended Fibonacci-like steganography by bit-plane(s) mapping. It was secured and robust but higher time complexity was the major drawback of this method.

Many Steganography methods work in the transform domain. In 2011, Bhattacharyya et al. in [5] exploited Discrete Fourier Transformation (DFT) with LSB technique by dividing the cover image in a $2 \times 2$ sliding window manner. But the time complexity was high for this method. In the transform domain, several integer transformation-based methods have been exploited for embedding in the last few years such as Laguerre Transform [18], Catalan Transform [33], Lah transforms [16] etc., which tried to increase the security of the secret data. The major shortcoming of these methods was higher time complexity than spatial domain methods. In more recent years, Wang et al. [44] proposed a new distortion function-based algorithm to use the Discrete Cosine Transform (DCT) in JPEG Steganography. However, the payload was low in this method.

Several methods in the spatial domain have also been proposed. In the year 2014, a high-capacity index function-based data hiding method was proposed by Jung in [26]. The cover image was divided into sub-blocks that did not overlap with each other and the basis pixel was calculated using the proposed index function. The proposed method had a higher payload capacity but the PSNR was low. Xie et al. in [42] proposed a new method for edge detection known as the holistically nested edge detection algorithm in 2015. It mainly focussed on two significant aspects namely the holistic image training and multi-scale feature learning. This method claimed that it outperformed many Convolutional Neural Network (CNN) based edge detection algorithms in terms of speed i.e., orders of magnitude. The method was not secure against steganalysis attacks. One year later, Hosam in [22] proposed a novel Steganography technique based on PVD where the cover image is divided into blocks of size $3 \times 3$ and edge-based embedding is exploited to achieve a higher payload. The disadvantage of the above mentioned PVD method was that the image distortion was high. In 2019, Younus et al. in [48] stated a novel data hiding technique based on EMD. They used Knight Tour Algorithm for efficient embedding. The payload and PSNR obtained in the process are high, but the computational complexity of this method was also high. In the same year, Shanthakumari in [37] proposed dual-layer security of data using LSB inversion image Steganography with elliptic curve cryptography encryption algorithm. To upgrade the embedding limit and to give a stego image indistinct to human vision, a structure for concealing huge volumes of information in pictures by combining Cryptography and Steganography while causing negligible perceptual debasement was proposed. PSNR was very low concerning payload. In more recent years, Mukherjee in [32] introduced an efficient multi-bit Steganography algorithm in the spatial domain with two-layer security. The method introduced a technique to first encode the information to be embedded using the Galois field. Then the encoded information was embedded in the cover multimedia in the spatial domain. The applicability of the method in the lossy domain remained a major problem. A year later in 2019, Pak et al. in [34] put forward a novel colour image LSB steganography using the improved 1D chaotic map. The method proposed an improved dimensional chaotic model which gave better results than other chaotic models. However, it had a lesser payload capacity as compared to its PSNR, also the security could have been further strengthened.

In the last few years, researchers have shown interest in edge-based embedding techniques to achieve a higher payload with less distortion in the stego-images. In this regard, several edge detection techniques such as “Sobel”, “Prewitt” and “Canny” etc. have been used by the researchers. In the past few decades, deep learning based edge detection methods have made a

huge breakthrough bringing out better edge image outputs than the traditional edge detection methods. Some of such works are as follows. In 2013, El-Sayad et al. [14] introduced a novel edge detection method following CNN based architecture. A holistically nested approach to edge detection was proposed by Xie et al. [42] in 2015. Whereas, Xue et al. in [46] performed an extensive analysis on edge detection operator of a CNN model. In the year 2020, Wibisono et al. [21] proposed a traditional deep neural network (DNN) for edge detection. Poma et al. [35] introduced a new deep learning edge detection method which was inspired by both Holistically-Nested Edge Detection (HED) and Xception networks. In the initial years of edge-based steganography methods, Lee in [28] proposed a data hiding scheme with high embedding capacity and good visual quality based on edge detection. The Edge Detection method based on the Maximizing Objective Function (ED-MOF) was applied to detect the edge map of the 4-MSB plane of the original image. The secret message was embedded in the LSB plane. Computation complexity was high and payload was low for this method. In the year 2014, Tseng in [41] proposed a high-payload block-based data hiding scheme using a hybrid edge detector with minimal distortion. The method used a block-based fuzzy logic based algorithm. The block-based design selects the appropriate number of non-edge/edge LSBs that achieve minimal distortion for each block. The method had a low PSNR concerning payload. In the same year, Islam et al. in [24] put forward an edge detection based Steganography method in which the number of pixels belonging to edges depends upon the payload to be embedded. Two years later, Dube et al. in [11] suggested a novel edge-based Steganography technique in which only the sharp edge regions of the gray-scale images were used to hide the data. For this, the image was clustered into groups of two pixels each and the difference in their pixel values was recorded. If this difference was more than a certain threshold value, the pair was used to hide the data. This technique showed better results than previous LSB and PVD techniques and also improved the quality of the stego image. This method failed to resist external stego attacks. A year later in 2017, Suneetha in [27] came up with another edge-based Steganography for secured transmission of cloud data. In this technique, the Canny edge detection method was used. The message information was then encoded using encryption algorithms and the key was obtained. To hide this key in the cover image, its edge image was obtained. Fibonacci pixels were selected from this edge image. The encryption key was then hidden in these edge pixels. The major shortcomings of the method were higher complexity due to Fibonacci calculation and lower PSNR. Chen et al. in [9] proposed a novel embedding technique based on a hybrid edge detector. It offered a high payload and was secure against Steganalysis attacks, but the time complexity was high. In 2018, Ghosal et al. in [15] proposed a Steganography method based on the Laplacian of Gaussian (LoG) edge detector. A novel embedding technique was used to achieve a higher payload. The advantage of this method is that it achieved a higher payload. But higher computational complexity was the main disadvantage of this method. Also in the same year, Setiadi et al. in [36] proposed an Enhanced LSB image Steganography using the hybrid Canny-Sobel edge detection. The pixels were embedded in the edge areas which were found by a hybrid Canny-Sobel detector. But, the method had a lower payload capacity. Kich et al. in [25] proposed a new technique in the same year which dissimulated the secret data in the edge pixels. It was based on over-segmentation using Modified Simple Linear Iterative Clustering which made it possible to segment the images into K regions known as a superpixel. Experimental results showed that the method was superior in terms of embedding capacity and imperceptibility. However, the method was not secure against all types of steganalysis attacks, which was the main disadvantage of this method. In recent years, Dhargupta et al. in [10] suggested an algorithm of Fuzzy edge

detection based on Steganography using modified Gaussian distribution. The scheme offered variable payload in the stego image. But mathematical interpretation was a bit difficult and it was not stable against Steganalysis and also PSNR was low for payload. Gujjunoori et al. in [20] proposed a difference expansion (DE) based reversible data embedding and edge detection method in the year 2019. Three proposed schemes were used namely the DER scheme, DER layer-2 scheme, and DEED scheme. Though the method was simple, the payload and the PSNR were low, complexity was high. In the same year, Zou et al. in [51] introduced a deep learning-based steganography method comprised of a feature learning method that mainly focused on the importance of global information. A major drawback of this method was neglecting local information features that increased Visual distortion in stego-image.

## 3 Motivation

In the Literature Survey section, we discuss the problems associated with existing Steganography methods. These methods mainly have less embedding capacity, high distortion, the neediness of more pre-processing which make the methods computationally expensive. In general, the embedding of secret information is done more in the edge area and less in the non-edge areas of any cover image. Based on the analysis of the contents present in an image, it is seen that the distortion occurring in the contents of an image due to a specific attack is much lesser in edge areas as compared to the non-edge areas. As a result, it is found that many researchers have relied upon edge detection based Steganography techniques in recent times. However, the edge detectors such as Sobel [31], Prewitt [47], Laplacian of Gaussian [43] and Canny [29] sometimes fail to achieve satisfactory results either by producing a fewer number of edges or by generating some noises. For quick reference, some limitations of the conventionally used edge detection techniques are mentioned in Table 1.

**Table 1** Some limitations of conventionally used edge detection techniques

| Edge detection technique | Limitations |
|---|---|
| **Canny** [29] | • This does not perform well on images captured using varying lighting condition<br>• This requires many pre-processing steps and manual tuning of parameters. [8] |
| **Sobel** [31] | • The major disadvantage of Sobel is the signal to noise ratio. With the increase in noise, the gradient magnitude of the edges also degrades which produces inaccurate results. [8]<br>• In the images containing textures, this technique fails drastically. |
| **Prewitt** [47] | • This technique is sensitive to noise.<br>• The magnitude of the edges degrades as the level of noise present in the image increases. [39] |
| **LoG** [43] | • The disadvantage is sensitivity to the noise. In detecting the edges and their orientations are increased in the noise to the image this will eventually degrade the magnitude of the edges.<br>• The second disadvantage is that the operation gets diffracted by some of the existing edges in the noisy image. [39] |
| **Haar** [49] | • The method helps in finding the edge in the high contrast region of an image, however, in the low contrast region, edge detection becomes difficult.<br>• Calculation of wavelet function (DWT) is time-consuming. |
| **Kirch** [17] | • It is very much sensitive to noise. As a result, inaccurate calculation makes it difficult in finding proper edge images. |

To encounter the problems reported in Table 1, in the past few years, deep learning based models such as CNNs have dominated as the edge detection methods due to their ability to capture the spatial distribution of the images efficiently. Deep learning based edge detection methods take care of these problems by generating a larger number of edge pixels and also by controlling the problem of noise efficiently by leaving only a little amount of noise for the other conventionally used edge detectors.

## 4 Edge detection based on CNN with deep supervision

Convolutional networks (ConvNets) have recently shown great success in large-scale image recognition which has become possible due to the large public image repositories, such as ImageNet and high-performance computing systems, such as GPUs or large-scale distributed clusters.VGG16 is a well-established CNN model proposed by K. Simonyan [50]. The main characteristic of VGG16 is that it replaces the large-sized filters i.e., 11 and 5 in the first and second convolutional (Conv.) layers, respectively with multiple $3 \times 3$ sized filters one after another. The input to cov1 layer is of fixed size $224 \times 224$ RGB image. The image is passed through a stack of Conv. layers, where the filters are used with a very small receptive field: $3 \times 3$. The convolution stride is fixed to 1 pixel; the spatial padding of Conv. layer is such that the spatial resolution is preserved after convolution, i.e., the padding is 1-pixel for $3 \times 3$ Conv. layers. Spatial pooling is carried out by five max-pooling layers, which follows some of the Conv. layers. Max-pooling is performed over a $2 \times 2$ pixel window, with stride 2. Three Fully-Connected (FC) layers follow a stack of Conv. layers. The final layer is the soft-max layer. All hidden layers are equipped with the rectification (ReLU) non-linearity. The VGGNet used here has five stages, with strides 1, 2, 4, 8 and 16, respectively, and with different receptive field sizes, all nested in the VGGNet. Table 2 shows a summary of the configurations of the receptive fields and strides.

We have used a kernel size of $3 \times 3$ which is the least possible size for capturing the notion of left/right, up/down, and centre. By stacking two $3 \times 3$ Conv. layers (without spatial pooling or max-pooling in between) an effective receptive field of $5 \times 5$ and thus next receptive layers are achieved by the same process of stacking. The receptive field size of each of these Conv. layers is equal to the corresponding side-output layer.

