Fully Connected Neural Network Based Carrier Estimation Mechanism on Encrypted Images for Data Hiding in Cloud Network

[1]Prasad C N and [2]Suchithra R
[1,2]Department of Computer Science, Chirashree Institute of Research and Development (CIRD), University of Mysore, Karnataka, India.
[1]prasad_achar@sju.edu.in, [2]suchithra.suriya@gmail.com

Correspondence should be addressed to Prasad C N : prasad_achar@sju.edu.in

Article Info
Journal of Machine and Computing (https://anapub.co.ke/journals/jmc/jmc.html)
Doi: https://doi.org/10.53759/7669/jmc202505069.
Received 05 March 2024; Revised from 14 November 2024; Accepted 21 February 2024.
Available online 05 April 2025.

Abstract – The work proposes a fully connected neural network (FCNN) based approach for detecting the Carrier blocks for embedding the data in encrypted images in the cloud network. In a data embedding process, the determination of non-carrier pixels that provide underflow and overflow during the data embedding process plays a major role. The location map for the non-carrier blocks is usually compressed and embedded in the encrypted image along with the hidden data. The embedding rate and peak signal-to-noise ratio (PSNR) are limited due to the storage of huge location map information on the image. Therefore, the proposed approach uses the FCNN network to detect the Carrier blocks /non-carrier blocks which highly minimizes the additional location map information to be embedded. In the embedding phase, a trained FCNN network is utilized to detect the carrier blocks, in which the FCNN network is trained with the labels that are generated by trial 0's and 1's embedding process. Two approaches are utilized in training the FCNN that includes FCNN with predictor only (FCNN-PO) and FCNN with sub-block fully (FCNN-SF) schemes in detecting the carrier blocks. In the extraction phase, the same FCNN model is used to detect the carrier blocks from which the data and actual encrypted image can be reconstructed. The performance of the carrier detection process was evaluated using measures such as precision, recall, and accuracy, while the data hiding process was evaluated using measures such as structural similarity index measurement (SSIM), PSNR, and embedding rate. The FCNN-PO carrier/non-carrier classification process results in an average accuracy of 98.59% in detecting the carrier while providing an SSIM, PSNR, and embedding rate of 0.9926, 58.86db and 1.97bpp respectively during the data embedding process when evaluated using the Bows-2 dataset.

Keywords – Data Hiding, Fully Connected Neural Network, Image Encryption, Embedding Rate, Prediction Error Expansion.

I. INTRODUCTION

The enhanced storage capacity of cloud computing [1] has drawn a number of consumers and researchers. Nevertheless, there are a number of issues with cloud storage, such as integrity, secrecy, and authentication. Data encryption [2] and data hiding [3] are employed to address these issues. The pixel content is altered by the data embedding process in order to hold the hidden material. Through the creation of a cipher picture, data encryption is employed which preserves the content of the plain image.

Schemes like difference expansion [4], integer transform [5], prediction error expansion [6], and histogram shifting [7] are the foundation of the widely used reversible data embedding technique. After data extraction, the original image cannot be recreated in the non-reversible approach [8], but in the reversible data hiding approach [9], both the carrier image and the hidden data can be recreated without any modifications. Before uploading the picture to the cloud, the user usually encrypts it to protect its privacy content. To stop an unauthorized individual from seeing the actual image content, partial or full encryption can be employed. In the patch level representation [10], the original image is transformed into sparse coefficients based on a dictionary, providing an adequate amount of carrier for embedding the data. For the residual errors to be embedded, this method necessitates the use of an extra reversible data-hiding strategy.

In [11] developed two data embedding approaches in which a public cryptosystem is used to encrypt the images based on their homomorphic and probabilistic features. This approach uses non-reversible and reversible embedding mechanisms. In the reversible method, the homomorphic cryptosystem is utilized to encrypt the final image after the histogram has first been reduced. The non-reversible method uses multilayer wet paper coding for embedding the data on the image which was initially encrypted. From the image's most significant bit (MSB) planes, the block-based MSB plain rearrangement technique [12] calculates the highly compressible bit streams, where the author Yi et al. suggested parametric binary tree labelling [13], in which the data is encoded in discrete encrypted pieces.

Two general categories can be used to categorize the process of maintaining the private content of an image: reserving a room before encryption (RRBE) and vacating the room after encryption (VRAE). According to the RRBE technique [14], the picture owner must preprocess the image so that encryption can take place after that some space has been left for data embedding. However, the owner of the VRAE technique [15], can upload the image to the cloud and encrypt it without performing any additional pre-processing. To conceal the data, the cloud might alter the encrypted picture. The RRBE strategy can produce a higher payload than the VRAE approach, however, it necessitates an extra preprocessing step that adds to the workload for the owner of the image.

The [14] presented the RRBE scheme for the first time. This method is based on the reversible data hiding technique known as histogram shifting, in which a blank space is produced in order to incorporate a small number of pixels least significant bits (LSB). While the remaining areas are encrypted and sent to the cloud, the unoccupied space is left intact. This method yields an embedding rate of 0.5bpp and uses the unoccupied region to conceal the secret data. In [15] used a second MSB plane in addition to the first MSB plane for data embedding, modifying the predictor used in [16]. An average payload of 1.35bpp is provided by this method. The [17] employ the bit planes iteratively from MSB to LSB, resulting in an average embedding rate of 1.84bpp. When compared to a difference expansion strategy established by [18], the performance of the prediction error expansion proposed by [19] delivers a superior performance.

In [20] introduced the two-dimensional (2D) prediction error histogram (PEH) and the one-dimensional (1D) prediction error histogram with the 2D approach offering a better correlation between the prediction errors. 2D-PEH and 1D-PEH were merged by the author Zhang et al. [21] to create an effective map for data embedding. According to [20], the picture pixel is divided into two areas: the flat area and the rough region. The flat area can embed two bits per pixel, while the rough area can embed one bit per pixel. The prediction value [23] is estimated by averaging four neighboring pixels, which enhances the quality of the marked image. The histogram's prediction error is shifted to the right and left, producing a number of zero and peak spots. Because of this, a higher capacity is achieved when histogram shifting and prediction error expansion are combined [21].

The scheme [20] uses different predictors in embedding the data. However, a Convolutional neural network model (CNN) was utilized to choose the best predictor, in which the CNN model was trained by embedding the trial data using different predictors. An adaptive approach was used with a block embedding process that increases the Laplacian type distribution in data embedding. To improve the correlation between the intra block the encryption was done by a bit plane selection approach which improve the performance to a certain extent. The data was embedded in a hierarchical approach that classifies the pixels of the image based on multiple linear regression. However, this approach results in huge location map information which is compressed and embedded in the encrypted image.

