Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Reversible data hiding in encrypted images based on additive secret sharing and additive joint coding using an intelligent predictor*#

Ziyi ZHOU†, Chengyue WANG, Kexun YAN, Hui SHI†‡, Xin PANG†

School of Computer Science and Artificial Intelligence, Liaoning Normal University, Dalian 116029, China

†E-mail: zzy_163361@163.com; shihui_jiayou@lnnu.edu.cn; pang.xin@163.com

Received Nov. 5, 2023; Revision accepted Feb. 20, 2024; Crosschecked July 4, 2024; Published online Aug. 3, 2024

Abstract: Reversible data hiding in encrypted images (RDHEI) is essential for safeguarding sensitive information within the encrypted domain. In this study, we propose an intelligent pixel predictor based on a residual group block and a spatial attention module, showing superior pixel prediction performance compared to existing predictors. Additionally, we introduce an adaptive joint coding method that leverages bit-plane characteristics and intra-block pixel correlations to maximize embedding space, outperforming single coding approaches. The image owner employs the presented intelligent predictor to forecast the original image, followed by encryption through additive secret sharing before conveying the encrypted image to data hiders. Subsequently, data hiders encrypt secret data and embed them within the encrypted image before transmitting the image to the receiver. The receiver can extract secret data and recover the original image losslessly, with the processes of data extraction and image recovery being separable. Our innovative approach combines an intelligent predictor with additive secret sharing, achieving reversible data embedding and extraction while ensuring security and lossless recovery. Experimental results demonstrate that the predictor performs well and has a substantial embedding capacity. For the Lena image, the number of prediction errors within the range of [−5, 5] is as high as 242 500 and our predictor achieves an embedding capacity of 4.39 bpp.

Key words: Reversible data hiding in encrypted images (RDHEI); Additive secret sharing; Adaptive joint coding; Intelligent predictor

https://doi.org/10.1631/FITEE.2300750 **CLC number:** TP309

1 Introduction

Reversible data hiding in encrypted images (RDHEI) is a technique used in the field of information security and image processing. RDHEI combines the concepts of reversible data hiding and encryption to protect sensitive information within digital images while maintaining the confidentiality and integrity of the data.

RDHEI techniques can be categorized into three main types, which are related to the order of data hiding, encryption, and image processing. These three categories are: vacating room after encryption (VRAE) (Qin et al., 2021; Hua et al., 2022), vacating room by encryption (VRBE) (Chen et al., 2022), and reserving room before encryption (RRBE) (Wu et al., 2020; Yin et al., 2020, 2022; Yu et al., 2022). Yin et al. (2020) used predicted pixel values to generate labeling tags for reserving embedding space before encryption. The predicted pixel values were obtained using the median edge detector (MED) (Weinberger et al.,

‡ Corresponding author

* Project supported by the Scientific Research Project of Liaoning Provincial Department of Education, China (No. JYTMS20231039) and the Liaoning Provincial Educational Science Planning Project, China (No. JG22CB252)

Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.2300750) contains supplementary materials, which are available to authorized users

ORCID: Ziyi ZHOU, https://orcid.org/0009-0009-0367-7256; Chengyue WANG, https://orcid.org/0009-0000-6957-5468; Kexun YAN, https://orcid.org/0009-0000-4917-2872; Hui SHI, https://orcid.org/0000-0001-5029-7461; Xin PANG, https://orcid.org/0009-0008-0468-7563

2000). A random matrix was then generated and bitwise XOR operations were applied to the pixel values for image encryption. Wu et al. (2020) employed MED (Thodi and Rodríguez, 2007) to estimate prediction errors and employed parametric binary tree labeling to preserve spatial correlations within the entire original image for reserving space for hidden data. Image encryption was performed by bitwise XOR operations with random matrices and pixel values. Yu et al. (2022) used MED (Weinberger et al., 2000) to implement a hierarchical prediction error magnitude division strategy, generating hierarchical bit-plane labels to reserve embedding space. Stream cipher encryption was then applied. Yin et al. (2022) employed prediction error manipulation, bit-plane rearrangement, and compression using the original image's prediction errors to reserve embedding space. Image encryption was performed through bitwise XOR operations with a pseudo-random matrix. Reserving embedding space through pixel prediction and compression methods can have superior embedding capabilities.

Qi et al. (2023) pioneered a method to enhance RDHEI embedding capacity using adaptive quadtree partitioning and most significant bit (MSB) prediction. Their approach tailored partitioning to image smoothness, encrypting and scrambling blocks to resist analysis. Notably, they used the upper–left pixel in each block for prediction, maximizing embedding space. Integrating quadtree partitioning into RDHEI distinguishes their method. However, neural network integration was expected to further enhance the performance (Wang YM et al., 2023), promising significant advancements in image embedding technology.

Note that these methods currently rely on pseudo-random matrix based encryption, which does not offer the same level of security as RDHEI schemes based on secret sharing (Qin et al., 2021; Chen et al., 2022; Hua et al., 2022) and homomorphic encryption (Xiang and Luo, 2018), but homomorphic encryption tends to come with high computational costs.

Additive secret sharing robustly distributes confidential information among shares, enhancing encrypted image security. It deters collusion attacks by requiring collaborative effort for reconstruction, ensuring lossless recovery. This aligns with our study's reversible data hiding objective. Additionally, it strengthens resistance against statistical attacks, as the sharing process complexity hampers attackers' attempts to extract meaningful information. Overall, additive secret sharing bolsters the security robustness of our RDHEI scheme.

Yan et al. (2023) introduced a novel public key based bidirectional shadow image authentication method via image secret sharing, eliminating pixel expansion. Tailored for a (k, n) threshold, it achieved bidirectional authentication without loss or auxiliary information. Acknowledging risks of pixel modifications, Hua et al. (2023) proposed a secure (r, n)-threshold preprocessing-free matrix secret sharing technique for image secret sharing, enabling direct sharing of m-bit data via matrix multiplication without preprocessing. Similarly, Yu et al. (2023) proposed Chinese remainder theorem based secret sharing (CRTSS) with hybrid coding for high embedding capacity, stressing the importance of recording randomness for complete reversibility.

To bolster security without significant computational cost, we propose to combine additive secret sharing encryption with pixel prediction and compression techniques. Our method integrates an intelligent predictor based on a ResNet architecture, enhancing embedding capacity by improving the accuracy. Leveraging neural network capabilities allows for precise pixel correlation anticipation, reducing prediction errors and optimizing embedding space utilization. Additionally, three-dimensional (3D) chaotic mapping and multilayer randomness strengthen additive secret sharing, enhancing security and resilience against advanced attacks. Our adaptive joint encoding improves embedding rates and mitigates risks associated with pixel modifications at specified coordinates.

2 The proposed method

2.1 Intelligent predictor

To predict pixel values, we construct an intelligent predictor based on a residual group block and a spatial attention module (SAM) (Woo et al., 2018). The residual group block is selected because its skip connection structure can better solve the problems of gradient disappearance and gradient explosion, so that the network can learn the feature representation of the image at a deeper level and through the stacking of multilayer residual blocks, gradually increasing and finally decreasing the number of channels in the network, to

increase the receptive field of the network and to improve the ability to extract features of different scales. This multi-scale feature fusion helps understand the semantic information of the image more comprehensively. The formula for the proposed intelligent predictor is shown in Eq. (1):

$$P_r=F(C_\eta, \psi), \tag{1}$$

where P_r represents the pixel value's prediction result obtained after training the network model F, C_η is the input data with a size of 512×512 obtained through preprocessing, and ψ is a set of network learning parameters including the learning rate and the number of iterations.

The architecture of the proposed intelligent predictor is depicted in Fig. 1. The predictor consists of 53 convolutional layers and four modules: input module, feature extraction module, prediction module, and output module.

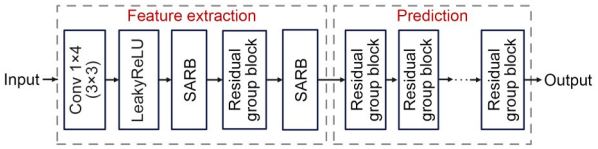

Fig. 1 Architecture of the proposed predictor (SARB: spatial attention residual block)

The input image size of the network is 512×512. The feature extraction module deployed after the input layer consists of 14 convolutional layers. The first convolutional layer is a shallow 1×4 layer (1-channel input, 4-channel output) with a stride of 1 and a padding of 1, used to capture basic features of the input image. The extracted features then pass through a 4×4 spatial attention residual block (SARB) after applying the LeakyReLU activation function. We combine the residual block with the SAM (Woo et al., 2018) to form the SARB with residual connections, as illustrated in Fig. 2. The SARBs can adaptively learn the spatial and channel relationships of the image, enabling the network to pay more attention to important regions and features in the image and improving the model ability to capture image details and key information. Each attention residual block consists of a residual block containing two convolutional layers and a channel attention mechanism module. The channel attention mechanism module is used to weight features based on the importance of the channel. This module performs global maximum pooling and global average pooling to obtain channel attention features and multiplies them with input features to obtain the final generated features.

