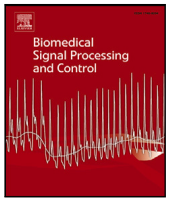
Biomedical Signal Processing and Control 93 (2024) 106209

Contents lists available at ScienceDirect

Biomedical Signal Processing and Control

journal homepage: www.elsevier.com/locate/bspc



Reversible data hiding in encrypted image with secure multi-party for telemedicine applications☆

Lingfeng Qu, Mohan Li *, Peng Chen

Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China

ARTICLE INFO

Keywords:
Reversible data hiding
Telemedicine applications
Secret sharing
Encrypted image

ABSTRACT

Privacy protection of electronic patient information (EPI) is a crucial concern in the telemedicine applications, especially when it comes to communication between patients and doctors. In this paper, we propose a high-capacity reversible data hiding (RDH) algorithm combined with secure multi-party computation for telemedicine applications. The pixel values within a 3 × 3-size image block are divided into embedded pixels (EPs) and sampled pixels (SPs), secret data is embedded into the prediction errors of EPs and SPs in two stages. In the first stage, two edge servers predict EPs using the four SPs in the block to obtain the prediction error of EPs. The secret data is embedded into EPs through addition operation and histogram shift method. In the second stage, the prediction error of SPs is calculated as the difference between EPs and SPs, and the embedding of secret data follows the same method as used in the first stage. The experimental results demonstrate that the proposed algorithm achieves a higher embedding rate compared to classical and state-of-the-art algorithms, with an average embedding rate of 0.47bpp on the UCID dataset. Furthermore, the proposed algorithm exhibits robustness against existing attacks.

1. Introduction

Reversible data hiding(RDH) algorithm has emerged as a prominent research area in multimedia information security [1]. One of its key attributes is the ability to extract confidential data at the decoding end without causing any distortion and reconstructing the original carrier without any loss of quality [2]. This feature is particularly crucial in certain domains, such as military, telemedicine, and legal forensics, where preserving the integrity of the original data is of utmost importance.

After almost two decades of research, RDH algorithm can be classified into three main categories: (1) Lossless compression based RDH algorithm. The first lossless compression-based RDH algorithm was proposed by Barton [3], and later, Celik et al. [4] enhanced the algorithm's compression efficiency. However, RDH algorithms based on lossless compression have certain limitations, such as low embedding capacity and significant distortion, which have hindered their adoption in practical applications. (2) Difference expansion(DE) based RDH algorithm [5–8]. The secret data is embedded into the carrier image in a reversible manner by utilizing the mean and difference calculation between two adjacent pixels. To enhance the embedding capacity of the DE algorithm, researchers proposed a prediction error expansion technique (PEE) [9], which embeds secret data into the expanded prediction error to obtain higher embedding capacity and lower embedding distortion. (3) Histogram Shift (HS) based RDH algorithm [10–12]. HS is a RDH algorithm proposed by Ni et al. [10] that involves shifting the pixel histogram to embed secret data. Typically, HS is used in conjunction with PEE technology, known as PEE-HS [11,12].

In the realm of cloud computing, ensuring reliable storage and secure transmission of electronic patient information (EPI) is crucial. To meet users' needs for personal data privacy protection, researchers have explored the use of reversible data hiding (RDH) algorithm for encrypted images (RDH-EI). Based on the timing of data embedding, the existing RDH-EI can be classified into two categories: vacating room before encryption (VRBE) [13–19] and vacating room after encryption (VRAE) [20–26]. For VRBE-based RDH-EI, users vacate the room from original image before encrypting and embedding the data. The VRBE based RDH-EI algorithm boasts a high embedding capacity due to its efficient utilization of the redundancy present in the original pixels. Ma et al. [13] suggested employing HS technology to create room within the image before encryption. Xu et al. [14] applied the prediction error histogram shift technique to the encrypted domain, embedding data through the HS method. To enhance the algorithm's embedding capacity even further, Puteaux et al. [15] proposed a RDH-EI algorithm based on the most significant bitplane(MSB) predictive coding, with an

☆ This work was supported by the National Natural Science Foundation of China (NSFC) under Grants 62272119, U20B2046.

* Corresponding author.
E-mail addresses: qlf@gzhu.edu.cn (L. Qu), limohan@gzhu.edu.cn (M. Li), ChenPeng@gzhu.edu (P. Chen).

https://doi.org/10.1016/j.bspc.2024.106209
Received 3 October 2023; Received in revised form 6 February 2024; Accepted 9 March 2024
Available online 13 March 2024

embedding rate close to 1 bit per pixel (bpp). Yin et al. [16] introduced an advanced high capacity RDH-EI scheme that utilizes multiple MSB prediction and Huffman coding. Recently, Chen et al. [17] proposed an RDH-EI algorithm based on multiple MSB prediction and bit plane compression, which has an embedding capacity close to 3 bpp. Although VRBE algorithm has high embedding capacity, there are many limitations in practical applications. On the one hand, preprocessing the original image increases the computational burden of the content owner. On the other hand, in order to ensure the reversibility of the algorithm, auxiliary information may be generated during image preprocessing, and the transmission of auxiliary information may pose security risks [20].

The RDH-EI algorithm, which is based on the VRAE framework, enables users to perform encryption operations on original images without requiring any preprocessing. This feature can significantly reduce the computational burden on users. However, it is important to note that encryption maximizes image entropy, making it challenging to embed any data in the encrypted images. In an effort to increase the embedding capacity of the VRAE-based RDH-EI algorithm, researchers need to involve special encryption schemes. Stream cipher XOR encryption [21,22], Block Permutation and Co-XOR (BPCX) encryption [23,24], Block Permutation and Co-Modulation (BPCM) encryption [25,26] are commonly used in VRAE-based RDH-EI. Zhang [21] proposed the first RDH-EI system based on stream cipher XOR encryption. By inverting the LSB of pixels in an image block to embed 1-bit information, secret data is extracted based on the fluctuation function. Due to the inability of the fluctuation function to accurately determine whether the LSB of an image block is flipped, the extracted data and image restoration are both lossy. Subsequently, Hong et al. [22] improved Zhang's algorithm based on block edge matching, achieving higher recovery quality. The RDH-EI algorithm employing stream cipher XOR encryption exhibits drawbacks such as diminished security and limited embedding capacity.

The existing VRAE based RDH-EI mostly uses BPCX encryption and BPCM encryption. Huang et al. [23] designed a general algorithm framework for RDH-EI based on BPCX encryption. In this framework, RDH algorithms are independent of image encryption algorithms. Therefore, most of the previously proposed DE and PEE-HS based RDH schemes can be directly completed in the encryption domain. Ge et al. [24] proposed a RDH algorithm based on BPCX encryption. This algorithm selects peak pixels from each encrypted block, and embeds data in the peak pixels of each block based on the HS method. Yi et al. [25] proposed a parametric binary tree labeling method based on BPCM encryption to encode prediction errors. Recently, Yu et al. [26] proposed an adaptive differential recovery RDH-EI algorithm based on BPCM encryption, which encodes and compresses the difference information of an image and replaces the remaining bits of the difference to achieve data hiding. The maximum embedding capacity is close to 3 bpp.

By designing encryption schemes, the embedding capacity of the VRAE based RDH-EI algorithm has been equivalent to that of the VRBE based RDH-EI algorithm. However, stream cipher XOR encryption, BPCX encryption, and BPCM encryption have been pointed out by researchers to have security risks [27–29], moreover, the current RDH-EI schemes are not well-suited for multi-party applications. In multi-party copyright protection applications, the embedding of secret data requires the cooperation of multiple servers, which can enhance the security of the algorithm. Recently, Xiong et al. [30] designed an RDH-EI algorithm for multi-party secure computing based on lightweight cryptography, which divides encrypted sharing into 2×2-size non overlapping image blocks, and embeds 1-bit into a shared image block using the PEE-HS method. Under the first level embedding condition, each image block is only embedded by 1 bit, which limits the embedding capacity of the algorithm. In order to further improve the embedding capacity, we propose a large capacity RDH-EI algorithm for multi-party secure computing combined with telemedicine applications [30]. The contributions of this paper are as follows:

(i) We propose a multi party secure computing framework for telemedicine applications. The additional data is embedded by two different edge servers in collaboration, improving the security of the RDH-EI algorithm in practical applications.
(ii) We propose a two-stage strategy for prediction error and embedding data. The pixels in the encrypted image block are categorized as embedded pixels (EPs) and sampled pixels (SPs). The EPs and SPs undergo two-stage prediction, with additional data embedded at the peak of the prediction error. This approach enhances the embedding capacity of the algorithm.
(iii) Our experimental results demonstrate that the proposed algorithm exhibits a high level of resistance against existing ciphertext-only and known-plaintext attacks, as compared to the comparison method. Furthermore, the algorithm demonstrates a superior embedding capacity for both natural and medical images.

The remaining parts of the paper are arranged as follows: Section 2 is the problem formulation, Section 3 is the preliminaries, Section 4 is the proposed algorithm, Section 5 is the experimental analysis.

2. Problem formulation

Within this section, we initially present an introduction and comprehensive analysis of the security risks associated with current RDH-EI frameworks, and then propose the application of multi-party secure cloud storage based on RDH-EI in telemedicine scenarios.

2.1. Analysis of existing schemes

The goal of this paper is to solve the problem of user privacy data leakage caused by untrusted cloud in the RDH-EI privacy protection system. In existing frameworks, content owners (users) encrypt the image based on key_1 and upload the encrypted image to cloud for storage. The data hider in the cloud reversible hides data in the encrypted image based on key_2 without knowing the original image content, to obtain the marked encrypted image. When the legitimate receiver only owns key_1, the marked encrypted image can be directly decrypted to obtain a marked original image similar to the original image. When the legitimate receiver only owns key_2, data can be extracted from the marked encrypted image. When the receiver has both key_1 and key_2, not only data can be extracted, but also the original image can be restored. The existing RDH-EI system still has the following security issues:

(1) Balance between security and embedded capacity. In existing research, the cloud environment is considered an untrusted third party. The security of encryption algorithms is crucial to the success of image content protection, which is very important for the algorithm design of RDH-EI. The VRAE based RDH-EI algorithm achieves high embedding capacity by designing special encryption schemes such as stream cipher XOR, BPCX, and BPCM encryption [31–33], but these encryption schemes have been proven to pose security risks. How to design a secure encryption scheme while ensuring the embedded capacity has become one of the urgent issues for RDH-EI.

(2) Auxiliary information processing. In order to ensure the reversibility of the algorithm, the RDH-EI algorithm based on the VRBE framework will generate auxiliary information during the preprocessing of the original image. One way to process the auxiliary information is to directly transmit the auxiliary information to the cloud [29,34], and the other way is to embed the auxiliary information into the vacated room. Both strategies for processing auxiliary information have security risks [34], and the processing of auxiliary information limits the application of RDH-EI.

To address the security threats faced by existing RDH-EI algorithms and increase the embedding capacity, we proposes a secure RDH-EI scheme with high embedding capacity for multi-party computing. The content owner does not need to transmit any auxiliary information to the cloud, and image restoration and data extraction are independent and do not affect each other. These characteristics can improve the application range of the algorithm.

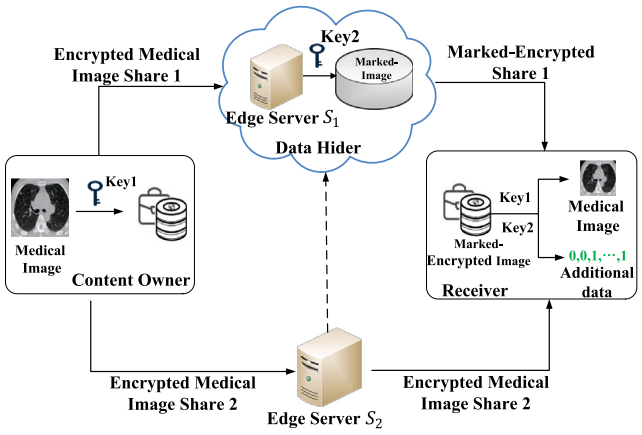


Fig. 1. RDH-EI algorithm for secure multi-party computing in telemedicine applications.

