Richer convolutional features for robust image steganography

Erfan Darzidehkalani[1,2], Mohsen Gharib-Naseri[2], Hamid Soltanian-Zadeh[2,3]

[1]Machine Learning Lab, Data Science Center in Health (DASH),University Medical Center Groningen, University of Groningen, Hanzeplein 1, the Netherlands
[2]Control and Intelligence Processing Center of Excellence (CIPCE), School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran
[3]Medical Image Analysis Laboratory, Henry Ford Health System, Detroit, MI, USA
e.darzidehkalani@umcg.nl ,mohsen.ghnaseri@ut.ac.ir, hszadeh@ut.ac.ir

***Abstract*—In this paper, we have introduced a new method for enhancing the payload capacity and the security of the coded message in an encoded image. This method works based on a deep neural network with the architecture of Resnet101 for edge detection. Edges are found using the most significant bits of the cover image and then expanded using the dilation operator. Also, there is no need to send additional data or a matrix to decode the image, improving steganography efficiency. This technique has improved the qualitative and quantitative factors of the encoded message, increasing the similarity of the input image and encoded one. The encoded message's imperceptibility has improved based on the PSNR index, increasing about 1 dB. Besides, 50-11% less MSE between input and stego image has been achieved, with the highest increase in 1,096 bits of the message, which shows a better quality. In addition, more meaningful pixels are found as edges which indicates a better payload capacity with better qualitative imperceptibility.**

***Index Terms*—Steganography, Edge detection, Residual networks**

I. Introduction

NOwadays, owing to the rapid expansion of computer systems, transmitting data with networks is getting more widespread. Individuals use the internet to send data, contributing to the increment of the risk of data theft during data transmission. Hiding information has been a crucial issue in information exchange.

There are two requirements that should be considered in hiding information. First and foremost, the number of bits of the hidden message embedded in every pixel of an image, known as the payload capacity. Higher payload capacities indicate a higher amount of information allowed to be embedded into the cover image. The message within a cover image will affect the cover image's pixel values. Notably, as the number of bits of the message increases, the ability to hide the message or imperceptibility decreases. Another characteristic one should care about is the quality of the image after the message is embedded into it. Higher quality stego images are more similar to the cover image and thus, have a higher imperceptibility.

There are several ways to measure the difference between a cover and a stego image. The more common one is the peak signal-to-noise ratio or PSNR. High PSNR values indicate the high quality of stego images and little visual difference between the cover and the stego image. Some of the most common approaches to protect data in image processing are cryptography, steganography, and watermarking.

Cryptography is converting data from a readable state to nonsense ones. Only entities with access to the encryption key can reverse the process and return the coded data to its initial understandable situation. For example, images can be split into two different sub-images, components. Each component has a pair of pixels for every pixel in the original images. In some cases, the data protection method is known as data hiding, a data security technique by covering data in particular media designated as containers or covers media. Two of the most popular approaches for hiding data are steganography and watermarking, which have similar procedures.

The watermarking approach plans to preserve the media cover, whereas the steganography technique aims to preserve and guard messages embedded in cover media. Generally, steganography aims to embed a message in an image or other data types so that unintended third parties do not notice the image or data containing confidential data. Both digital watermarking and steganography utilize steganographic procedures to embed information covertly in noisy signals. Nonetheless, steganography aims for imperceptibility to human senses, and digital watermarking tries to control robustness as the top priority because messages are a crucial part of steganography. Incorrect message extraction can give rise to the disruption of the meaning of messages. Thus, coded messages must be extracted in the best way possible until the message reaches the recipient correctly.

In this paper, a practical and straightforward image-steganography method is proposed. This method is based on identifying edge location on a cover image using deep neural networks and incorporating the XOR coding function. Numerous studies have developed on the payload capacity and security while improving the imperceptibility concept. The

All of the authors are with the Control and Intelligence Processing Center of Excellence (CIPCE), School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran (e-mail: erfandarzi@ut.ac.ir).

least-significant-bit (LSB) technique is a common approach for steganographic algorithms in the spatial domain. It is simple but has excellent imperceptibility and can also enhance the payload capacity. This method embeds each bit of a message in LSBs[1] of pixels of an image. Edges of an image have a much higher tolerance than other parts. This is why noises are usually close to edge areas; hence, embedding messages in edge areas can minimize the decrement of stego images' imperceptibility.

In this article, we expanded the edge areas by utilizing a new method to increase payload capacity and imperceptibility. We used the 3-bit MSB bits of the original image where 5-LSB-bits are set to zero. Hence, the edge areas generated using the 3 MSB bits allow us to continue embedding messages up to 5 bits.

II. Related research

Chen et al. [2] introduced a high embedding capacity steganography scheme with a hybrid edge detector. They used a fuzzy edge operator to generate an edge image from a grayscale image. They then segmented the edge image into a collection of pieces inside an n-pixels block; for each block, the first pixel stores the edge data of the rest n-1 pixels.

Bai et al., [3] organized a steganography technique employing the edge detection procedure, which provides a more substantial capacity. Traditionally, edge detection utilizes all the pixel values (8-bits); but, they employed the three most significant bits on the cover image, where the five least significant bits were set to zero. Then, the edge zones are formed by the three most significant bits. In the research, they tested three edge-detectors, such as Sobel, Canny, and Fuzzy. Through the method, all edge detectors generate more edge regions. This method was declared to be more reliable than the two earlier studies proposed by [2], although testing outcomes hold the increase in the payload capacity.

Tseng and Leng [4] proposed a block-based design that stretched to attain a minor distortion. Their scheme utilized one parameter x for the x-LSB replacement in non-edge pixels. The second element in each bracket denotes the number of secret bits hidden in edge pixels. In order to obtain the least amount of deformity and distortion, one of the four cases is determined by measuring the least mean square error for each 4 x 4 pixels block.

Gaurav et al. [5] introduced an edge detection approach, using Canny edge detector. Somewhat modified from the method proposed by [3], the edge detection method uses the 4-bits MSBs. By employing edge detection on the 4-bits MSBs, the detected edges in the steganography image will be the same as the initial cover image. In short, the edge area in the message insertion and extraction processes will be similar, and there is no need to include the edge locations in the message. They also improved the payload capacity by using the dilation operator. They also XORed the message to improve security.