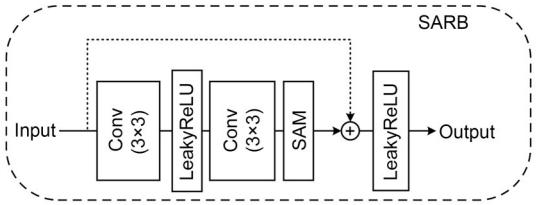

Fig. 2 Architecture of SARB (SARB: spatial attention residual block; SAM: spatial attention module)

Subsequently, the connected 4×8 residual group block further processes the features by stacking multiple residual blocks to capture higher-level image information. By stacking multiple residual blocks, the number of channels in the network is reduced, and then increased, and gradually reduced again, thereby increasing the receptive field of the network and improving the ability to extract features at different scales. This multi-scale feature fusion contributes to a more comprehensive understanding of the semantic information of images. Finally, the connected 8×8 attention residual block reintroduces the spatial attention mechanism to weight the features, further enhancing the focus on important information.

The prediction module is connected after the feature extraction module to process the features obtained from the feature extraction module for pixel prediction. The prediction module consists of seven residual group blocks. Each residual group block is composed of a residual block containing three convolutional layers and a residual block containing two convolutional layers connected, as shown in Fig. 3 (the parameters of the prediction module are provided in the supplementary materials).

Our predictor surpasses conventional counterparts, enhancing the embedding capacity. Integrating a neural network pixel prediction model refines precision and accuracy. Leveraging neural networks anticipates pixel correlations, reducing errors and optimizing embedding space. This refinement elevates algorithm efficacy and adaptability across diverse images, enhancing practicality and versatility.

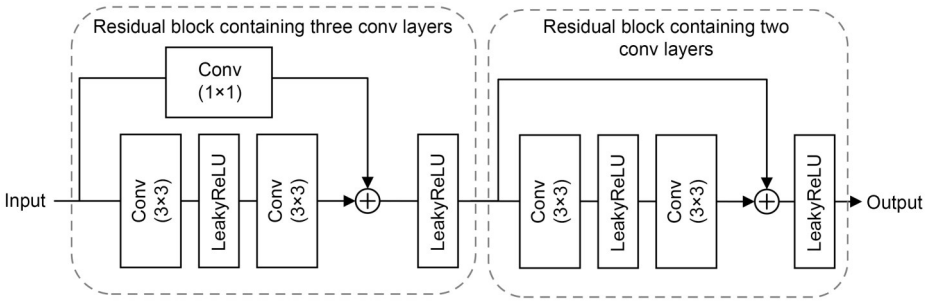

Fig. 3 Architecture of the residual group blocks

2.2 Adaptive joint coding

For compressing the cover image to vacate embedding space, we propose an adaptive joint coding method. When maxe≤τ&&(3+3×maxd)>T, the encoding method based on the prediction error is used, where maxe represents the maximum prediction error within a block, τ represents the threshold for pixel values, maxd represents the highest differing bit in bit-plane comparison, and T represents the threshold for the prediction error. When (maxe≤τ&&(3+3×maxd)≤T)‖(maxd≤τ&&maxe>τ), the encoding method based on bit-plane comparison is used (the threshold derivation is provided in the supplementary materials).

2.2.1 Bit-plane comparison based coding

sp_1, sp_2, sp_3 and spp_1, spp_2, spp_3 are decomposed into bit-planes using Eq. (2):

$$b_n^k(i,j)=\left\lfloor \frac{p_n(i,j)}{2^k} \right\rfloor \bmod 2, \tag{2}$$

where $b_n^k(i, j)$ denotes the k^{th} bit-plane, $p_n(i, j)$ represents the n^{th} pixel in the block at (i, j), and $k\in\{7, 6, 5, 4, 3, 2, 1, 0\}$. k=0 represents the least significant bit-plane. The highest differing bit within each block is calculated using Eq. (3). The function dif(,) compares two elements and returns k+1 if they are different; otherwise, it returns 0.

$$\begin{aligned}\text{maxd}=\max(&\text{dif}(b_{\text{spp}_1}^k, b_{\text{sp}_1}^k), \text{dif}(b_{\text{spp}_2}^k, b_{\text{sp}_2}^k),\\ &\text{dif}(b_{\text{spp}_3}^k, b_{\text{sp}_3}^k)).\end{aligned} \tag{3}$$

Maxd is represented using a three-bit binary value, and the least significant bit-planes (LSBs) of sp_1, sp_2, sp_3 and the error bit-planes "elsb" are extracted. The three-bit binary representation of maxd and the maxd

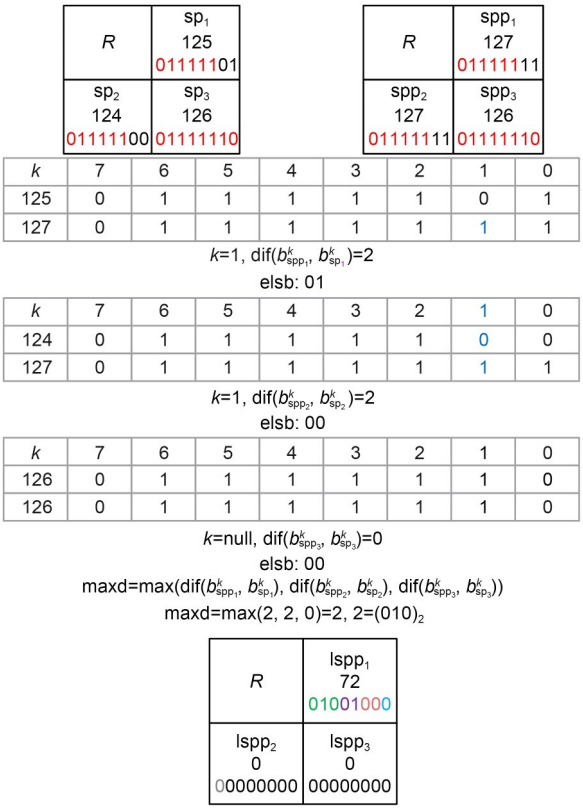

Fig. 4 Bit-plane comparison coding example

LSBs of sp_1, sp_2, sp_3 are sequentially replaced by sp_1, sp_2, sp_3. Fig. 4 illustrates this process with an example. Let sp_1, sp_2, sp_3 be 125, 124, 126, and spp_1, spp_2, spp_3 be 127, 127, 126, respectively. maxd=max(2, 2, 0)=2, and then maxd is converted to its three-bit binary representation as "010," which serves as the first three bits of the label. LSBs of the original pixel values sp_1, sp_2, sp_3 are recorded as "elsb," resulting in 01, 00, 00, respectively. The label and elsb are stored, while the remaining bit-planes are set to 0. Thus, the compressed pixel values using bit-plane comparison based coding are $lspp_1$ 72, $lspp_2$ 0, and $lspp_3$ 0.

2.2.2 Prediction error based encoding

In contrast to the encoding based on bit-plane comparison, prediction error based encoding represents prediction errors using four-bit binary codes. Instead of using the same encoding for each block, the encoding is performed individually for each pixel value. First, the cover image and the original image are compared on a block basis, and the prediction error value e_φ is computed using Eq. (4) with φ=1, 2, 3:

$$e_\varphi=\mathrm{spp}_\varphi-\mathrm{sp}_\varphi. \tag{4}$$

Next, when the prediction error is $<T$, the prediction error is represented using a four-bit code, where the first bit represents the sign of the error (0 for positive, 1 for negative) and the remaining three bits represent the binary representation of the value $|e_\varphi|$. T is a threshold value that determines the encoding method, which is explained in the supplementary materials. The obtained four-bit codes are then replaced in the high-order four-bit-planes of sp_1, sp_2, sp_3.

Fig. 5 illustrates an example using numerical values, where sp_1, sp_2, sp_3 are taken as 126, 129, 128, and spp_1, spp_2, spp_3 are taken as 127, 127, 127, respectively. e_1=127−126=1, e_2=127−129=−2, e_3=127−128=−1. These values $|e_\varphi|$ are converted into a three-bit binary representation, resulting in 001, 010, and 001. When combined with the sign codes, we obtain 0001, 1010, and 1001. The four LSBs of each pixel are then set to 0, creating space. The compressed pixel values based on the prediction error encoding are thus obtained as pspp_1 16, pspp_2 160, pspp_3 144.