2.2. System architecture

This paper proposes an RDH-EI algorithm by applying lightweight multi-party encryption strategy [30] to medical image privacy protection scenarios. The system architecture of secure multi party computing applied to telemedicine is shown in Fig. 1. First, the content owner encrypts the original medical image X based on key_1 to obtain the encrypted image E. The encrypted image E is randomly divided into two parts, which are represented as encryptedshare1 E^1 and encrypted share2 E^2, respectively. Next, E^1 and E^2 are sent to the two edge servers S_1 and S_2, respectively. Employing a secure addition protocol, the two edge servers collaborate to seamlessly embed secret data (e.g., copyright details, patient data, etc.) into their respective shares. The legitimate receiver adds the marked encrypted share1 and encrypted share2 to obtain the marked encrypted image M. Depending on the secret key owned by the legitimate receiver, different operations such as secret data extraction and image restoration can be performed.

We embrace a semi-honest (also referred to as passive or honest but curious) security model. [35]. In other words, each edge server will fulfill the protocol content according to regulations, and may attempt to gather as much confidential information of interest as possible by accessing the encrypted data on hand. In addition, suppose that two edge servers, S_1 and S_2, are independent and do not collude. This means that one edge server will not disclose more information than protocol messages to another server, which is a practical assumption because two edge servers can be deployed and managed by two different or even competing service providers.

3. Preliminaries

This section provides an overview of Xiong et al. [30] proposed lightweight secret sharing encryption scheme. For a grayscale image X with a size of $W \times H$, $X = \{x(i,j)|i = 1,2,\dots,W, j = 1,2,\dots,H\}$, X is divided into N non overlapping image blocks, each image block has a size of 2×2. To break the correlation between image blocks, the image blocks are scrambled based on key_1 to obtain a preliminary encrypted image E. Block scrambling encryption only changes the position of pixels without changing their values, and the correlation between pixels within the block is preserved. Taking pixel$x(i,j)$ as an example, $x(i,j)$ can be divided into n least significant bits (LSBs) and (8-n) higher significant bits (HSBs) according to Eqs. (1).

$$x(i,j) = x_{HSB}(i,j) \times 2^n + x_{LSB}(i,j) \tag{1}$$

Here, the calculation of $x_{HSB}(i,j)$ and $x_{LSB}(i,j)$ can refer to the description in Ref. [30]. $x_{HSB}(i,j)$ represents the decimal value of (8-n) HSBs of pixel $x(i,j)$, $x_{LSB}(i,j)$ represents the decimal value of n LSBs of pixel $x(i,j)$. $x_{HSB}(i,j)$ and $x_{LSB}(i,j)$ are randomly split by the addition operation based on the encryption key, as shown in Eqs. (2) and (3)

$$x_{HSB}(i,j) = x^1_{HSB}(i,j) + x^2_{HSB}(i,j) \tag{2}$$

$$x_{LSB}(i,j) = x^1_{LSB}(i,j) + x^2_{LSB}(i,j) \tag{3}$$

By using Eqs. (4) and (5), the pixel value $x(i,j)$ is divided into two encryption shares $e_1(i,j)$ and $e_2(i,j)$,

$$e_1(i,j) = x^1_{HSB}(i,j) \times 2^n + x^1_{LSB}(i,j) \tag{4}$$

$$e_2(i,j) = x^2_{HSB}(i,j) \times 2^n + x^2_{LSB}(i,j) \tag{5}$$

Since the HSB and LSB of the pixel value $x(i,j)$ are randomly divided into two parts, the pixel value of the two encrypted shares $e_1(i,j)$ and $e_2(i,j)$ are independent of each other and are not equal to $x(i,j)$. In other words, through secret sharing and block scrambling, not only the value of image pixels is changed, but also the position of pixels is changed. Finally, the content owner sends the two encrypted shares $E^1 = \{E^1_i|i = 1,2,\dots,N\}$, $E^1_i = \{e_1(i,j)\}$ and $E^2 = \{E^2_i|i = 1,2,\dots,N\}$, $E^2_i = \{e_2(i,j)\}$ to the two edge servers, respectively. The relationship between the initial encrypted image E and the two shared images is:

$$E = E^1 + E^2 \tag{6}$$

Lightweight secret sharing schemes have advantages such as speed, security, and more importantly, retaining enough redundancy in the ciphertext can enhance the embedding capacity of the algorithm.

4. Proposed algorithm

As shown in Fig. 2. Firstly, the original image has undergone encryption using the lightweight secret sharing technique. In the data embedding stage, the prediction errors of the original image can be obtained by adding the prediction errors of the image blocks in share1 and share2. The secret data is embedded into the encrypted sharing through the prediction error expansion method. By employing an addition operation on the two indicated encrypted shares, it becomes feasible to acquire the marked encrypted image. Based on the secret key owned by the legitimate receiver, the marked encrypted image can be directly decrypted, and then the operation of extracting data and restoring the original image can be performed. Alternatively, first, the secret data can be extracted, followed by decrypting the encrypted image to retrieve the original image.

4.1. Lightweight secret sharing encryption

Unlike Xiong's encryption algorithm, which divides the image into 2×2 block size, in order to adapt to the data embedding strategy proposed in this paper, we divide the image X into N image blocks, each image block has a size of 3×3. In addition, based on the experimental performance and results in literature [30], we fixed the value of n (the number of least significant bit planes) to 3 in this paper.

In Fig. 3, we take medical image as an example to show the two shares of encrypted images (E^1 and E^2). It can be seen that the changes we made to Xiong's encryption algorithm will not affect the visual effect of encryption. The encrypted sharing presents random noise distribution, which can protect image content information and has good security.

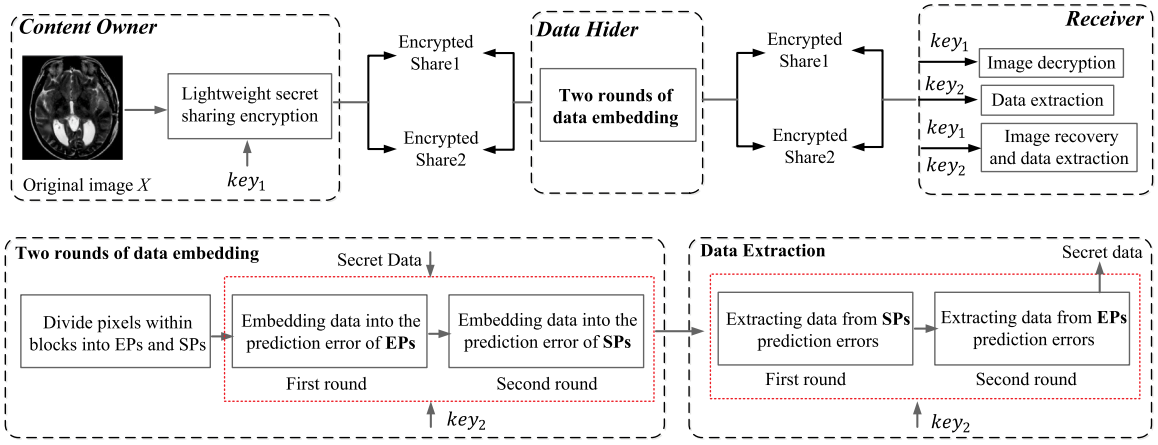


Fig. 2. Algorithm framework.

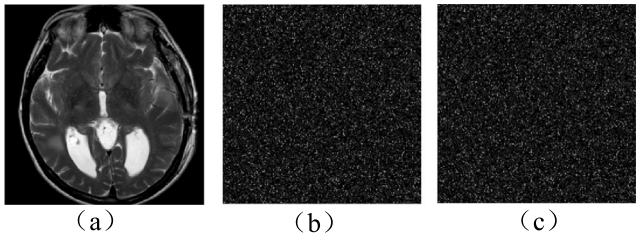


Fig. 3. Original medical image and two encrypted shares((a) Original medical image, (b) Encrypted share1, (c) Encrypted share2).

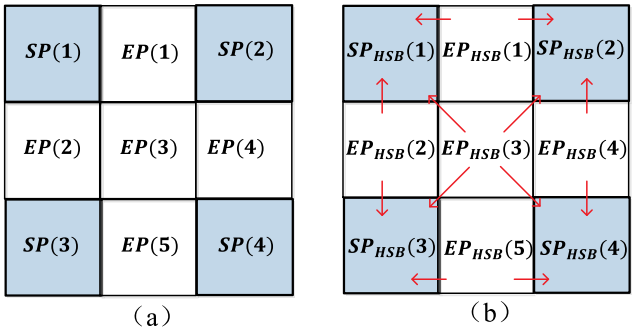


Fig. 4. Pixel value division and prediction relationship in blocks.

4.2. Two-stage data embedding strategy

The secret data waiting to be embedded is first encrypted based on key_1 to increase the security. In order to increase the embedding capacity, we adopt a method based on high significant bit PEE-HS to embed additional data in two stages. Taking encrypted image E as an example, the pixel values in the kth ($k = 1, 2, \ldots, N$) image block with a size of 3×3 are divided into two categories: sampling pixels (SPs) $SP_k = \{SP(i) | i = 1, 2, 3, 4\}$ and embeddable pixels (EPs) $EP_k = \{EP(i) | i = 1, 2, \ldots, 5\}$. The distribution of sampling pixels and embedded pixels is shown in Fig. 4(a). The two stages of data embedding are embedding additional data in EPs and embedding additional data in SPs.

4.2.1. Data embedding in EPs

In this stage, the HSBs of the EPs is predicted by the HSBs of the SPs in the block, and the prediction value $\overline{EP}_{HSB}(i)$ is obtained. The HSBs prediction error of the EPs in the kth image block is expressed as $D_{HSB}(k)$, $D_{HSB}(k) = \{d_{HSB}(i) | i = 1, 2, \ldots, 5\}$.

$$d_{HSB}(i) = EP_{HSB}(i) - \overline{EP}_{HSB}(i) \tag{7}$$

In the pixel prediction process, for the pixel $EP(3)$ located at the center of the image block, we use the average value of the four SPs's HSBs as the prediction value of $EP_{HSB}(3)$. For pixels at non central locations, the average value of two adjacent SPs's HSBs is used as the predicted value of $EP_{HSB}(i), (i = 1, 2, 4, 5)$, as shown in Fig. 4(b). The calculation process of $\overline{EP}_{HSB}(i)$ is shown in Eqs. (8) to (9).

$$\overline{EP}_{HSB}(3) = \frac{1}{4} \times \left(\sum_{i=1}^{4} SP_{HSB}(i)\right) \tag{8}$$

$$\begin{cases} \overline{EP}_{HSB}(1) = \frac{1}{2} \times (SP_{HSB}(1) + SP_{HSB}(2)) \\ \overline{EP}_{HSB}(2) = \frac{1}{2} \times (SP_{HSB}(1) + SP_{HSB}(3)) \\ \overline{EP}_{HSB}(4) = \frac{1}{2} \times (SP_{HSB}(2) + SP_{HSB}(4)) \\ \overline{EP}_{HSB}(5) = \frac{1}{2} \times (SP_{HSB}(3) + SP_{HSB}(4)) \end{cases} \tag{9}$$

In the proposed algorithm, the data hider needs to calculate $d_{HSB}(i)$ from two shares, that is, two edge servers calculate a portion of $d_{HSB}(i)$ respectively. Taking encryption share 1 as an example, we express the HSBs prediction error of EPs in the kth encrypted image block as $D^1_{HSB}(k)$, $D^1_{HSB}(k) = \{d^1_{HSB}(i) | i = 1, 2, \ldots, 5\}$. The prediction error of $EP^1_{HSB}(3)$ at the center position is calculated as,

$$d^1_{HSB}(3) = 4 \times EP^1_{HSB}(3) - \sum_{i=1}^{4} SP^1_{HSB}(i) \tag{10}$$

The prediction error of $EP^1_{HSB}(i), (i = 1, 2, 4, 5)$ is calculated as,

$$\begin{cases} d^1_{HSB}(1) = 2 \times EP^1_{HSB}(1) - (SP^1_{HSB}(1) + SP^1_{HSB}(2)) \\ d^1_{HSB}(2) = 2 \times EP^1_{HSB}(2) - (SP^1_{HSB}(1) + SP^1_{HSB}(3)) \\ d^1_{HSB}(4) = 2 \times EP^1_{HSB}(4) - (SP^1_{HSB}(2) + SP^1_{HSB}(4)) \\ d^1_{HSB}(5) = 2 \times EP^1_{HSB}(5) - (SP^1_{HSB}(3) + SP^1_{HSB}(4)) \end{cases} \tag{11}$$