Another article, [6] combined the LSB procedure and the Exploiting Modification Direction (EMD) method intending to improve the capacity and the quality of Steganography imagery. Another article [7] introduced a mixture of LSB replacement methods and modulo functions. It is just a modulo function utilized to compress messages. Before the message is embedded, it is broken into two sections; additionally, the components are inserted into the cover image using a modulo function. This study's results show increased capacity and message security to maintain the imperceptibility quality of Steganography images.

Setaidi et al. [8] proposed a method quite similar to [5], for example, the LSB method with a dilation procedure on the edge area. The difference in the edge detection method is made on both the 4-bits of MSBs value and also on the entire bits of the pixel value. Hence, since the message embedding approach can alter the cover image's pixels value, the results of edge detection of the cover image before and after message insertion will be different. The image edge area must be stored to have an accurate message extraction and decoding. The XOR operator is done on the MSB bits of the cover image to embed messages.

Data hiding with neural networks: With the advent of neural networks, many researchers utilized them in steganography [9]. Until recently, prior work has typically used them for one stage of a more extensive pipeline, such as determining watermarking strength per image region.

Most of the works use neural networks as a part of a more complex procedure. For example, Jin et al. [10] used neural networks to find image parts that are more suitable for watermarking. Other examples include using neural networks as the encoder [11] or the decoder [12] part within a larger framework.

There are also other usages of neural networks in information hiding. Abadi and Anderson [13] used adversarial networks to protect communications; Uchida et al. [14] hid information in the weights of neural networks and retrieved them successfully; Fang et al. [15] designed a steganography system that enables secret text communication using LSTM networks. Current top classical machine learning approaches are based on Ensemble Classifiers, such as SVM based models, [16], basic Rich Models [17], [18] or Selection-Channel based Rich Models [19], [20], [20].

Deep neural networks were first applied to steganalysis [21] with auto-encoders. Research works focusing on the application of CNNs in steganalysis are [22] [23], their results were promising but still far from current methods. Xu reached an excellent accuracy in steganography with multiple CNNs [24] [25], which can be regarded as a baseline for the performance of CNNs. Published works on applying convolutional neural networks (CNNs) to image steganalysis mainly focus on detecting steganography embedding in the original spatial domain.

All of those researches hid the secret message in spatial domain using CNNs [22] [23]. However, since images are more prevalent in JPEG format, studies began to apply CNNs in the JPEG domain. For example, Zeng et al.[26] used a two-stage CNN with convolutional and pooling layers. This architecture of CNN, however, is not limited to the JPEG domain and was already used in the spatial domain as well [22] [25].

Other works primarily include a data preparation stage and

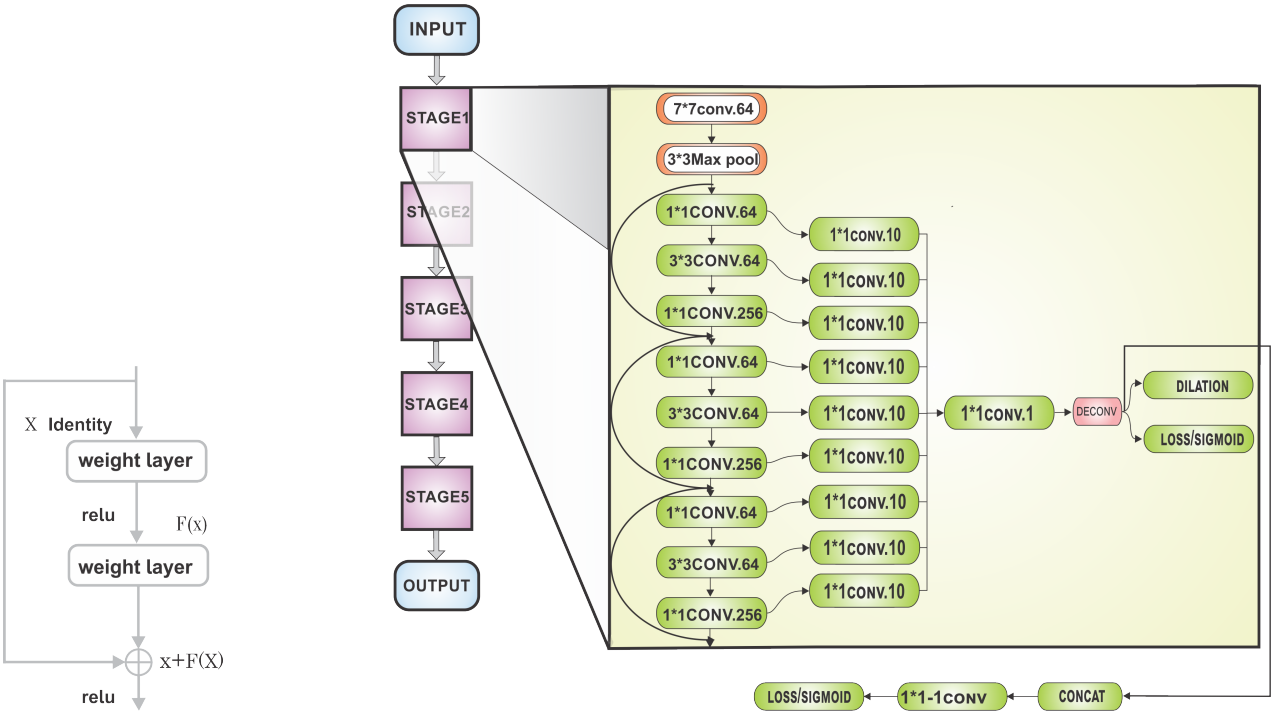

Fig. 1: Above images show a scheme of our network. We use Resnet-101 as the backbone net,then for each $conv$ layer, we feed their outputs to $1x1$ $conv$ layers with 10 filters, then add them together. Since the edge maps should be represented in one channel,we apply a $conv$ layer with one filter and a $deconv$ layer to get the final image in one channel

training model based on a large amount of data, which is proven to be effective in steganalysis [27] [26]. In [28], authors tried to integrate the JPEG compression method in the design of the CNN model. The idea of deep Residual networks [29] influenced the field of steganography. Xu-Net-Jpeg [30] is a neural network based on ResNet and reported an improved accuracy. ResDet was also another network based on Resnet with similar results[31]. The above results are promising; however, deep learning made a breakthrough in many other fields of research [32], while in the field of steganalysis, deep learning is still unable to imporove the performance of the machine learning methods more than 10%,