3 RDHEI

In this section, we detail our RDHEI scheme, which is based on an intelligent predictor and additive secret sharing and is structured around three executing parties, image owner, data hider, and receiver, as illustrated in Fig. 6. We preprocess images using a

R	sp_1 126 01111110	R	spp_1 127 01111111
sp_2 129 10000001	sp_3 128 10000000	spp_2 127 01111111	spp_3 127 01111111

sp_φ	126	129	128
spp_φ	127	127	127
$e_\varphi=\mathrm{spp}_\varphi-\mathrm{sp}_\varphi$	1	−2	−1
Codeword	0001	1010	1001

R	pspp_1 16 00010000
pspp_2 160 10100000	pspp_3 144 10010000

Fig. 5 An example of prediction error based encoding

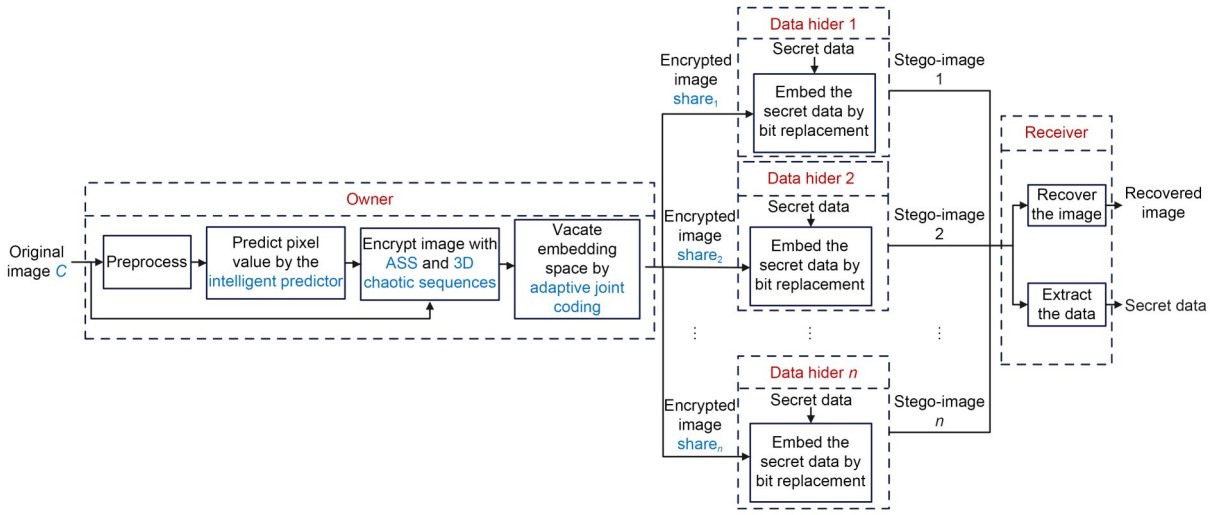

Fig. 6 An overview of the proposed reversible data hiding in encrypted images (ASS: additive secret sharing)

division method (Section 3.1) to enhance the predictor accuracy. Our intelligent predictor (Section 2.1) forecasts pixel values. Additive secret sharing encryption (Section 3.1) secures images using 3D chaotic mapping for enhanced security against collusion and statistical attacks. Adaptive joint coding (Section 2.2) compresses images to boost the embedding capacity. Encrypted images are distributed to data hiding parties for secret information embedding via bit replacement. At the receiver end, lossless image recovery and data extraction take place separately. Table S1 in the supplementary materials displays the symbol representation.

3.1 Image owner

3.1.1 Image preprocessing

The original image C with a size of $H\times W$ is divided into four sets, namely, the cross set C_1, circle set C_2, triangle set C_3, and square set C_4, as illustrated in Fig. 7.

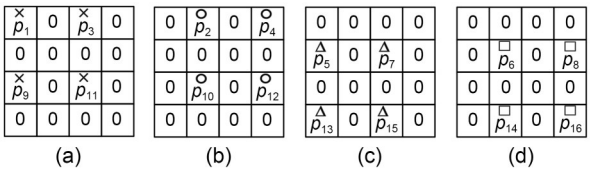

Fig. 7 Sample set graph obtained from preprocessing: (a) cross set C_1; (b) circle set C_2; (c) triangle set C_3; (d) square set C_4

3.1.2 Constructing input data

As illustrated in Fig. 8, we form input images by combining the cross set C_1, circle set C_2, triangle set C_3, and square set C_4 in groups of three. We merge sets C_1, C_2, and C_3 to obtain the input image in_square. Similarly, we merge sets C_1, C_2, and C_4 to obtain the input image in_triangle, merge sets C_1, C_3, and C_4 to obtain the input image in_circle, and merge sets C_2, C_3, and C_4 to obtain the input image in_cross.

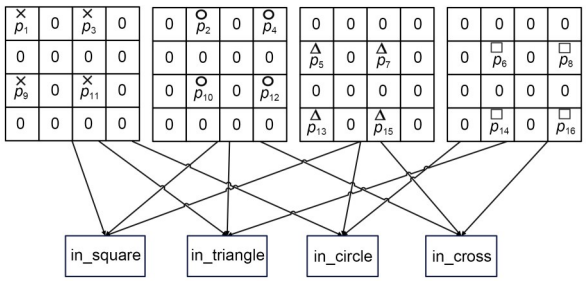

Fig. 8 Schematic of constructing input data

3.1.3 Predicted values

To predict the pixel values, the constructed input data are fed into the intelligent predictor individually. The resulting predictions are then merged to obtain the predicted image. Specifically, the input images, namely in_square, in_triangle, in_circle, and in_cross, are used as the inputs to the intelligent predictor. The corresponding outputs, namely out_square, out_triangle, out_ circle, and out_cross, represent the predicted pixel values for the sets of C_4, C_3, C_2, and C_1, respectively. Merging the outputs, out_square, out_triangle, out_circle, and out_cross produces the predicted image pre_C for the original image C.

3.1.4 Generation of the cover images

To generate the cover images, we replace the original pixel values of the intersection sets C_1–C_4 from the original image C with their corresponding positions in the predicted image pre_C. This process is illustrated in Fig. 9, where the preserved sets are highlighted in black. Consequently, we obtain cover images, namely C_{cross}, C_{circle}, C_{triangle}, and C_{square}, which represent the true values of the preserved sets C_1, C_2, C_3, and C_4, respectively.

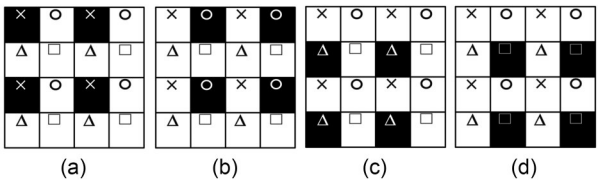

Fig. 9 Illustration of the cover image construction: (a) cover image C_{cross}; (b) cover image C_{circle}; (c) cover image C_{triangle}; (d) cover image C_{square}

3.1.5 Generation of the chaotic sequences and block shuffling

Three random initial values A_0, B_0, and C_0 are selected as keys, and the 3D chaotic image encryption method (Khade and Narnaware, 2012) is employed to generate three chaotic sequences A_α, B_β, and C_γ. Here, $A_0, B_0, C_0 \in (0, 1)$.

The original image C and the cover images are divided into non-overlapping blocks of size 2×2. Block shuffling is performed using the chaotic sequence A_α to obtain C', C'_{cross}, C'_{circle}, C'_{triangle}, and C'_{square}.

3.1.6 Additive secret sharing encryption

First, C', C'_{cross}, C'_{circle}, C'_{triangle}, and C'_{square} are decomposed into bit-planes using Eq. (2). Then, the decomposed bit-planes are randomly combined using Eqs. (5)–(9) to form bit-plane combination cb_n:

$$\text{cb}_n^1(i,j)=\sum_{k=0}^{a} b_n^k(i,j)\times 2^k, \tag{5}$$

$$\text{cb}_n^2(i,j)=\sum_{k=a+1}^{q} b_n^k(i,j)\times 2^k, \tag{6}$$

$$\text{cb}_n^3(i,j)=\sum_{k=q+1}^{t} b_n^k(i,j)\times 2^k, \tag{7}$$

$$\text{cb}_n^4(i,j)=\sum_{k=t+1}^{7} b_n^k(i,j)\times 2^k, \tag{8}$$

$$\begin{aligned} p_n(i,j)=&\sum_{k=0}^{a} b_n^k(i,j)\times 2^k+\sum_{k=a+1}^{q} b_n^k(i,j)\times 2^k\\ &+\sum_{k=q+1}^{t} b_n^k(i,j)\times 2^k+\sum_{k=t+1}^{7} b_n^k(i,j)\times 2^k, \end{aligned} \tag{9}$$

where $\text{cb}_n^1(i,j)$–$\text{cb}_n^4(i,j)$ represent the combination of the first to fourth bit-planes of the n^{th} pixel in the block at (i, j), and a, q, $t\in\{0, 1, 2, 3, 4, 5, 6, 7\}$. The randomness of the combination is achieved using chaotic sequences generated by the 3D chaotic mapping with A_0. Subsequently, using Eq. (10), the random combinations cb_n are split into maxm addends add, where add is a positive integer. The randomness of the split is achieved using chaotic sequences generated by the 3D chaotic mapping with B_0.

$$\text{cb}_n^r=\sum_{m=1}^{\text{maxm}} \text{add}_m^r. \tag{10}$$

Finally, Eq. (11) is used to randomly select the addends for merging to obtain the share C_share_n of C' and the shares cross_share_n, circle_share_n, triangle_share_n, and square_share_n of C'_{cross}, C'_{circle}, C'_{triangle}, C'_{square}, respectively:

$$\begin{aligned} \text{share}_n(i,j)=&\text{add}_m^1(i,j)+\text{add}_m^2(i,j)\\ &+\text{add}_m^3(i,j)+\text{add}_m^4(i,j), \end{aligned} \tag{11}$$

where n=1, 2, 3, ⋯. Note that the same additive component does not appear in the same combined bit-plane cb_n^r across different shares. The shared encrypted cover image generated from the original image is denoted as C_share_n, while the shared encrypted cover images generated from the predicted images C'_{cross}, C'_{circle}, C'_{triangle}, and C'_{square} are denoted as cross_share_n, circle_share_n, triangle_share_n, and square_share_n, respectively. The randomness of the selection is achieved using chaotic sequences generated by the 3D chaotic mapping with C_0.