Most of the data embedding approaches that are discussed above do not uses an artificial intelligence based carrier and non-carrier detection approach that pre-determines the carrier blocks for embedding the data. The carrier and non-carrier blocks are only detected after embedding the data, which results in a huge location map information. Therefore, the proposed approach uses a fully connected neural network to pre-determine the carrier blocks before the actual embedding process. This further facilitate to pre-determine the carrier blocks during the data extraction process in which the data is embedded. Detection of carrier and non-carrier without the use of location map information will highly reduce the computational burden of the data-hiding process while improving the embedding rate.

The Contributions of The Work Are Mentioned Below

- The work proposes a fully connected neural network (FCNN) based carrier and non-carrier classification approach which facilitate to pre-determine the carrier blocks before the actual embedding process.
- Two different types of carriers and non-carrier classification approaches namely FCNN with sub-block fully (FCNN-SF) and FCNN with predictor only (FCNN-PO) were proposed to detect the carrier blocks.
- The approach uses a prediction error expansion approach to embed the data, while in the extraction phase, the carrier is again detected using either the same FCNN-SF or FCNN-PO approach.
- A block-based approach is utilized in the encryption/decryption process, where the classification process involved in the proposed approach was evaluated with the metrics recall, precision, and accuracy. The embedding process involved in the proposed approach was evaluated with the measures namely, SSIM, embedding rate, and PSNR with the datasets namely BOWS-2 and Bossbase.

The upcoming sections of the paper are structured as follows. A detailed description of the proposed approach is presented in Section 2, while the experimental results of the proposed scheme are provided in Section 3. Finally, the conclusion is framed in Section 4.

II. PROPOSED METHOD

The three major steps involved in the proposed data embedding approach are (i) the Data embedding process with Carrier detection, (ii) the Data extraction process with carrier detection, and (iii) the Training process of FCNN with label generation.

Data Embedding Process with Carrier Detection

The illustration of the data embedding process is shown in **Fig 1**. In the embedding process, the image was initially encrypted by a block-based encryption approach.

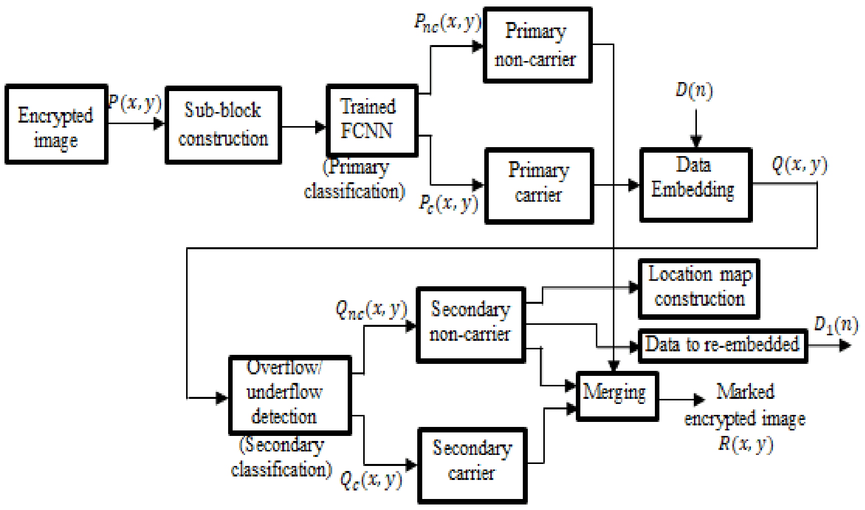


Fig 1. Flow Representation of Data Embedding.

Encryption Process

Let $G(x,y)$ represents the actual image having a size of $J \times K$. The image $G(x,y)$ is initially grouped into sub-blocks each having a size of 3×3. Thus, the total number of sub-blocks be $N_t = \frac{J \times K}{9}$. Let Δ represent the key that is used to perform encryption. The key $\Delta = \{\Delta_1, \Delta_2\}$ has two components. The key Δ_1 is used to perform the inter-block permutation process, while the key Δ_2 is used to perform the intra-block permutation process.

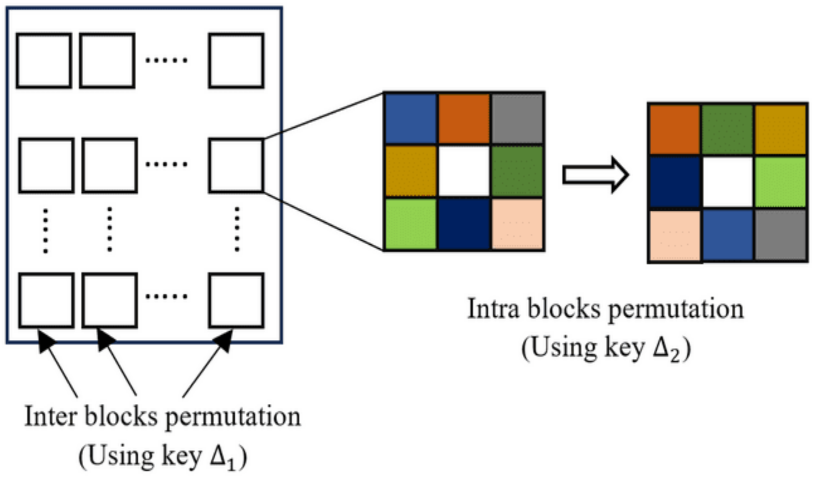


Fig 2. Inter-Block and Intra-Block Permutation-Based Encryption.

In the proposed encryption process, two permutation processes are involved namely (i) Inter-block permutation and (ii) intra-block permutation. In the inter-block permutation process N_t number of random sequences between 1 and N_t is generated utilizing the key Δ_1. Let the inter-block random sequence be represented by S_1. With the sequence S_1, the position of each block is shuffled. Let the sub-blocks after shuffling be represented as $G_1(x,y)$. In the intra-block permutation, the key Δ_2 is used to derive N_t number of intermediate keys using the pseudo-random sequence whose values lie between 1 and 8. Using the intermediate keys the pixels in the sub-blocks are scrambled to perform intra-block permutation as illustrated in **Fig 2**.