III. Proposed scheme

Inspired by many other research pieces, deep learning is utilized to perform edge detection. In the first place, a simple network is built to create side outputs of middle layers using Resnet-101. We can see that the information obtained by different convolution layers gradually becomes coarser. It can be seen that output images of initial stages are smoother compared to further ones. Later, we use the edge information provided by different convolution layers to combine complementary information from all layers of CNNs and, thus, obtain accurate representations for objects parts in different scales.
After edge detection, we divide our image into the edge and non-edge areas. We then insert our message in those two regions according to the message insertion approach discussed later. Successively, rather than utilizing plain message codes inside the image, we will apply an XOR operator to intensify coding security.
Previous studies suggested many methods of edge detection and embedding message on edges of images. Based on previous discussions, edges are more suitable for embedding secret messages since they are more tolerant of changes.
In this paper, a deep neural network has been utilized to extract the edges of the image. Further, like the earlier papers, the LSB technique for embedding messages has been employed. The XOR method has been used to enhance the security of hidden messages as the main goal is to improve the payload capacity and imperceptibility quality. The steps proposed for the embedding phase are:

A. Pre processing

Reading the cover image as a matrix M1, then the copied matrix is stored in another matrix M2. Converting the input image to an 8-bit integer value to binary, then setting the first five least significant bits (LSBs) of M2 to zero; the reason why we have chosen to use 5 LSBs is that based on many papers, the edge found using 3 MSBs is reasonably similar to the edge found in the original picture. Hence, zeroing in the first 5 LSBs, we will be able to enhance the payload capacity. At the same time, the image's appearances in human eyes have not noticeably changed.
Further, the three remaining bits of the message will be coded in non-edge bits: the first LSB non-edge pixel will be zeroed, and it will be supplanted by the XOR of the first LSB of the message and the MSB of the pixel. And the first bit of the second non-edge pixel will be supplanted by the 2nd

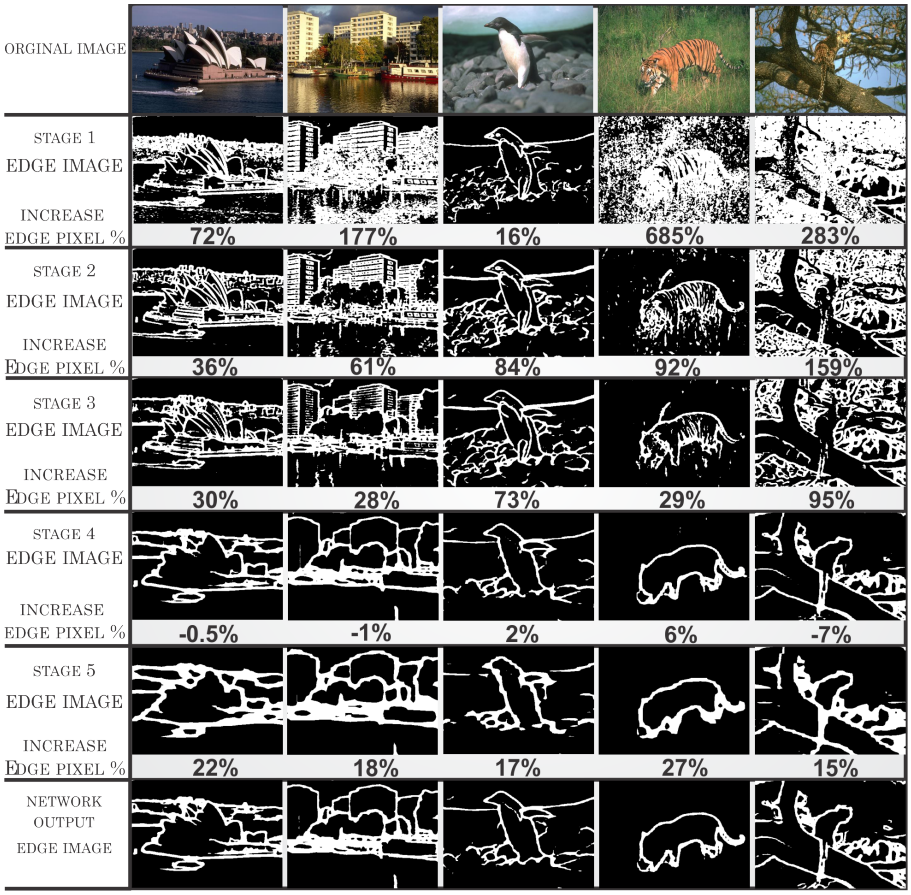

Fig. 2: Five sample images, and outputs of different ResNet-101 stages are shown here. As you can see, in early stages, more edges are found than in final stages.

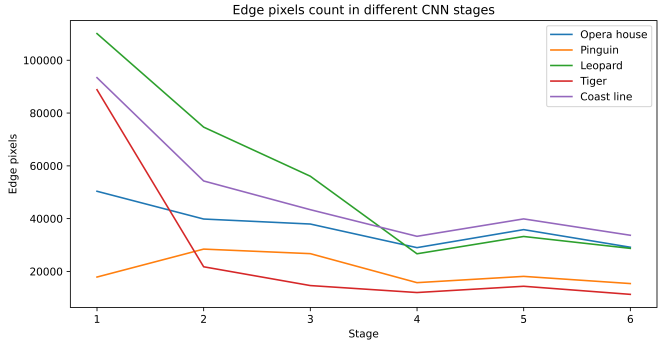

Fig. 3: This graph shows the number of pixels found in each stages of the Resnet101. As it is obvious, the higher stages discover a lower amount of edge pixels.

LSB of the first character of the message. In fact, for the 3 LSBs of a character, 3 LSBs of the first three non-edge pixels will be utilized. If the first LSBs of all the non-edged pixels get occupied, the second LSBs of non-edged pixels will be utilized. The reason behind this procedure lies behind the fact that through this process, just the lowest bits of almost all the non-edged pixels will be used, which will, in turn, decrease the noticeable-by-human-eye changes; as a result, discovering the hidden message inside the picture will become increasingly arduous.