Similar to the calculation method of prediction error for encryption share 1, the prediction error for image block EPs in encryption share 2 is expressed as: $D^2_{HSB}(k)$, $D^2_{HSB}(k) = \{d^2_{HSB}(i) | i = 1, 2, \ldots, 5\}$. After the data hider in the edge server S_1 receives the prediction error $d^2_{HSB}(i)$ sent by the edge server S_2, the prediction error $d_{HSB}(i)$ can be calculated by Eqs. (12) to (13).

$$d_{HSB}(3) = \left\lfloor \frac{1}{4} \times (d^1_{HSB}(3) + d^2_{HSB}(3)) \right\rfloor \tag{12}$$

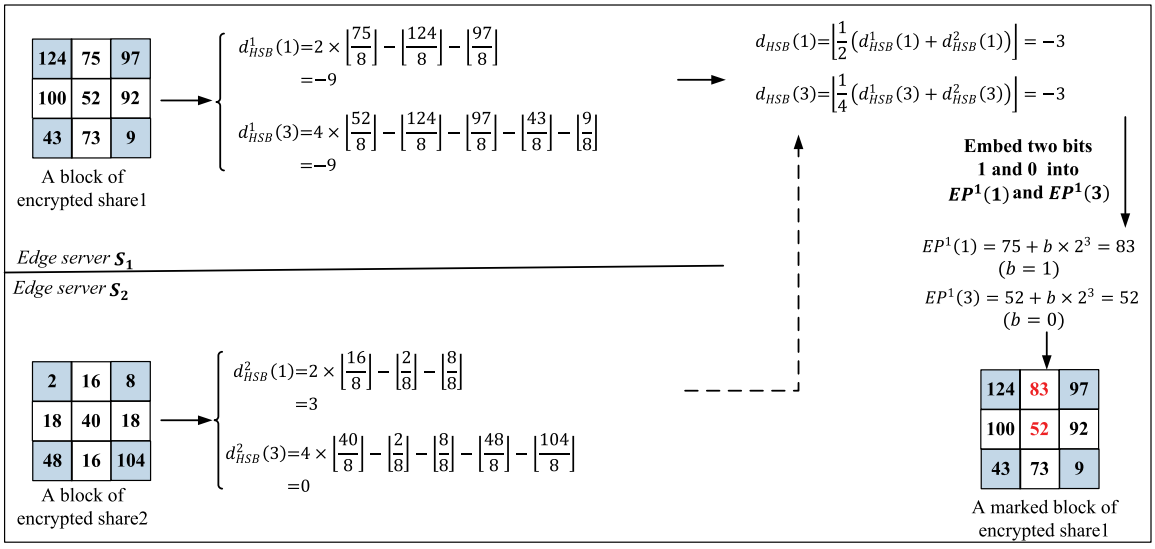


Fig. 5. An example of embedding data in EPs.

$$d_{HSB}(i)=\left\lfloor\frac{1}{2}\times(d^1_{HSB}(i)+d^2_{HSB}(i))\right\rfloor, i\neq 3 \tag{13}$$

we can obtained the set of predictive errors $ED=\{D_{HSB}(k)|k=1,2,\ldots,N\}$. Using the predictive errors ED derived from the HSBs, the secret data can be embedded into shares through the combination of PEE and HS, as illustrated in Eq. (14).

$$\widehat{EP}^{\theta}(i)=\begin{cases} EP^{\theta}(i)+b\times 2^n & d_{HSB}(i)=M_E \\ EP^{\theta}(i)+2^n & d_{HSB}(i)>M_E \\ EP^{\theta}(i) & d_{HSB}(i)<M_E \end{cases} \tag{14}$$

In the context of the equation, $b\in\{0,1\}$ denotes a single bit of the secret data, while $\theta\in\{1,2\}$ indicates the share number. M_E is defined as the value $M_E=\arg\max(h_E(d_{HSB}(i)))$, where $h_E(x)$ represents the count of occurrences when the prediction-error values in the sequence D are equal to x. It should be noted that the range of pixel values is [0, 255], so the location of overflow pixels caused by histogram shifting needs to be recorded. The position of the overflow pixels are compressed and embedded as secret data into the encrypted share.

In Fig. 5, the data embedding process is illustrated by taking $EP^{\theta}(1)$ and $EP^{\theta}(3)$ ($\theta\in\{1,2\}$) in a block of encrypted share 1 and share 2 as an example. First, calculate $d^1_{HSB}(1)$, $d^1_{HSB}(3)$ and $d^2_{HSB}(1)$, $d^2_{HSB}(3)$ according to Eqs. (10) and (11), respectively. According to Eqs. (12) and (13), the data hider can calculate $d_{HSB}(1)$ and $d_{HSB}(3)$. In Fig. 5, we assume that additional data is embedded in encrypted share 1, and $M_E=-3$. From Eq. (14), it can be seen that both $EP^1(1)$ and $EP^1(3)$ can be used for data hiding. We embed bit '1' and bit '0' in these two EPs to obtain the marked encrypted share 1.

4.2.2. Data embedding in SPs

For any encrypted block E_k, $k=1,2,\ldots,N$, we express the EPs embedded with additional data as: $\widehat{EP}_k=\{\widehat{EP}(i)|i=1,2,\ldots,5\}$ (Fig. 6(a)). In the second stage of data embedding, we use the difference between the HSBs of SPs $SP(i)$ in the block and the HSBs of EPs $\widehat{EP}(i)$ as the prediction error (Fig. 6(b)).

The HSBs prediction error of SPs in the encrypted image block E_k is expressed as $SD_{HSB}(k)$, $SD_{HSB}(k)=\{sd_{HSB}(i)|i=1,2,3,4\}$.

$$sd_{HSB}(i)=SP_{HSB}(i)-\widehat{EP}_{HSB}(i) \tag{15}$$

Analogous to the data embedding procedure in the initial stage, the data concealer needs to calculate $sd_{HSB}(i)$ from two shares, and each of the two edge servers calculates a portion of the $sd_{HSB}(i)$. Taking encryption share E^1 as an example, the HSBs prediction error of the

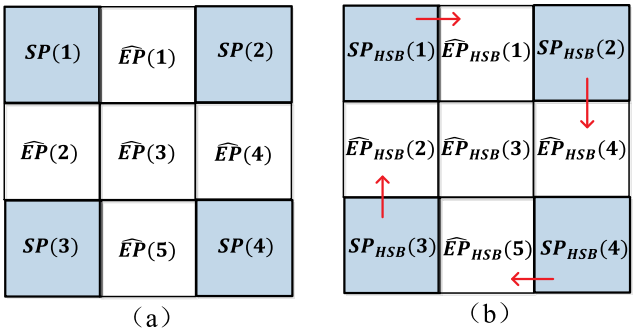


Fig. 6. Pixel value division and prediction relationship in block after EPs embeds additional data.

EPs in the kth encrypted image block E^1_k can be obtained through Eq. (16), the prediction error of $SP^1_{HSB}(i)$ is expressed as $SD^1_{HSB}(k)$, $SD^1_{HSB}(k)=\{sd^1_{HSB}(i)|i=1,2,3,4\}$.

$$sd^1_{HSB}(i)=SP^1_{HSB}(i)-\widehat{EP}^1_{HSB}(i) \tag{16}$$

In the encryption share E^2, the HSBs prediction error of SPs in E^2_k is expressed as $SD^2_{HSB}(k)$, $SD^2_{HSB}(k)=\{sd^2_{HSB}(i)|i=1,2,3,4\}$. After receiving the prediction error $sd^2_{HSB}(i)$ sent by the edge server S_2, the data hider in S_1 can calculate the prediction error$sd_{HSB}(i)$ using Eq. (17).

$$sd_{HSB}(i)=sd^1_{HSB}(i)+sd^2_{HSB}(i) \tag{17}$$

The collection of predictive errors, denoted as $SD=\{SD_{HSB}(k)|k=1,2,\ldots,N\}$. The secret data can be embedded into $SP^{\theta}(i)$ ($\theta\in\{1,2\}$) by Eq. (18).

$$\widehat{SP}^{\theta}(i)=\begin{cases} SP^{\theta}(i)+b_2\times 2^n & sd_{HSB}(i)=\widetilde{M}_E \\ SP^{\theta}(i)+2^n & sd_{HSB}(i)>\widetilde{M}_E \\ SP^{\theta}(i) & sd_{HSB}(i)<\widetilde{M}_E \end{cases} \tag{18}$$

Here, $b_2\in\{0,1\}$ is the embedded data, $\widetilde{M}_E$ and M_E have the same meaning. The processing of overflow pixels is the same as the first stage.

In Fig. 7, the data embedding process is illustrated by taking $SP^{\theta}(1)$ and $SP^{\theta}(2)$ ($\theta\in\{1,2\}$) in a block of encrypted share 1 and share 2 as an example. First, calculate $sd^1_{HSB}(1)$, $sd^1_{HSB}(2)$ and $sd^2_{HSB}(1)$, $sd^2_{HSB}(2)$ according to Eqs. (16). According to Eqs. (17), the data hider can

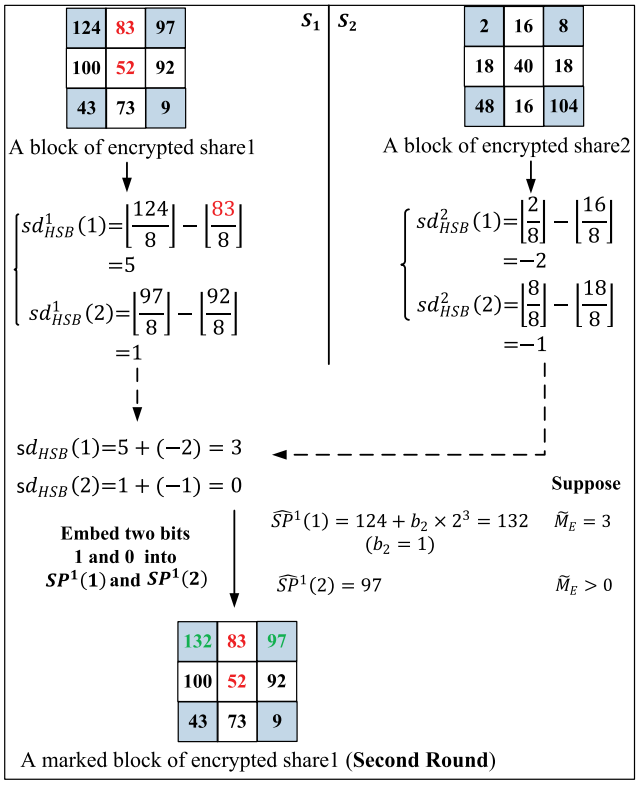


Fig. 7. An example of embedding data in SPs.

calculate $sd_{HSB}(1)$ and $sd_{HSB}(2)$. In Fig. 7, we assume that additional data is embedded in encrypted share 1, and $\widetilde{M}_E=3$. It can be seen that only $SP^1(1)$ can be used for data hiding. We embed bit '1' in $SP^1(1)$ to obtain the marked encrypted share 1.

Secret data is embedded into the prediction error peaks of the $EP_{HSB}(i)$ and the $SP_{HSB}(i)$, respectively. Therefore, the maximum embedding capacity $C_{\max}$ (bit) is,

$$C_{\max}=M_E+\widetilde{M}_E-\varphi \tag{19}$$

φ is the position information of the overflow pixel, for images with a size of $W\times H$, the maximum embedding rate $ER_{\max}$ (bpp) is,

$$ER_{\max}=C_{\max}/(W\times H) \tag{20}$$

4.3. Data extraction and image recovery

According to the key owned by the legitimate receiver, there are two possible scenarios for data extraction and image restoration. (1) Case1: The receiver first extracts the data, then decrypts the image and restores the original image. (2) Case2: Directly decrypt the image to obtain the marked decrypted image, and then extract the data and restore the original image.