When applying a convolution with the kernel size k, the padding size p, and the stride size s, the attributes of the output layer can be calculated by the following equations:

**Table 2** The receptive field and stride size used in VGGNet. The last convolutional layers of each stage are linked to additional side-output layers

| Layer | Receptive field | Stride |
|---|---|---|
| 2nd layer of Conv1 | 5 | 1 |
| Pooling1 | 6 | 2 |
| 2nd layer of Conv2 | 14 | 2 |
| Pooling2 | 16 | 4 |
| 3rd layer of Conv3 | 40 | 4 |
| Pooling3 | 44 | 8 |
| 3rd layer of Conv4 | 92 | 8 |
| Pooling4 | 100 | 16 |
| 3rd layer of Conv5 | 196 | 16 |

$$n_{out} = \left[\frac{n_{in} + 2p - k}{s}\right] + 1 \tag{1}$$

$$j_{out} = j_{in} \times s \tag{2}$$

$$r_{out} = r_{in} + (k-1) \times j_{\text{in}} \tag{3}$$

$$start_{out} = start_{in} + \left(\frac{k-1}{2} - p\right) \times j_{in} \tag{4}$$

Here, eq. (1) calculates the number of output features based on the number of input features and the convolution properties. Equation (2) calculates the jump in the output feature map, which is equal to the jump in the input map times the number of input features that are jumped over when applying the convolution (the stride size). Eq. (3) calculates the receptive field size of the output feature map, which is equal to the area that is covered by $k$ input features $(k-1) \times j_{in}$ plus the extra area that is covered by the receptive field of the input feature that is on the border. Eq. (4) calculates the centre position of the receptive field of the first output feature, which is equal to the centre position of the first input feature plus the distance from the location of the first input feature to the centre of the first convolution $(k-1) \times j_{in}$ minus the padding space $p \times j_{in}$.

The proposed edge detection technique is based on *CNN with Deep Supervision*. The role of *Deep Supervision* is to generate natural and inherent side outputs for each layer. The resultant side outputs from each layer are aggregated with fused weight output to find the final output. The architecture of the proposed model is represented in Fig. 1. The proposed network consists of a VGG Net model whose final stage is trimmed along with the 5th max-pooling layer to reduce the computational complexity in terms of memory and time, and also to produce better side outputs, which cannot be achieved with a higher stride.

### 4.1 Model parameters

To find the optimal number of training iterations the PSNR value is used to evaluate the network output by increasing the number of iterations during training.

From the graph shown in Fig. 2, it can be visualized that the best PSNR value is found when the number of epochs is 10,000. Thus, during training 10,000 epoch value has been set. The other hyper-parameter values have been given default values. The strides for the 5-staged model are 1, 2, 4, 8 and 16, respectively. The hyper-parameters include batch size (20), learning rate (1e-6), loss-weight $\alpha_m$ for each side output layer (1), momentum (0.92), initial values of the fusion layer weights (1/5), weight decay (0.0002), and several training iterations (10,000). The 16 layer VGGNet model has been used whose final stage is trimmed along with the 5th max-pooling layer to reduce the computational complexity in terms of memory and time requirements.

**Training phase** The formulation for the training phase is described here.

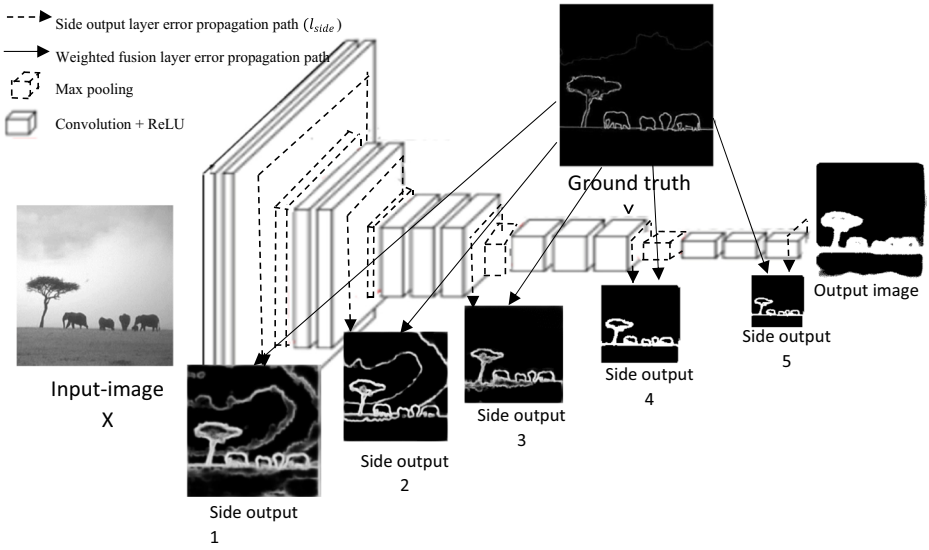



**Fig. 1** Illustrating the error propagation path of the proposed edge detector model called CNN with Deep Supervision. The model uses a VGG Net whose final stage is trimmed along with the 5th max pooling layer to reduce the computational memory and time loss also to produce meaningful side output which cannot be achieved with a higher stride yield. Deep Supervision is implemented at each side output layer, guiding the side outputs towards better edge predictions. The outputs of the model are both multi-scaled, with side output size becoming smaller and an increase in the receptive field. A weighted-fusion layer is added to learn how the combination of outputs from multiple scales works. The entire network is trained with multiple error propagation path forms the side outputs

For the input dataset

$$T = \{(X_n, Y_n)\}, n = 1, 2, \ldots . N \tag{5}$$

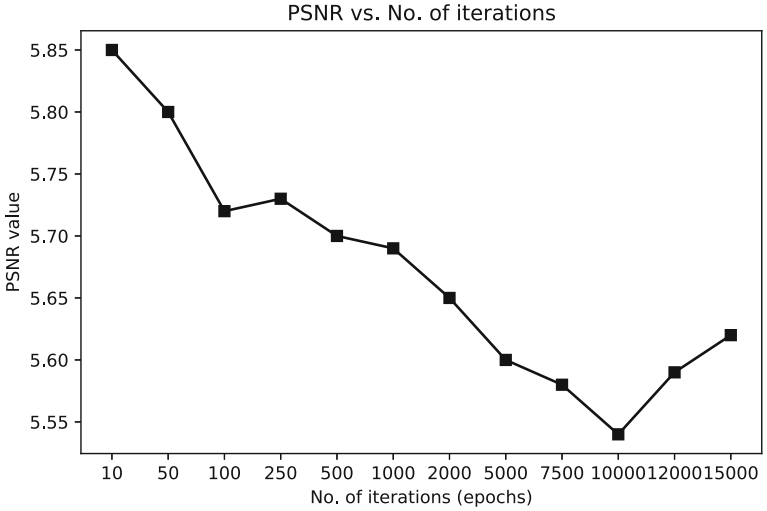



**Fig. 2** Showing the changes in PSNR value for the number of iterations during training the CNN model

where, the raw input image: $X_n = x_j^n, j = 1, \ldots \mid X_n \mid$ taking $x_j$ as each pixel of an input image $X$.

and the ground edge map for $X_n$: $Y_n = y_j^n, j = 1, \ldots \mid X_n \mid$ and $y_j^n \in \{0, 1\}$,

taking $y_j$ as each pixel for a ground edge map $Y$

$$\text{The objective function is : } f_{side}(P, w) = \sum_{m=1}^{M} \alpha_m l_{side}{}^{(m)}(P, w_m) \tag{6}$$

where$l_{side}$ denotes the loss function of the side outputs created at each layer.

$P$ denotes the collection of standard network layer parameters.

Each side output layer is also associated with a classifier, in which the corresponding weights are denoted as $w = (w^{(1)}, \ldots. w^{(M)})$

M is the number of side output layers.

and $\alpha_m$ is the loss-weight for each side output layer

At each side output, an edge prediction of $Y_{side} = \sigma(A_{side}{}^{(m)})$ is obtained, where the activation function of the side output for the layer m is $A_{side}{}^{(m)} = \{a_i, i = 1, \ldots \mid Y \mid\}$

To directly utilize side outputs, a weighted-fusion layer is added to the network and learning of the fusion weight is done simultaneously during training.

The loss function for the fusion layer is defined as follows:

$$f_{fusion}(P, w, h) = Dist\left(Y, Y_{fusion}\right) \tag{7}$$

where, $Y_{fusion} = \sigma(\sum_{m=1}^{M} g_m A_{side}{}^{(m)})$, $g = h_1, \ldots, h_M$ is the fusion weight.

$Dist(..)$ is the distance between the fused output and the ground truth map

Hence after taking into account the calculated $f_{fusion}(P, w, h)$,the objective function can be minimized by back propagation:

$$R(P, w, h)' = argmin\left(f_{side}(P, w) + f_{fusion}(P, w, h)\right) \tag{8}$$

**Testing phase:** Prediction of the edge maps for a given image I, from side outputs generated in each layer along with the edge prediction from the weighted fusion output, is obtained.

$$\left(Y_{fusion}, Y_{side}{}^{(1)}, \ldots, Y_{side}{}^{(M)}\right) = F\left(I, (P, w, h)'\right) \tag{9}$$

where $F(\ldots)$ represents the edge maps generated by the proposed model.

The model is trained on the Berkeley Segmentation Dataset and Benchmark (BSDS 500) [30]. It consists of 200 training, 100 validation and 200 testing images. The ground truth boundaries are found by manual segmentation of the dataset. Some customizations are made by training the model with 8 variants of binary edge patterns along with other binary non-edge patterns. The images of the different binary edge patterns as shown in Fig. 3 are used as training pattern for the model along with other non-edge binary image patterns.

The side outputs generated for each layer give more natural edge images. The final output is then found out by merging these side outputs with the weighted fusion layer output which are in gray-scale formats. To get the edges for masking, it is converted to a binary image using different thresholding approaches. To prove the robustness of the edge maps produced by the CNN with Deep Supervision four different types of thresholding – two global and two adaptive approaches are considered.

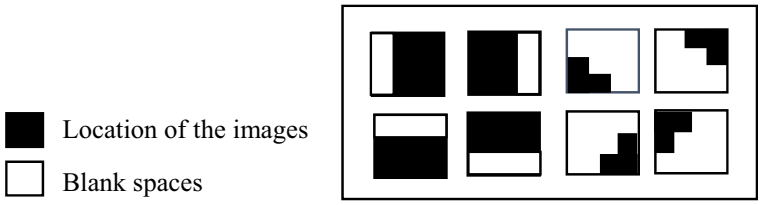



**Fig. 3** Showing the 8 binary edge patterns used to train CNN with Deep Supervision edge detector model

**Hard thresholding** Pixels with an intensity greater than 127 are converted to white (255), and all other pixels are converted to black (0).

**Otsu thresholding** This is one of the most popular and standard global thresholding approaches used for binarization.

**Adaptive mean thresholding** The mean of the local intensity distribution is taken for the threshold function.

**Gaussian adaptive thresholding** Here the threshold value is the weighted sum of neighbourhood values where weights are a Gaussian window

The Gaussian window is defined as:

$$w(n) = e^{\frac{-1}{2}\left(\frac{n}{\sigma}\right)^2} \tag{10}$$

where σ is the standard deviation.