Data Embedding

Let $P(x, y)$ represents the image that was encrypted by the user (image that was uploaded to the cloud). The image $P(x, y)$ was partitioned into non-overlapping blocks where each block has size of 3×3. Each sub-blocks are fed to the trained FCNN for primary classification, the trained FCNN categorizes the sub-blocks either as primary non-carriers $P_{nc}(x, y)$or primary carriers $P_c(x, y)$. Let the 3×3 neighborhood pixels with center pixel h be represented by the matrix

$$\lambda = \begin{bmatrix} g_1 & g_2 & g_3 \\ g_4 & h & g_5 \\ g_6 & g_7 & g_8 \end{bmatrix} \tag{1}$$

From the neighbor pixels $p = \{g_1, g_2 \dots \dots g_8\}$, the predictor $\hat{\varphi}$ is estimated. The proposed predictor is derived from the predictor φ used in [25] which has the following relations.

$$Predictor\ \varphi = \begin{cases} \min(\gamma_1, \gamma_2) & \gamma_1 \geq \max(g_7, \gamma_2) \\ \max(\gamma_1, \gamma_2) & \gamma_1 \geq \min(g_7, \gamma_2) \\ g_7 + \gamma_2 - \gamma_2 & otherwise \end{cases} \tag{2}$$

Where, $\gamma_1, = \left\lfloor \frac{(g_1 + g_2 + g_3)}{3} \right\rfloor$, and $\gamma_2 = \left\lfloor \frac{(g_4 + g_5 + g_6)}{3} \right\rfloor$

For the determination of the proposed predictor $\hat{\varphi}$, the pixels $p = \{g_1, g_2 \dots \dots g_8\}$ is sorted in ascending order. The sorted pixel sequence is represented as $f = \{b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8\}$. After sorting, three different boundary values are estimated. The first boundary values are estimated from the alternate sorted values starting from the first value b_1

$$\hat{\gamma}_1 = \left\lfloor \frac{(b_1 + b_3 + b_5 + b_7)}{4} \right\rfloor \tag{3}$$

The second boundary value is estimated from the alternate sorted values started from the second value b_2

$$\hat{\gamma}_2 = \left\lfloor \frac{(b_2 + b_4 + b_6 + b_8)}{4} \right\rfloor \tag{4}$$

The third boundary represents the median values which can be estimated by

$$\hat{\gamma}_3 = \left\lfloor \frac{(b_4 + b_5)}{2} \right\rfloor \tag{5}$$

Using the three boundaries $\hat{\gamma}_1$, $\hat{\gamma}_2$ and $\hat{\gamma}_3$ the predictor can be estimated as,

$$\hat{\varphi} = \begin{cases} \min(\hat{\gamma}_1, \hat{\gamma}_2) & \hat{\gamma}_3 \geq \max(\hat{\gamma}_1, \hat{\gamma}_2) \\ \max(\hat{\gamma}_1, \hat{\gamma}_2) & \hat{\gamma}_3 \geq \min(\hat{\gamma}_1, \hat{\gamma}_2) \\ \hat{\gamma}_1 + \hat{\gamma}_2 - \hat{\gamma}_3 & otherwise \end{cases} \tag{6}$$

During the data embedding the prediction error is estimated using the predictor $\hat{\varphi}$ and the centre pixel h as,

$$\delta = h - \hat{\varphi} \tag{7}$$

The error δ can be expanded using the relation,

$$\hat{\delta} = \begin{cases} 2\delta + \rho & \text{if } \delta \in (\rho, \infty) \\ 2\delta - \rho & \text{if } \delta \in (-\infty, -\rho) \\ D + 2\delta & \text{if } \delta \in (-\rho, \rho) \end{cases} \tag{8}$$

Here D represent the data and the embedding capacity can be controlled by the parameter ρ. Utilizing the expanded error and the predictor value $\hat{\varphi}$, the marked pixel can be obtained as

$$\hat{h} = \hat{\varphi} + \hat{\delta} \tag{9}$$

The primary carrier blocks $P_c(x, y)$ is used to embed the data D to obtain the data embedded carrier blocks $Q(x, y)$. The data-embedded carrier blocks $Q(x, y)$ is checked for overflow/underflow. This overflow/underflow detection categorizes the data-embedded blocks as secondary non-carriers $Q_{nc}(x, y)$ or secondary carriers $Q_c(x, y)$. The secondary carrier blocks do not have any overflow/underflow, while the secondary non-carrier blocks have overflow/underflow. The primary non-carrier blocks $P_{nc}(x,y)$, secondary non-carriers blocks $Q_{nc}(x,y)$, and secondary carrier blocks $Q_c(x,y)$ are merged to obtain the marked encrypted image. Eventhough, the secondary non-carriers also hold the secret data information, the data embedded in this non-carrier are re-embedded (Data D_1) in another carrier location. The location map of secondary non-carrier blocks is compressed and embedded in the marked encrypted image. However, the number of location map information is highly reduced since the majority of the non-carrier can be detected by the FCNN without its location map in the extraction phase. Let the resultant marked encrypted image be represented as $R(x,y)$.

Data Extraction Process with Carrier Detection

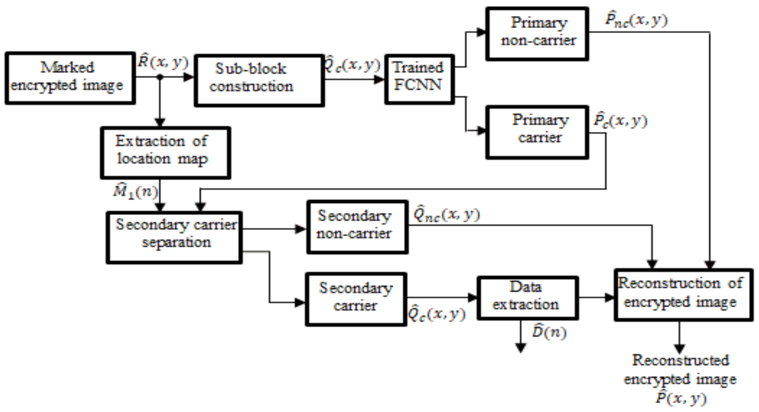


Fig 3. Flow Representation of Data Extraction.

During the data extraction process, the cloud server will extract the data that was embedded in the marked encrypted image and reconstruct the actual encrypted image which was uploaded to the cloud by the user. The marked encrypted image $\hat{R}(x,y)$ is initially partitioned into sub-blocks where each block has a size of 3×3. From the marked encrypted image, the additional location map information regarding the secondary non-carriers is initially extracted as illustrated in **Fig 3**. The trained FCNN is utilized to classify the sub-blocks into primary carrier and non-carrier classes. From the extracted location map information, the secondary carrier blocks $\hat{Q}_c(x,y)$, and secondary non-carrier blocks $\hat{Q}_{nc}(x,y)$ are identified. The secondary non-carrier blocks $\hat{Q}_{nc}(x,y)$ which are identified by location map information are not considered for data extraction, since it does not hold data. The data from the secondary carrier blocks are extracted using the prediction error expansion approach using the proposed predictor provided in equation (6). Let the predictor estimated from the 3x3 sub-blocks be $\hat{\varphi}$, while $\hat{h}$ represents the marked center pixels. Using the predictor $\hat{\varphi}$ and center pixel $\hat{h}$, the expanded error can be estimated as

$$\hat{\delta} = \hat{h} - \hat{\varphi} \tag{10}$$

Using the expanded error $\hat{\delta}$, the hidden data D and the actual center pixel h can be reconstructed [28]. The same procedure is followed in all sub-blocks to obtain the encrypted image and the complete data D. The reconstructed encrypted image $\hat{P}(x,y)$ is decrypted using the same inter-intra permutation approach to obtain the actual image by the user. The same key Δ must be used in the decryption process for the exact reconstruction of plain image.