4.3.1. Case1

The data extraction process is the inverse process of data embedding. The receiver first adds the marked encrypted share 1 and encrypted share 2 to obtain the marked encrypted image M. Divide M into N non overlapping image blocks of 3×3 sizes, and divide the pixel values in the block into SPs and embedded pixel EPs according to Fig. 3(a). First, extract the additional data embedded in $SP(i)$ and restore the value of $SP(i)$. Calculate the prediction error for each block according to Eqs. (16) and (17). The extraction process of the data

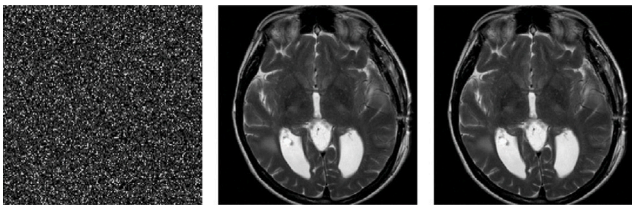


Fig. 8. Visual effects of the decrypted images. ((a) Marked encrypted image, (b) decrypted image, and (c) restored original image.).

embedded in the second stage is shown in Eq. (21). Overflow pixels will be skipped during the data extraction process.

$$b_2=\begin{cases}0 & sd_{HSB}(i)=\widetilde{M}_E\\ 1 & sd_{HSB}(i)=\widetilde{M}_E+1\end{cases} \tag{21}$$

The pixel $SP(i)$ can be restored by Eq. (22),

$$SP(i)=\begin{cases}\widehat{SP}(i)-2^n & sd_{HSB}(i)>\widetilde{M}_E\\ \widehat{SP}(i) & sd_{HSB}(i)\le\widetilde{M}_E\end{cases} \tag{22}$$

Then, extract additional data b embedded in $EP(i)$ using the same method. Through the above data extraction and pixel recovery process, the receiver can obtain a block scrambling encrypted image E. The original secret data can be obtained based on key_2. Finally, image X is obtained by inverting E based on key_1.

4.3.2. Case2

Using key_1, the receiver can perform a direct decryption on the marked encrypted image, resulting in the decrypted image O, which exhibits visual similarity to the X. Then, secret data is extracted from O according to Eqs. (21) to (22) and the original image X is restored. Because the secret data is embedded in the image after block scrambling, the secret data extracted in Case 2 is scrambled. Therefore, it is necessary to reverse scramble the additional data based on key_1 and decrypt it based on $key_2$to obtain the original secret data. Taking Fig. 8 as an example, Fig. 8(a) shows the marked encrypted image M obtained by adding two shares, Fig. 8(b) shows the image O obtained by directly decrypting M, and Fig. 8(c) shows the restored image X.

The similarity between the image O and the image X is measured using Peak Signal-to-Noise Ratio (PSNR). For a grayscale image (pixel values encoded by 8 bits) with $W\times H$ size, the pixel value range of the image is [0, 255]. The calculation of PSNR is shown in Eq. (23).

$$PSNR=10\times\log_{10}\frac{255^2}{MSE} \tag{23}$$

The computation involves evaluating the mean square error (MSE) between the two images.

$$MSE=\frac{1}{WH}\times\sum_{i=0}^{W-1}\sum_{j=0}^{H-1}(X(i,j)-O(i,j))^2 \tag{24}$$

Generally speaking, when the PSNR value exceeds 30 dB, it becomes challenging for human vision to discern any differences in detail between two images. Fig. 8 illustrates that the PSNR value between (b) and (c) is 34.17 dB, which implies that it would be difficult for human vision to detect any intricate variations between these two images.

5. Experimental results and analysis

We select four natural images (as shown in Fig. 9(a) to (d)) and medical images (as shown in Fig. 9(e) to (h)) with different texture complexity as test images, each with a size of 512×512. The names of Fig. 9(a) to (d) are: 'Lena', 'Pepper', 'Lake', and 'Baboon', and the names of (e) to (h) are: 'Med1', 'Med2', 'Med3', and 'Med4'. The

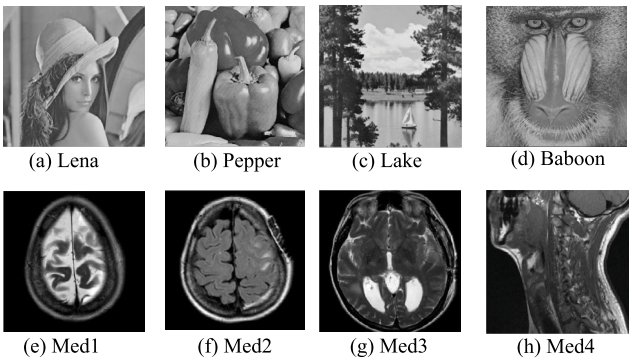


Fig. 9. Test images.

performance evaluation of the algorithm encompasses its embedding capacity, visual quality of marked image, and the security of encrypted images. Furthermore, a comparative analysis is conducted between the proposed scheme and classic as well as state-of-the-art schemes.

5.1. Algorithm performance analysis

In this section, we initially analyze the algorithm's embedding capacity and the visual quality of the decrypted image. Subsequently, we examine the impact of parameter selection on the overall algorithm performance.

5.1.1. Embedding capacity and the visual quality

One of the crucial metrics for assessing the RDH-EI algorithm is its maximum embedding capacity. To exclude the impact of image size on embedding capacity, the embedding rate (ER) is usually used to represent the embedding capacity of the algorithm. We analyze the ER of the algorithm and the similarity between the image O and X. In Table 1, we tested the maximum ER of the test images, the PSNR of the decrypted images and the structural similarity(SSIM) of the decrypted image, respectively.

Unlike PSNR, SSIM places more emphasis on the perception of images by the human eye. The value range of SSIM is between [−1, 1], and the closer the value is to 1, the more similar the structure of the two images is. Generally speaking, when $0.9 \leq SSIM < 1$, it indicates very high image similarity. When $0.7 \leq SSIM < 0.9$, it indicates high image similarity, but there may be some slight differences that may not be significant under normal observation. When $SSIM < 0.7$, it indicates low image similarity. The results in Table 1 show that for the test images, the SSIM values of the decrypted images obtained by the proposed algorithm are all higher than 0.7, indicating that the visual quality of the decrypted images is satisfactory. In addition, we compared the experimental results of the proposed algorithm with classic and latest algorithms [14,21,22,24,30–33].

Among them, early algorithms such as Zhang et al. [21] and Hong et al. [22] embed data by flipping the least significant bit (LSB) of pixels. Because only 1 bit data can be embedded in each image block, these algorithms have a low embedding rate. In addition, due to the inability of the fluctuation function to accurately determine whether the LSB of a pixel is inverted, the image cannot be completely reversible restored. In Table 1, we tested the ER and PSNR of the algorithms proposed by Zhang and Hong et al. at 8 × 8 block sizes. For the test image, the maximum ER of Zhang and Hong's algorithm is 0.015, and the maximum PSNR is 38.13 dB.

To enhance the embedding rate, Xu et al. [14] divided image pixels into two categories: 1/4 sampled pixels and 3/4 non sampled pixels, and obtained prediction errors by using sampled pixels to predict non sampled pixels. Combining stream cipher XOR encryption and difference encryption methods to obtain encrypted images. Due

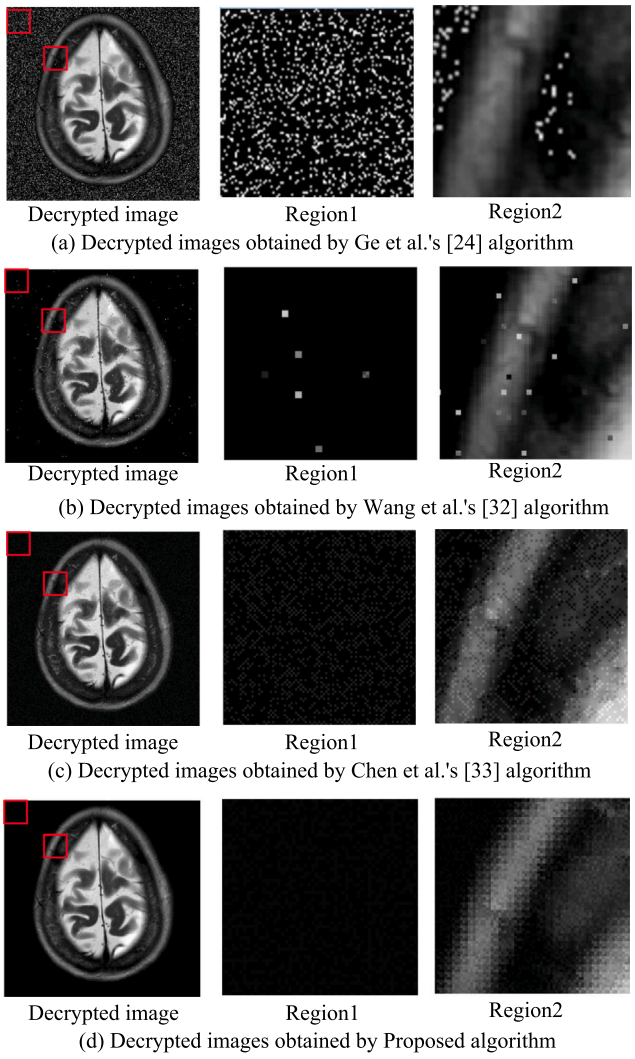


Fig. 10. Visual quality comparison of decrypted images.

to the preservation of a fraction of the prediction error from the original image in the encrypted image, the algorithm employs the histogram shift method to embed secret data into the prediction error of non-sampled pixels. Compared to Zhang and Hong's algorithms, Xu's algorithm achieves better PSNR and higher embedding capacity. However, due to the prediction error only for non sampled pixels, this limits the maximum ER of the algorithm to not exceed 0.75bpp. In addition, the encrypted image generated by the difference encryption method proposed by Xu et al. discloses the contour information of the original image, which poses a security risk [14]. In Table 1, we tested the performance of the Xu's algorithm at a threshold of [−2,1]. Xu's algorithm has the highest ER for the test image 'Med1', which is 0.51bpp, and the PSNR of the decrypted image is 46.92 dB.

Recently, Ge et al. [24] proposed a multilevel embedded RDH-EI algorithm. The encrypted images are generated based on the BPCX encryption method, then randomly select two peaks from each encrypted image block, and embed data into the peaks of each encrypted image block through histogram shift method. Because XOR encryption breaks the correlation between pixels, the peaks in each image block are very low, and the randomly selected peaks are usually not true peaks, resulting in a low embedding rate of the algorithm. With the aim of enhancing the quality of decrypted images, Ge et al. only perform block XOR encryption on the MSB of the image. To ensure a fair comparison, Table 1 only incorporates one level of embedding for the algorithm

Table 1
Maximum embedding rate and PSNR,SSIM for test images under different algorithms.

Test images	Parameters	Methods								
		Zhang [21]	Hong [22]	Xu [14]	Ge [24]	Xiong [30]	Rupali [31]	Wang [32]	Chen [33]	Our
Lena	ER(bpp)	0.015	0.016	0.30	0.12	0.12	0.25	0.49	0.30	0.46
	PSNR(dB)	37.92	37.93	44.69	51.24	40.25	40.34	26.09	24.67	34.17
	SSIM	0.92	0.92	0.98	0.99	0.95	0.95	0.86	0.48	0.89
Pepper	ER(bpp)	0.015	0.015	0.24	0.09	0.11	0.25	0.49	0.27	0.40
	PSNR(dB)	37.91	37.90	44.39	50.72	40.36	40.30	24.52	24.66	34.05
	SSIM	0.93	0.92	0.98	0.99	0.96	0.94	0.81	0.49	0.90
Lake	ER(bpp)	0.015	0.015	0.21	0.08	0.10	0.25	0.49	0.25	0.35
	PSNR(dB)	37.91	37.91	44.24	50.67	40.14	40.34	22.74	24.69	33.95
	SSIM	0.94	0.94	0.98	0.99	0.97	0.95	0.80	0.57	0.92
Baboon	ER(bpp)	0.014	0.014	0.12	0.03	0.06	0.25	0.49	0.10	0.25
	PSNR(dB)	37.92	37.92	43.87	50.73	39.78	40.31	20.65	24.69	33.73
	SSIM	0.97	0.97	0.99	0.99	0.98	0.95	0.77	0.74	0.95
Med1	ER(bpp)	0.011	0.012	0.51	0.09	0.19	0.25	0.49	0.30	0.63
	PSNR(dB)	35.73	35.74	46.92	**12.11**	39.59	40.34	24.7	24.88	34.24
	SSIM	0.68	0.64	0.98	**0.48**	0.87	0.83	0.89	0.31	0.73
Med2	ER(bpp)	0.012	0.012	0.50	0.09	0.18	0.25	0.49	0.32	0.62
	PSNR(dB)	35.96	35.97	46.73	**12.63**	39.57	40.34	26.28	24.62	34.23
	SSIM	0.71	0.68	0.98	**0.54**	0.88	0.85	0.89	0.35	0.76
Med3	ER(bpp)	0.013	0.013	0.43	0.10	0.17	0.25	0.49	0.30	0.57
	PSNR(dB)	36.44	36.45	45.93	**14.04**	39.68	40.33	24.71	24.88	34.22
	SSIM	0.79	0.76	0.98	**0.65**	0.91	0.89	0.87	0.42	0.75
Med4	ER(bpp)	0.015	0.015	0.38	0.12	0.15	0.25	0.49	0.32	0.54
	PSNR(dB)	38.1	38.13	45.13	44.5	40.03	40.34	26.32	24.62	34.29
	SSIM	0.93	0.93	0.98	0.99	0.95	0.95	0.88	0.49	0.89

proposed by Ge et al. since the proposed algorithm also performs only one level of embedding.