It is to be noted that all the four binary versions of a cover image are used separately for data embedding to get an idea about the usefulness of the proposed edge detection method, irrespective of the thresholding approach taken, in developing the edge-based Steganography method.

## 5 Proposed method

In this section, we have designed a security framework by which an authorized person, say 'Alice' embeds secret image (SEI) within a cover image (CI) and obtains the stego image (SI). To accomplish this, the CI is pre-processed through masking (i.e., by setting 5-LSBs of each pixel to '0') and in turn, CNN with Deep Supervision based edge detection method followed by a binarization scheme is applied over the masked image (MI) to obtain the edge or reference image (RI). Since the edge pixels are more tolerable than non-edge pixels against distortion, more bits of SEI are embedded into the edge pixels and fewer bits are embedded into the non-edge pixels of CI. At the receiving end, another authorized person, say 'Bob' receives the SI, re-calculates RI and performs the reverse procedures to recover SEI from SI.

The major steps are summarized here:

a) Alice chooses CI and SEI from her image dataset
b) Obtain MI by clearing 5-LSBs of each pixel of CI

c) Obtain RI by applying CNN with Deep Supervision based edge detector followed by a binarization scheme over CI
d) Obtain SI by embedding bits of SEI into CI based on RI. Edge pixels of RI ensures embedding of more bits than non-edge pixels
e) Re-obtain MI by clearing 5-LSBs of each pixel of SI
f) Re-obtain RI by applying CNN with Deep Supervision based edge detector followed by a binarization scheme over SI
g) Bob extracts out SEI from SI.

The block diagram of the proposed framework is shown in Fig. 4.

To understand the concept in great detail, the proposed method is segmented into three major sections: Pre-processing and edge detection, Embedding and Extraction. The Pre-processing and edge detection phase is discussed in Section 5.1. The embedding and extraction phases are elaborately described in Section 5.2 and Section 5.3 respectively.

### 5.1 Pre-processing and edge detection

In this section, our proposed Steganography method, which uses **CNN with Deep Supervision based edge detection** is reported. Initially, the *CI* is taken as input and *MI* is produced by masking the last 5-bits of each pixel of CI. Then from MI, the reference edge image is calculated by applying **CNN with Deep Supervision based edge detection** which is discussed in Section 3. Then the gray-scale edge image is binarized using **Hard thresholding, Otsu thresholding, Adaptive mean thresholding** and **Gaussian adaptive thresholding** to get four different reference images (*RI*). Then more numbers of bits are embedded in edge pixels and a lesser number of bits in non-edge pixels to get the SI. In the receiver's end, SI is undergone through the same pre-processing (i.e., masking) and then the same edge detection method is applied to re-generate the identical RI. From RI, secret bits of SEI are extracted out. Figure 5 depicts the flow diagram of the pre-processing and edge detection process.

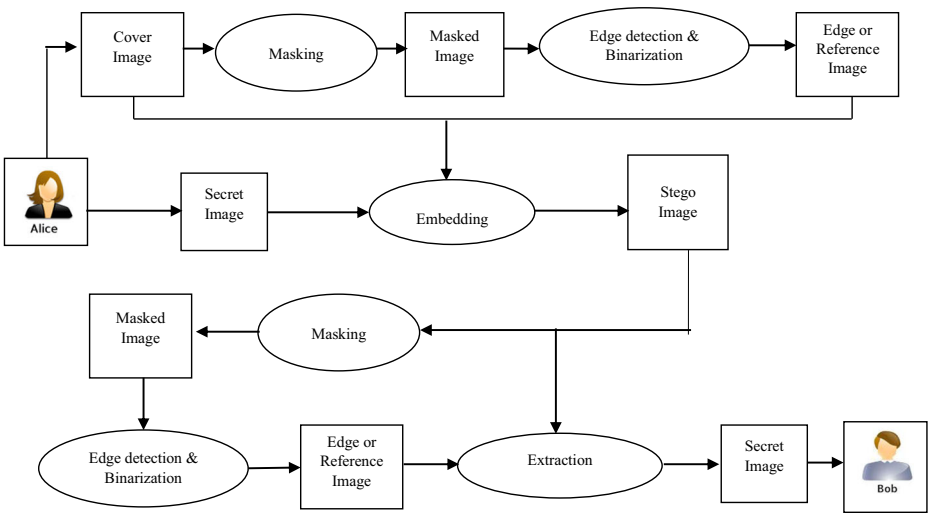



**Fig. 4** Block diagram of proposed steganography framework

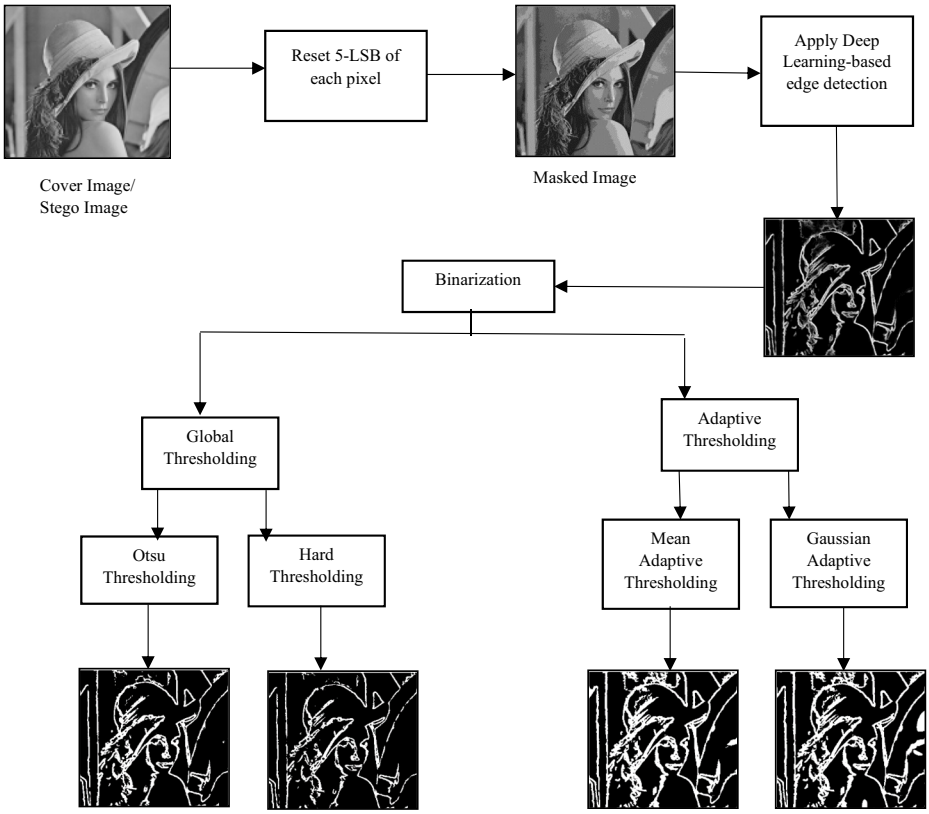



**Fig. 5** Flow diagram of the Pre-processing and edge detection phase

**Pseudocode of Pre-processing and edge detection** ($CI_{m\times n}/SI_{m\times n}$)

**1. Start**
**2. for** i ← 1 **to** m **do**
**3. for** j ←1 **to** n **do**
**4.** $PI_{i\times j} = CI_{i\times j}/SI_{m\times n}$**&** 0b11100000;//**unsettling 5 LSBs in pre-processing step.**
**5. end**
**6. end**
**7.** $RI_{m\times n}$ = **CNN with Deep Supervision based edge detection and Binarization** ($PI_{m\times n}$);
**8. Stop**

## 5.2 Embedding method

In the previous section, we get the four RIs by applying the binarization techniques. With these RIs, we have a clear idea of edge and non-edge pixels. We convert the secret image (*SEI*) into a 1D bitstream called 's'. Then we embed 'x' number of bits from s in each edge pixel of CI and 'y' number of bits in each non-edge pixel of CI by the LSB method. Here x > y because more number of bits are embedded in edge pixels to get a less distorted stego image (*SI*′). A pixel

adjustment process is also applied to decrease distortion and enhance the visual quality of the *SI* without hampering the secret data. Figure 6 depicts the flow diagram of the embedding process.

### 5.3 Extraction methods

The RI is obtained from the SI by the pre-processing step illustrated in the Section 5.1. From RI, we can easily detect, which pixel of SI is an edge pixel or which is of background pixel. So x numbers of LSBs are extracted from edge pixels and y numbers of LSBs are extracted from a non-edge pixel of SI. The extracted bits are stored in 1D array s. From s, we can easily detect the Secret image (SEI). Figure 7 shows the flow diagram of the extraction process. The pseudo-code of extraction is given below:

### 5.4 Result, comparison and analysis

To showcase the superiority of the proposed edge detection technique over classical edge detectors

(Canny [29], Sobel [31], Prewitt [47], Laplacian of Gaussian [43]) in the field of steganography.

The number of edge pixels for each technique for the 'Cameraman' image of dimension $128 \times 128$ is calculated. A higher number of edge pixels ensures higher embedding in edge pixels, which in turn results in less distortion of the stego image when being attacked. The number of edge pixels for each of the methods along with their resultant edge image is compared with the edge image of the proposed method in Fig. 8.

The deep learning-based edge detection technique, used in the proposed steganographic method, is efficient in generating edge images of various environmental conditions like cloudy, foggy, night, rainy and shining. In Fig. 9, we have shown the edge images of different environmental situations and compared them with Canny and Sobel methods in terms of the number of edge pixels generated.