We employ additive secret sharing for encryption, leveraging its robust distribution of confidential data across shares to reduce unauthorized access risks and enhance the overall security. This is crucial, especially in scenarios of betrayal or collusion attacks, because one, two, or three parties alone cannot access the information, and only four parties working together can reconstruct the information, which helps improve the security. Additive secret sharing ensures lossless image recovery during decryption, aligning seamlessly with our reversible data hiding goal. Additionally, it strengthens resistance to statistical attacks by introducing complexity, making it challenging for attackers to deduce meaningful information. The scheme security is further fortified by introducing randomness through 3D chaotic mapping, making it resilient against cryptographic attacks and resistant to reverse engineering based on knowledge of the randomness.

3.1.7 Vacating room using adaptive joint coding

C_share_n of the original image and cross_share_n, circle_share_n, triangle_share_n, and square_share_n of the cover images are divided into 2×2 non-overlapping blocks. The pixels in the C_share_n blocks are denoted as R, sp_1, sp_2, and sp_3 in raster scan order. The pixels in cross_share_n, circle_share_n, triangle_share_n, and square_share_n blocks are denoted as R, spp_1, spp_2, and spp_3 in raster scan order. The shared encrypted cover images cross_share_n, circle_share_n, triangle_share_n, and square_share_n are compared with the corresponding encrypted original image C_share_n on a block-by-block basis. The maximum prediction error within each block, denoted as maxe, is calculated using Eq. (12):

$$\text{maxe}=\max\left(\left|\text{spp}_1-\text{sp}_1\right|,\left|\text{spp}_2-\text{sp}_2\right|,\left|\text{spp}_3-\text{sp}_3\right|\right). \tag{12}$$

If the current block satisfies maxe≤τ&&(3+3×maxd)>T, encoding based on prediction error is employed. Otherwise, if the current block satisfies (maxe≤

τ&&(3+3×maxd)≤T)||(maxd≤τ&&maxe> τ), encoding based on bit-plane comparison is used (T=12, τ=7).

Our proposed adaptive joint encoding demonstrates an enhanced ability to adapt to diverse images, resulting in improved embedding rates and mitigating the risk associated with modifying pixels at specified coordinates. This feature underscores the flexibility and effectiveness of our approach in accommodating a wide range of image characteristics while maintaining robustness against potential challenges.

3.1.8 Generation of the position maps

If none of the above conditions are met, the block is considered non-embeddable. We generate a position non-embedable location map (NLP) of size $H\times W/4$ for the non-embeddable blocks. The corresponding positions in the map are marked as 0. The embeddable blocks are marked as 1. Additionally, we generate a position compression location map (CLP) to indicate the compression method used for each block. If the current block is compressed using prediction error based encoding, it is marked as 1. If the current block is compressed using bit-plane comparison based encoding, it is marked as 0. Other cases are not recorded. We use arithmetic coding to compress NLM and CLM into cNLM and cCLM, respectively. The lengths of cNLM and cCLM are denoted as length_cNLM and length_cCLM in $\log_2(M\times N/4)$ bits, respectively.

3.1.9 Pixel block rearrangement

All compressible blocks, compressed using prediction error based encoding, are rearranged at the beginning of the image. The compressed blocks obtained using bit-plane comparison based encoding are placed next in the arrangement. Non-embeddable blocks are placed at the end. Fig. 10 illustrates this arrangement, where compressed embeddable blocks obtained using prediction error based encoding are marked in gray, those obtained using bit-plane comparison based encoding are marked in white, and non-embeddable blocks are marked in black.

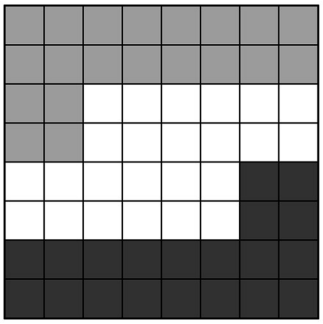

Fig. 10 Block rearrangement schematic

3.1.10 Auxiliary information embedding

The first pixel of each modified cover image after block permutation is used to record the image number. The top two bits represent the reserved set, where "00" represents cross_share$_n$, "01" represents circle_share$_n$, "10" represents triangle_share$_n$, and "11" represents square_share$_n$. The remaining six bits indicate that the cover image is the n^{th} share. If more shares need to be generated, additional pixels can be used to record the encoding. The values of the first pixel, length_cNLM, cNLM, length_cCLM, cCLM, and the coordinates of the last embeddable block are represented using 32 bits and are embedded into an available space. The resulting encrypted cover images cross_share$'_n$, circle_share$'_n$, triangle_share$'_n$, and square_share$'_n$ are obtained. Finally, the shares are distributed.

3.1.11 Implementation of the image owner algorithm

Step 1: Image preprocessing is implemented as in Section 3.1.1, and the original image C is divided into sets: cross set C_1, circle set C_2, triangle set C_3, and square set C_4 (Fig. 7).

Step 2: Constructing input data involves a combination of sets: in_square, in_triangle, in_circle, and in_cross (Fig. 8) in Section 3.1.2.

Step 3: Pixel value prediction involves individual predictions for sets C_1–C_4 with reference to Section 3.1.3.

Step 4: The generation of cover images is accomplished in Section 3.1.4. To generate the cover images, we replace the original pixel values of the intersection sets C_1-C_4 from the original image C with their corresponding positions in the predicted image pre_C (Fig. 9).

Step 5: Generating chaotic sequences involves initially selecting three random initial values A_0, B_0, and C_0 as keys and generating three chaotic sequences A_α, B_β, and C_γ in Section 3.1.5.

Step 6: The original image C and the cover images are divided into non-overlapping blocks of size 2×2. Block shuffling is performed using the chaotic sequence A_α to obtain C', C'_{cross}, C'_{circle}, C'_{triangle}, and C'_{square}.

Step 7: Additive secret sharing is used to encrypt the shuffled image C' and the cover images C'_{cross}, C'_{circle}, C'_{triangle}, and C'_{square}, thus obtaining encrypted images. Initially, the shuffled image C' and the cover images C'_{cross}, C'_{circle}, C'_{triangle}, and C'_{square} are decomposed into bit-planes using Eq. (2). Next, the decomposed bit-planes are randomly combined using Eqs. (5)–(9) to create bit-plane combinations cb_n. Subsequently, Eq. (10) is employed to split the random combinations cb_n into n addends add. Finally, Eq. (11) is used to randomly select addends for merging to obtain the shares C_share_n of the shuffled image C' and the shares cross_share_n, circle_share_n, triangle_ share_n, and square_share_n of the cover images C'_{cross}, C'_{circle}, C'_{triangle}, and C'_{square}.

Step 8: Adaptive joint coding is implemented as in Section 3.1.7 to compress the encrypted images, creating space for embedding and obtaining compressed images. If the current block satisfies $\text{maxe}\le\tau\&\&(3+3\times\text{maxd})>T$, encoding based on prediction error is employed; if the current block satisfies $(\text{maxe}\le\tau\&\&(3+3\times\text{maxd})\le T)\|(\text{maxd}\le\tau\&\&\text{maxe}>\tau)$, encoding based on bit-plane comparison is employed (T=12, τ=7).

Step 9: Position maps are generated as in Section 3.1.8.

Step 10: The rearranged images are obtained by pixel block rearrangement with compressed images as described in Section 3.1.9. Compressible blocks are placed using prediction error based encoding at the beginning, followed by blocks compressed through bit-plane comparison based encoding.

Step 11: Shared images are obtained by embedding auxiliary information in the rearranged images in Section 3.1.10.

Step 12: Shared images are distributed.

3.2 Data hiders

Step 1: Extract and decompress the location map.