As shown in Table 1, for natural images (Fig. 9(a) to (d)), the decrypted image quality of Ge's algorithm exceeds 50 dB. However, for medical images (Fig. 9(e) to (h)), the decrypted image quality of Ge's algorithm is very poor, with PSNR values of 'Med1', 'Med2', and 'Med3' being no more than 15 dB. In contrast to natural images, medical images contain a significant number of pixels with a value of 0. Due to the XOR encryption applied on the most significant bit plane of the image, pixels with a value of 0 are encrypted to either 0 or 128. Pixels with a value of 0 will be marked as overflow values and modified to 1. However, in medical images, which contain an excessive amount of labeled location information, this modification can significantly reduce the embedding capacity.

When a pixel with value of 128 is encountered while embedding data using Ge's algorithm, it is not considered as an overflow value and is left unprocessed. In this case, when bit 1 is embedded into the pixel value of 128, it is modified to 127 using histogram movement. This means that the lower 7 bits of the pixel value are inverted, and only the MSB remains as 0. However, if the same decryption secret key is used to directly XOR the modified pixel value of 127, the decrypted pixel value will be 255, which is different from the original pixel value of 0. Therefore, it is evident that direct decryption of medical images using Ge's algorithm may lead to significant errors.

Recently, Rupali et al. [31] proposed a joint RDH-EI algorithm. They adopt stream cipher XOR encryption to encrypt the image, dividing the encrypted image into non overlapping blocks and embedding the secret data by flipping the LSB of the image blocks. In the algorithm proposed by Rupali et al. when the block size is 4×4, the algorithm can achieve a maximum ER of 0.25 bpp. Wang et al. [32] proposed an RDH-EI method based on pixel rotation within image blocks. Firstly, users encrypt the original image bitwise XOR. Then, the XOR encrypted image is divided into 2×2-size non overlapping image blocks, which are divided into four categories: unique block (UB), two fold block (TB), three fold block (ThB), and unusable block. The data hider embeds secret data by rotating the pixel values in the first three types of blocks. Wang et al.'s algorithm exhibits a noteworthy enhancement with a maximum embedding capacity of up to 0.5 bits per pixel (bpp), showcasing a significant leap from Rupali's algorithm. However, an issue arises as the PSNR for the directly decrypted image remains suboptimal, falling short of 30 dB.

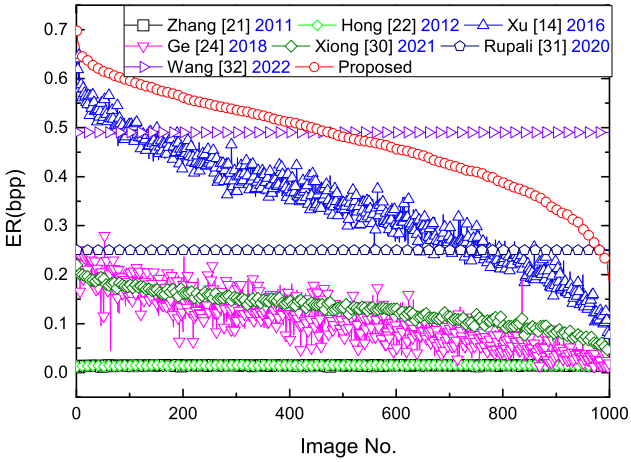


Fig. 11. Embedding rate of 1000 test images in UCID dataset.

The latest algorithms such as Chen et al.'s RDH-EI algorithm based on binary symmetric channel model and polar code [33] and the robust reversible watermarking in encrypted image with secure multi party (RRWEI-SM) algorithm proposed by Xiong et al. [30]. In [33], the user generates encrypted images through stream cipher XOR encryption, while the data-hider encodes the secret information for embedding using polarization code technology and achieves data embedding through LSB flipping. Leveraging the strong error correction capabilities of polarization codes, errors during information extraction and image restoration can be identified and corrected. In Table 1, we select the 6-th bit plane for data embedding and calculate the maximum ER and PSNR of the decrypted image of the algorithm under the optimal parameter settings. It can be seen that Chen's algorithm has low maximum ER and decryption image quality.

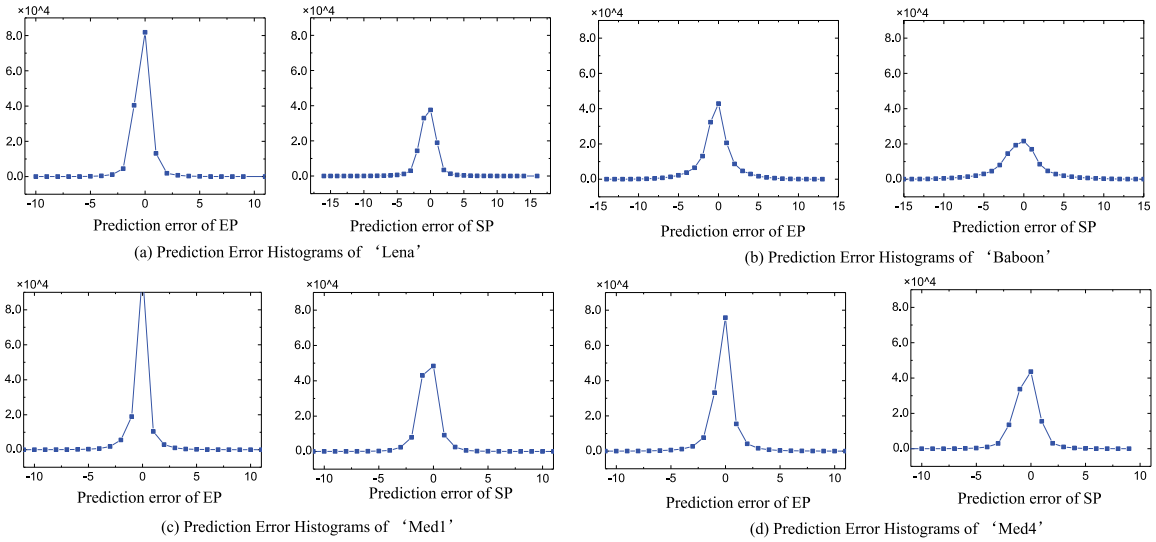


Fig. 12. HSBs prediction errors for natural images and medical images.

In Fig. 10, we take the test medical images 'Med1' as examples to compare the decrypted image O of the algorithm proposed by Ge et al. [24], Wang et al. [32], Chen et al. [33] and this paper. Observing the two regions within the red boxes in the 'Med1' image, these two areas correspond to the background and texture information in the image, respectively. The algorithms proposed by Ge et al. Wang et al. and Chen et al. exhibit noise points (pixels with decryption errors) in the pixel area of the directly decrypted image. In contrast, our proposed algorithm demonstrates superiority in medical image processing as it visually lacks noticeable noise points.

In order to improve the robustness of the algorithm, Xiong et al. [30] proposed RRWEI and Modified RRWEI strategy. Due to the higher embedding capacity under the RRWEI scheme than the Modified RRWEI scheme, we compare the RRWEI scheme with the proposed algorithm. Table 1 tests the ER and PSNR of the RRWEI scheme when it performs only one level of embedding. The RRWE algorithm embeds only 1 bit of additional data into a 2×2 size encrypted block, so the maximum ER does not exceed 0.25 bpp. Unlike Xiong's algorithm, the proposed algorithm computes prediction errors for all pixels in a 3×3-block size, so the difference utilization rate is higher than Xiong's algorithm. Based on the data presented in Table 1, it is evident that the proposed algorithm exhibits a higher embedding rate compared to Xiong's algorithm.

In order not to lose generality, Fig. 11 tests the maximum ER of this paper and the comparison algorithm under 1000 natural images in the UCID database. In Fig. 11, we rank the ERs of all comparison methods in descending order based on the ER of the proposed algorithm. The results presented in Fig. 11 indicate that the proposed algorithm holds a distinct advantage in terms of maximum ER compared to the literature used for comparison. The average ER value of Zhang's algorithm and Hong's algorithm under 1000 test images is 0.014bpp. Ge et al.'s algorithm that achieves an average ER of 0.1 bpp when only performing one level embedding, while Xiong et al. proposed the RRWEI algorithm, which achieves an average ER of 0.13 bpp when only performing one level embedding. The average ER of the algorithm proposed by Xu et al. under 1000 datasets is 0.34bpp, while the average ER of the proposed algorithm is 0.47bpp, which is higher than that of the comparative literature. Although for some images, the maximum ER of the proposed algorithm is lower than Wang et al.'s algorithm, however, the decrypted image quality of the proposed algorithm is higher than Wang's method.

The high embedding capacity of this paper is partly due to the high correlation between pixels, and we only use the HSBs prediction error of pixels to achieve higher peak values. On the other hand, the proposed algorithm obtains more prediction errors not only for EPs prediction errors, but also for SPs prediction errors, further improving the embedding capacity. Fig. 12 displays the histograms of HSBs prediction errors for both EPs and SPs, using 'Lena', 'Baboon' and 'Med1', 'Med4' images as representative examples.

As can be seen from Fig. 12, compared to the prediction error of SPs, the prediction error of EPs is more concentrated, because the prediction process of EPs utilizes the original pixel values and has a higher correlation. The prediction error of SPs is based on the EPs embedded with additional data, and the correlation between pixels is reduced. Compared to 'Lena' and 'Baboon' images, the prediction error histogram of 'Lena' with lower texture complexity is more concentrated because of the stronger correlation between pixels. Compared to natural images, the prediction errors of medical images such as 'Med1' and 'Med2' are more concentrated. Even for 'Med4', which exhibits high texture complexity, its peak prediction error remains higher than that of the natural image Lena. This implies that medical images can embed more secret data and achieve better decrypted image quality.

Medical images typically have narrower pixel value ranges because they usually represent specific types of human tissue or structures. In contrast, natural images have broader pixel value ranges as they encompass diverse scenes and colors. For any pixel value $x(i,j)$, the formula for its upper $8-n$ bits is:

$$x_{HSB}(i,j) = \frac{x(i,j) - mod(x(i,j), 2^n)}{2^n} \tag{25}$$

We can discuss in two situations:

Case 1: When the pixel value $x(i,j)$ is less than 2^n. In this case, $mod(x(i,j), 2^n) = x(i,j)$, therefore, $x_{HSB}(i,j) = 0$. That is to say, no matter what value $x(i,j)$ takes, $x_{HSB}(i,j)$ is equal to 0, and the distribution of $x_{HSB}(i,j)$ is more concentrated, which is conducive to pixel prediction.

Case 2: When the pixel value satisfies $x(i,j) \geq 2^n$. In this case, $mod(x(i,j), 2^n)$ will be a non-zero value less than 2^n, therefore, $x_{HSB}(i,j)$ will produce a non-zero result. When $x(i,j)$ starts to increase from 2^n, the value of $mod(x(i,j), 2^n)$ starts to increase from 0 until $2^n - 1$, and the growth rate of $x(i,j)$ is faster than that of 2^n, the value of $x_{HSB}(i,j)$ will continue to increase, resulting in a wider distribution of $x_{HSB}(i,j)$.