To analyze the performance of the stated Steganography scheme, the two most widely metrics in the literature namely, payload and stego image quality, have been taken into account. The payload metric reveals the information of how many bits of secret information

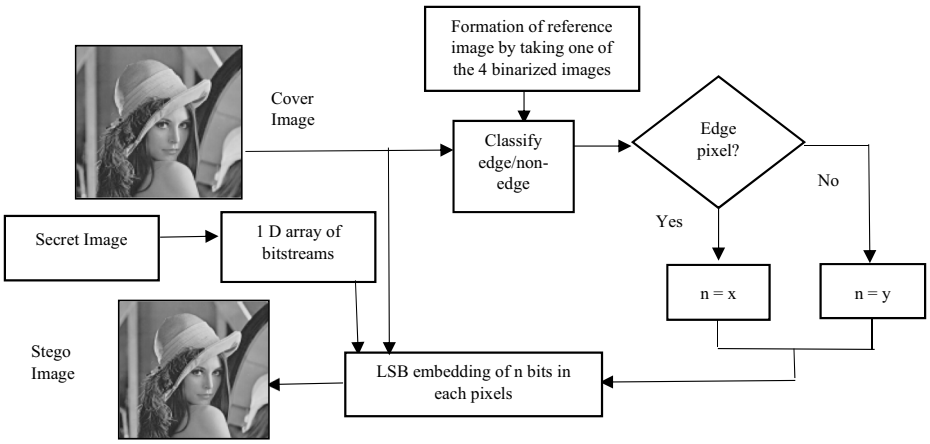



**Fig. 6** Flow diagram of the embedding process

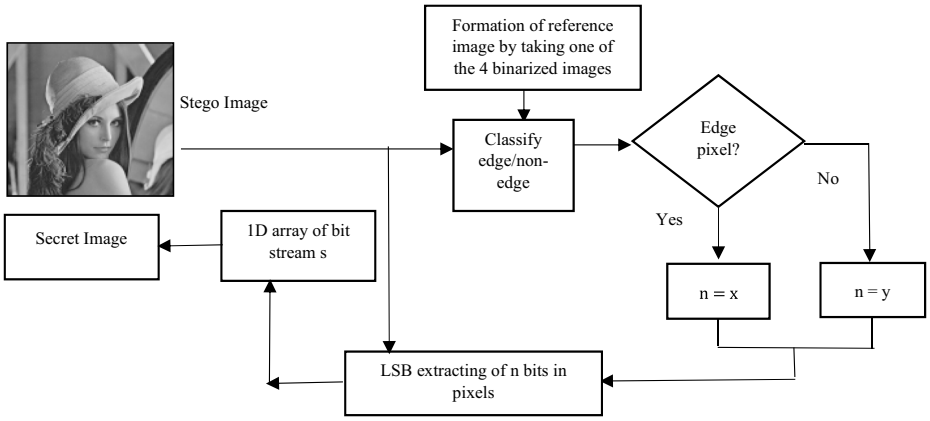



**Fig. 7** Flow diagram of the extraction process

**Pseudocode of Embedding technique** ($RI_{m\times n}, SEI_{l\times k}$)

**1. Start**
**2. for**i←1 **to** l **do**
**3. for**i←1 **to** k **do**
**4.** s[w] =$SEI_{i\times j}$; **//make secret 1D bitstream.**
**5.** w=w+1;
**6. end**
**7. end**
**8.**
**9. for** i ← 1 **to** m **do**
**10. for** j←1 **to** n **do**
**11. if** ($RI_{i\times j}$==255)
**12.** $SI'_{i\times j}$=**LSB**($CI_{i\times j}$,s,x); **//LSB embedding of x bit of s in edge pixel**
**13. else**
**14.** $SI'_{i\times j}$=**LSB**($CI_{i\times j}$,s,y); **//LSB embedding of y bit of s in non-edge pixel**
**15. endif**
**16. end**
**17. end**
**18.**
**19. for** i ← 1 **to** m **do**
**20. for** j←1 **to** n **do**
**21.** $SI_{i\times j}$=Pixel Adjustment($SI'_{i\times j}$);
**22. end**
**23. end**
**24. Stop**

is embedded into each pixel of the cover image of dimension A × B. On the other hand, the quality of a stego image has been assessed utilizing its PSNR value which is calculated as:

$$\mathrm{PSNR} = 10 \times \log\left(\frac{255^2}{MSE}\right) \tag{11}$$

$$\text{where Mean Square Error(MSE)} = \frac{\sum_{i=1}^{A}\sum_{j=1}^{B}[I_a(i,j) - I_b(i,j)]^2}{A \times B} \tag{12}$$

$I_a$ and $I_b$are two images of dimension A × B.

Payload is measured in terms of bits per pixel (Bpp) and PSNR as dB.

SSIM (Structural Similarity Index for measuring) an indicator to measures changes in brightness, contrast and structure of an image has also been assessed to measure the similarity between the stego and the cover image. Hence, the higher value of SSIM ensures minimum distortion of stego images from the cover images from which they were generated. The SSIM index between two images of $x$ (cover images) and $y$ (stego images) is calculated as:

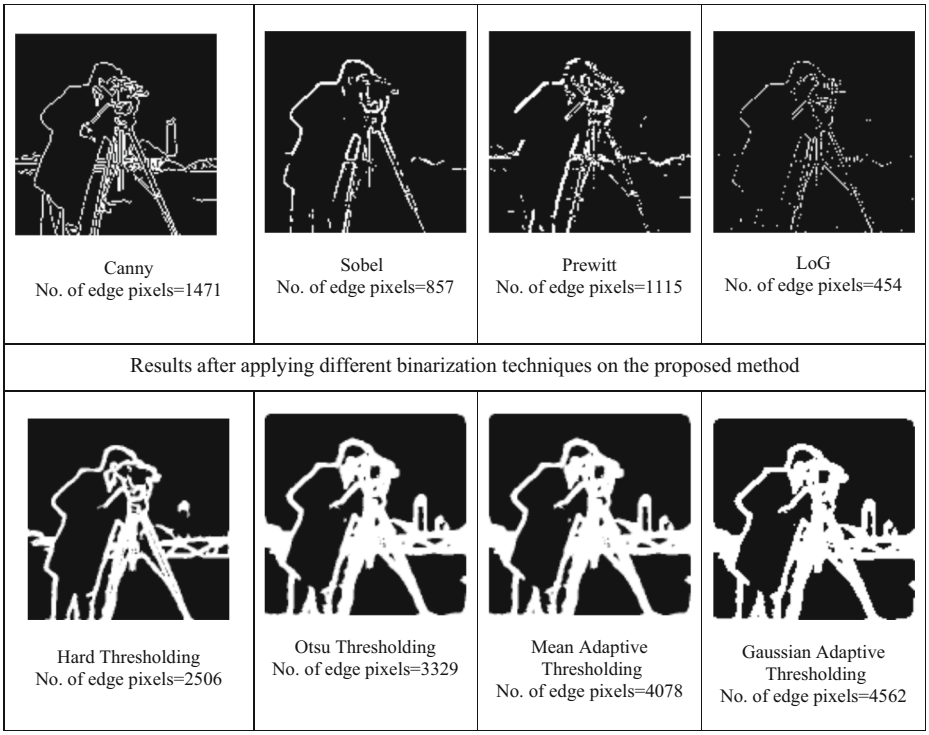



**Fig. 8** Displaying the resultant edge output of different edge detection techniques for the Cameraman image of dimension 128 × 128. The number of edge pixels for each edge detection method is also noted

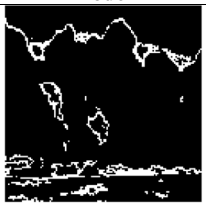
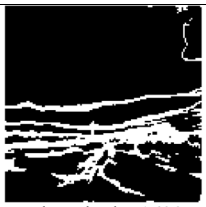
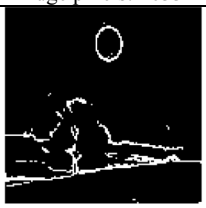
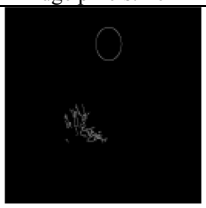
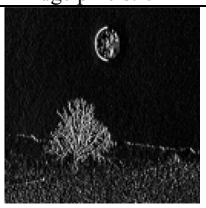
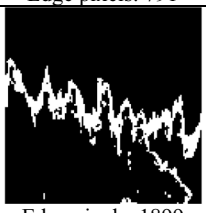
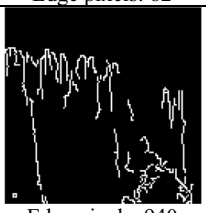
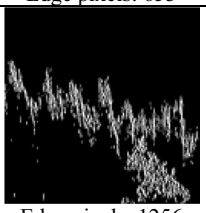
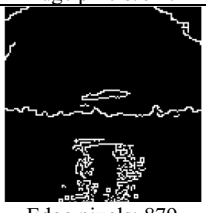
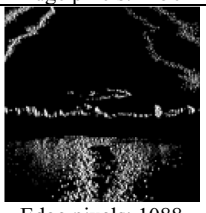
| Environmental condition | Edge detection by | | |
|---|---|---|---|
| | Proposed CNN model | Canny | Sobel |
|  Cloudy |  Edge pixels: 1368 |  Edge pixels: 867 |  Edge pixels: 1137 |
|  Foggy |  Edge pixels: 1633 |  Edge pixels: 261 |  Edge pixels: 612 |
|  Night |  Edge pixels: 791 |  Edge pixels: 82 |  Edge pixels: 653 |
|  Rain |  Edge pixels: 1899 |  Edge pixels: 940 |  Edge pixels: 1256 |
|  Day (sunshine) |  Edge pixels: 1466 |  Edge pixels: 879 |  Edge pixels: 1088 |

**Fig. 9** Comparison of the edge images generated by Deep learning-based methods with Canny and Sobel methods in terms of the number of edge pixels (images of various weather conditions are considered)

$$SSIM(x,y) = \frac{\left(2\mu_x\mu_y + K_1\right)\left(2\sigma_{xy} + K_2\right)}{\left(\mu_x{}^2 + \mu_y{}^2 + K_1\right)\left(\sigma_x{}^2 + \sigma_y{}^2 + K_2\right)} \tag{13}$$

where $K_1$ & $K_2$ are constants obtained by the equations:

$K_1 = (c_1L)^2$, $K_2 = (c_2L)^2$ and $L = 255$ for an image with 8 bits/pixel and $c_1 \ll 1$, $c_2 \ll 1$ are very small constants.

And, $\mu_x$, $\mu_y$ are the averages, $\sigma_x$, $\sigma_y$ are the variances, $\sigma_{xy}$ is the covariance of $x$ & $y$.

The experimentation has been conducted on 10 benchmark gray-scale images of USC-SIPI [45] image database as shown in Figs. 10, and 5 images of different weather conditions taken from Kaggle [23]. The dimensions of the images are 128 × 128. The images are (i) Lena, (ii) Baboon, (iii) Pepper, (iv)Splash, (v) Sailboat, (vi) Stream and bridge, (vii) Tank, (viii) Fishing boat, (ix) Aerial and (x) Airplane. The size of the secret image (here "Male") is varied and is being embedded in the cover images as shown in the above embedding process.