Training Process of FCNN With Label Generation

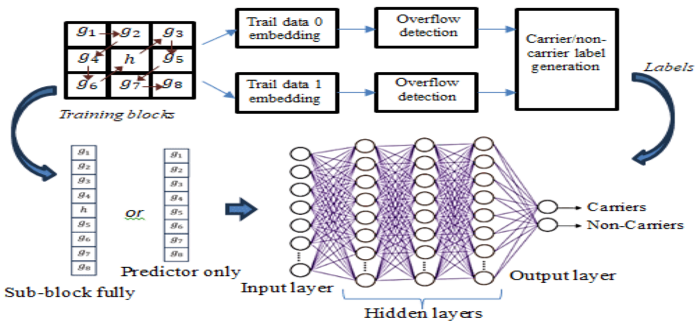


Fig 4. Flow Diagram Representation of Training Process in FCNN Model.

For the training of the FCNN [29], the labels are initially generated. In the FCNN with sub-block fully (FCNN-SF) approach the FCNN uses the number of input layers as 9, while in the FCNN with predictor only (FCNN-PO) approach the FCNN uses the number of input layers as 8. In the FCNN-PO approach the pixels that are utilized for the predictor estimation are used for training, while in the FCNN-SF approach the complete pixels in the 3×3 sub-blocks are used for training. The labels are generated by embedding the trial data on the center pixels. The sub-block is considered to have carrier as label, if the block does not result in overflow or underflow while embedding trial 0's and 1's during data embedding. Conversely, the block is considered to have non-carrier as label if it results in overflow or underflow during trial 0's and 1's during data embedding. The FCNN is model is trained using the pixels of the sub-blocks as the input and generated labels as the actual labels as illustrated in **Fig 4**. The FCNN input is normalized between -1 to 1 before the actual training/ classification process. The pixel input x to the input layer is normalized using the relation $\hat{x} = \frac{(x-128)}{128}$ which results in both positive and negative values. The proposed FCNN uses the LCL activation function [30] which gives equal weightage to both positive and negative values. The FCNN uses 2 hidden layers each having 12 neurons in both the FCNN-SF and FCNN-PO approach.

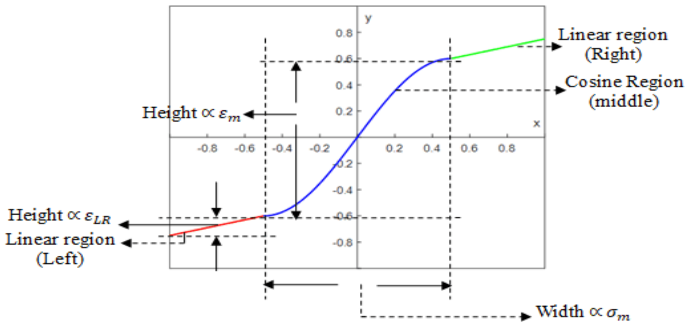


Fig 5. Representation of LCL Activation Function.

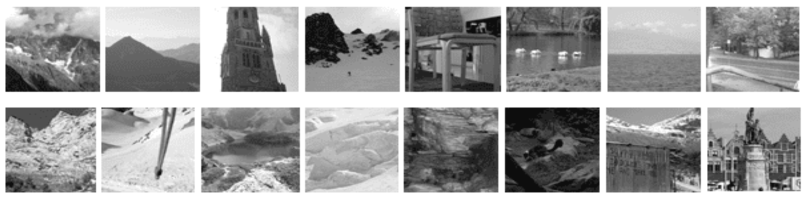
The structure of the LCL activation function is illustrated in **Fig 5**which contains a cosine region between two linear regions. Let ε_m and ε_{LR} represents the amplitude or height of the middle cosine and linear regions respectively. The width of the mid-cosine region is represented as σ_m. The output of the activation function $y_{act}(x)$ for input x can be represented as

$$y_{act}(x) = \begin{cases} \varepsilon_m cos\left(2\pi\left(0.75 + \frac{0.25x}{\sigma_m}\right)\right) & -\sigma_m \le x \le \sigma_m \\ \varepsilon_{LR}x + y_{act}(\sigma_m) - \varepsilon_{LR}\sigma_m & x > \sigma_m \\ \varepsilon_{LR}x + y_{act}(-\sigma_m) + \varepsilon_{LR}\sigma_m & x < -\sigma_m \end{cases} \quad (11)$$

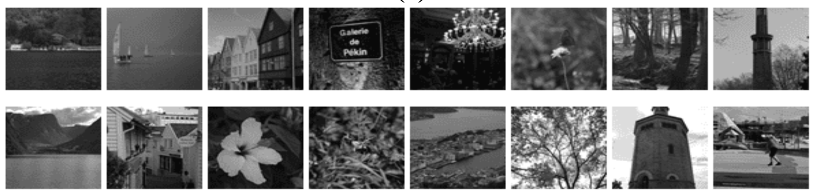
The use of the LCL activation function bounds the values between -1 to 1 while training or classifying the model. In the proposed approach the activation function uses the factors $\varepsilon_m = 0.6$, $\varepsilon_{LR} = 0.4$ and $\sigma_m = 0.6$. The next section provides the experimental results of the FCNN-SF and FCNN-PO data hiding schemes.

III. EXPERIMENTAL RESULTS

Two databases such as BOWS-2 [31] and Bossbase databases [32] are used for evaluation. The images of these two databases are 8 bit grayscale images each having a size of 512 × 512. Each dataset has 10,000 images, where a few sample images are illustrated in **Fig 6**. To detect the carrier block, the FCNN is trained with 60% of the images from each dataset. The remaining 40% of the images are used to test the carrier detection process and the actual data embedding. The tool MATLAB 2018a is used to implement the data embedding algorithm.



(a)



(b)

Fig 6. Sample Images Used for Analysis (A) Bows-2 Dataset (B) Bossbase Dataset.

Let the true positives, false negatives, true negatives, and false positives be represented by the variables ρ_{tp}, ρ_{fn}, ρ_{tn} and ρ_{fp} respectively. The performance of the proposed fully connected network-based carrier detection process was evaluated using measures such as precision, recall, and accuracy which can be estimated with the relations that are provided below,

$$Precision = \frac{\rho_{tp}}{\rho_{tp}+\rho_{fp}} \tag{12}$$

$$\text{Recall} = \frac{\rho_{\text{tp}}}{\rho_{\text{tp}}+\rho_{\text{fn}}} \tag{13}$$

$$Accuracy = \frac{\rho_{tp}+\rho_{tn}}{\rho_{tp}+\rho_{tn}+\rho_{fp}+\rho_{fn}} \tag{14}$$

The performance of the proposed data hiding approach was evaluated with the parameters namely structural similarity index measurement (SSIM), embedding rate, and peak signal-to-noise ratio (PSNR) that can be estimated using the following relations.