In addition, when $n \geq 1$, due to the pixel value range of $x_{HSB}(i,j)$ being less than [0,255], adjacent $x_{HSB}(i,j)$ pixels have stronger correlation, which is more conducive to pixel prediction compared to the original pixel value. Due to the unique characteristics of medical images, on one hand, medical images exhibit a greater abundance of pixel values satisfying $x(i,j) < 2^n$. On the other hand, the correlation

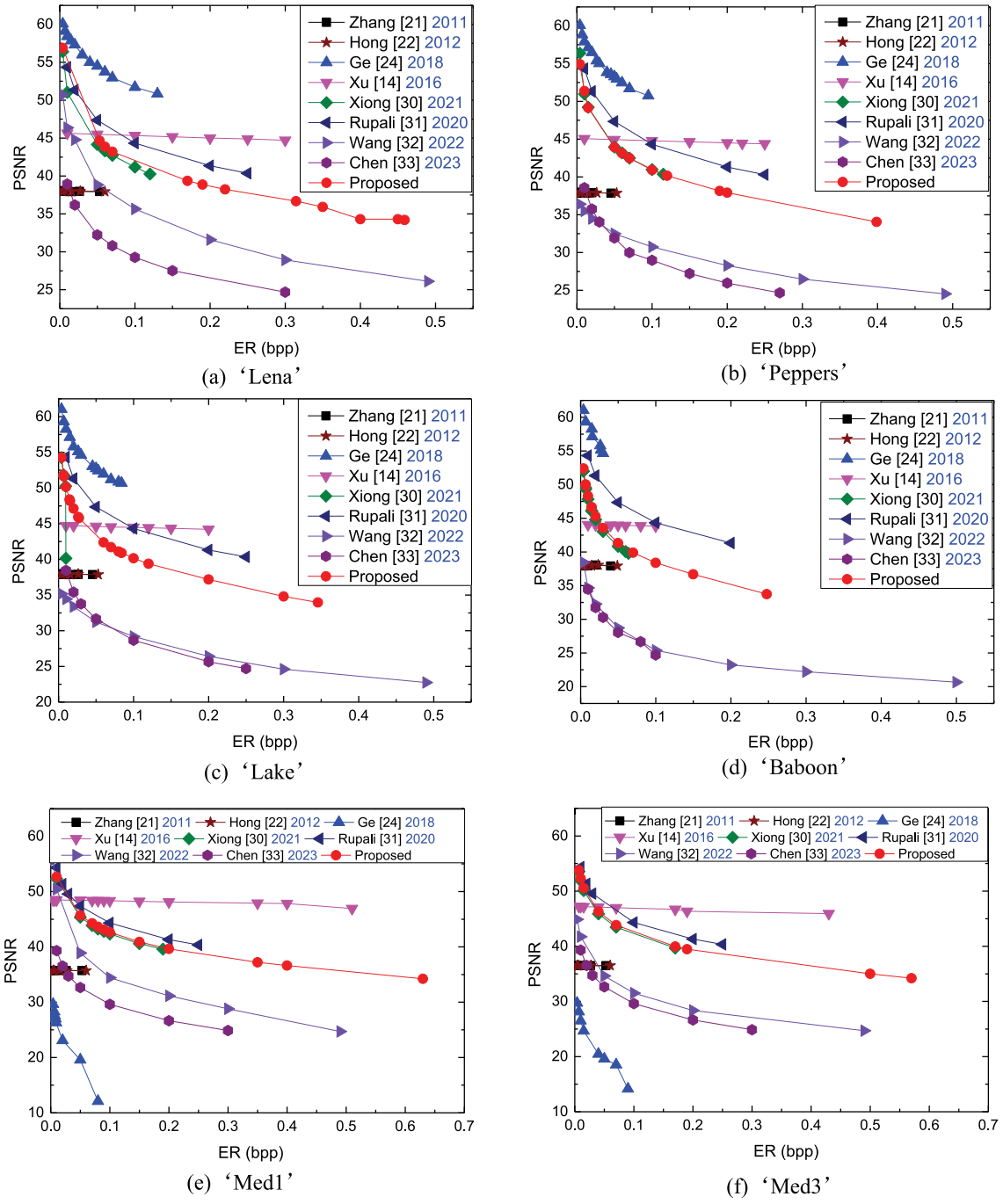


Fig. 13. PSNR values under different embedding rates.

between the original pixels in medical images is stronger compared to that of natural images.

Considering the factors mentioned above, the proposed algorithm can achieve higher peak prediction errors for HSB in medical images, thereby enhancing embedding capacity and decryption image quality compared to natural images.

Fig. 13 tests the PSNR values of four natural images 'Lena', 'Pepper', 'Lake', 'Baboon' and two medical images'Med1', 'Med3' under different ERs, and compares them with the comparative literature. In Fig. 13(a) to (f), as the embedding rate rises, a corresponding decrease in the PSNR value becomes apparent. For four natural images, the PSNR values of the proposed algorithm are all higher than 30 dB. The PSNR value of the proposed algorithm under different embedding rates is much higher than Zhang's algorithm and Hong's algorithm, and slightly higher than Xiong's algorithm. The algorithm proposed by Ge et al. has the highest PSNR value at the expense of algorithm ER and security. Rupali et al. [31] embedded secret data by modifying the LSBs of encrypted image blocks, limiting their maximum embedding rate to 0.25bpp. Despite this constraint, the algorithm produced high-quality decrypted images, with a PSNR exceeding 40 dB for natural test images. The algorithm proposed by Wang et al. [32] and Chen et al. [33] has a high ER, but the quality of the decrypted images is lower than the proposed algorithm.

In the comparative literature, Zhang et al. [21], Hong et al. [22], and Chen et al. [33] used LSB flipping method to embed data, resulting in poor decrypted image quality and low data embedding rate. The main reason is that stream cipher XOR encryption completely removes pixel redundancy in the image. Ge et al. [24], Xu et al. [14], and Rupali

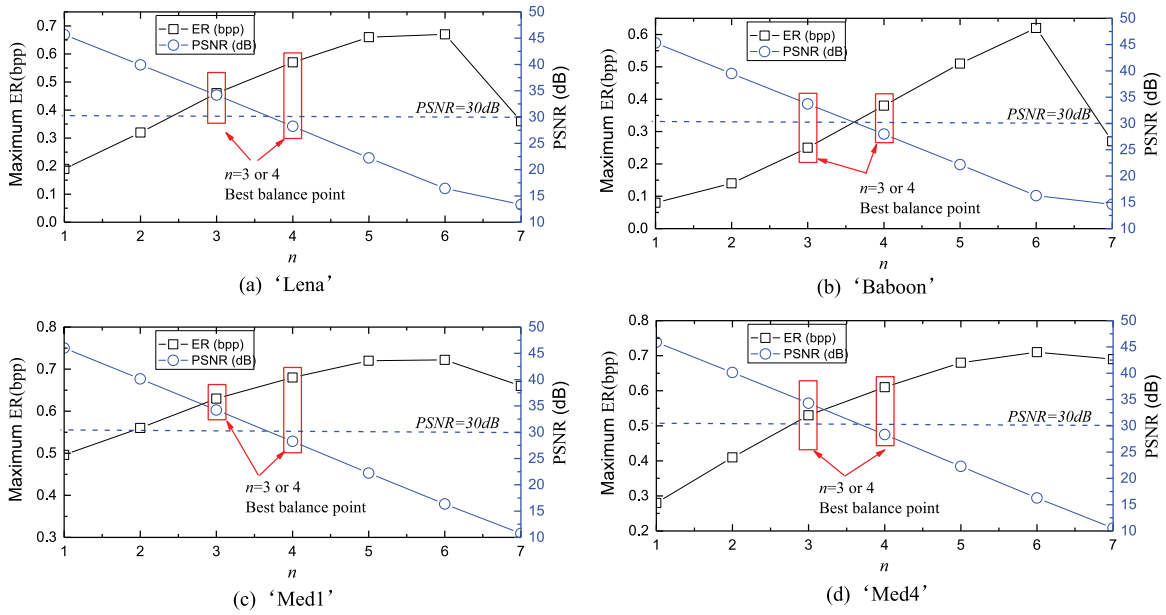


Fig. 14. The selection of parameter n influences the algorithm's performance.

et al. [31] achieved data embedding by shifting the original histogram of the image. Due to the higher peak value of the HSB histogram compared to the original histogram of the image, our proposed strategy can achieve higher embedding capacity compared to them. Although Xiong et al. [30] employed the method of modifying the HSB histogram of images for data embedding, the embedding capacity of their algorithm was limited due to the small block size. We found that the addition invariance property in lightweight secret sharing encryption is also applicable to specific prediction methods in 3 × 3 block size. Based on this motivation, we designed an RDH-EI technique with two rounds of HSB prediction error extension, and achieved better embedding performance and decrypted image quality.

For medical images, our algorithm demonstrates significant superiority compared to the contrastive algorithms, both in terms of decrypted image quality and embedding capacity. This is because medical images have higher peak prediction errors. Compared to embedding data through methods like flipping the LSBs of encrypted image blocks [21,22,33] and pixel rotation [32], using prediction error expansion for data embedding more effectively utilizes redundancy within the image. In other words, the algorithm we propose is more suitable for medical images.

5.1.2. Parameter selection on algorithm performance

The parameters in the proposed algorithm mainly include block size and n (the number of LSB bit planes).

For block size parameters: Block size plays a crucial role in determining both the embedding capacity and security of the algorithm. On the one hand, the block size cannot be too small, as small block sizes can lead to lower embedding capacity. On the other hand, the block size should not be too large. Excessive block size can weaken the correlation between pixels within the block. In addition, larger block sizes pose security risks and are susceptible to KPA attacks [28]. Therefore, in the proposed algorithm, we set the block size parameter to 3 × 3. In contrast to the 2 × 2 size utilized by Xiong et al. our algorithm is better suited for the chessboard prediction method. This choice results in more prediction errors within each block, enabling the embedding of additional information. Furthermore, compared to larger block sizes like 4 × 4, a 3 × 3 block size offers enhanced security. Taking all these factors into consideration, we fix the block size at 3 × 3.

For n parameters: The selection of parameter n can determine the embedding capacity and decryption image quality of the algorithm. In Fig. 14, we take natural images 'Lena', 'Baboon', and medical images 'Med1', 'Med4' as examples to test the ER and decrypted image quality of the algorithm under different n values (PSNR of the original image and decrypted image). Fig. 14 reveals that with an increasing n, the embedding capacity initially rises and then declines, accompanied by a gradual decrease in decrypted image quality. Generally, a PSNR above or close to 30 dB indicates satisfactory visual results for decrypted images. The experimental findings suggest that the algorithm achieves an optimal balance between embedding capacity and decrypted image quality at n values of 3 or 4. Specifically, when n is set to 3, the decrypted image quality consistently exceeds 30 dB. Setting n to 4 results in slightly lower but close to 30 dB decrypted image quality, coupled with a higher embedding capacity than n equals 3.

In summary, to balance the algorithm's embedding capacity and decrypted image quality, we recommend choosing n as either 3 or 4. If prioritizing embedding capacity, n can be set to 4; for a greater emphasis on decrypted image quality, setting n to 3 is advisable.

Table 2
Accuracy of random key estimation under COA.

Image number	Methods	$\beta(q)$							
		0	1	2	3	4	5	6	7
20	XOR-Only	0.51	0.52	0.48	0.67	0.92	0.99	1.00	1.00
	Proposed	0.51	0.50	0.50	0.52	0.49	0.50	0.50	0.49
50	XOR-Only	0.50	0.51	0.59	0.81	0.99	1.00	1.00	1.00
	Proposed	0.50	0.51	0.40	0.51	0.50	0.50	0.51	0.50
60	XOR-Only	0.50	0.53	0.66	0.85	0.99	1.00	1.00	1.00
	Proposed	0.50	0.50	0.51	0.50	0.49	0.50	0.50	0.50
80	XOR-Only	0.50	0.54	0.69	0.91	1.00	1.00	1.00	1.00
	Proposed	0.50	0.50	0.50	0.51	0.50	0.50	0.50	0.50
100	XOR-Only	0.51	0.54	0.65	0.93	1.00	1.00	1.00	1.00
	Proposed	0.50	0.51	0.49	0.50	0.50	0.50	0.50	0.51

5.2. Security analysis

In this section, we analyze the security of algorithms from two perspectives: the ability to resist attacks and statistical analysis.