Step 2: Encrypt the secret data by performing a bitwise XOR operation between the secret data and a chaotic sequence generated using A_0. The result of this operation is the encrypted secret data, denoted as ss.

Step 3: Embed the secret data. Use the location map to embed len_ss (the length of ss) and ss into the cover images using a bitwise substitution method, resulting in stego-images, namely $\text{ecross\_share}'_n$, $\text{ecircle\_share}'_n$, $\text{etriangle\_share}'_n$, and $\text{esquare\_share}'_n$.

Step 4: Transmit the stego-images to the receiver.

3.3 Receiver

Step 1: Extract the code, the first pixel value, and the position map.

Step 2: Extract the length embedding len_ss and the secret data ss embedded in the received encrypted images, and perform chaotic decryption to obtain the original secret data s.

Step 3: Recover the first pixel value and use the position NLM and CLM to recover the original arrangement of blocks. Using the image code, add the corresponding pixel values of the images with the first two bits as 00, 01, 10, and 11 to obtain $\text{re}C'_{\text{cross}}$, $\text{re}C'_{\text{circle}}$, $\text{re}C'_{\text{triangle}}$, $\text{re}C'_{\text{square}}$, respectively. Combine the corresponding reserved sets $\text{re}C'_{\text{cross}}$, $\text{re}C'_{\text{circle}}$, $\text{re}C'_{\text{triangle}}$, and $\text{re}C'_{\text{square}}$ and use the inverse block scrambling with key A_0 to obtain the original image.

Step 4: Recover the image in the case of partial share loss. If a certain reserved set is lost, extract the other three sets based on the image code and merge them into one image, setting the pixel values of the lost set to 0. Use the merged image as the input to the intelligent predictor to obtain the prediction results for the missing parts. Perform the same encryption process as the image owner on the prediction results of the missing parts to obtain shares of the prediction results. Extract error codes based on the position map and use error correction to recover the original pixel values of the lost parts. Add up the recovered shares to obtain the original values of the lost set, thus achieving lossless restoration of the cover images. Note that since data embedding and image encryption are separable, the data extraction in step 2 and the image restoration in steps 3 and 4 are independent.

4 Experiments

4.1 Experimental setting

Secret sharing based RDHEI offers superior security compared to stream cipher encryption based RDHEI and lower algorithmic complexity compared to homomorphic encryption based RDHEI. Additionally, it provides fault tolerance and distributed storage advantages, effectively addressing data corruption or loss scenarios and thwarting collaborative attacks. This approach is well-suited for privacy protection or decentralized storage needs. Our proposed RDHEI

scheme based on additive secret sharing not only ensures high confidentiality but also simplifies image recovery, significantly enhancing embedding efficiency and capacity for hiding parties.

To better evaluate the performance and demonstrate the advancement of the proposed RDHEI scheme, we conducted experiments on four aspects: performance of the proposed intelligent predictor, ablation experiment, encryption performance, and embedding capacity. To better demonstrate the effectiveness of our proposed scheme and compare it with similar advanced methods, we selected nine 512×512 grayscale images from the test dataset for the experimental result display. The nine images are shown in Fig. 11, namely Airplane, Baboon, Barbara, Boat, Jetplane, Lena, Pepper, Man, and Tiffany. Additionally, a grayscale image of size 256×256 is shown as the secret data. Furthermore, to enhance the persuasiveness of the experimental results and confirm the generality of the proposed method, we conducted experiments on three datasets: BOSSBase (Bas et al., 2011), BOWS2 (Ankita Gupta, 2023), and UCID.v2 (Schaefer and Stich, 2004). As neural network models are used for prediction, the time complexity of the neural network can be calculated as $O\left(\sum_{l=1}^{D} F_l^2 K_l^2 C_{l-1} C_l\right)$, where D represents the number of convolution layers in the neural network, F signifies the side length of the output feature map from each convolution kernel, K denotes the side length of each convolution kernel, and C represents the number of output channels in the l^{th} convolution layer.

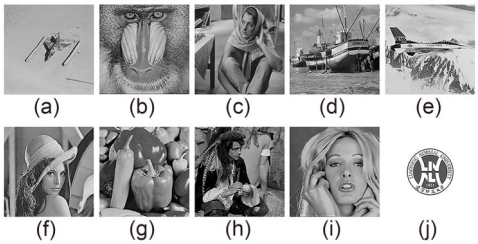

Fig. 11 Standard grayscale images of size 512×512 for the experiments: (a) Airplane; (b) Baboon; (c) Barbara; (d) Boat; (e) Jetplane; (f) Lena; (g) Pepper; (h) Man; (i) Tiffany; (j) secret data sized 256×256

The time complexity of the compression module, embedding module, and extraction module is determined by the algorithm and data characteristics as follows:

Compression module: $O(N^2)$;

Embedding module: $O(N^2)$;

Extraction module: $O(N^2)$;

Total complexity: $O\left(\sum_{l=1}^{D} F_l^2 K_l^2 C_{l-1} C_l\right)+O(N^2)$.

4.2 Experimental results

The experimental results of the proposed RDHEI method with threshold selection $(9, 12)$ using the Lena image as the cover image are shown in Fig. 12. Fig. 12a depicts the standard grayscale image Lena with the size of 512×512. Fig. 12b represents the secret data as a grayscale image with the size of 256×256. Figs. 12c–12e depict the stego-images, denoted as ecross_share$'_1$, ecross_share$'_2$, and ecross_share$'_3$, obtained by embedding the secret data from Fig. 12b into images cross_share$'_1$, cross_share$'_2$, and cross_share$'_3$, respectively. The peak signal-to-noise ratio (PSNR) values for Figs. 12c–12e are measured as 8.0686, 7.6982, and 8.7648 respectively, while the structural

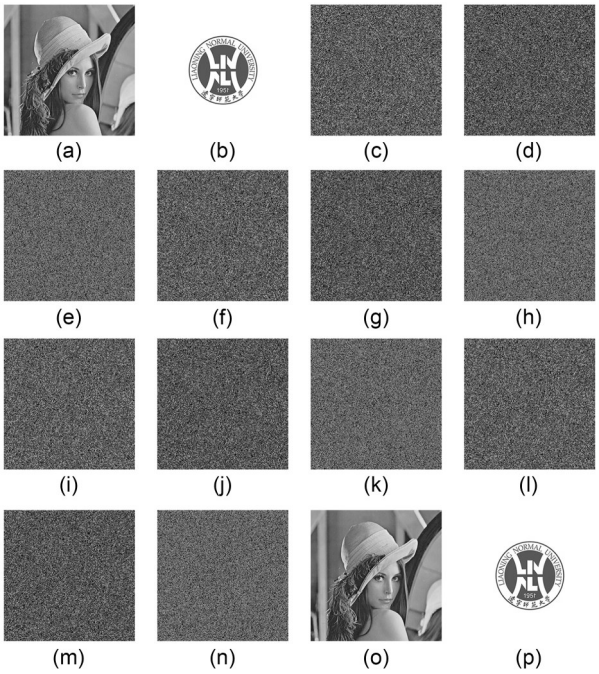

Fig. 12 Experimental results of Lena image as a cover image for threshold selection (9, 12): (a) original image Lena sized 512×512; (b) secret data sized 256×256; (c) ecross_share$'_1$; (d) ecross_share$'_2$; (e) ecross_share$'_3$; (f) ecircle_share$'_1$; (g) ecircle_share$'_2$; (h) ecircle_share$'_3$; (i) etriangle_share$'_1$; (j) etriangle_share$'_2$; (k) etriangle_share$'_3$; (l) esquare_share$'_1$; (m) esquare_share$'_2$; (n) esquare_share$'_3$; (o) restored image sized 512×512; (p) extracted secret data sized 256×256

similarity (SSIM) values are calculated as 0.008 41, 0.007 70, and 0.008 60, respectively. Figs. 12f–12h show the stego-images, represented as ecircle_share$'_1$, ecircle_share$'_2$, and ecircle_share$'_3$, resulting from the embedding of the secret data from Fig. 12b into images circle_share$'_1$, circle_share$'_2$, and circle_share$'_3$, respectively. The PSNR values for Figs. 12f–12h are determined as 8.0706, 7.6944, and 8.7591, with the corresponding SSIM values of 0.007 67, 0.007 69, and 0.009 45, respectively.

Figs. 12i–12k show stego-images (etriangle_share$'_1$, etriangle_share$'_2$, and etriangle_share$'_3$) created by embedding secret data from Fig. 12b into the corresponding images (triangle_share$'_1$, triangle_share$'_2$, and triangle_share$'_3$). PSNR values are 8.0785, 7.6974, and 8.7778, with the SSIM values of 0.008 48, 0.007 56, and 0.009 57, respectively.