$$\text{SSIM}(\text{P},\text{R}) = \frac{(2\sigma_{\text{P,R}}+\omega_1)(2\mu_\text{P}\mu_\text{R}+\omega_2)}{(\sigma_\text{P}^2+\sigma_\text{R}^2+\omega_1)(\mu_\text{P}^2+\mu_\text{R}^2+\omega_2)} \tag{15}$$

In the above equation of SSIM σ_P^2 and σ_R^2 represents the variance of the encrypted and marked encrypted image respectively, while μ_P and μ_R represents the mean of the encrypted and marked encrypted image respectively.

$$Embedding\ rate = \frac{Z}{J\times K} \tag{16}$$

In above equation Z represents the total number of bits embedded and $J \times K$ represents the size of the image

$$\text{PSNR} = 10\log_{10}\frac{255^2}{\tau} \tag{17}$$

Where τ resembles the mean square error estimated using the encrypted image $P(x,y)$ and the marked encrypted image $R(x,y)$ as

$$\tau = \frac{1}{J\times K}\sum_{x=1}^{J}\sum_{y=1}^{K}[P(x,y) - R(x,y)] \tag{18}$$

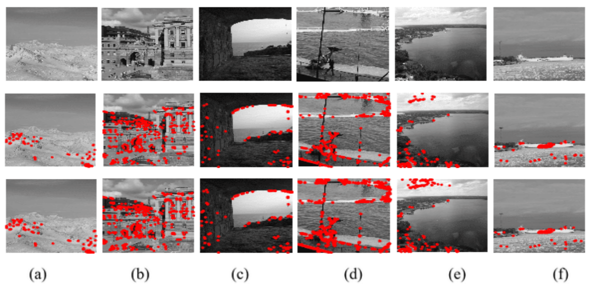


Fig 7. Results Obtained by The Proposed Approach in Carrier Detection on Plain Images (A) Input Images (B) Encrypted Images (C) Carriers Detected By FCNN-SF (D) Carriers Detected By FCNN-PO.

Fig 7 illustrates the results obtained by the proposed approach in detecting the carriers and the non-carriers on the plain images without encryption. The red color indication shows the non-carrier blocks which are detected by the FCNN-SF and FCNN-PO approach. Both the FCNN-SF and FCNN-PO approach almost detect an equal number of non-carriers. This non-carrier block indicates that these blocks will overflow or underflow if the data either 0 or 1 is embedded.

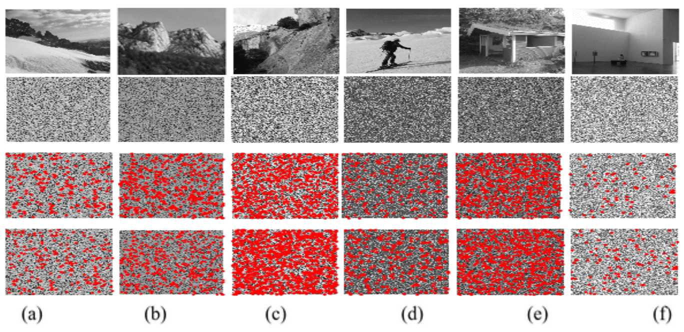


Fig 8. Results Obtained During the Proposed Approach (A) Input Images (B) Encrypted Images (C) Carriers Detected By FCNN-SF (D) Carriers Detected By FCNN-PO.

Fig 8 illustrates the results obtained by the proposed approach in detecting the carriers and the non-carriers on the encrypted images. The red color indication shows the non-carrier blocks that are detected by the FCNN-SF and FCNN-PO on the encrypted images. For this implementation, in both the schemes FCNN-SF and FCNN-PO the plain images are encrypted with the same key to compare the location of the FCNN-SF and FCNN-PO non-carriers. Both the FCNN-SF and FCNN-PO approach almost detect an equal number of non-carriers. This non-carrier block indicates that these blocks will overflow or underflow if the data either 0 or 1 is embedded.

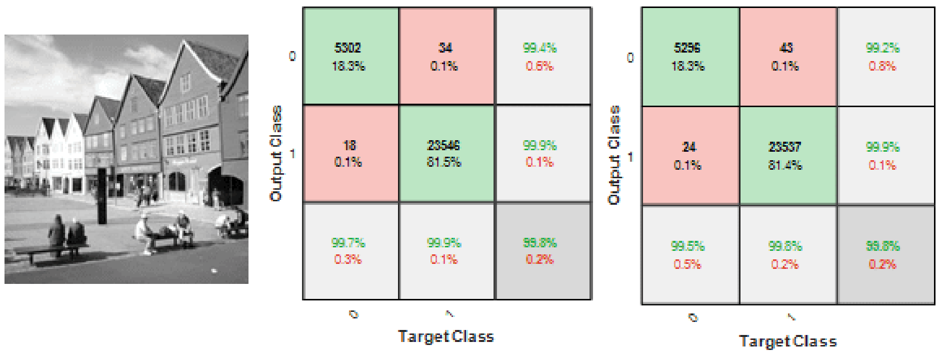


Fig 9. Confusion Matrices Obtained During the Classification of Carrier and Non-Carrier (A) Input Image (B) FCNN-SF (C) FCNN-PO.

Fig 9 (b) and (c) show the confusion matrices obtained by the FCNN-SF and FCNN-PO schemes respectively in the classification of carrier and non-carrier. In this confusion matrices the class '0' represents the non-carrier blocks while the class '1' represents the classified carrier blocks.