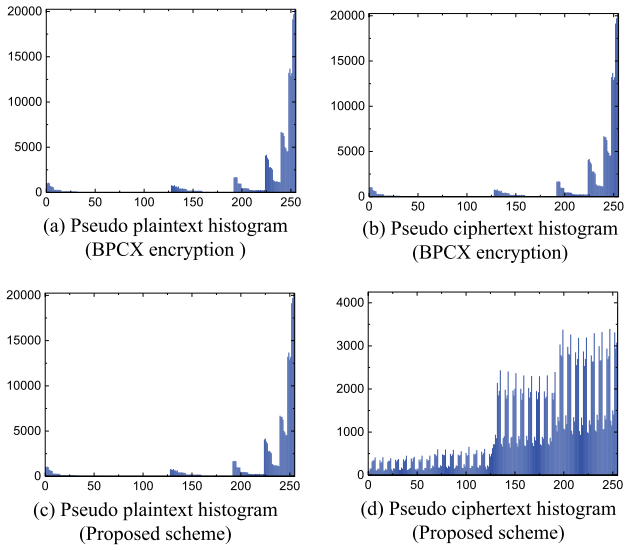


Fig. 15. Pseudo plaintext pseudo ciphertext histogram for proposed algorithm and BPCX encryption.

5.2.1. Ability to resist attacks

We conduct a security analysis of the proposed algorithm and perform a comparative assessment with the stream cipher XOR encryption method employed by Zhang [21], Hong et al. [22], Xu et al. [14], Wang et al. [32], Chen et al. [33] and the BPCX encryption adopted by Ge et al. [24]. By conducting experiments, we evaluate the proposed algorithm's ability to withstand existing ciphertext-only attacks (COA), known plaintext attacks (KPA) and Chosen plaintext attacks (CPA) in the context of RDH-EI algorithms.

(a) Ciphertext-only attack

COA is a commonly used method in cryptography to analyze ciphertext. This attack involves an attacker analyzing the contents of a ciphertext image and attempting to crack the encryption secret key, based on the knowledge of some ciphertext. In 2018, Khelifi [27] analyzed the security of stream cipher XOR encryption commonly used in RDH-EI under COA. An attacker can obtain a random matrix generated based on the encryption key and decrypt the ciphertext image with only a limited number of ciphertext known, resulting in partial disclosure of the original image. It is proved that image stream cipher XOR encryption does not have security under COA attack. We use the method proposed by Khelifi to analyze the security of XOR encryption and proposed algorithms in [21,22], and [14,32,33] under COA.

The calculation formula for estimating the accuracy of the random matrix generated based on the secret key is shown in Eq. (25), here, $\beta(q)$ represents the correct rate of each bit plane of a random matrix, where $q = [0, 1, 2 \cdots, 7]$. The XOR encryption random matrix is expressed byψ_q and the estimated random matrix is ψ'_q,

$$\beta(q) = \frac{1}{W \times H} \times \sum_{i=1}^{W} \sum_{j=1}^{H} (1 - |\psi_q(i,j) - \psi'_q(i,j)|) \tag{26}$$

Table 2 shows the estimated accuracy of each bit plane of the random matrix for XOR only encryption under the COA attack. Table 2 show that for XOR only encryption, the estimation accuracy of each bit plane of the random matrix increases as the number of known ciphertext images increases. The higher the bit plane, the higher the estimation accuracy. The most significant bit plane ($q = 7$) can be completely estimated by only 20 encrypted images.

Compared with XOR-only encryption adopt in [21,22], and [14,32,33], the algorithm proposed in this paper adopts scrambling encryption, and randomly assigns the HSB and LSB of pixels to change the value of the encrypted image pixels. Table 2 demonstrates the application of Khelifi's COA method for estimating the random key of each bit plane for the encrypted share1 or 2. The accuracy of estimating each bit plane's random key is reported as 0.5, this means that the estimated random keys have a uniform distribution of 0 bits and 1 bits. Because scrambling encryption destroys the correlation between adjacent pixels, the COA attack proposed by Khelifi cannot crack the content of the encrypted image in this paper.

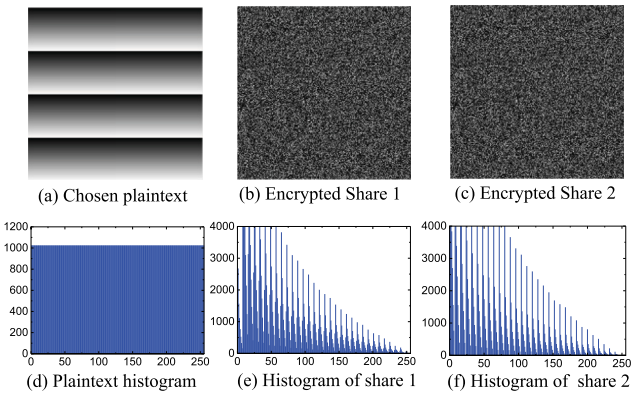


Fig. 16. Plaintext Construction, Encryption Shares, and Histogram Distributions.

(b) Known-plaintext attack

KPA refers to an attack where the attacker attempts to estimate a partial encryption key using a limited number of known plaintext–ciphertext pairs. The encryption algorithm applied to RDH-EI should be designed to be resistant to KPA attacks. This means that the algorithm should not leak any information about the key even when a small number of plaintext–ciphertext pairs are known to the attacker.

In the comparative literature, Ge et al. used BPCX encryption method, which has been widely used in RDH-EI. Recently, Qu et al. [28] proposed a KPA method for BPCX encryption based on image block mean equivalent division. Theoretical analysis and experiments prove that BPCX encryption does not have security under KPA. The main reason why BPCX encryption cannot resist KPA attacks is that the pixel values within the image block are XOR encrypted with the same random number, which makes it possible to construct pseudo plain-ciphertext with consistent pixel values.

The encryption scheme of the proposed algorithm randomly assigns HSBs and LSBs to each pixel value, changing the value of the pixel. Different pixel values are independent of each other and do not affect each other. Therefore, when an attacker only obtains some plaintext images and a corresponding encrypted shares, it is not possible to use the KPA method of Qu et al. to construct pseudoplaintext and pseudociphertext with consistent pixel values. Taking 'Lena' image as an example, we used Qu's KPA method to construct pseudoplaintext and pseudociphertext for BPCX encryption and the proposed encryption method, respectively. The pseudoplaintext and pseudociphertext of 'Lena' image are shown in Fig. 15.

Fig. 15(a) and (b) show the pseudo plaintext and pseudo ciphertext histograms generated by BPCX encryption, the pseudo plain-ciphertext pixel histograms are completely consistent. Fig. 15(c) and (d) show the pseudo plain-ciphertext histograms generated by the proposed scheme. The pseudo plain-ciphertext pixel histograms are significantly different, which proves that the proposed algorithm can resist the KPA attack proposed by Qu et al.

(c) Chosen-plaintext attack

CPA is a cryptographic method where an attacker, having access to encryption and decryption algorithms but not the secret key, selects specific plaintexts to observe their corresponding ciphertexts. The goal is to analyze patterns and vulnerabilities to eventually deduce the secret key.

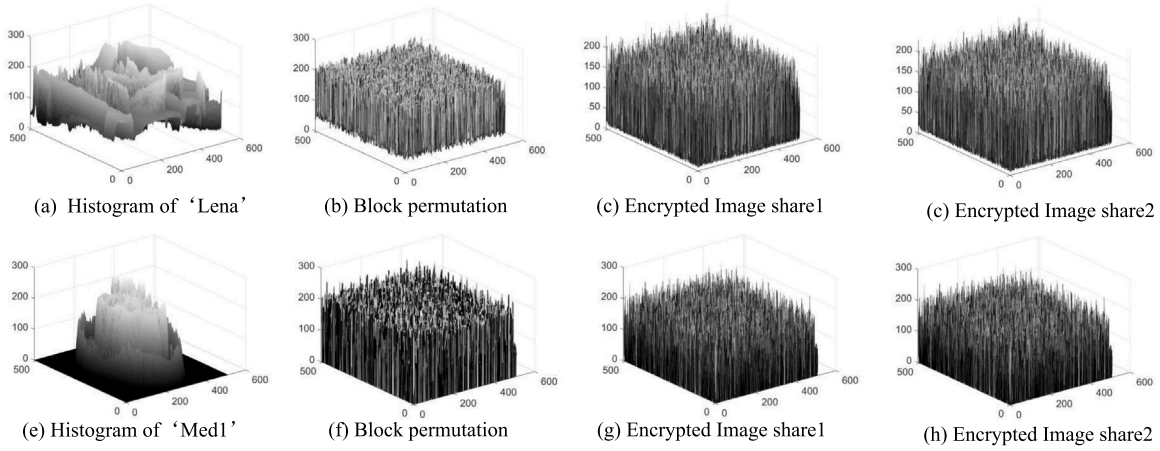


Fig. 17. Histogram distribution of original image and encrypted shares.

In 2016, Jolfaei [36] proposed a CPA method specifically designed for permutation-only encrypted images. For an image of size 512×512, the method involves constructing three specific plaintext images and encrypting them using a designated encryption algorithm, resulting in three pairs of plaintext–ciphertext images. By utilizing these three pairs of plaintext–ciphertext images, an attacker can analyze and deduce the permutation encryption key.

In this section, we utilize Jolfaei's plaintext construction method to create a specific plaintext image, as depicted in Fig. 16(a). The constructed plaintext is then encrypted into two shares, denoted as share1 and share2, illustrated in Fig. 16(b) and (c), respectively. It is evident that the encrypted shares exhibit random noise. Fig. 16(d)–(f) display the histograms corresponding to the chosen plaintext, encryption share 1, and encryption share 2.

The histogram of the original chosen plaintext image is completely different from the histograms of the encryption shares, signifying that the proposed algorithm alters the original image's pixel values. As concluded in the previous section on KPA, simultaneous changes in pixel values and positions make it impossible for attackers to deduce the key through plaintext–ciphertext comparison. Thus, the proposed algorithm proves resilient against CPA.

5.2.2. *Statistic analysis*

In this section, we analyze the algorithm's security from the perspectives of histogram distribution, pixel correlation in encrypted images, and key space.

(a) Histogram distribution and pixel correlation

In Fig. 17, we use natural image 'Lena' and medical image 'Med1' as examples to showcase the three-dimensional histogram distributions of the original image, block-permutation encrypted image, and encryption shares. Whether for natural or medical images, the encrypted images and encryption shares exhibit a uniform distribution in their three-dimensional histograms. This uniform distribution underscores the algorithm's effectiveness in safeguarding image content and eliminating pixel correlation, making it more challenging for attackers to extract useful information and enhancing overall security.

To visually display the correlation between pixels, in the encrypted image share, 10,000 pixel values labeled as x are randomly selected, along with their corresponding 10,000 horizontally adjacent pixel values labeled as y. These pairs (x, y) form coordinate points plotted on a two-dimensional system, creating a scatter plot with 10,000 points, as shown in Fig. 18. It is evident that the greater the concentration of these points, the stronger the pixel correlation in the horizontal direction.

From the distribution of adjacent pixel correlations in Fig. 18, it can be observed that the encryption shares effectively break the pixel correlations in the original image. This indicates that after encryption,

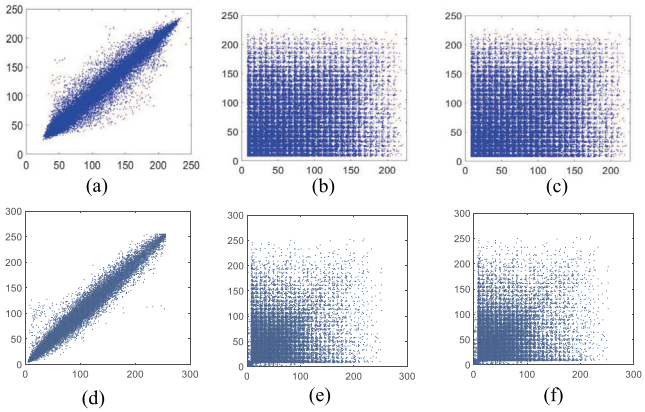


Fig. 18. Scatter plot of pixel correlation for original image and encrypted shares ((a) to (c): Scatter plot distribution of pixel correlation for 'Lena', and encrypted shares 1 and 2. (d) to (f): Scatter plot distribution of pixel correlation for 'Med1', and encrypted shares 1 and 2.).

the correlation between adjacent pixels in the image is significantly weakened, enhancing the overall security of the image.

(b) Key space analysis

Key space analysis refers to the evaluation of possible combinations and variations within the cryptographic key space of a given algorithm or system. It is generally considered that when the key space exceeds2^{128}, the encryption algorithm can resist exhaustive searches or brute-force attacks.