B. Edge detection

1) Network architecture: Resnet 101 [29] consists of 5 different stages, each of them having a number of *conv* layers. Table 1 shows Resnet-101[29] architecture. As the stages increase, Resnet-network the layers take a more holistic view of the image, capturing information in a wider receptive field. We have also modified the Resnet-101 network. Our modifications are as follows:

- Each *conv* layer is connected to a layer with kernel size 1x1 and channel depth 10
- We added the outputs of each stage together
- A dilation operator block was added after the output of each stage for enhancing the edges found by each stage
- We accumulated the information from different layers, concatenating them.
- We added more loss functions to the output of each stage
- We made a feature map using a cross-entropy loss function

In this network, all the information from different image scales is used to obtain a general view of data. Different representations in different scales could be accumulated and trained via back-propagation. As edge detection requires multilevel representation of data, taking large-scale information from the

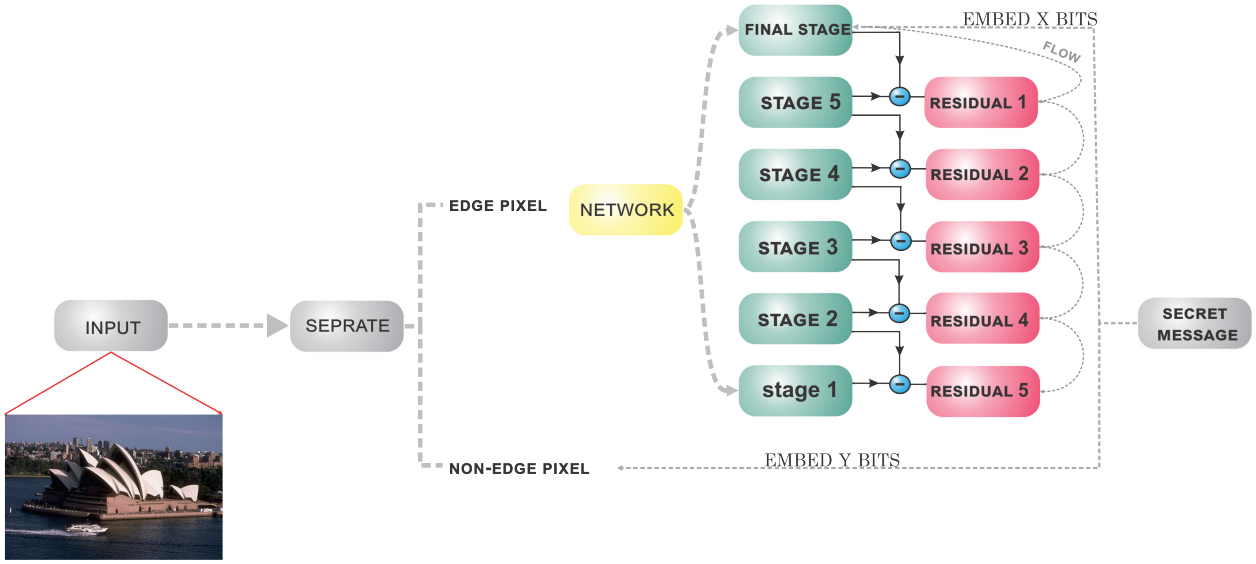

Fig. 4: A schema of the six stages of Resnet101, for edge detection. The network consists of 6 stages, and in each stage, different edges are extracted. The input of the network is an image that its' first 5 LSBs have become zeroed. Each stage has an output matrix showing edges.

last layers and intricate edge details from the first layers is necessary to find edges and further code the image.

2) Loss function: Similar to many other types of research on edge detection, We use the BSDS500[33]. Each image in this dataset is annotated by 4 to 6 different persons. To utilize the labeling information from all annotators, we summed over all the ground truth images. We then normalized the resulting values from 0 to 1 to find general ground truth. If a pixel value is 0, this means that none of the annotators labeled that pixel as an edge. Conversely, if a pixel value is 1, it means that it was labeled as edge by all annotators. Thus, as η is closer to 1, we can take it as an edge with a higher confidence level. Similar to HED [34], we assumed a threshold of 0.5 for edge parts, meaning that pixels with η below 0.5 are considered non-edge and, thus, negative samples. The loss of each pixel is computed as follows:

$$l(X_i; W) = \begin{cases} \alpha \, log(1 - P(X_i; W)), & if \ y_i < \eta \\ \beta \, log P(X_i; W), & otherwise \end{cases} \quad (1)$$

in which

$$\alpha = \lambda . \frac{|Y^+|}{|Y^-| + |Y^+|} \quad (2)$$

$$\beta = \frac{|Y^-|}{|Y^-| + |Y^+|} \quad (3)$$

In which Y^+ and Y^- denote that positive sample set and negative sample set, respectively. The hyper-parameter λ was used to balance the number of the activation value (CNN feature vector), and ground truth edge probability at pixel i are presented by X_i and y_i, respectively. P(X) is the standard $sigmoid$ function, and W denotes all the parameters learned in our architecture. Therefore, our improved loss function can be formulated as

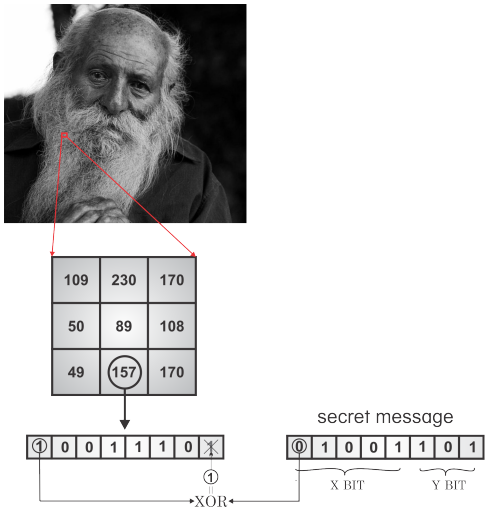

Fig. 5: The process of encoding the message is illustrated here. First, edge pixels are found. Then, for each pixel, we find its 8-bit value. Then we find the ASCII code of the character we want to embed in that pixel. Assuming that the selected pixel with a value of 157 is an edge pixel, we XORed the MSB of the secret message with the MSB of that pixel and embedded the resulting value into the LSB of that pixel

$$L(W) = \sum_{i=1}^{|I|} \Big(\sum_{k=1}^{|K|} l(X_i^{(k)}; W) + l(X_i^{fuse}; W) \Big) \quad (4)$$

Where $X_i^{(k)}$ is the activation value from stage k while X_i^{fuse} is the final output from the last layer, in which complementary information from all layers are merged in one general representation. Moreover, $|I|$ is the number of pixels in Image I,

and K is the number of stages in residual architecture.