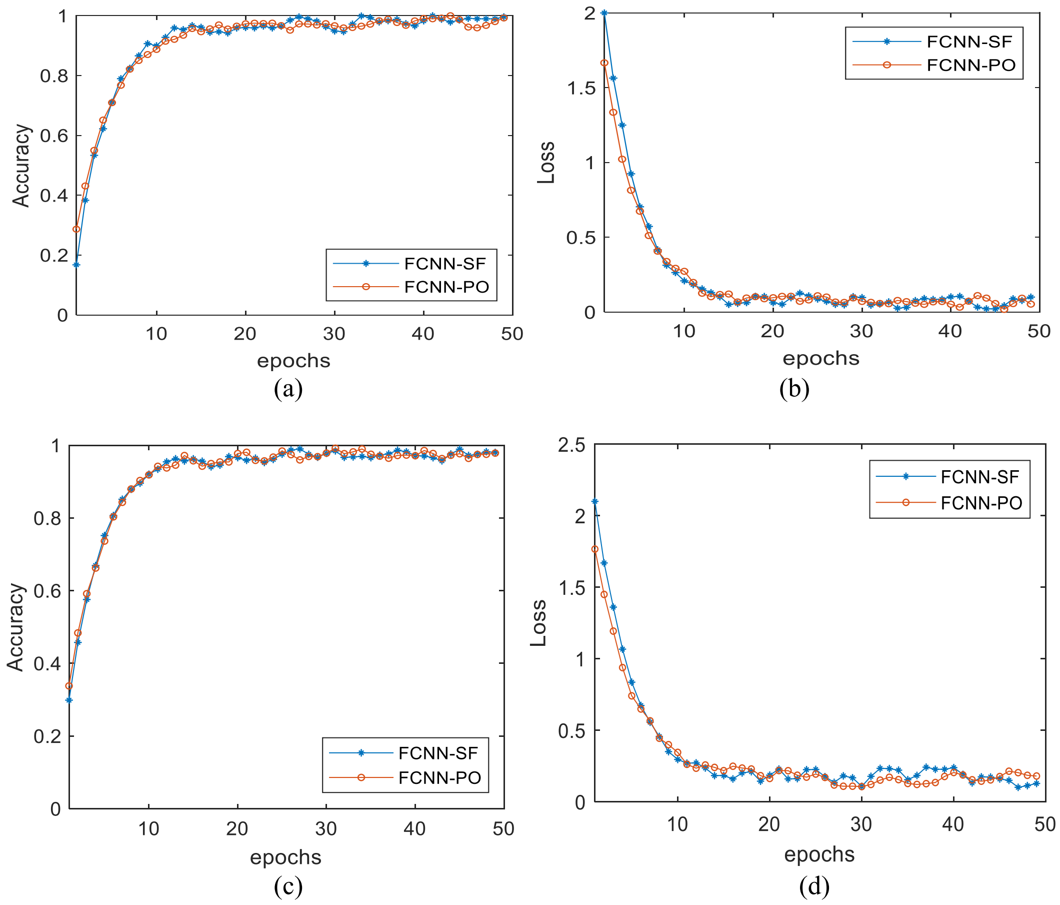


Fig 10. Accuracy And Loss Comparison for The Two Approaches FCNN-SF And FCNN-PO During the Training Process (A) Accuracy Plot in Bossbase Dataset (B) Loss Plot in Bossbase Dataset (C) Accuracy Plot in Bows-2 Dataset (B) Loss Plot in Bows-2 Dataset.

Fig 10 illustrates the accuracy and loss curves obtained during the training of the model by the FCNN-SF and FCNN-PO approaches. The FCNN model was trained with 50 epochs. The accuracy and loss almost stabilize as the number of epochs reaches 25. The training accuracy obtained while training with the Bows-2 dataset is less than the Bossbase dataset. Also, the loss obtained while training with the Bows-2 dataset is higher than the Bossbase dataset.

Table 1. Performance Comparison of the FCNN-SF And FCNN-PO Schemes in The Classification of Carrier and Non-Carrier.

Database	**Class**	**FCNN-SF**			**FCNN-PO**		
		Precision	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**
Bossbase	Carrier	98.73	98.12	98.46	99.12	98.96	99.24
	Non-carrier	98.23	98.04	98.54	98.96	98.74	99.01
	Average	**98.48**	**98.08**	**98.50**	**99.04**	**98.85**	**99.13**
Bows-2	Carrier	98.21	97.96	98.37	98.76	98.43	98.74
	Non-carrier	98.01	97.58	97.91	98.54	98.26	98.43
	Average	**98.11**	**97.77**	**98.14**	**98.65**	**98.35**	**98.59**

Table 1 illustrates the performance comparison of the FCNN-SF and FCNN-PO schemes in detecting the carrier and non-carrier. For the evaluation of the FCNN-SF, the FCNN model is trained with the complete sub-blocks before and after embedding the data. Therefore, the performance was evaluated on the complete sub-block before and after embedding the data. The FCNN-SF approach results in an average precision, recall, and accuracy of 98.48%, 98.08%, and 98.50% respectively when evaluated using Bossbase dataset. The same approach results in precision, recall, and accuracy of 98.11%, 97.77%, and 98.14% respectively using Bows-2 dataset. For the evaluation of the FCNN-PO approach, the FCNN model is trained with only the predictor pixels without including the pixel in which the data is embedded. The FCNN-PO approach results in an average precision, recall, and accuracy of 99.04%, 98.85%, and 99.13% respectively when evaluated using the Bossbase dataset. The same approach results in precision, recall, and accuracy of 98.65%, 98.35%, and 98.59% respectively when evaluated using the Bows-2 dataset.

Fig 11 illustrates the graphical comparison of the proposed schemes FCNN-SF and FCNN-PO approach in both the Bossbase and Bows-2 dataset. The FCNN-PO approach results in higher precision, recall, and accuracy of 0.56%, 0.77%, 0.63% respectively in Bossbase dataset. The FCNN-PO approach results in higher precision, recall, and accuracy of 0.54%, 0.58%, 0.45% respectively in Bows-2 dataset.

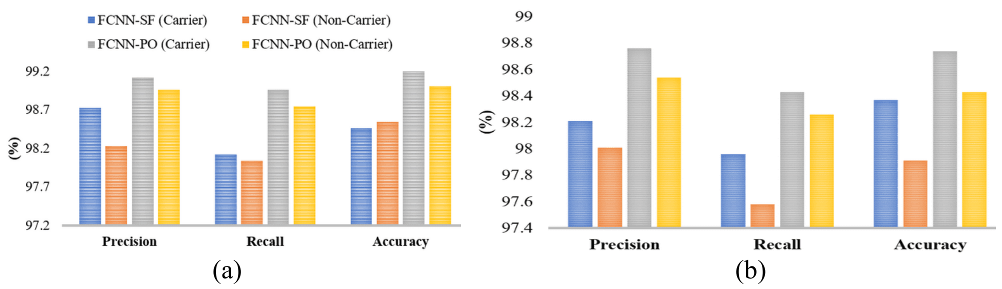


Fig 11. Performance Comparison for The Proposed Schemes FCNN-SF And FCNN-PO In the Classification of Carrier and Non-Carrier (a) Bossbase Database (b) Bows-2 Database.

Table 2. Performance Comparison Of FCNN-SF And FCNN-PO Approach with Recent Data-Hiding Approaches

Methods	**SSIM**	**PSNR (dB)**	**ER (bpp)**
Efficient prediction [16]	0.9988	60.84	1.42
EPE approach [17]	0.9909	55.63	1.81
Binary tree [13]	0.9871	51.36	1.92
Extended Run-length [12]	0.9892	53.47	1.93
Recursive RDH [33]	0.9921	57.81	1.7
CNN-Predictor [25]	0.9915	55.82	1.95
FCNN-SF (BossBase)	0.9929	58.97	1.99
FCNN-SF (Bows-2)	0.9923	58.42	1.98
FCNN-PO (BossBase)	0.9931	59.01	1.99
FCNN-PO (Bows-2)	0.9926	58.86	1.97

The comparison interms of SSIM, PSNR, and ER for the proposed approach was made with other data hiding schemes such as efficient prediction [16], EPE approach [17], Binary tree [13], Extended Run-length [12], Recursive RDH and CNN-predictor approach [25]. The data was embedded by selecting the carrier using the two proposed carrier selection processes namely FCNN-SF and FCNN-PO. The performance of the proposed FCNN-PO is better when compared to the FCNN-SF approach in both the Bossbase and Bows-2 datasets as illustrated in **Table 2**. In case of the BossBase dataset, the FCNN-PO results in SSIM, PSNR, and embedding rates of 0.9931, $59.01dB$ and $1.99bpp$ respectively. In case of the Bows-2 dataset, the FCNN-PO results in SSIM, PSNR, and embedding rate of 0.9926, 58.86dB, and $1.97bpp$ respectively. The proposed FCNN-PO approach results in higher performance than the traditional schemes including the FCNN-SF approach.