For $W \times H$ sized grayscale image is divided into N image blocks, and the key space for scrambling the blocks during the encryption phase is $(N)!$. The process of randomly splitting the image encrypted after block permutation into two shares can be viewed as encrypting the lower n-bit planes of the image through bitwise XOR with random numbers. The key space for this process is $(2^n)^{W \times H}$. Therefore, the total key space for the encryption process is $(N)! \times (2^n)^{W \times H}$. For a grayscale image of size 510×510, divided into 3×3 blocks with n=3, the proposed algorithm's key space is $(2.89 \times 10^4)! \times (2^3)^{510 \times 510}$, significantly exceeding 2^{128} and demonstrating its ability to resist exhaustive attacks.

5.3. *Time complexity analysis*

The time complexity of algorithms serves as a pivotal indicator of algorithmic performance, particularly in the context of processing substantial amounts of image data within cloud storage scenarios. In this paper, we adopt a lightweight encryption scheme. Here, lightweight

Table 3
Running time in seconds for each stage of different methods.

Test	Image encryption				Data embedding and extracting				Image recover			
Images	[14]	[32]	[33]	Our	[14]	[32]	[33]	Our	[14]	[32]	[33]	Our
Lena	0.51	0.38	0.38	0.29	0.52	1.05	0.16	0.69	0.8	3.43	2.43	5.48
Peppers	0.44	0.4	0.4	0.29	0.34	1.07	0.76	0.79	0.56	3.42	2.42	6.81
Lake	0.47	0.38	0.38	0.3	0.32	1.04	0.78	0.66	0.66	3.68	2.45	5.45
Baboon	0.49	0.38	0.38	0.29	0.33	1.05	0.78	0.65	0.58	3.41	2.45	5.42
Med1	0.44	0.38	0.38	0.29	0.36	1.18	0.77	0.74	0.83	5.76	2.43	6.69
Med2	0.45	0.37	0.37	0.29	0.39	1.07	0.79	0.73	0.62	6.98	2.44	5.45
Med3	0.44	0.38	0.38	0.29	0.37	1.05	0.76	0.72	0.63	3.59	2.42	5.46
Med4	0.43	0.39	0.39	0.3	0.4	1.06	0.77	0.7	0.58	3.5	2.43	5.51
Average	0.45	0.38	0.38	0.29	0.37	1.07	0.69	0.71	0.65	4.22	2.43	5.78

image encryption refers to reducing the demand for computational resources while ensuring the security of image content. This is primarily characterized by a fast algorithmic processing speed and low computational complexity. Lightweight encryption helps alleviate the computational burden on users during both the encryption and decryption processes.

The commonly used lightweight encryption algorithm in RDH-ED is stream cipher XOR encryption, such as the latest comparative literature [14,32,33] all using stream cipher XOR encryption. While the stream cipher XOR encryption method offers high speed, its security is notably weak and may not withstand COA attack, as illustrated in Table 2. The proposed algorithm adopts a lightweight secret sharing encryption scheme, which can achieve fast encryption and decryption while ensuring security.

In Table 3, we compare the running time in seconds of the state-of-the-art methods [14,32,33] with the proposed algorithm at each stage. Table 3 reveals that the proposed algorithm's runtime at various stages closely approaches the state-of-the-art lightweight RDH-ED algorithm. Particularly noteworthy is the encryption phase, where the runtime is less than existing methods. In the decryption phase, the proposed approach exhibits a slightly longer runtime compared to [32], although still within an acceptable range.

5.4. Limitation

Although the proposed algorithm has good performance in embedding capacity, decryption image quality, and security, there are also some limitations. On the one hand, lightweight secret sharing encryption consumes excessive storage resources, which may pose challenges to resource limited environments or devices with limited storage capacity, limiting the feasibility of this algorithm in some practical applications. On the other hand, similar to existing methods, our algorithm cannot guarantee lossless restoration of the marked encrypted image when it is attacked during transmission or storage, and watermark extraction embedded in the image may result in errors. In the future, we will consider balancing the relationship between embedding capacity and storage overhead, and consider methods to improve the robustness of the proposed algorithm.

6. Conclusion

This paper introduces a novel approach to tackle security and communication concerns in telemedicine applications. The proposed solution involves a high-capacity RDH algorithm designed for multi-party secure computing. By dividing the pixel values in the image block into two categories: embedded pixels(EPs) and sampled pixels(SPs), two-stage data embedding is achieved in encrypted images. In the first stage of data embedding, two edge servers predict EPs using SPs in the image blocks, and obtain the HSBs prediction error of EPs. The additional data is embedded into HSBs prediction error of EPs through addition operation and histogram shift method. In the second stage of data embedding, two edge servers use the difference between the HSB values of SPs and EPs as the HSBs prediction error of SPs. By employing a two-stage data embedding process, the proposed algorithm achieves a significant increase in embedding capacity, all the while maintaining the quality of decrypted images at a satisfactory level. Furthermore, the proposed algorithm demonstrates the capability to withstand existing attack.

CRediT authorship contribution statement

Lingfeng Qu: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Mohan Li:** Writing – review & editing, Project administration, Funding acquisition. **Peng Chen:** Investigation, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Rupali Bhardwaj, An enhanced reversible patient data hiding algorithm for e-healthcare, Biomed. Signal Process. Control 64 (2021) 102276.
[2] Yun-Qing Shi, Xiaolong Li, Xinpeng Zhang, Hao-Tian Wu, Bin Ma, Reversible data hiding: advances in the past two decades, IEEE Access 4 (2016) 3210–3237.
[3] James M. Barton, Method and apparatus for embedding authentication information within digital data, 1997, United States Patent, 5 646 997.
[4] Mehmet Utku Celik, Gaurav Sharma, A Murat Tekalp, Eli Saber, Lossless generalized-LSB data embedding, IEEE Trans. Image Process. 14 (2) (2005) 253–266.
[5] Jun Tian, Reversible data embedding using a difference expansion, IEEE Trans. Circuits Syst. Video Technol. 13 (8) (2003) 890–896.
[6] Yongjian Hu, Heung-Kyu Lee, Jianwei Li, DE-based reversible data hiding with improved overflow location map, IEEE Trans. Circuits Syst. Video Technol. 19 (2) (2008) 250–260.
[7] Adnan M. Alattar, Reversible watermark using the difference expansion of a generalized integer transform, IEEE Trans. Image Process. 13 (8) (2004) 1147–1156.
[8] Lute Kamstra, Henk J.A.M. Heijmans, Reversible data embedding into images using wavelet techniques and sorting, IEEE Trans. Image Process. 14 (12) (2005) 2082–2090.
[9] Diljith M. Thodi, Jeffrey J. Rodriguez, Prediction-error based reversible watermarking, in: 2004 International Conference on Image Processing, 2004, Vol. 3, ICIP'04, IEEE, 2004, pp. 1549–1552.
[10] Zhicheng Ni, Yun-Qing Shi, Nirwan Ansari, Wei Su, Reversible data hiding, IEEE Trans. Circuits Syst. Video Technol. 16 (3) (2006) 354–362.
[11] Diljith M. Thodi, Jeffrey J. Rodríguez, Expansion embedding techniques for reversible watermarking, IEEE Trans. Image Process. 16 (3) (2007) 721–730.
[12] Dinu Coltuc, Improved embedding for prediction-based reversible watermarking, IEEE Trans. Inf. Forensics Secur. 6 (3) (2011) 873–882.

[13] Kede Ma, Weiming Zhang, Xianfeng Zhao, Nenghai Yu, Fenghua Li, Reversible data hiding in encrypted images by reserving room before encryption, IEEE Trans. Inf. Forensics Secur. 8 (3) (2013) 553–562.
[14] Dawen Xu, Rangding Wang, Separable and error-free reversible data hiding in encrypted images, Signal Process. 123 (2016) 9–21.
[15] Pauline Puteaux, William Puech, An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images, IEEE Trans. Inf. Forensics Secur. 13 (7) (2018) 1670–1681.
[16] Zhaoxia Yin, Youzhi Xiang, Xinpeng Zhang, Reversible data hiding in encrypted images based on multi-MSB prediction and Huffman coding, IEEE Trans. Multimed. 22 (4) (2019) 874–884.
[17] Fan Chen, Yuan Yuan, Hongjie He, Miao Tian, Heng-Ming Tai, Multi-MSB compression based reversible data hiding scheme in encrypted images, IEEE Trans. Circuits Syst. Video Technol. 31 (3) (2020) 905–916.
[18] Chunqiang Yu, Xianquan Zhang, Xinpeng Zhang, Guoxiang Li, Zhenjun Tang, Reversible data hiding with hierarchical embedding for encrypted images, IEEE Trans. Circuits Syst. Video Technol. 32 (2) (2021) 451–466.
[19] Xiaoshuai Wu, Tong Qiao, Ming Xu, Ning Zheng, Secure reversible data hiding in encrypted images based on adaptive prediction-error labeling, Signal Process. 188 (2021) 108200.
[20] Assad Abbas, Samee U. Khan, A review on the state-of-the-art privacy-preserving approaches in the e-health clouds, IEEE J. Biomed. Health Inform. 18 (4) (2014) 1431–1441.
[21] Xinpeng Zhang, Reversible data hiding in encrypted image, IEEE Signal Process. Lett. 18 (4) (2011) 255–258.
[22] Wien Hong, Tung-Shou Chen, Han-Yan Wu, An improved reversible data hiding in encrypted images using side match, IEEE Signal Process. Lett. 19 (4) (2012) 199–202.
[23] Fangjun Huang, Jiwu Huang, Yun-Qing Shi, New framework for reversible data hiding in encrypted domain, IEEE Trans. Inf. Forensics Secur. 11 (12) (2016) 2777–2789.
[24] Haoli Ge, Yan Chen, Zhenxing Qian, Jianjun Wang, A high capacity multi-level approach for reversible data hiding in encrypted images, IEEE Trans. Circuits Syst. Video Technol. 29 (8) (2018) 2285–2295.
[25] Shuang Yi, Yicong Zhou, Separable and reversible data hiding in encrypted images using parametric binary tree labeling, IEEE Trans. Multimed. 21 (1) (2018) 51–64.
[26] Chunqiang Yu, Xianquan Zhang, Guoxiang Li, Shanhua Zhan, Zhenjun Tang, Reversible data hiding with adaptive difference recovery for encrypted images, Inform. Sci. 584 (2022) 89–110.
[27] Fouad Khelifi, On the security of a stream cipher in reversible data hiding schemes operating in the encrypted domain, Signal Process. 143 (2018) 336–345.
[28] Lingfeng Qu, Hongjie He, Fan Chen, On the security of block permutation and Co-XOR in reversible data hiding, IEEE Trans. Circuits Syst. Video Technol. 32 (3) (2021) 920–932.
[29] Lingfeng Qu, Fan Chen, Shanjun Zhang, Hongjie He, Cryptanalysis of reversible data hiding in encrypted images by block permutation and co-modulation, IEEE Trans. Multimed. 24 (2021) 2924–2937.
[30] Lizhi Xiong, Xiao Han, Ching-Nung Yang, Yun-Qing Shi, Robust reversible watermarking in encrypted image with secure multi-party based on lightweight cryptography, IEEE Trans. Circuits Syst. Video Technol. 32 (1) (2021) 75–91.
[31] Rupali Bhardwaj, Ashutosh Aggarwal, An improved block based joint reversible data hiding in encrypted images by symmetric cryptosystem, Pattern Recognit. Lett. 139 (2020) 60–68.
[32] Xu Wang, Ching-Chun Chang, Chia-Chen Lin, Chin-Chen Chang, Reversal of pixel rotation: A reversible data hiding system towards cybersecurity in encrypted images, J. Vis. Commun. Image Represent. 82 (2022) 103421.
[33] Kaimeng Chen, Qingxiao Guan, Weiming Zhang, Nenghai Yu, Reversible data hiding in encrypted images based on binary symmetric channel model and polar code, IEEE Trans. Dependable Secure Comput. (2022).
[34] Ioan Catalin Dragoi, Dinu Coltuc, On the security of reversible data hiding in encrypted images by MSB prediction, IEEE Trans. Inf. Forensics Secur. 16 (2020) 187–189.
[35] Zhihong Tian, Mohan Li, Meikang Qiu, Yanbin Sun, Shen Su, Block-DEF: A secure digital evidence framework using blockchain, Inform. Sci. 491 (2019) 151–165.
[36] Alireza Jolfaei, Xin-Wen Wu, Vallipuram Muthukkumarasamy, On the security of permutation-only image encryption schemes, IEEE Trans. Inf. Forensics Secur. 11 (2) (2015) 235–246.