(i) Lena, (ii) Baboon, (iii) Pepper, (iv) Splash, (v) Sailboat, (vi) Stream and bridge, (vii) Tank, (viii) Fishing boat, (ix) Aerial, (x) Airplane

**Maximum size of SEI embedded in the cover image** Let us consider that the "Male" (let's say n × n) image is to be embedded within the R × C cover image. Let cover image has $N_e$ number of edge pixels and ((R × C)-$N_e$) number of non-edge pixels. We consider x and y numbers of LSBs for replacement in edge and non-edge pixels respectively. The hidden image of dimension n × n must satisfy the following condition:

$$n \times \mathrm{n} \times 8 \leq (N_e \times \mathrm{x}) + ((\mathrm{R} \times \mathrm{C}) - N_e) \times \mathrm{y}) \tag{14}$$

In case of maximum payload, n should be $n_{max}$

$$n_{max} = floor\left(\sqrt{\frac{(R \times C \times y) + N_e(x-y)}{8}}\right) \tag{15}$$

**Maximum payload offered by the cover image in bpp** In this portion, we will develop a formula to calculate the payload of maximum embedding in bpp. The number of total pixel in the cover image is R × C. For max embedding, The number of embedded bits should be

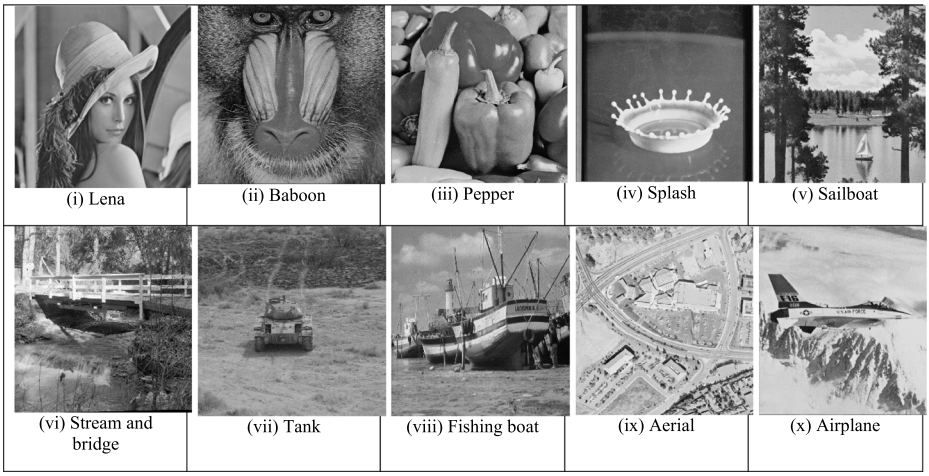



**Fig. 10** Different gray-scale cover images of the USC-SIPI database

| CI | Binarization technique | SEI | SI (128×128) (x=2,y=1) | Extracted SEI (x=2,y=1) |
|---|---|---|---|---|
|  Lena (128× 128) | Otsu Threshold |  Male (56×56) |  *PSNR:47.86 dB* |  Male (56×56) |
| | Hard threshold |  Male (54×54) |  *PSNR: 48.32 dB* |  Male (54×54) |
| | Adaptive mean threshold |  Male (55×55) |  *PSNR: 48.10 dB* |  Male (55×55) |
| | Adaptive Gaussian Threshold |  Male (56×56) |  *PSNR: 47.99 dB* |  Male (56×56) |

**Fig. 11** Cover, hidden, stego and extracted images for different binarization methods used to achieve edge image

$$B_{embed} = (R \times C \times y) + N_e(x-y). \tag{16}$$

Then payload is calculated as follows:

$$\begin{aligned} Payload &= \frac{(R \times C \times y) + N_e(x-y)}{R \times C} \; bpp \\ &= \left[ y + \frac{N_e(x-y)}{R \times C} \right] bpp \\ &= \left[ \left( 1 - \frac{N_e}{R \times C} \right) y + \frac{N_e}{R \times C} x \right] bpp \end{aligned} \tag{17}$$

The cover, hidden, stego and extracted images are shown in Fig. 11. The PSNR and MSE of the respective images are also reported. The PSNR and MSE values are infinity and zero which ensures lossless extraction. Further, we calculate BER $= \frac{\text{Error bits number} \times 100\%}{\text{Total secret bits}}$, which is 0%. From this, we can conclude that perfect extraction is possible by our method.

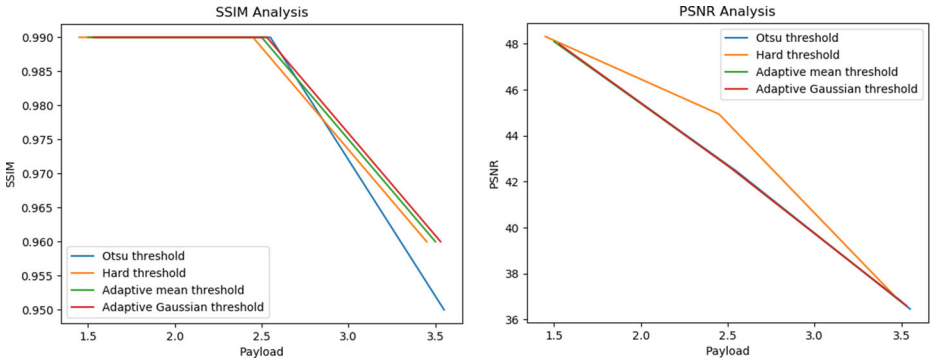
**Table 3** PSNR values given by stego images of the proposed Steganographic scheme for different binarization threshold with different payload

| Cover image | Application of different binarization methods after the proposed CNN based edge detection method used in the proposed steganographic scheme | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Otsu threshold | | Hard threshold | | Adaptive mean threshold | | Adaptive Gaussian threshold | |
| | Payload | PSNR | Payload | PSNR | Payload | PSNR | Payload | PSNR |
| Lena | 1.55 | 47.86 | 1.45 | 48.32 | 1.50 | 48.10 | 1.53 | 47.99 |
| | 2.55 | 42.46 | 2.45 | 44.94 | 2.50 | 42.70 | 2.53 | 42.53 |
| | 3.55 | 36.46 | 3.45 | 37.11 | 3.50 | 36.79 | 3.53 | 36.60 |
| Baboon | 1.30 | 49.07 | 1.20 | 49.61 | 1.38 | 48.64 | 1.38 | 48.67 |
| | 2.30 | 43.76 | 2.20 | 44.47 | 2.38 | 43.34 | 2.38 | 43.32 |
| | 3.30 | 37.98 | 3.20 | 38.68 | 3.38 | 37.37 | 3.38 | 37.47 |
| Pepper | 1.39 | 48.63 | 1.33 | 48.91 | 1.42 | 48.40 | 1.41 | 48.46 |
| | 2.39 | 43.31 | 2.33 | 43.53 | 2.42 | 43.09 | 2.41 | 43.14 |
| | 3.39 | 37.40 | 3.33 | 37.76 | 3.42 | 37.23 | 3.41 | 37.25 |
| Splash | 1.15 | 49.99 | 1.12 | 50.12 | 1.18 | 49.76 | 1.18 | 49.73 |
| | 2.15 | 44.76 | 2.12 | 45.13 | 2.18 | 44.63 | 2.18 | 44.60 |
| | 3.15 | 39.11 | 3.12 | 39.40 | 2.18 | 38.80 | 3.18 | 38.85 |
| Sailboat | 1.33 | 48.88 | 1.23 | 49.43 | 1.36 | 48.75 | 1.37 | 48.77 |
| | 2.33 | 43.59 | 2.23 | 44.21 | 2.36 | 43.37 | 2.37 | 43.35 |
| | 3.33 | 37.77 | 3.23 | 38.48 | 3.36 | 37.60 | 3.37 | 37.50 |
| Fishing boat | 1.30 | 49.12 | 1.21 | 49.64 | 1.35 | 48.82 | 1.34 | 48.85 |
| | 2.30 | 43.73 | 2.21 | 44.37 | 2.35 | 43.50 | 2.34 | 43.51 |
| | 3.30 | 37.94 | 3.21 | 38.68 | 3.35 | 37.63 | 3.34 | 37.70 |
| Airplane | 1.39 | 48.62 | 1.27 | 49.20 | 1.32 | 48.98 | 1.37 | 48.71 |
| | 2.39 | 43.25 | 2.27 | 43.98 | 2.32 | 43.55 | 2.37 | 43.34 |
| | 3.39 | 37.43 | 3.27 | 38.11 | 3.32 | 37.84 | 3.37 | 37.51 |
| Aerial | 1.31 | 49.05 | 1.06 | 50.64 | 1.21 | 49.59 | 1.32 | 48.95 |
| | 2.31 | 43.71 | 2.06 | 45.61 | 2.21 | 44.42 | 2.32 | 43.65 |
| | 3.31 | 37.87 | 3.06 | 39.93 | 3.21 | 38.54 | 3.32 | 37.82 |
| Tank | 1.18 | 49.75 | 1.12 | 50.12 | 1.20 | 49.64 | 1.23 | 49.50 |
| | 2.18 | 44.66 | 2.12 | 45.19 | 2.20 | 44.53 | 2.23 | 44.27 |
| | 3.18 | 38.83 | 3.12 | 39.41 | 3.20 | 38.75 | 3.23 | 38.41 |
| Stream and Bridge | 1.33 | 48.82 | 1.18 | 49.81 | 1.35 | 48.83 | 1.36 | 48.74 |
| | 2.33 | 43.62 | 2.18 | 44.62 | 2.35 | 43.47 | 2.36 | 43.40 |
| | 3.33 | 37.80 | 3.18 | 38.88 | 3.35 | 37.71 | 3.36 | 37.57 |
| Cloudy | 1.30 | 49.12 | 1.21 | 49.64 | 1.35 | 48.82 | 1.34 | 48.85 |
| | 2.30 | 43.73 | 2.21 | 44.37 | 2.35 | 43.50 | 2.34 | 43.51 |
| | 3.30 | 37.94 | 3.21 | 38.68 | 3.35 | 37.63 | 3.34 | 37.70 |
| Foggy | 1.39 | 48.62 | 1.27 | 49.20 | 1.32 | 48.98 | 1.37 | 48.71 |
| | 2.39 | 43.25 | 2.27 | 43.98 | 2.32 | 43.55 | 2.37 | 43.34 |
| | 3.39 | 37.43 | 3.27 | 38.11 | 3.32 | 37.84 | 3.37 | 37.51 |
| Night | 1.31 | 49.05 | 1.06 | 50.64 | 1.21 | 49.59 | 1.32 | 48.95 |
| | 2.31 | 43.71 | 2.06 | 45.61 | 2.21 | 44.42 | 2.32 | 43.65 |
| | 3.31 | 37.87 | 3.06 | 39.93 | 3.21 | 38.54 | 3.32 | 37.82 |
| Rainy | 1.18 | 49.75 | 1.12 | 50.12 | 1.20 | 49.64 | 1.23 | 49.50 |
| | 2.18 | 44.66 | 2.12 | 45.19 | 2.20 | 44.53 | 2.23 | 44.27 |
| | 3.18 | 38.83 | 3.12 | 39.41 | 3.20 | 38.75 | 3.23 | 38.41 |
| Shiny day | 1.33 | 48.82 | 1.18 | 49.81 | 1.35 | 48.83 | 1.36 | 48.74 |
| | 2.33 | 43.62 | 2.18 | 44.62 | 2.35 | 43.47 | 2.36 | 43.40 |
| | 3.33 | 37.80 | 3.18 | 38.88 | 3.35 | 37.71 | 3.36 | 37.57 |

In Table 3, PSNR values obtained at different bpps of payload for different binarization techniques based on stego images are reported. Here, the average PSNR obtained by our