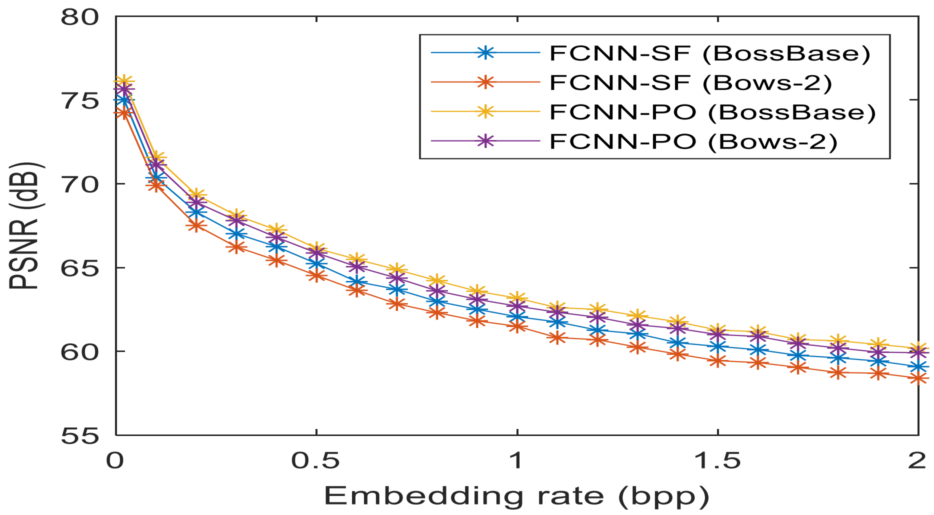


Fig 12. Variation of PSNR At Different Embedding Rates for The Proposed Two Approaches FCNN-SF And FCNN-PO.

Fig 12 illustrates the variation of PSNR at different embedding rates between 0.02bpp to 2bpp. As the bpp is increased, the PSNR gradually reduces in both the approaches namely FCNN-SF and FCNN-PO. This characteristic is almost linear in using both the BossBase and Bows-2 dataset. While comparing the FCNN-SF and FCNN-PO schemes, the FCNN-PO approach results in higher PSNR performance at different embedding rates.

Let $T_{c,em}$, $T_{d,em}$, and $T_{l,em}$, be the time of classification, data embedding, and location map embedding respectively during the data embedding process. Thus, the total time during the embedding phase is

$$T_{t,em} = T_{c,em} + T_{d,em} + T_{l,em} \tag{19}$$

Let $T_{c,ex}$, $T_{d,ex}$, and $T_{l,ex}$, be the time of classification, data extraction, and location map extraction respectively during the data extraction process. Thus, the total time during the extraction phase is

$$T_{t,ex} = T_{c,ex} + T_{d,ex} + T_{l,ex} \tag{20}$$

Table 3. Time Complexity of The Schemes FCNN-SF and FCNN-PO

Dataset	Scheme	Time complexity (s)							
		Embedding phase				Extraction phase			
		$T_{c,em}$	$T_{d,em}$	$T_{l,em}$	$T_{t,em}$	$T_{c,ex}$	$T_{l,ex}$	$T_{d,ex}$	$T_{t,ex}$
Bossbase	FCNN-SF	0.478	0.597	0.397	1.472	0.416	0.423	0.341	1.18
	FCNN-PO	0.517	0.623	0.439	1.579	0.452	0.463	0.387	1.302
Bows-2	FCNN-SF	0.493	0.612	0.413	1.518	0.431	0.448	0.368	1.247
	FCNN-PO	0.526	0.647	0.451	1.624	0.479	0.495	0.408	1.382

Table 3 illustrates the time complexity comparison of the proposed approach when evaluated using the Bossbase and Bows-2 datasets. The total time in the embedding phase by the FCNN-SF and FCNN-PO approach when evaluated in the Bossbase dataset is estimated as 1.472s and 1.518s respectively. The total time in the extraction phase by the FCNN-SF and FCNN-PO approach when evaluated in the Bossbase dataset is estimated as 1.18s and 1.302s respectively. In both the

Bossbase and Bows-2 datasets the FCNN-PO approach has a higher computation time than the FCNN-SF approach as illustrated in **Fig 13**.

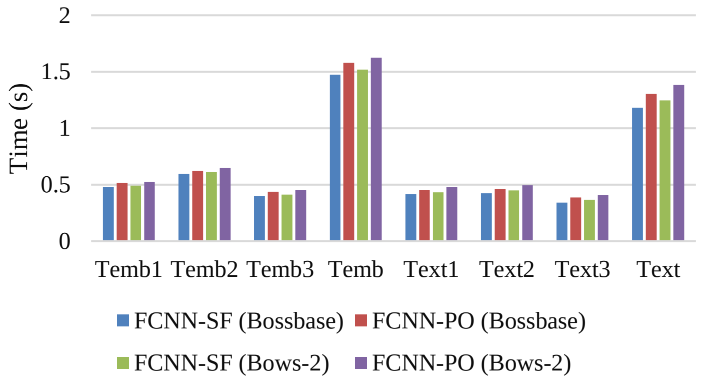


Fig 13. Graphical Comparison of Time Complexity in Proposed Data Embedding.

Even though the FCNN-PO approach provides a higher computation time, its performance is higher in both the Bossbase and Bows-2 datasets.

IV. CONCLUSION

The work proposed a data hiding approach along with a carrier/ non-carrier detection process. Two different carrier detection process namely FCNN-SF and FCNN-PO are proposed that uses fully connected neural networks. The FCNN-SF differentiates the carrier and non-carrier blocks with the complete information from the sub-block including the pixel in which the data is embedded. The FCNN-PO uses only the predictor pixels without considering the pixel that is used to embed data. The performance of the FCNN-SF and FCNN-PO approach in classifying the carrier and non-carrier was evaluated using the measures such as precision, recall, and accuracy while the performance of the FCNN-SF and FCNN-PO data embedding process was evaluated using the measures such as SSIM, embedding rate and PSNR. The proposed FCNN-SF-based carrier detection process results in precision, recall, and accuracy of 98.48%, 98.08%, and 98.50% respectively, while the FCNN-PO-based carrier detection results in precision, recall, and accuracy of 99.04%, 98.85%, 99.13% respectively when evaluated in Bossbase dataset. In the process of data hiding the FCNN-PO results in SSIM, PSNR, and embedding rate of 0.9931, 59.01dB, and 1.99bpp respectively which is better than the FCNN-SF approach and other recent data embedding schemes.