C. Embedding

1) According to the discovered edges in the last step, both edge, and non-edge pixels will be flagged on the input image.
2) Changing the massage will be coded to an ASCII code and generating a binary code based on it. Successively creating a matrix by utilizing the first five most significant bits of each character of the message and saving the remaining 3 bits of each character in another matrix.;
3) In the algorithm's embedding part, for each edge pixel, the MSB of the pixel will be XORed with all the five MSBs of the massage. The result will be stored in the first least significant bit. The same approach will be made for the non-edge pixels' MSBs and the matrix of the three least significant of the soon-will-be-coded characters of the message; this will be done for all the pixels of the input image.;

Fig. 4 in page 6 shows a schema of embedding method. An input image is fed to the network; the network's input is an image that its' first 5 LSBs have been set to zero. Each stage has an output matrix showing edges. Its edge and non-edge pixels are separated, then we get the final output of our network. This output is a binary mask image in which the final edges are found. We also have other edge maps from previous stages. We can fuse the information from those steps by calculating the residuals between every consecutive stage, i.e., finding the edge pixels that exist in one stage but are removed in the next stage.

After obtaining the residual masks, we first put our code in the mask obtained from the final stage, and then we put the remaining parts of the code in the residual masks. This is the encoding procedure for X bits. The remaining Y bits of the message are embedded into the non-edge pixels.

In the lower stages, almost all edges are extracted regardless of whether they are actual edges or fake edges because of noise. In the second stage, due to the use of some pulling and convolution layers, a considerable amount of noise has been removed, which, in turn, gave us fewer edges pixels in which their being-edge probability is higher than the first stage.

Similarly, edges found in higher stages, like the final stage, gave us edges that we are confident are actual edges. Consecutively, we create a matrix based on the output of the final stage; then, we compare it to the output of the 5th stage; the additional edges, Residual 1, which exist in the 5th stage's output but not in the final's, will be flagged in the matrix mentioned before.

Likewise, residual 2 to 5 will be added, respectively. We consider a higher priority for the output of the higher stages: the final layer output has the highest priority, and residual5 or stage1 has the lowest priority. Based on this priority, we encode the image: pixels in the final layer's output will be used sooner than the edges in Residual1. This priority contributes to two significant results; the first result is that no one knows about such a priority, and even if they understand that the image has been coded and they concatenate the codes, a meaningless message will be discovered since they do not know how to and based on what priority they should decode the message.

The second result of this priority is that by just learning about the existence of such priority, we do not need to send receivers a matrix that shows how pixels' message should be concatenated, and this improves the efficiency of the method by decreasing the amount of data that should be sent to the receiver.

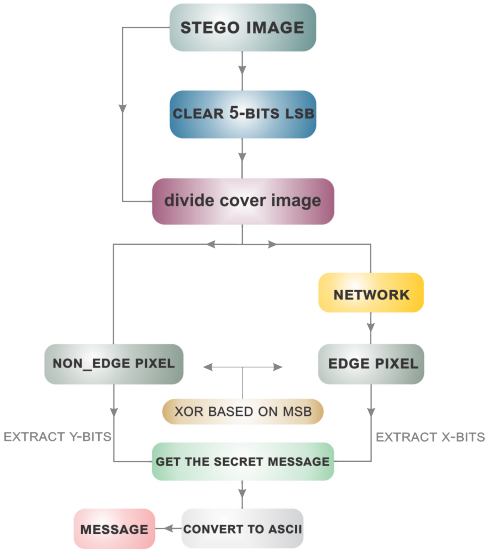

Fig. 6: The diagram of the decoding process. The first 5 bits of the stego image are cleared and divided into edge and non-edge pixels. LSB and MSB for each pixel are XORed to find the hidden binary values. After doing this one time for every pixel, we then XOR the second LSB with MSB and go forth. After finding all bits of the message, we can concatenate them to find the 8-bit binary value representing the message, then it is converted to a decimal value, and the corresponding ASCII code and character is found from it

D. decoding

The diagram of decoding is shown in Fig 6 In the first place, we create a duplicate image of the coded image and assign its first five LSBs of each pixel zero. Then, the Resnet-101 network, having been trained, will be utilized for edge detection. Five stages have been chosen in the network to give some outputs that will be dilated; in fact, the network has six output matrices. One of them is the final output of the network, with the least edge pixels. Further, the other stages' edge pixels will generate five other matrices. Accordingly, a search from left to right and up to down of matrix will be done so as to find and label edge pixels; hence, so the locations of edge pixels of the six matrices will be stored, the priority is on the final matrix, the final stage on the Resnet-101 network. Then, we duplicate the final matrix, which is called the "Label matrix." Then, the label matrix will be compared to the five other matrices, and their additional edge pixels will be added to the label matrix, and the result is called the modified matric.

The pixels in the modified matrix with zero values will be called non-edge pixels containing one LSB of the coded message. Using the modified matrix, we can find the pixels containing the first 5 MSBs of the coded message. Thus, we calculate the XOR of the first five LSBs of each pixel and the MSB of the pixel; through this process, the first 5 MSBs of the message will be decoded. Likewise, the LSB of each non-edge pixel will be XORed with the MSB of the pixel, and the result will be one LSB of the coded message.

The coded message will be found by concatenating the 5 MSBs found from edge pixels and 3 LSBs of the non-edge pixels. These found 8 bits are the ASCII codes of characters of the message, which can be easily converted to a readable-by-human message.

IV. Experiment

The dataset used in this paper is BSDS500, and its images are RGB images; however, we changed it to grayscale images with a size of 321*481 and 8 bits (256 discrete levels.) This dataset contains 200 training and 100 validation and 200 test images utilized to learn the network. The secret message that was embedded had 10000 bits; in fact, it is a stream of data that results from changing characters of the message to ASCII code based on English letters' ASCII table.