**Table 4** SSIM values given by stego images of the proposed Steganographic scheme for different binarization threshold values having different payloads

| Cover image | Application of different binarization methods after the proposed CNN based edge detection method used in the proposed steganographic scheme | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Otsu threshold | | Hard threshold | | Adaptive mean threshold | | Adaptive Gaussian threshold | |
| | Payload | SSIM | Payload | SSIM | Payload | SSIM | Payload | SSIM |
| Lena | 1.55 | 0.99 | 1.45 | 0.99 | 1.50 | 0.99 | 1.53 | 0.99 |
| | 2.55 | 0.99 | 2.45 | 0.99 | 2.50 | 0.99 | 2.53 | 0.99 |
| | 3.55 | 0.95 | 3.45 | 0.96 | 3.50 | 0.96 | 3.53 | 0.96 |
| Baboon | 1.30 | 0.99 | 1.20 | 0.99 | 1.38 | 0.99 | 1.38 | 0.99 |
| | 2.30 | 0.99 | 2.20 | 0.99 | 2.38 | 0.99 | 2.38 | 0.99 |
| | 3.30 | 0.96 | 3.20 | 0.96 | 3.38 | 0.96 | 3.38 | 0.96 |
| Pepper | 1.39 | 0.99 | 1.33 | 0.99 | 1.42 | 0.99 | 1.41 | 0.99 |
| | 2.39 | 0.99 | 2.33 | 0.99 | 2.42 | 0.99 | 2.41 | 0.99 |
| | 3.39 | 0.96 | 3.33 | 0.96 | 3.42 | 0.96 | 3.41 | 0.96 |
| Splash | 1.15 | 0.99 | 1.12 | 0.99 | 1.18 | 0.99 | 1.18 | 0.99 |
| | 2.15 | 0.99 | 2.12 | 0.99 | 2.18 | 0.99 | 2.18 | 0.99 |
| | 3.15 | 0.95 | 3.12 | 0.95 | 2.18 | 0.95 | 3.18 | 0.95 |
| Sailboat | 1.33 | 0.99 | 1.23 | 0.99 | 1.36 | 0.99 | 1.37 | 0.99 |
| | 2.33 | 0.99 | 2.23 | 0.99 | 2.36 | 0.99 | 2.37 | 0.99 |
| | 3.33 | 0.96 | 3.23 | 0.96 | 3.36 | 0.96 | 3.37 | 0.96 |
| Fishing boat | 1.30 | 0.99 | 1.21 | 0.99 | 1.35 | 0.99 | 1.34 | 0.99 |
| | 2.30 | 0.99 | 2.21 | 0.99 | 2.35 | 0.99 | 2.34 | 0.99 |
| | 3.30 | 0.96 | 3.21 | 0.96 | 3.35 | 0.96 | 3.34 | 0.96 |
| Airplane | 1.39 | 0.99 | 1.27 | 0.99 | 1.32 | 0.99 | 1.37 | 0.99 |
| | 2.39 | 0.99 | 2.27 | 0.99 | 2.32 | 0.99 | 2.37 | 0.99 |
| | 3.39 | 0.95 | 3.27 | 0.95 | 3.32 | 0.95 | 3.37 | 0.95 |
| Aerial | 1.31 | 0.99 | 1.06 | 0.99 | 1.21 | 0.99 | 1.32 | 0.99 |
| | 2.31 | 0.99 | 2.06 | 0.99 | 2.21 | 0.99 | 2.32 | 0.99 |
| | 3.31 | 0.96 | 3.06 | 0.96 | 3.21 | 0.96 | 3.32 | 0.96 |
| Tank | 1.18 | 0.99 | 1.12 | 0.99 | 1.20 | 0.99 | 1.23 | 0.99 |
| | 2.18 | 0.99 | 2.12 | 0.99 | 2.20 | 0.99 | 2.23 | 0.99 |
| | 3.18 | 0.96 | 3.12 | 0.96 | 3.20 | 0.96 | 3.23 | 0.96 |
| Stream and Bridge | 1.33 | 0.99 | 1.18 | 0.99 | 1.35 | 0.99 | 1.36 | 0.99 |
| | 2.33 | 0.99 | 2.18 | 0.99 | 2.35 | 0.99 | 2.36 | 0.99 |
| | 3.33 | 0.96 | 3.18 | 0.96 | 3.35 | 0.96 | 3.36 | 0.96 |
| Cloudy | 1.30 | 0.99 | 1.21 | 0.99 | 1.35 | 0.99 | 1.34 | 0.99 |
| | 2.30 | 0.99 | 2.21 | 0.99 | 2.35 | 0.99 | 2.34 | 0.99 |
| | 3.30 | 0.96 | 3.21 | 0.96 | 3.35 | 0.96 | 3.34 | 0.96 |
| Foggy | 1.39 | 0.99 | 1.27 | 0.99 | 1.32 | 0.99 | 1.37 | 0.99 |
| | 2.39 | 0.99 | 2.27 | 0.99 | 2.32 | 0.99 | 2.37 | 0.99 |
| | 3.39 | 0.95 | 3.27 | 0.95 | 3.32 | 0.95 | 3.37 | 0.95 |
| Night | 1.31 | 0.99 | 1.06 | 0.99 | 1.21 | 0.99 | 1.32 | 0.99 |
| | 2.31 | 0.99 | 2.06 | 0.99 | 2.21 | 0.99 | 2.32 | 0.99 |
| | 3.31 | 0.96 | 3.06 | 0.96 | 3.21 | 0.96 | 3.32 | 0.96 |
| Rainy | 1.18 | 0.99 | 1.12 | 0.99 | 1.20 | 0.99 | 1.23 | 0.99 |
| | 2.18 | 0.99 | 2.12 | 0.99 | 2.20 | 0.99 | 2.23 | 0.99 |
| | 3.18 | 0.96 | 3.12 | 0.96 | 3.20 | 0.96 | 3.23 | 0.96 |
| Shiny day | 1.33 | 0.99 | 1.18 | 0.99 | 1.35 | 0.99 | 1.36 | 0.99 |
| | 2.33 | 0.99 | 2.18 | 0.99 | 2.35 | 0.99 | 2.36 | 0.99 |
| | 3.33 | 0.96 | 3.18 | 0.96 | 3.35 | 0.96 | 3.36 | 0.96 |




(a) (b)

**Fig. 12** (a) The graph between SSIM v/s Payload values for 'Lena' stego image of the proposed method for different binarization threshold values, (b) Graph between PSNR v/s Payload values for 'Lena' stego image of the proposed method for different binarization threshold values

method is between 36.16 dB to 50.12 dB, however, the PSNR value lies above 30 dB is considered a good quality stego image. From the table, we can easily observe that the PSNR of no stego image of 'Lena' falls under 30 dB. It is also observed from Table 2 that the variation in binarization techniques of cover images has minimal impact on the deviation of the PSNR values. In other words, we can say that the proposed method offers uniform PSNR values for all 10 images, and as the payload increases, the corresponding PSNR value reduces. We have added the PSNR values of different payloads of the images having various weather conditions to prove the effectiveness of our method.

**Table 5** Performance comparison of the proposed Steganography method with some spatial domain methods in terms of Payload and PSNR

| Image | Method | Payload (bpp) | PSNR (dB) |
|---|---|---|---|
| Lena | Mukherjee et al. [32] | 2.21 | 39.70 |
| | Pak et al. [34] | 1.0 | 41.59 |
| | Shanthakumari and malliga's scheme [37] | 1.5 | 48.13 |
| | Jung and Yoo's scheme [26] | 2.34 | 31.94 |
| | ***Proposed scheme*** | ***1.45*** | ***48.32*** |
| | | ***2.45*** | ***44.94*** |
| | | ***3.45*** | ***37.11*** |
| Baboon | Mukherjee et al. [32] | 2.25 | 35.59 |
| | Pak et al. [34] | 1.0 | 38.45 |
| | Shanthakumari and malliga's scheme [37] | 1.5 | 48.13 |
| | Jung and Yoo's scheme [26] | 2.62 | 25.96 |
| | ***Proposed scheme*** | ***1.20*** | ***49.61*** |
| | | ***2.20*** | ***44.47*** |
| | | ***3.20*** | ***38.68*** |
| Pepper | Mukherjee et al. [32] | 2.2 | 38.55 |
| | Shanthakumari and malliga's scheme [37] | 1.5 | 47.5 |
| | Jung and Yoo's scheme [26] | 2.33 | 30.42 |
| | ***Proposed scheme*** | ***1.33*** | ***48.91*** |
| | | ***2.33*** | ***43.53*** |
| | | ***3.33*** | ***37.76*** |

**Table 6** Performance comparison of the proposed Steganography method with some edge detection based Steganography methods in terms of Payload and PSNR

| Image | Method | Payload (bpp) | PSNR (dB) |
|---|---|---|---|
| Lena | Setiadi et al. [36] | 1.102 (x-y-z hybrid) | 48.08 |
| | Lee et al. [28] | 2.14 | 40.39 |
| | Dhargupta et al. [10] | 1.39 | 41.40 |
| | | 2.39 | 37.02 |
| | | 3.39 | 28.16 |
| | Tseng et al. [41] | 0.91 | 42.18 |
| | | 1.66 | 41.03 |
| | | 2.41 | 38.18 |
| | | 3.16 | 33.58 |
| | ***Proposed scheme*** | ***1.45*** | ***48.32*** |
| | | ***2.45*** | ***44.94*** |
| | | ***3.45*** | ***37.11*** |
| Baboon | Setiadi et al. [36] | 1.208 (x-y-z hybrid) | 46.577 |
| | Dhargupta et al. [10] | 1.99 | 39.10 |
| | | 2.99 | 34.07 |
| | | 3.99 | 25.71 |
| | Tseng et al. [41] | 1.06 | 41.47 |
| | | 1.80 | 40.22 |
| | | 2.56 | 37.04 |
| | | 3.32 | 32.47 |
| | ***Proposed scheme*** | ***1.20*** | ***49.61*** |
| | | ***2.20*** | ***44.47*** |
| | | ***3.20*** | ***38.68*** |
| Pepper | Setiadi et al. [36] | 1.094 (x-y-z hybrid) | 48.24 |
| | Dhargupta et al. [10] | 1.47 | 41.38 |
| | | 2.47 | 36.83 |
| | | 3.47 | 28.12 |
| | Tseng et al. [41] | 0.90 | 41.99 |
| | | 1.65 | 40.88 |
| | | 2.40 | 38.16 |
| | | 3.16 | 33.60 |
| | ***Proposed scheme*** | ***1.33*** | ***48.91*** |
| | | ***2.33*** | ***43.53*** |
| | | ***3.33*** | ***37.76*** |
| Airplane | Setiadi et al. [36] | 1.108 (x-y-z hybrid) | 47.82 |
| | Lee et al. [28] | 2.167 | 39.27 |
| | Dhargupta et al. [10] | 1.49 | 41.29 |
| | | 2.49 | 36.84 |
| | | 3.49 | 28.18 |
| | Tseng et al. [41] | 0.90 | 42.10 |
| | | 1.65 | 41.02 |
| | | 2.40 | 38.16 |
| | | 3.15 | 33.60 |
| | ***Proposed scheme*** | ***1.27*** | ***49.20*** |
| | | ***2.27*** | ***43.98*** |
| | | ***3.27*** | ***38.11*** |
| Sailboat | Tseng et al. [41] | 0.92 | 42.03 |
| | | 1.67 | 40.95 |
| | | 2.41 | 38.12 |
| | | 3.16 | 33.40 |
| | ***Proposed scheme*** | ***1.23*** | ***49.43*** |
| | | ***2.23*** | ***44.21*** |
| | | ***3.23*** | ***38.48*** |

In Table 4, SSIM values obtained at different bpps of payload for different binarization techniques based on stego images are noted. Here, the SSIM obtained are between 0.95 to 0.99. Hence, we can conclude that the proposed method offers uniform SSIM values for all 15 images, and as the payload increases, the corresponding SSIM value reduces.

For better visualization, the effect of payload on the PSNR and the SSIM for a stego image generated by the proposed scheme, graphs representing the PSNR v/s Payload, and SSIM v/s Payload for the 'Lena' stego image of the proposed scheme for different binarization techniques are shown in Fig.12.