Figs. 12l–12n present the stego-images, represented as esquare_share$'_1$, esquare_share$'_2$, and esquare_share$'_3$, resulting from the embedding of the secret data from Fig. 12b into images square_share$'_1$, square_share$'_2$, and square_share$'_3$, respectively. The PSNR values for Figs. 12l–12n are determined as 8.0755, 7.6940, and 8.7555, with the corresponding SSIM values of 0.008 16, 0.007 56, and 0.008 78, respectively. It is observed that the quality of the encrypted images is relatively low, with PSNR values close to 8 and low SSIM values. This implies that the additive secret sharing encryption process introduces significant random noise and disrupts the structural similarity of the images, achieving the desired effect of concealing the content of the cover image. Fig. 12o represents the recovered image, while Fig. 12p displays the extracted secret data. The PSNR value between Fig. 12b and Fig. 12p is infinite, indicating that the extracted secret data remain unchanged and identical to the original secret data, thus demonstrating the complete reversibility of our approach. Additionally, the PSNR value between Fig. 12a and Fig. 12o is infinite, which signifies that the recovered image is identical to the original image without any distortion or differences, thus verifying the lossless recovery capability of our scheme.

4.3 Training

The training dataset was selected from the widely used ImageNet dataset (Deng et al., 2009). The test sets come from BOSSBase, BOWS2, and UCID.v2. For optimization, we employed the backpropagation technique (LeCun et al., 1998) along with the Adam optimizer (Kingma and Ba, 2017). These methods allow us to iteratively refine the model performance. We performed several training iterations to enhance the predictive capabilities of the proposed ResNet-based predictor. The training process involved adjusting the model's internal parameters to minimize the prediction errors, using the training dataset (the training specifics are detailed in the supplementary materials).

4.4 Prediction accuracy

We compared our approach with state-of-the-art conventional prediction methods (Chang et al., 2021; Wang XY et al., 2021, 2023; Fu et al., 2022; Jeena and Shreelekshmi, 2023; Ni and Bi, 2023; Zhang et al., 2023) to evaluate its accuracy. The experimental results are illustrated in Fig. 13. The predictor yielded the numbers of prediction errors within the range of [−5, 5] as 139 930, 208 600, 238 700, and 242 500 for Baboon, Barbara, Boat, and Lena, respectively. Compared to the conventional prediction methods (Chang et al., 2021; Wang XY et al., 2021, 2023; Fu et al., 2022; Jeena and Shreelekshmi, 2023; Ni and Bi, 2023; Zhang et al., 2023), our predictor exhibited a higher number of prediction errors within the range of [−5, 5], particularly within the range of [−1, 1]. This indicates that our predictor can approach the original pixel values more closely and demonstrates superiority in accurately predicting pixel values. Based on experimental results, we concluded that the intelligent predictor outperforms traditional methods in pixel value approximation and prediction.

4.5 Ablation experiment

To assess spatial attention effect, we deleted the spatial attention residual block and checked the model's new predictions. In Table 1, we compared the number of zero prediction errors before and after removing the spatial attention blocks, highlighting their significance in reducing errors. The numbers of instances with zero prediction errors on nine test images Airplane, Baboon, Barbara, Boat, Jetplane, Lena, Pepper, Man, and Tiffany after removing the SARBs were 33 585, 13 863, 29 471, 33 296, 17 955, 33 283, 34 929, 15 317, and 21 576, respectively, and compared to our proposed intelligent predictor, they decreased by 4852, 6396,

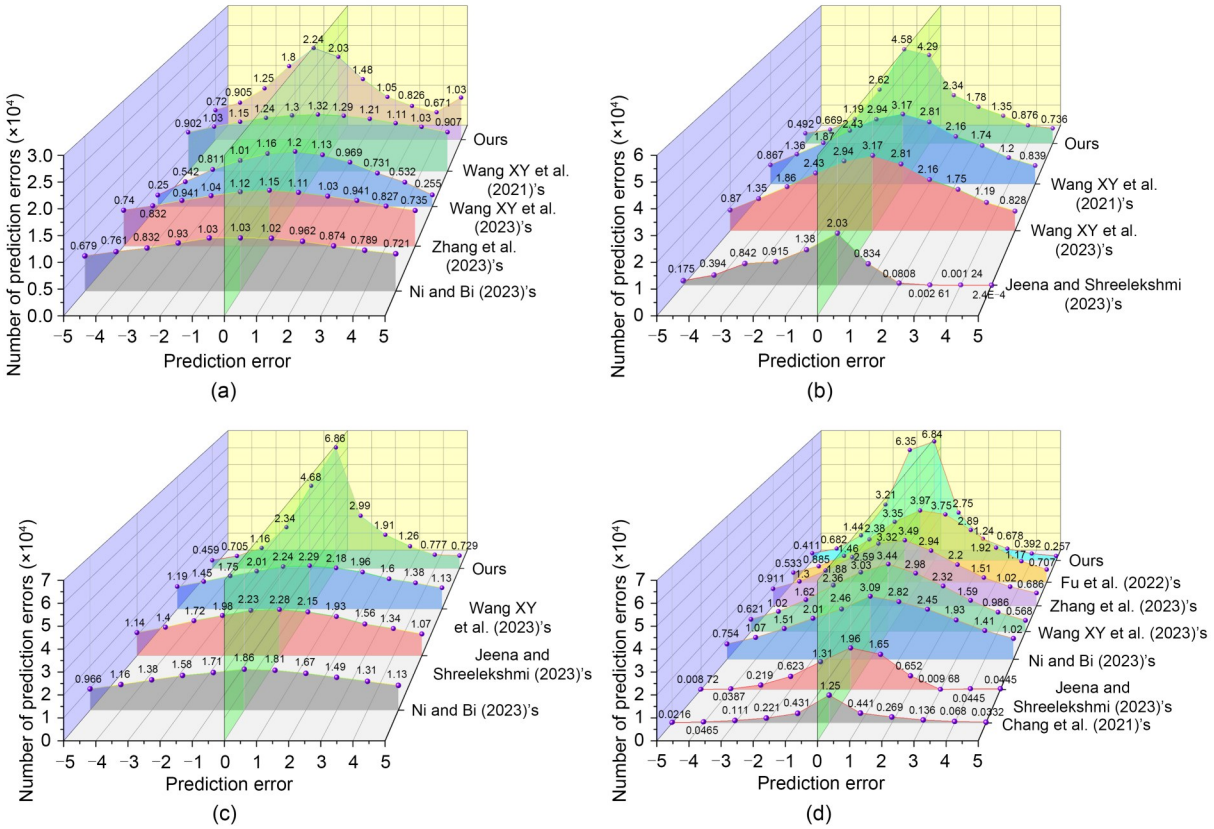

Fig. 13 Comparison of the number of prediction errors within the range of [−5, 5] with advanced methods: (a) Baboon; (b) Barbara; (c) Boat; (d) Lena

Table 1 Comparison of the number of zero prediction errors between the model without SARB and the proposed intelligent predictor

Image	Number of zero prediction errors	
	The predictor without SARB	The proposed intelligent predictor
Airplane	33 585	38 437
Baboon	13 863	20 259
Barbara	29 471	42 941
Boat	33 296	68 561
Jetplane	17 955	21 760
Lena	33 283	68 416
Pepper	34 929	53 339
Man	15 317	26 139
Tiffany	21 576	25 216

13 470, 35 265, 3805, 35 133, 18 410, 10 822, and 3640, respectively. Based on these results, we can conclude that SARBs have a positive impact on the prediction performance of the model, and removing these blocks will result in more prediction errors.

4.6 Encryption performance

Our proposed RDHEI scheme encrypts images using additive secret sharing, safeguarding them from unauthorized access. This ensures confidentiality as the ciphertext cannot be restored to the original image without decryption.

We evaluated scheme performance using various image quality assessment metrics: (1) PSNR, to measure noise distortion between the encrypted and original images (Lower PSNR values indicate lower similarity); (2) SSIM, to evaluate structural similarity between the encrypted and original images (Closer-to-zero SSIM values suggest greater dissimilarity); (3) number of pixels change rate (NPCR), to measure variation between two encrypted images (Higher NPCR values indicate greater variation); (4) unified average changed intensity (UACI), to assesse average pixel intensity variation between encrypted images (Higher UACI values signify larger average changes). Evaluation metrics for the proposed RDHEI scheme on the Lena image, with n

set to 12 for the generation of 12 shared images, are presented in Table 2. The average PSNR value was 8.1779 dB, which is below 10. The average SSIM value was 0.008 30, indicating a significant dissimilarity to the original image. The average NPCR value was 0.996 98, approaching 1, indicating significant differences between the encrypted images. The average UACI value was 0.557 23, suggesting a substantial average pixel intensity variation between the encrypted images. These metrics demonstrate the sufficient chaotic properties of the images.