The function used for this procedure is "unit8" in the Python programming language. Edge detection has been done using the Pytorch library for the implementation of Resnet101. In convolution layers, the initializing coefficients of the network are based on the Gaussian distribution with an average of zero; also, the convolution layers' biases are zero. For other layers, the pre-trained coefficients of Resnet101 have been used. These coefficients are the result of 40000 iterations. We implement our model using the ResNet-101 as the backbone network. The GPU used for this project is an NVIDIA TITAN provided by Google Colab. For edge detection, the probability map has been used to assign a value from zero to one, the probability of a pixel being an edge pixel. Based on this map, a threshold has been used to divide pixels to edge and non-edge pixels. For the edge-detection part, a 3*3 kernel dilation has been done to enhance edge areas.

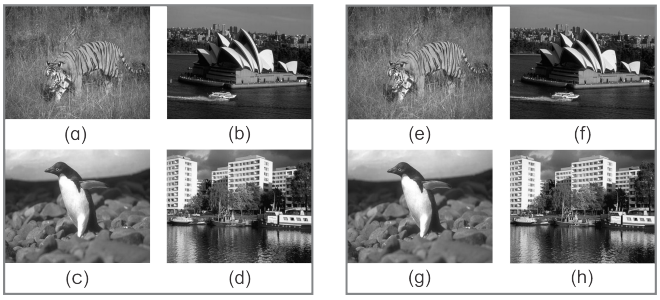

Fig. 7: The differences of the coded images. The left (a (Tiger) , b (Boat), c (Penguin), d (apartments)) pictures show original images that have not decoded yet. The right images (E (Tiger) , F (Boat), G (Penguin), H (apartments) show coded images. The changes in the images are not noticeable.

V. Results

The subjectivity of visual quality of a digital image cannot be ignored, saying that one method presents a more excellent quality image that could vary from person to person; as a result, it is inevitable to establish quantitative/empirical and qualitative measurements to differentiate the effects of adding a secretly added message on the image quality. Having such factors, we can systematically compare a particular algorithm of a coding message with other methods. There are three quantitative metrics under examination: PSNR (peak signal-to-noise ratio), MSE (Mean Squared Error), and SSIM (structural similarity index). This ability to show that a technique or procedure can produce an image containing a coded message with the minimum amount of changes is tantamount to the resemblance of coded Image and original Image, which will help us decide which procedure is more accurately reliable.

The MSE is the cumulative squared of the changes between the coded and the original image for each pixel. While I(x,y) is the original image, I'(x,y) is the coded version, and M, N are the dimensions of original and coded images. A lower value for MSE means a higher resemblance of a coded image and the original one.

$$MSE = \frac{1}{M * N} \sum_{M,N} \big(I_o(m,n) - I_c(m,n)\big)^2 \tag{5}$$

Where M and N are the numbers of rows and columns in the original image.
I_o represents the original image, and I_c represents the coded image. PSNR is an analysis of the ratio between the maximum possible value (power) of a signal and the power of distorting noise, which is a coded message, that affects the quality of an image. PSNR is usually denoted in a logarithmic decibel scale, as images have an extended dynamic range (proportion of the highest and lowest amounts of a possibly changeable quantity).

$$PSNR = 10 * \log_{10} \frac{R^2}{MSE} \tag{6}$$

In which R is the maximum fluctuation in the input image. The Structural Similarity Index (SSIM) is a mathematical factor that quantifies image quality depravity caused by processing, such as coding a message inside an image or data compression or by losses in data transmission. It is a full reference factor operating using two images, a reference and a processed or coded image. SSIM is well-known in video production but has robust applications for steganography. For images I_o and I_c, the SSIM is calculated as follows:

$$SSIM(I_o, I_c) = \frac{(2\mu_{I_o}\mu_{I_c} + C_1) + (2\sigma_{I_o y} + C_2)}{(\mu_{I_o}^2 + \mu_{I_c}^2 + C_1)(\sigma_{I_o}^2 + \sigma_{I_c}^2 + C_2)} \tag{7}$$

In which μ_{I_o} and μ_{I_c} are the averages, and $\sigma_{I_o}^2$ and $\sigma_{I_c}^2$ are the variances of I_o and I_c, respectively. The covariance of I_o and I_c is denoted as σ_{I_c,I_o}. We also have $C_1 = (k_1 L)^2$, and C_2=$(k_2 L)^2$. L is the dynamic range of the pixel-values (typically L=2). For k_1 and k_2 we have: k_1 = 0.01 , and k_2 = 0.03 .

The inverse relation of MSE and PSNR declares that a high value of PSNR is equivalent to a low amount of MSE.

TABLE I: Comparison of PSNR (dB) in various message lengths, among different methods

Payload of images	The method in [3]	The method in [8]	The method in [5]	The method in [35]	Proposed method
1,024	66.2163	69.2603	69.2797	67.8120	**70.0102**
4,096	63.1718	66.2462	66.2631	64.7703	**66.7539**
8,192	60.1287	63.2481	63.1555	61.8675	**63.8757**
16,384	overflow	60.3134	60.2261	overflow	**60.8302**

TABLE II: Comparison of MSE results in various message lengths, among different methods

Payload of images	The method in [3]	The method in [8]	The method in [5]	The method in [35]	Proposed method
1,024	0.0155	0.0077	0.0077	0.0108	**0.0032**
4,096	0.0313	0.0154	0.0154	0.0217	**0.0137**
8,192	0.0631	0.0308	0.0315	0.0423	**0.0266**
16,384	overflow	0.0605	0.0618	overflow	**0.0537**