CRediT Author Statement

The authors confirm contribution to the paper as follows:
Conceptualization: Prasad C N and Suchithra R; **Methodology:** Prasad C N and Suchithra R; **Software:** Prasad C N; **Data Curation:** Prasad C N and Suchithra R; **Writing- Original Draft Preparation:** Prasad C N and Suchithra R; **Investigation:** Prasad C N and Suchithra R; **Supervision:** Suchithra R; **Validation:** Prasad C N and Suchithra R; **Writing- Reviewing and Editing:** Prasad C N and Suchithra R; All authors reviewed the results and approved the final version of the manuscript.

Data Availability

No data was used to support this study.

Conflicts of Interests

The author(s) declare(s) that they have no conflicts of interest.

Funding

No funding agency is associated with this research.

Competing Interests

There are no competing interests

References

[1]. M. Yesilyurt and Y. Yalman, "New approach for ensuring cloud computing security: using data hiding methods," Sādhanā, vol. 41, no. 11, pp. 1289–1298, Nov. 2016, doi: 10.1007/s12046-016-0558-8.

[2]. K. P. S. Shijin and Dhas. D. Edwin, "Simulated attack-based feature region selection for efficient digital image watermarking," 2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET), pp. 1128–1132, Mar. 2012, doi: 10.1109/icceet.2012.6203846.

[3]. M. Y. Valandar, M. J. Barani, P. Ayubi, and M. Aghazadeh, "An integer wavelet transforms image steganography method based on 3D sine chaotic map," Multimedia Tools and Applications, vol. 78, no. 8, pp. 9971–9989, Sep. 2018, doi: 10.1007/s11042-018-6584-2.

[4]. Sao, Nguyen Kim, Cao Thi Luyen, and Pham Van At. "Efficient reversible data hiding using block histogram shifting with invariant peak points." J. Inf. Hiding Multim. Signal Process. 13.1 (2022): 78-97.
[5]. A. Zulehner and R. Wille, "Make it reversible: Efficient embedding of non-reversible functions," Design, Automation &amp; Test in Europe Conference &amp; Exhibition (DATE), 2017, pp. 458–463, Mar. 2017, doi: 10.23919/date.2017.7927033.
[6]. X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High-Capacity Reversible Data Hiding in Encrypted Images by Patch-Level Sparse Representation," IEEE Transactions on Cybernetics, vol. 46, no. 5, pp. 1132–1143, May 2016, doi: 10.1109/tcyb.2015.2423678.
[7]. K. Chen and C.-C. Chang, "High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement," Journal of Visual Communication and Image Representation, vol. 58, pp. 334–344, Jan. 2019, doi: 10.1016/j.jvcir.2018.12.023.
[8]. K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible Data Hiding in Encrypted Images by Reserving Room Before Encryption," IEEE Transactions on Information Forensics and Security, vol. 8, no. 3, pp. 553–562, Mar. 2013, doi: 10.1109/tifs.2013.2248725.
[9]. Y. Puyang, Z. Yin, and Z. Qian, "Reversible Data Hiding in Encrypted Images with Two-MSB Prediction," 2018 IEEE International Workshop on Information Forensics and Security (WIFS), pp. 1–7, Dec. 2018, doi: 10.1109/wifs.2018.8630785.
[10]. Jun Tian, "Reversible data embedding using a difference expansion," IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 8, pp. 890–896, Aug. 2003, doi: 10.1109/tcsvt.2003.815962.
[11]. D. M. Thodi and J. J. Rodriguez, "Expansion Embedding Techniques for Reversible Watermarking," IEEE Transactions on Image Processing, vol. 16, no. 3, pp. 721–730, Mar. 2007, doi: 10.1109/tip.2006.891046.
[12]. B. Ou, X. Li, Y. Zhao, R. Ni, and Y.-Q. Shi, "Pairwise Prediction-Error Expansion for Efficient Reversible Data Hiding," IEEE Transactions on Image Processing, vol. 22, no. 12, pp. 5010–5021, Dec. 2013, doi: 10.1109/tip.2013.2281422.
[13]. Xiaolong Li, Bin Yang, and Tieyong Zeng, "Efficient Reversible Watermarking Based on Adaptive Prediction-Error Expansion and Pixel Selection," IEEE Transactions on Image Processing, vol. 20, no. 12, pp. 3524–3533, Dec. 2011, doi: 10.1109/tip.2011.2150233.
[14]. V. Sachnev, Hyoung Joong Kim, Jeho Nam, S. Suresh, and Yun Qing Shi, "Reversible Watermarking Algorithm Using Sorting and Prediction," IEEE Transactions on Circuits and Systems for Video Technology, vol. 19, no. 7, pp. 989–999, Jul. 2009, doi: 10.1109/tcsvt.2009.2020257.
[15]. W. Wang, J. Ye, T. Wang, and W. Wang, "A high-capacity reversible data hiding scheme based on right-left shift," Signal Processing, vol. 150, pp. 102–115, Sep. 2018, doi: 10.1016/j.sigpro.2018.04.008.
[16]. Z. Fu, X. Chai, Z. Tang, X. He, Z. Gan, and G. Cao, "Adaptive embedding combining LBE and IBBE for high-capacity reversible data hiding in encrypted images," Signal Processing, vol. 216, p. 109299, Mar. 2024, doi: 10.1016/j.sigpro.2023.109299.
[17]. H. Gao, X. Zhang, and T. Gao, "Hierarchical reversible data hiding in encrypted images based on multiple linear regressions and multiple bits prediction," Multimedia Tools and Applications, vol. 83, no. 3, pp. 8757–8783, Jun. 2023, doi: 10.1007/s11042-023-15939-0.
[18]. W. He and Z. Cai, "An Insight into Pixel Value Ordering Prediction Based Prediction-error Expansion," IEEE Transactions on Information Forensics and Security, pp. 1–1, 2020, doi: 10.1109/tifs.2020.3002377.
[19]. H. Wang et al., "A high-precision arrhythmia classification method based on dual fully connected neural network," Biomedical Signal Processing and Control, vol. 58, p. 101874, Apr. 2020, doi: 10.1016/j.bspc.2020.101874.
[20]. J. Naren and A. R. Babu, "EEG stress classification based on Doppler spectral features for ensemble 1D-CNN with LCL activation function," Journal of King Saud University - Computer and Information Sciences, vol. 36, no. 4, p. 102013, Apr. 2024, doi: 10.1016/j.jksuci.2024.102013.
[21]. P. Bas, T. Filler, and T. Pevný, "" Break Our Steganographic System": The Ins and Outs of Organizing BOSS," Information Hiding, pp. 5970, 2011, doi: 10.1007/978-3-642-24178-9_5.