To ensure the superiority of the proposed method in terms of payload and/or PSNR, performance analysis has been made in comparison with the state-of-the-art techniques [32] [34] [37] [26]. From Table 3 it can be observed that the proposed method could conceal more bits of data with reduced image quality distortion. Firstly, we compare our proposed method with some recent spatial domain data hiding methods in terms of PSNR and payload (bpp), which are summarized in Table 5. In Mukherjee's scheme [32] it is seen that, for 2.2 bpp of average payload, the maximum PSNR is 39.7 dB which is on an average 5 dB lesser than our method. Also for 'Lena', the payload is 0.24 dB lesser with a much lower PSNR. In Pak's [34] scheme we find that for 1 bpp of payload, the maximum PSNR is 41.59 dB which is on an average 8 dB less than the PSNR of our method even with a lesser payload. On comparing

**Table 7** Time is taken by the proposed edge detection method to generate an edge image for 10 benchmark gray-scale images of the USC-SIPI database with variable dimensions

| Image | Dimensions | Time Taken (in seconds) |
|---|---|---|
| Lena | 128×128 | 0.338065 |
| | 256×256 | 0.691117 |
| | 512×512 | 1.790223 |
| Baboon | 128×128 | 0.341054 |
| | 256×256 | 0.672169 |
| | 512×512 | 1.874955 |
| Pepper | 128×128 | 0.34803 |
| | 256×256 | 0.690115 |
| | 512×512 | 1.738352 |
| Splash | 128×128 | 0.349035 |
| | 256×256 | 0.666187 |
| | 512×512 | 1.878942 |
| Sailboat | 128×128 | 0.357015 |
| | 256×256 | 0.685169 |
| | 512×512 | 1.874954 |
| Fishing boat | 128×128 | 0.35601 |
| | 256×256 | 0.675154 |
| | 512×512 | 1.84204 |
| Airplane | 128×128 | 0.34408 |
| | 256×256 | 0.685061 |
| | 512×512 | 1.811124 |
| Aerial | 128×128 | 0.34704 |
| | 256×256 | 0.67616 |
| | 512×512 | 1.810158 |
| Tank | 128×128 | 0.34504 |
| | 256×256 | 0.686672 |
| | 512×512 | 1.841043 |
| Stream and Bridge | 128×128 | 0.343049 |
| | 256×256 | 0.671173 |
| | 512×512 | 1.850057 |

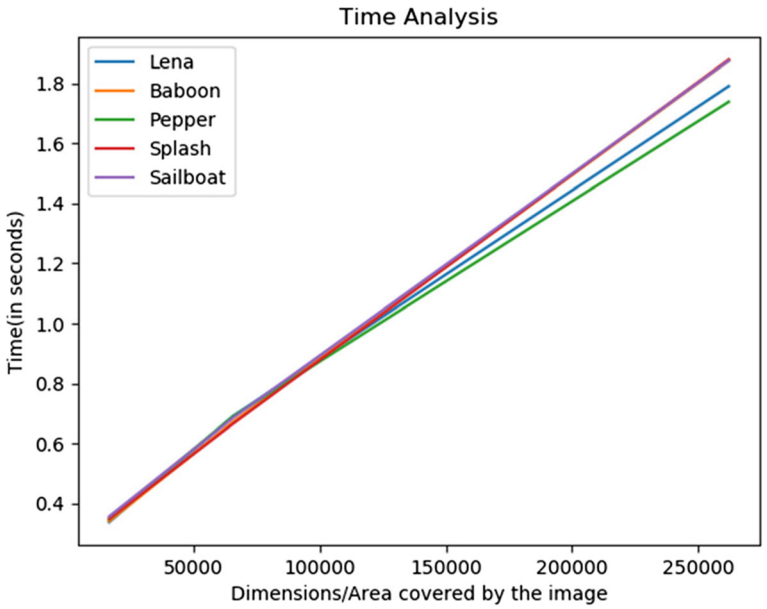



**Fig. 13** A graph between the time taken to generate an edge image (in seconds) v/s Area enclosed by the input image for 5 different images

'Lena', our method has a 0.45 dB higher payload. In Shanthakumari's scheme [37] it is seen that, for 1.5bpp of payload, the maximum PSNR is 48.13 dB which is 0.2 dB less for a comparable payload in 'Lena' for the proposed scheme. In Jung's scheme [26], for the 2.3bpp payload, the maximum PSNR is 31.94 dB which on an average is13dB lesser than the PSNR of our proposed method, also when we compare it with 'Lena' the proposed scheme has a 0.11 dB higher payload.

Now, we compare the PSNR and payload (bpp) of our proposed technique with some edge detection based data hiding schemes, and the results are summarized in Table 6. In Setiadi's scheme [36], for an average 1.1 bpp of payload, the maximum PSNR is 48.24 dB which is on an average 1 dB less than the PSNR of our proposed method. When we compare it with 'Lena' our payload is 0.35 dB more than that.

In Lee et al. [28], for an average 2.1 bpp payload, the maximum PSNR is 40.39 dB which is on an average 4 dB less than the PSNR of our proposed method, also when we compare it with 'Lena' our payload is 0.3 dB more than that. In Dhargupta's scheme [10] for the payloads 1.39 bpp, 2.39 bpp and 3.39 bpp, our proposed method outperforms the scheme for all the payload values, also having a greater PSNR for each payload value. In Tseng et al. [41] our proposed method has a much greater PSNR for each payload value. Also, Tseng's scheme has average payloads of 0.9 bpp, 1.6 bpp, 2.4 bpp and 3.2 bpp, our proposed method outperforms the scheme for all the payload values.

**Time analysis** The time taken by the proposed Deep Learning-based edge detector to generate the edge images for the 10 benchmark gray-scale images of the USC-SIPI database for dimensions $128 \times 128$, $256 \times 256$, $512 \times 512$ are noted in Table 7. It can be easily seen from the table that the time taken for the images are less than 0.36 s for images with $128 \times 128$, less than 0.7 s for images with $256 \times 256$, less than 1.9 s for images with $512 \times 512$ dimensions.

For better visualization of the effect of image dimension on the time taken by the proposed edge detector in generating a stego image a graph among the time taken (in seconds) v/s the area enclosed by the 5 different input image is shown in Fig. 13. From the graph, it can be seen that with an increase in the dimension of the image there is also an increase in the time taken by the edge detector to generate an edge image.

## 6 Security analysis

### 6.1 StegExpose analysis

This section is very important as here we judge the efficacy of our proposed Steganographic method through some steganalysis tools. We use the StegExpose tool [6], which incorporates a fusion of many well-known steganalysis methods and this assists us to decide whether a given stego image is above the threshold or not.

**Brief description of different steganalysis methods:**

- **RS analysis** [12]**:** It differentiates between the number of singular and regular groups for the LSB and shifted LSB plane.
- **Primary Sets** [19]**:** It recognizes LSB embedding by forming subsets of pixels whose cardinality differs due to the insertion of hidden message.
- **Chi-square** [40]**:** It is a simple and popular method used to test the robustness of a security system against attacks. It depends on probability analysis of stego image after putting the information inside it using the LSB method and then compares it with the probability analysis of the original image to check the difference between them. If the difference is near zero, it means no information inside the image is present, otherwise, if it is near one, it implies that the image holds the information.
- **Sample pair analysis** [13]**:** In this method, selected multi-sets form the state of a finite state machine as its imprecise change of state causes between these multi-sets based on LSB flipping.

**Table 8** Results of security analysis on different Stego image for different attacks

| Binarization technique | Payload (bpp) | Above stego threshold? | Secret message size in bytes (ignore for clean files) | Primary Sets | Chi-square | Sample pairs | RS analysis | Fusion (mean) |
|---|---|---|---|---|---|---|---|---|
| **Gaussian** | 1.53 | **False** | 231 | 0.079722 | Null | 0.015189 | 0.02437 | 0.03976 |
| **Adaptive** | 2.53 | **False** | 307 | 0.085355 | Null | 0.001038 | 0.07177 | 0.05272 |
| | 3.53 | **False** | 275 | 0.019145 | Null | 0.076371 | 0.04605 | 0.04718 |
| **Mean** | 1.50 | **False** | 298 | 0.106712 | Null | 0.030772 | 0.01552 | 0.05180 |
| **Adaptive** | 2.50 | **False** | 102 | 0.00646 | Null | 0.011855 | 0.02063 | 0.05896 |
| | 3.50 | **False** | 158 | 0.02966 | Null | 0.022464 | 0.20256 | 0.23464 |
| **Hard** | 1.45 | **False** | 301 | 0.098807 | Null | 0.041060 | 0.01627 | 0.05125 |
| | 2.45 | **False** | 343 | 0.115821 | Null | 0.040426 | 0.03404 | 0.01745 |
| | 3.45 | **True** | 1366 | 0.270663 | Null | 0.230694 | 0.02932 | 0.02715 |
| **Otsu** | 1.55 | **False** | 475 | 0.117331 | Null | 0.054026 | 0.07325 | 0.08153 |
| | 2.55 | **False** | 701 | 0.172017 | Null | 0.137317 | 0.05195 | 0.12043 |
| | 3.55 | **True** | 1244 | 0.29747 | Null | 0.22176 | 0.12173 | 0.21366 |

**Table 9** Probabilities of different Stego image to contain hidden information as calculated by SR-Net

| Binarization Technique | Payload (bpp) | Probability of an image to contain hidden information |
|---|---|---|
| Gaussian Adaptive | 1.53 | 0.00007161912 |
| | 2.53 | 0.000062491825 |
| | 3.53 | 0.000024740268 |
| Mean Adaptive | 1.50 | 0.000083623345 |
| | 2.50 | 0.00006367562 |
| | 3.50 | 0.00002220086 |
| Hard | 1.45 | 0.00007202594 |
| | 2.45 | 0.000054932243 |
| | 3.45 | 0.000029396779 |
| Otsu | 1.55 | 0.0000730547 |
| | 2.55 | 0.00006260316 |
| | 3.55 | 0.000029423087 |

StegExpose performs a fast fusion of all the above-mentioned techniques by only looking at the suspicious files. As the results are shown in Table 8, out of the total 12 stego images of 'Lena' (dimension 128 × 128), generated by different binarization threshold values, 10 lies below the stego-threshold. This implies that the proposed algorithm creates the stego images which are robust to steganalysis attacks and can assist external encroachment.

### 6.2 SR-net Steganalysis

We have also used a state-of-the-art neural network-based SR-Net (Steganalysis Residual Network) [7] Steganalysis tool. It is based on Deep Residual Network for Steganalysis of images. The architecture of the SR-Net is a concatenation of three segments: the front segment is responsible for extracting the noise residuals, the middle segment reduces the dimensionality of the feature maps, and the last segment is a standard fully connected layer followed by a softmax node. For the testing purpose, we have used a pretrained SR-Net model. The stego images along with their probabilities containing hidden information (calculated by the softmax function of the network) are recorded in Table 9. From the results, it can be visualized that all of the total 12 stego images of 'Lena' (dimension 128 × 128) have a very small value for the probability produced by SR-Net (in the range of $10^{-5}$). This concludes that the proposed algorithm creates stego images that are highly secure as they are undetectable by SR-Net.