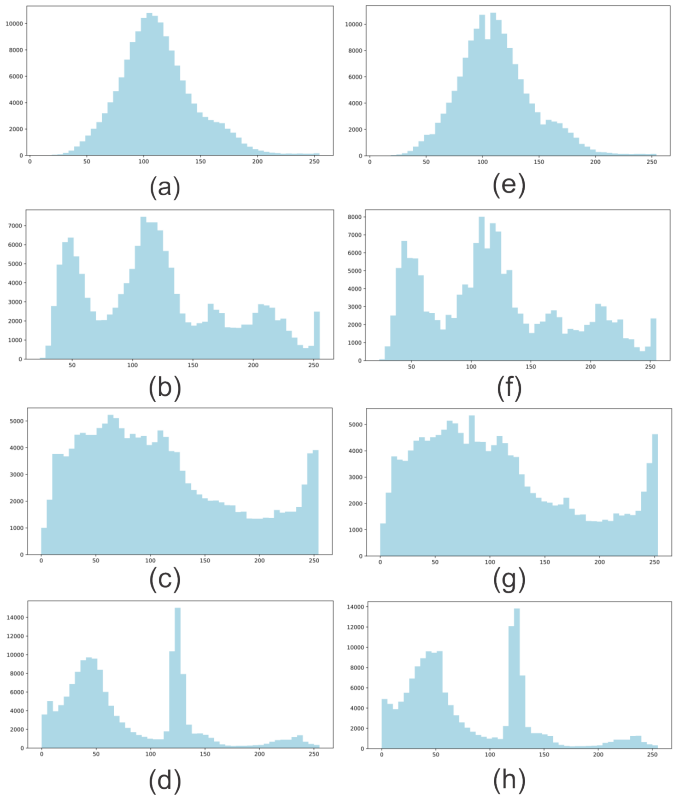

Fig. 8: Histograms of input images and the corresponding coded images. The left column is the histograms of the original images, and the right column is the histograms of coded images.

Accordingly, a higher value of PSNR is desirable, because it means that the ratio of changes due to adding a message and changing the original image is increasingly lower, which will, in turn, dramatically decline the possibility of detecting codes image.

Our results indicate that the PSNR value has been improved. Thus it is harder for the human eye to detect the difference between the initial images and coded ones. The payload capacity did not noticeably enhance since our goal was not to solely increase the number of edges. However, the edge detection proposed in our method improves the reliability of our procedure by spotting semantic and meaningful edges, not the made-by-noise ones.

TABLE III: Experiment results on sample images

IMAGE	MSE	PSNR	SSIM
OPERA	0.0130	65.4338 dB	0.9999
CITY	0.0188	65.8624 dB	0.9999
TIGER	0.0113	66.5647 dB	0.9999
PENGUIN	0.0193	66.8303 dB	0.9999

The comparison of the results of the proposed method with some of the methods proposed earlier was carried out by [3], [5], [35] , and [35]. Testing is done based on the methods proposed in the mentioned articles. The results are in the tables I, and II. These tables show the final values for two metrics, MSE and PSNR. The table indicates better performance with a small margin compared to the existing methods. PSNR has been increased in all the tried payload capacities. Besides, higher PSNR indicates better imperceptibility while preserving the image's capacity to carry messages of various sizes.

The output factors of our method, both qualitative and quantitative, have been improved compared to the previous studies. Final PSNR is over 60 dB, which is better in comparison to the works of [5], [3], and [8]. By visual inspection and quantitative evaluation, it is harder to distinguish the initial and coded images in our paper than previous methods.
In Table III, our algorithm was applied on four sample images, and the values of MSE, PSNR, and SSIM were calculated. The sample message length was 4,096 bits. Our other contribution was that our edge-detection network made us capable of prioritizing the edge pixels based on our level of confidence about them. This gives us three advantages.

First and foremost, such a priority will help us avoid choosing noisy pixels as edges, and this is inevitably crucial since if we change the values of the noisy pixels by embedding characters from secret messages, the change will be easily detectable by the human eye. Noise edges are not reliable spots in which we embed codes. As a result, such an approach contributes to the increment of message security.

Second, this priority enhances message security because

even if a person or a system detects a code inside an image, they will have no clue with what order the message has been embedded in pixels' bits. Even if they concatenate XOR of the first 5 LSBs of edge pixels and 3 LSBs of non-edge pixels, a vague message will be decoded, having no meaning.

Third, in other methods, the indexes of secret codes should be sent to receivers, but this method does not; thus, in this term, our method is more efficient than earlier ones. Noticeably, the qualitative factors of coded images, such as histogram resemblance, have been increased, based on Fig.8 the left column portrays the histograms of four input images and the right column does the same task for the coded images of the four images.

As it is evident from Fig. 8 in page 9, the histograms in the left column are extremely close to their corresponding histogram in the right column. This similarity is tantamount to the embedded code's security increment since the difference between the original and coded images is slightly noticeable. From this point of view, our method proposes a higher level of steganography based on edge detection.

VI. Conclusion

Mid-layers of Resnet101 represent a better understanding of the image since they can give us more information about the image. Our edge-detection method is fast and accurate, which helps us acquire more beneficial details on the initial image, improving the message's security. The encoded images are remarkably close to the original ones. Images extracted from the middle layers of ResNet are different representations of the edge image, and all of them were used since each edge image has some unique properties that others do not, so it will help us find more edges.

Accordingly, the payload capacity has been meaningfully enhanced since we prioritized edges for encoding messages based on our confidence that which pixel is more likely to be a semantic edge, giving rise to the final quality increment imperceptibly of the encoded images. No additional data is needed to be sent, unlike other methods, and due to this reason, the efficiency of our method is higher than similar papers since less data is required to be sent to receivers. The XOR operation also makes it harder for third parties to decode our message.

References

[1] Sri Shakthi Institute of Engineering and Technology, Institute of Electrical and Electronics Engineers. Madras Section, Institute of Electrical and Electronics Engineers, and Council of Scientific & Industrial Research (India), *2014 International Conference on Computer Communication and Informatics : January 03-05, 2014, Coimbatore, India.*

[2] W. J. Chen, C. C. Chang, and T. H. N. Le, "High payload steganography mechanism using hybrid edge detector," *Expert Systems with Applications*, vol. 37, no. 4, pp. 3292–3301, 4 2010.

[3] J. Bai, C. C. Chang, T. S. Nguyen, C. Zhu, and Y. Liu, "A high payload steganographic algorithm based on edge detection," *Displays*, vol. 46, pp. 42–51, 1 2017.

[4] H. W. Tseng and H. S. Leng, "High-payload block-based data hiding scheme using hybrid edge detector with minimal distortion," *IET Image Processing*, vol. 8, no. 11, pp. 647–654, 11 2014.

[5] K. Gaurav and U. Ghanekar, "Image steganography based on Canny edge detection, dilation operator and hybrid coding," *Journal of Information Security and Applications*, vol. 41, pp. 41–51, 8 2018.

[6] H. M. Sun, C. Y. Weng, and S. J. Wang, "A scheme of modulo-based capacity-improvement upon EMD systems," in *IIH-MSP 2009 - 2009 5th International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2009, pp. 365–368.