Table 2 Encryption performance of PSNR, SSIM, NPCR, and UACI on Lena image of (9, 12) threshold generating 12 shares

Image	PSNR (dB)	SSIM	NPCR	UACI
cross_share$'_1$	8.0686	0.008 41	0.997 52	0.542 14
cross_share$'_2$	7.6982	0.007 70	0.997 81	0.561 54
cross_share$'_3$	8.7648	0.008 60	0.995 59	0.567 75
circle_share$'_1$	8.0706	0.007 67	0.997 31	0.541 83
circle_share$'_2$	7.6944	0.007 69	0.997 91	0.561 94
circle_share$'_3$	8.7591	0.009 45	0.995 70	0.567 28
triangle_share$'_1$	8.0785	0.008 48	0.997 37	0.541 42
triangle_share$'_2$	7.6974	0.007 56	0.997 83	0.561 64
triangle_share$'_3$	8.7778	0.009 57	0.995 94	0.569 27
square_share$'_1$	8.0755	0.008 16	0.997 31	0.541 76
square_share$'_2$	7.6940	0.007 56	0.997 77	0.561 37
square_share$'_3$	8.7555	0.008 78	0.995 73	0.568 81

In addition to these metrics, correlation coefficients were employed to evaluate the performance of the RDHEI scheme. The correlation coefficients are depicted in Table S2 in the supplementary materials, showing the vertical correlation coefficient, horizontal correlation coefficient, and diagonal correlation coefficient for the shared encrypted images cross_share$'_1$, cross_share$'_2$, and cross_share$'_3$, preserving true values of the cross set. The average correlation coefficient values for Figs. 11a–11i are 0.1301, 0.1005, 0.1512, 0.1610, 0.1435, 0.1554, 0.1547, 0.1417, and 0.1399, respectively.

4.7 Embedding capacity

In our experimental results, the embedding capacity (EC) represents the net capacity excluding auxiliary information. Table 3 presents the EC and embedding rate (ER) on nine test images. It can be observed that all the ER values exceeded 3.9.

Table 3 Embedding capacity (EC) and embedding rate (ER) of the proposed method on nine test images

Image	ER (bpp)	EC (bit)
Airplane	4.2895	1 124 478
Baboon	3.9018	1 022 855
Barbara	4.2113	1 103 989
Boat	4.3770	1 147 408
Jetplane	4.3025	1 127 899
Lena	4.3966	1 152 545
Pepper	4.3601	1 142 979
Man	4.1089	1 077 132
Tiffany	4.0743	1 068 063

To better demonstrate the capacity advantage of our proposed scheme, a comparison was made with excellent RDHEI schemes (Wu et al., 2020; Yin et al., 2020, 2022; Qin et al., 2021; Chen et al., 2022; Hua et al., 2022, 2023; Yu et al., 2022, 2023; Wang YM et al., 2023) as shown in Fig. 14. In Fig. 14a, on Lena, Baboon, Airplane, Man, and Tiffany, Wu et al. (2020)'s method is an RRBE type with capacity values of 2.6447, 0.9692, 2.8578, 2.479, and 2.6515 respectively; Yu et al. (2022)'s method is an RRBE type with capacity values of 3.019, 1.4596, 3.9872, 2.651, and 3.0901 respectively; Yin et al. (2022)'s method is an RRBE type with capacity values of 3.075, 1.383, 3.402, 2.635, and 3.149 respectively; Yin et al. (2020)'s method is an RRBE type with capacity values of 2.583, 1.066, 3.725, 2.349, and 2.824 respectively. Our proposed method achieved capacity values of 4.3966, 3.9018, 4.2895, 4.1089, and 4.0743 on Lena, Baboon, Airplane, Man, and Tiffany, respectively. Compared to the advanced RRBE-type RDHEI schemes of the same category, our proposed scheme demonstrated a clear advantage in terms of capacity.

In Fig. 14b, on Lena, Baboon, Pepper, Boat, and Airplane, Hua et al. (2022)'s method is a VRAE type with capacity values of 2.9129, 1.2522, 2.5696, and 3.6386 respectively; Qin et al. (2021)'s method is a VRAE type with capacity values of 1.5812, 0.5490, 1.8883, 1.4521, and 1.5321 respectively; Chen et al. (2022)'s method is a VRBE type with a capacity of 3.5000. Our proposed method achieved capacity values of 4.3966, 3.9018, 4.3601, 4.3770, and 4.2895 on Lena, Baboon, Pepper, Boat, and Airplane, respectively. Compared to VRAE and VRBE type RDHEI schemes,

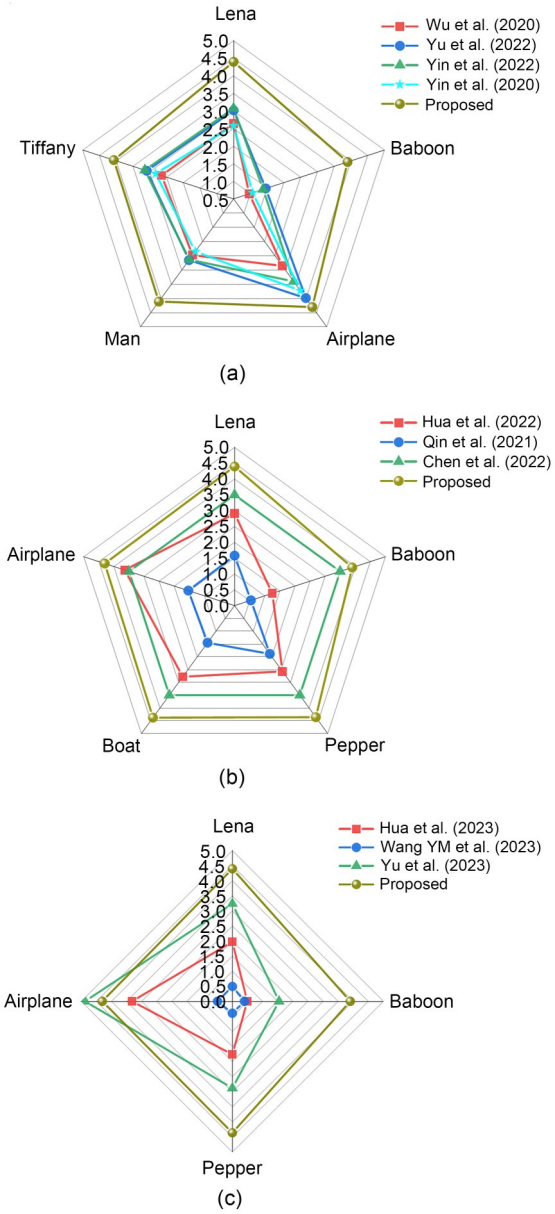

Fig. 14 Comparison of capacity with similar schemes: (a) comparison with methods (Wu et al., 2020; Yin et al., 2020, 2022; Yu et al., 2022) on Lena, Baboon, Airplane, Man, and Tiffany; (b) comparison with methods (Qin et al., 2021; Chen et al., 2022; Hua et al., 2022) on Lena, Baboon, Pepper, Boat, and Airplane; (c) comparison with methods (Hua et al., 2023; Wang YM et al., 2023; Yu et al., 2023) on Lena, Baboon, Pepper, and Airplane

our method maintained an advantage in terms of ER. In Fig. 14c, on Lena, Baboon, Pepper, and Airplane, Hua et al. (2023)'s method is a secret sharing method with capacity values of 1.9735, 0.5145, 1.7629, and 3.3212 respectively; Wang YM et al. (2023)'s method is a pixel-value-ordering method with capacity values of 0.4922, 0.3997, 0.3942, and 0.4889 respectively; Yu et al. (2023)'s method is a secret sharing method with capacity values of 3.2330, 1.5440, 2.8752, and 4.6865, respectively.

We conducted capacity experiments on three datasets, BOSSBase, BOWS2, and UCID.v2, with average capacities of 4.4094, 4.5920, and 4.3838, respectively, as shown in Table 4. Note that we used all the images from UCID.v2 and randomly selected 2000 images from the BOSSBase and BOWS2 datasets to calculate the average capacity.

Table 4 Average embedding capacity (EC) and embedding rate (ER) on datasets BOSSBase, BOWS2, and UCID.v2

Database	Average EC (bit)	Average ER (bpp)
BOSSBase	1 155 885.169	4.4094
BOWS2	1 203 773.084	4.5920
UCID.v2	1 149 189.185	4.3838

4.8 Security analysis

In this subsection, we explored the uncertainty and security of the additive secret sharing encryption method used in our proposed RDHEI method. The encryption system employed is a randomized and uncertain encryption scheme. When the same combination scheme is applied twice to the cover image for encryption, the resulting encrypted images exhibit significant differences.

These differences arise from the reliance on randomly generated chaotic sequences in Eqs. (5)–(11), where each execution of the encryption process uses different random values. Even within the same encryption operation, the random values for each encrypted pixel are distinct. All operations in this scheme, such as bit-plane separation, bit-plane combination, addition splitting, and share generation, possess randomness without the need to record or transmit these random values, thereby further enhancing security. In the experimental results, we encrypted Lena twice using two sets of random keys, as shown in Fig. 15 in the two rows. By examining the pixel values at the same position in these two sets of encrypted images, we can observe significant differences.