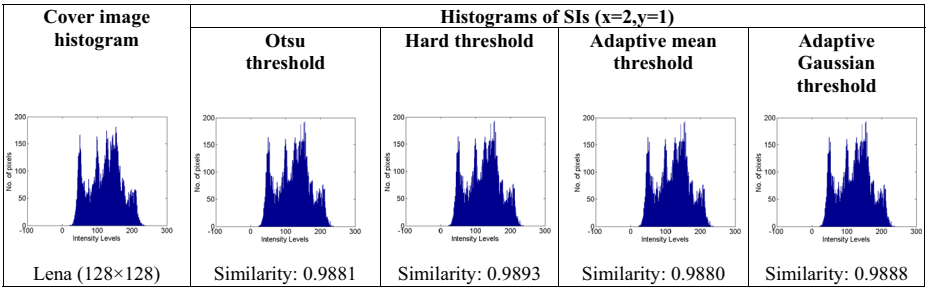



**Fig. 14** Depicts the global histograms and similarity scores of Lena CI and SIs

### 6.2.1 Histogram analysis

We further analyse the histograms. In this context, we have looked into the global histograms. We choose $x = 2$, $y = 1$, and plot the histograms of different SIs of Lena ($128 \times 128$) using different binarization techniques. The histogram similarity scores are 0.9881, 0.9893, 0.9880 and 0.9888 respectively for Otsu threshold, Hard threshold, Adaptive mean threshold and Adaptive Gaussian threshold. Figure 14 depicts the global histograms of Lena CI and SIs at different binarization methods used to achieve the edge images. The histogram similarity score of each case is nearly equal to 1, which proves the robustness of our method.

## 7 Conclusion

It is a fact that the human eye is susceptible to detect a sharp difference in an image pixel. This implies that the edges of an image can permit more distortion than the other areas of the cover image since the edge areas themselves to have a sharp change in pixel values than the surrounding areas. Therefore, in edge based Steganography, the maximum numbers of message bits are embedded in the edge pixels and as we traverse from the edge regions into the homogeneous regions of the cover image, we decrease the number of bits embedded to make it less perceptible for the human eye to detect the distortion. In this paper, we have proposed a new edge detection based Steganography method to achieve higher payload at the cost of minimum distortion in the stego images. Here edges of the cover images are computed using a *CNN with Deep Supervision* based edge detector. One of the notable advantages of the proposed edge detection technique is that it is less sensitive to the thresholding approaches applied during binarization which is evident from our results where it has been found that four different binarization approaches produce an almost equivalent outcome in terms of data hiding capability. PSNR obtained by the proposed method outperforms some existing techniques both in the spatial and edge-based Steganography methods, and at the same time maintains a reasonably well payload capacity. In our future work, the improvement of the results in terms of PSNR and Payload values will be considered. Besides, we will try methods other than the LSB method for embedding the secret data.

## References

1. Abdulla AA (2015) Exploiting similarities between secret and cover images for improved embedding efficiency and security in digital steganography. PhD dissertation, Dept. of applied computing, Buckingham Univ., Buckingham, UK. http://bear.buckingham.ac.uk/149/
2. Abdulla AA, Jassim SA, Sellahewa H (2013) Secure steganography technique based on Bitplane indexes. IEEE International Symposium on Multimedia 2013:287–291
3. Abdulla AA, Jassim SA, Sellahewa H (2013) Efficient high capacity steganography technique, Proceedings of SPIE - The International Society for Optical Engineering.
4. Akhtar N, Ahamad V, and Javed H (2017) A compressed LSB steganography method. IEEE. Ghaziabad, India.
5. Bhattacharyya D, Kim T (2011) Image data hiding technique using discrete Fourier transformation. Ubiquitous computing and multimedia applications: second international conference, UCMA 2011. Daejeon, Korea
6. Boehm B (2014) StegExpose - a tool for detecting LSB steganography. Multimedia, Cryptography and Security:1–11

7. Boroumand M, Chen M, Fridrich J (2018) Deep residual network for Steganalysis of digital images. IEEE Transactions on Information and security 14:1181–1193
8. Chandwadkar R, Dhole SP. (2013) Comparison of Edge Detection Techniques. 6th Annual Conference of IRAJ
9. Chen W-J, Chang C-C, Chang C-C (2010) High payload steganography mechanism using hybrid edge detector. Expert Syst Appl 37(4):3292–3301
10. Dhargupta S, Chakraborty A, Ghosal SK, Saha S, Sarkar R (2019) Fuzzy edge detection based steganography using modified Gaussian distribution. Multimed Tools Appl 78(4):17589–17606
11. Dube RR, Lalkot MA (2016) Improved edge based steganography scheme for GrayScale images in spatial domain. International Journal of Science and Research (IJSR) 5(6):1976–1978
12. Dumitrescu S, Wu X, Memon N (2002) On steganalysis of random LSB embedding in continuous-tone images. Proceedings of International Conference on Image Processing, IEEE 3:641–644
13. Dumitrescu S, Wu X, Wang Z (2002) Detection of LSB steganography via sample pair analysis. IEEE Trans Signal Process 51:1995–2007
14. El-Sayed MA, Estaitia YA, Khafagy MA (2013) Edge detection using convolutional neural network. Int J Adv Comput Sci Appl 4(10):11–17
15. Ghosal S, Mandal JK, Sarkar R (2018) High payload image steganography based on Laplacian of Gaussian (LoG) edge detector. Multimed Tools Appl 77(3–4):30403–30418
16. Ghosal S, Mukhopadhyay S, Hossain S, Sarkar R (2020) Application of Lah transform for security and privacy of data through information hiding in telecommunication. Transactions on Emerging Telecommunications Technologies
17. Ghosal SK, Chatterjee A, Sarkar R (2020) Image steganography based on kirsch edge detection. In: Multimedia Systems. Springer
18. Ghosal S, Mukhopadhyay S, Hossain S, Sarkar R (2021) Exploiting Laguerre transform in image steganography. Computers & Electrical Engineering 89:106964
19. Golijan M, Rui D, Fridrich J (2001) Detecting LSB steganography in color, and gray-scale images. Multimedia, IEEE 8:22–28
20. Gujjunoori S, Oruganti M (2019) Difference expansion based reversible data embedding and edge detection. Multimed Tools Appl 78:25889–25917
21. Han JKW, Hsueh-Ming (2020) Traditional method inspired deep neural network for edge detection. IEEE international conference on image processing (ICIP). Abu Dhabi, United Arab Emirates, United Arab Emirates.
22. Hosam O, Halima NB (2016) Adaptive block-based pixel value differencing steganography. Security and communication networks, Wiley 9(18):5036–5050
23. https://www.kaggle.com/vijaygiitk/multiclass-weather-dataset - Accessed on 20-Jan-2020
24. Islam S, Modi MR, Gupta P (2014) Edge-based image steganography. EURASIP J. on Info. Security , 8 (1). https://doi.org/10.1186/1687-417X-2014-8
25. Ismail K., El Bachir A and Taouil Y (2018) Image steganography based on edge detection algorithm. International conference on electronics, control, optimization and computer science (ICECOCS). Kenitra, Morocco.
26. Jung K, Yoo K (2015) High-capacity index based data hiding method. Multimed Tools Appl 74(6):2179–2193.
27. Kumar DS, Kiran R (2017) Data hiding using Fibonacci EDGE based steganography for cloud data. Int J Appl Eng Res 12(16):5565–5569
28. Lee CF, Chang C and Tsou P (2010) Data Hiding Scheme with High Embedding Capacity and Good Visual Quality Based on Edge Detection. 2010 Fourth International Conference on Genetic and Evolutionary Computing.
29. Ma X, Li B, Zhang Y, Yan M (2012) The Canny Edge Detection and Its Improvement. International Conference on Artificial Intelligence and Computational Intelligence. 50–58
30. Martin D, Fowlkes C, Tal D, Malik J (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. Proc. 8th Int'l Conf. Computer Vision 2:416–423
31. Mathura N, Mathur S, Mathur D (2016) A novel approach to improve Sobel edge detector. Procedia Computer Science 93:431–438
32. Mukherjee N, Paul G, Saha SK (2018) An efficient multi-bit steganography algorithm in spatial domain with two-layer security. Multimed Tools Appl 77(2):18451–18481
33. Mukhopadhyay S, Hossain S, Ghosal S, Sarkar R (2021) Secured image steganography based on Catalan transform. Multimed Tools Appl 80:14495–14520
34. Pak C, Kim J, Kwangil A, Kim C, Kim K (2019) A novel color image LSB steganography using improved 1D chaotic map. Multimed Tools Appl 79:1409–1425

35. Poma XS, Riba E, Sappa A (2020) Dense Extreme Inception Network: Towards a Robust CNN Model for Edge Detection. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)
36. Setiadi DRIM, Jumanto J (2018) An enhanced LSB-image steganography using the hybrid canny-Sobel edge detection. Cybernetics and information technologies 18(2):1314–4081
37. Shanthakumari R, Malliga S (2020) Dual layer security of data using LSB inversion image steganography with elliptic curve cryptography encryption algorithm. Multimed Tools Appl 79:3975–3991
38. Shin N (2000) One-Time Hash Steganography. In: Pfitzmann A. (eds) Information Hiding. IH 1999. Lecture notes in computer science, vol 1768. Springer, Berlin, Heidelberg https://doi.org/10.1007/10719724_2
39. Shrivakshan GT, Chandrasekar C (2012) A comparison of various edge detection techniques used in image processing. IJCSI International Journal of Computer Science Issues 9(5):269–276
40. Stanley CA (2005) Pairs of values and the Chi -Squared Attack. CiteSteer 1–45
41. Tseng H-W, Leng H-S (2014) High-payload block-based data hiding scheme using hybrid edge detector with minimal distortion. IET Image Process 8(11):647–654
42. Tu SX and Zhuowen (2015) Holistically-nested edge detection. IEEE international conference on computer vision (ICCV). Santiago, Chile
43. Wang X (2007) Laplacian operator-based edge detectors. IEEE Trans Pattern Anal Mach Intell 29(5):886–900
44. Wang Z, Yin Z, Zhang X (2017) Distortion function for JPEG steganography based on image texture and correlation in DCT domain. IETE Tech Rev 35(4):1–8
45. Weber AG (1997) USC-SIPI Image Database: Version 5, Original release: October 1997, Signal and image processing institute, University of Southern California, Department of Electrical Engineering.
46. Xue C, Zhang J, Xing J, Lei Y and Sun Y (2019) Research on edge detection operator of a convolutional neural network. IEEE 8th joint international information technology and artificial intelligence conference (ITAIC). Chongqing, China.
47. Yang L, Wu X , Zhao D , Li H, Zhai J (2011) An improved Prewitt algorithm for edge detection based on noised image. 2011 4th international congress on image and signal processing. Shanghai, China
48. Younus ZS, Hussain MK (2019) Image steganography using exploiting modification direction for compressed encrypted data. Journal of King Saud University –Computer and Information Sciences
49. Zhang Z, Ma S, Liu H, Gong Y (2009) An edge detection approach based on directional wavelet transform. Computers & Mathematics with Applications, Elsevier 57(8):1265–1271
50. Zisserman KS and Andrew (2014) Very deep convolutional networks for large-scale image recognition. Computer Vision and Pattern Recognition.
51. Zou Y, Zhang G, Liu L (2019) Research on image steganography analysis based on deep learning. J Vis Commun Image R 60:266–275