[7] ABES Engineering College and Institute of Electrical and Electronics Engineers, *3rd IEEE International Conference on "Computational Intelligence and Communication Technology" (IEEE-CICT 2017) : 9th & 10th February, 2017, ABES Engineering College, Ghaziabad.*

[8] D. R. I. M. Setiadi, "Improved payload capacity in LSB image steganography uses dilated hybrid edge detection," *Journal of King Saud University - Computer and Information Sciences*, 2019.

[9] B. Isac and V. Santhi, "A Study on Digital Image and Video Watermarking Schemes using Neural Networks," *International Journal of Computer Applications*, vol. 12, no. 9, pp. 1–6, 1 2011.

[10] C. Jin and S. Wang, "Applications of a neural network to estimate watermark embedding strength," in *8th International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2007*, 2007.

[11] H. Kandi, D. Mishra, and S. R. Gorthi, "Exploring the learning capabilities of convolutional neural networks for robust image watermarking," *Computers and Security*, vol. 65, pp. 247–268, 3 2017.

[12] S.-M. Mun, S.-H. Nam, H.-U. Jang, D. Kim, and H.-K. Lee, "A Robust Blind Watermarking Using Convolutional Neural Network," *Neurocomputing*, vol. 337, pp. 191–202, 4 2017. [Online]. Available: http://arxiv.org/abs/1704.03248http://dx.doi.org/10.1016/j.neucom.2019.01.067

[13] M. Abadi and D. G. Andersen, "Learning to Protect Communications with Adversarial Neural Cryptography," 10 2016. [Online]. Available: http://arxiv.org/abs/1610.06918

[14] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding Watermarks into Deep Neural Networks," *ICMR 2017 - Proceedings of the 2017 ACM International Conference on Multimedia Retrieval*, pp. 269–277, 1 2017. [Online]. Available: http://arxiv.org/abs/1701.04082http://dx.doi.org/10.1145/3078971.3078974

[15] T. Fang, M. Jaggi, and K. Argyraki, "Generating Steganographic Text with LSTMs," *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Student Research Workshop*, pp. 100–106, 5 2017. [Online]. Available: http://arxiv.org/abs/1705.10742

[16] J. Kodovský, J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," in *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, 4 2012, pp. 432–444.

[17] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012.

[18] C. Xia, Q. Guan, X. Zhao, Z. Xu, and Y. Ma, "Improving GFR steganalysis features by using Gabor symmetry and weighted histograms," in *IH and MMSec 2017 - Proceedings of the 2017 ACM Workshop on Information Hiding and Multimedia Security*. New York, New York, USA: Association for Computing Machinery, Inc, 6 2017, pp. 55–66. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3082031.3083243

[19] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, "Selection-channel-aware rich model for Steganalysis of digital images," in *2014 IEEE International Workshop on Information Forensics and Security, WIFS 2014*. Institute of Electrical and Electronics Engineers Inc., 4 2015, pp. 48–53.

[20] T. Denemark, J. Fridrich, and P. Comesaña-Alfaro, "Improving selection-channel-aware steganalysis features," in *IS and T International Symposium on Electronic Imaging Science and Technology*. Society for Imaging Science and Technology, 2016.

[21] S. Tan and B. Li, "Stacked convolutional auto-encoders for steganalysis of digital images," in *2014 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2014*. Institute of Electrical and Electronics Engineers Inc., 2 2014.

[22] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in *Media Watermarking, Security, and Forensics 2015*, A. M. Alattar, N. D. Memon, and C. D. Heitzenrater, Eds., vol. 9409. SPIE, 3 2015, p. 94090J. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2083479

[23] L. Pibre, P. Jérôme, D. Ienco, and M. Chaumont, "Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch," *IS and T International Symposium on Electronic Imaging Science and Technology*, 11 2015. [Online]. Available: http://arxiv.org/abs/1511.04855

[24] G. Xu, H. Z. Wu, and Y. Q. Shi, "Ensemble of CNNs for steganalysis: An empirical study," in *IH and MMSec 2016 - Proceedings of the 2016 ACM*

Information Hiding and Multimedia Security Workshop. Association for Computing Machinery, Inc, 2016, pp. 103–107.

[25] ——, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, 5 2016.

[26] J. Zeng, S. Member, S. Tan, S. Member, B. Li, and J. Huang, "Large-scale JPEG image steganalysis using hybrid deep-learning framework," Tech. Rep.

[27] J. Zeng, S. Tan, B. Li, and J. Huang, "Pre-training via fitting deep neural network to rich-model features extraction procedure and its effect on deep learning for steganalysis," in *IS and T International Symposium on Electronic Imaging Science and Technology*. Society for Imaging Science and Technology, 2017, pp. 44–49.

[28] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich, "JPEG-phase-aware convolutional neural network for steganalysis of JPEG images," in *IH and MMSec 2017 - Proceedings of the 2017 ACM Workshop on Information Hiding and Multimedia Security*, vol. 10. New York, New York, USA: Association for Computing Machinery, Inc, 6 2017, pp. 75–84. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3082031.3083248

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Tech. Rep. [Online]. Available: http://image-net.org/challenges/LSVRC/2015/

[30] G. Xu, "Deep convolutional neural network to detect J-UNIWARD," in *IH and MMSec 2017 - Proceedings of the 2017 ACM Workshop on Information Hiding and Multimedia Security*. New York, New York, USA: Association for Computing Machinery, Inc, 6 2017, pp. 67–73. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3082031.3083236

[31] X. Huang, S. Wang, T. Sun, G. Liu, and X. Lin, "Steganalysis of Adaptive JPEG Steganography Based on ResDet," in *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2018 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., 3 2019, pp. 549–553.

[32] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," pp. 436–444, 5 2015. [Online]. Available: https://www.nature.com/articles/nature14539

[33] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[34] S. Xie and Z. Tu, "Holistically-Nested Edge Detection," *International Journal of Computer Vision*, vol. 125, no. 1-3, pp. 3–18, 4 2015. [Online]. Available: http://arxiv.org/abs/1504.06375

[35] D. R. I. M. Setiadi and J. Jumanto, "An enhanced LSB-Image Steganography Using the Hybrid Canny-Sobel edge detection," *Cybernetics and Information Technologies*, vol. 18, no. 2, pp. 74–88, 6 2018. [Online]. Available: https://content.sciendo.com/view/journals/cait/18/2/article-p74.xml