It is evident that the pixel values at the same position obtained from two encrypted images are entirely different. This encryption system, characterized by

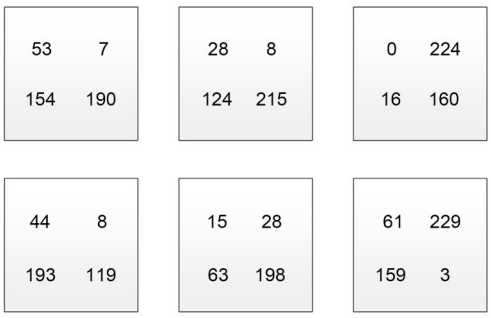

Fig. 15 Specific pixel values of the same area in Lena

uncertainty and randomness, is capable of effectively resisting numerous potential attacks.

5 Conclusions

We introduce a novel approach for reversible data hiding in encrypted data, combining intelligent prediction and additive secret sharing. Our method comprises key components: training an intelligent predictor, encrypted predictions, additive encryption, and joint encoding for embedding. It offers efficient hiding, adaptive encoding, and lossless recovery. The method possesses several notable advantages: first, it significantly enhances the efficiency of information concealment by employing an intelligent predictor; second, the utilization of additive secret sharing mechanism ensures robust encryption, achieving a good balance between security and efficiency; third, the application of adaptive joint encoding technology maximizes the capacity of hidden information; last, thanks to accurate prediction mechanisms, the method ensures lossless recovery of the original information even in the case of lost shares. Our method represents a significant advancement in reversible data hiding in encrypted data. Future enhancements aim to refine and extend its capabilities, ensuring relevance and impact in the field.

Contributors

Ziyi ZHOU designed the research. Chengyue WANG and Kexun YAN processed the data. Hui SHI drafted the paper. Xin PANG helped organize the paper. Ziyi ZHOU and Hui SHI revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Ankita Gupta NCU, 2023. BOWS2, Mendeley Data, V1. https://data.mendeley.com/datasets/kb3ngxfmjw/1 [Accessed on Oct. 1, 2023].

Bas P, Filler T, Pevný T, 2011. "Break our steganographic system": the ins and outs of organizing BOSS. Proc 13th Int Workshop on Information Hiding, p.59-70. https://doi.org/10.1007/978-3-642-24178-9_5

Chang J, Ding F, Li XL, et al., 2021. Hybrid prediction-based pixel-value-ordering method for reversible data hiding. *J Vis Commun Image Represent*, 77:103097. https://doi.org/10.1016/j.jvcir.2021.103097

Chen B, Lu W, Huang JW, et al., 2022. Secret sharing based reversible data hiding in encrypted images with multiple data-hiders. *IEEE Trans Depend Sec Comput*, 19(2):978-991. https://doi.org/10.1109/TDSC.2020.3011923

Deng J, Dong W, Socher R, et al., 2009. ImageNet: a large-scale hierarchical image database. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.248-255. https://doi.org/10.1109/CVPR.2009.5206848

Fu ZF, Gong MX, Long GQ, et al., 2022. Efficient capacity-distortion reversible data hiding based on combining multi-peak embedding with local complexity. *Appl Intell*, 52(11):13006-13026. https://doi.org/10.1007/s10489-022-03323-8

Hua ZY, Wang YX, Yi S, et al., 2022. Reversible data hiding in encrypted images using cipher-feedback secret sharing. *IEEE Trans Circ Syst Video Technol*, 32(8):4968-4982. https://doi.org/10.1109/TCSVT.2022.3140974

Hua ZY, Liu XY, Zheng YF, et al., 2023. Reversible data hiding over encrypted images via preprocessing-free matrix secret sharing. *IEEE Trans Circ Syst Video Technol*, 34(3):1799-1814. https://doi.org/10.1109/TCSVT.2023.3298803

Jeena P, Shreelekshmi R, 2023. High capacity reversible data hiding in encrypted images using block labeling. *Multim Tools Appl*, 82(17):25883-25898. https://doi.org/10.1007/s11042-023-14455-5

Khade PN, Narnaware M, 2012. 3D chaotic functions for image encryption. *IJCSI Int J Comput Sci Iss*, 9(3):323-328.

Kingma DP, Ba J, 2017. Adam: a method for stochastic optimization. https://arxiv.org/abs/1412.6980

LeCun Y, Bottou L, Bengio Y, et al., 1998. Gradient-based learning applied to document recognition. *Proc IEEE*, 86(11):2278-2324. https://doi.org/10.1109/5.726791

Ni BB, Bi WH, 2023. New predictor-based schemes for reversible data hiding. *Multim Tools Appl*, 82(4):5923-5948. https://doi.org/10.1007/s11042-022-13396-9

Qi KL, Zhang MQ, Di FQ, et al., 2023. High capacity reversible data hiding in encrypted images based on adaptive quadtree partitioning and MSB prediction. *Front Inform Technol Electron Eng*, 24(8):1156-1168. https://doi.org/10.1631/FITEE.2200501

Qin C, Jiang CY, Mo Q, et al., 2021. Reversible data hiding in

encrypted image via secret sharing based on $GF(p)$ and $GF(2^8)$. *IEEE Trans Circ Syst Video Technol*, 32(4):1928-1941. https://doi.org/10.1109/TCSVT.2021.3091319

Schaefer G, Stich M, 2004. UCID: an uncompressed color image database. Proc SPIE 5307, Storage and Retrieval Methods and Applications for Multimedia, p.472-480. https://doi.org/10.1117/12.525375

Thodi DM, Rodríguez JJ, 2007. Expansion embedding techniques for reversible watermarking. *IEEE Trans Image Process*, 16(3):721-730. https://doi.org/10.1109/TIP.2006.891046

Wang XY, Wang XY, Ma B, et al., 2021. High precision error prediction algorithm based on ridge regression predictor for reversible data hiding. *IEEE Signal Process Lett*, 28:1125-1129. https://doi.org/10.1109/LSP.2021.3080181

Wang XY, Wang XY, Ma B, et al., 2023. High-performance reversible data hiding based on ridge regression prediction algorithm. *Signal Process*, 204:108818. https://doi.org/10.1016/j.sigpro.2022.108818

Wang YM, Xiong GQ, He WG, 2023. High-capacity reversible data hiding in encrypted images based on pixel-value-ordering and histogram shifting. *Expert Syst Appl*, 211:118600. https://doi.org/10.1016/j.eswa.2022.118600

Weinberger MJ, Seroussi G, Sapiro G, 2000. The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS. *IEEE Trans Image Process*, 9(8):1309-1324. https://doi.org/10.1109/83.855427

Woo S, Park J, Lee JY, et al., 2018. CBAM: convolutional block attention module. Proc 15th European Conf on Computer Vision, p.3-19. https://doi.org/10.1007/978-3-030-01234-2_1

Wu YQ, Xiang YZ, Guo YT, et al., 2020. An improved reversible data hiding in encrypted images using parametric binary tree labeling. *IEEE Trans Multim*, 22(8):1929-1938. https://doi.org/10.1109/TMM.2019.2952979

Xiang SJ, Luo XR, 2018. Reversible data hiding in homomorphic encrypted domain by mirroring ciphertext group. *IEEE Trans Circ Syst Video Technol*, 28(11):3099-3110. https://doi.org/10.1109/TCSVT.2017.2742023

Yan XH, Li LL, Chen J, et al., 2023. Public key based bidirectional shadow image authentication without pixel expansion in image secret sharing. *Front Inform Technol Electron Eng*, 24(1):88-103. https://doi.org/10.1631/FITEE.2200118

Yin ZX, Xiang YZ, Zhang XP, 2020. Reversible data hiding in encrypted images based on multi-MSB prediction and Huffman coding. *IEEE Trans Multim*, 22(4):874-884. https://doi.org/10.1109/TMM.2019.2936314

Yin ZX, Peng YY, Xiang YZ, 2022. Reversible data hiding in encrypted images based on pixel prediction and bit-plane compression. *IEEE Trans Depend Sec Comput*, 19(2):992-1002. https://doi.org/10.1109/TDSC.2020.3019490

Yu CQ, Zhang XQ, Zhang XP, et al., 2022. Reversible data hiding with hierarchical embedding for encrypted images. *IEEE Trans Circ Syst Video Technol*, 32(2):451-466. https://doi.org/10.1109/TCSVT.2021.3062947

Yu CQ, Zhang XQ, Qin C, et al., 2023. Reversible data hiding in encrypted images with secret sharing and hybrid coding. *IEEE Trans Circ Syst Video Technol*, 33(11):6443-6458. https://doi.org/10.1109/TCSVT.2023.3270882

Zhang XR, Pan ZB, Zhou Q, et al., 2023. A novel two-level embedding pattern for grayscale-invariant reversible data hiding. *Multim Tools Appl*, 82(22):33911-33935. https://doi.org/10.1007/s11042-023-14789-0

List of supplementary materials