Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.085342

ARTICLE

VARStego: A Reversible Local Variance-Based Steganographic Method

Basten Andika Salim[1], Adifa Widyadhani Chanda D'Layla[1], Ntivuguruzwa Jean De La Croix[2], Tohari Ahmad[1,*] and Kambombo Mtonga[3]

[1]Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
[2]Faculty of Computing and Information Sciences, University of Lay Adventists of Kigali (UNILAK), Kigali, Rwanda
[3]School of Science and Technology, Malawi University of Business and Applied Sciences, Blantyre, Malawi

*Corresponding Author: Tohari Ahmad. Email: tohari@if.its.ac.id or tohari@its.ac.id

Received: 09 May 2026; Accepted: 29 July 2026; Published: 15 September 2026

ABSTRACT: The modernization of communication and healthcare environments has introduced critical security challenges, as all transmission is almost certainly done through digital networks prone to attacks. To counteract this, researchers have worked to create methods of data concealment, hiding the existence of sensitive data itself from prying eyes. However, many of these methods lack the necessary ability to balance imperceptibility and payload capacity. In addition, different from generic images, medical images must preserve their structural and visual integrity, requiring frameworks that prioritize maintaining high similarity between images or, at times, complete recovery of the original image. This paper introduces VARStego, a reversible dual-layered steganographic framework designed to conceal confidential patient information within medical images. VARStego uses the Huffman algorithm to compress all patient information in a separate layer of the framework, while a localized variance-based algorithm is employed to analyze medical images to find rough and complex regions. Utilizing these regions, the VARStego embeds data within selected pixels using a redundancy-based algorithm, further improving payload capacity. The proposed VARStego method is tested on the DICOM Library dataset, containing Digital Imaging and Communications in Medicine (DICOM) formatted files and the Kaggle dataset, providing computed tomography (CT) medical images. VARStego achieved a Peak Signal-to-Noise Ratio (PSNR) of 72.192 dB and a perfect Structural Similarity Index Measure of up to 1, ensuring that image quality is maintained. These scores show minimal degradation as the payload size increases from 1 to 50 kilobyte for American Standard Code for Information Interchange (ASCII) payload and 1 to 100 kilobit for bitstring payload, demonstrating the method's consistent performance.

KEYWORDS: Cybersecurity; information hiding; information security; electronic health records; ICT infrastructure; steganography

1 Introduction

The rapid advancement of software and hardware technologies has fundamentally transformed how data is generated, transmitted, and stored across nearly every professional domain. In the medical field, the digitalization of clinical workflows has accelerated the generation of massive volumes of sensitive patient data, ranging from personal identity records to high-resolution diagnostic imagery [1]. These Electronic Health Records (EHR) serve as the backbone of modern patient care. However, their digital nature introduces significant security vulnerabilities. Protecting the confidentiality of EHR while ensuring they remain accessible to authorized personnel has become a paramount challenge, as any breach compromises patient privacy and may result in serious legal and clinical consequences [2].

Several techniques have been developed to address these concerns, with cryptography and steganography emerging as the two principal approaches. Cryptography secures the content of a message by rendering it unintelligible; however, the resulting ciphertext often makes the presence of sensitive information conspicuous to observers [3], increasing the risk of targeted interception. Steganography, by contrast, conceals the very existence of the message by embedding it within innocent-looking cover media [4], as illustrated in Fig. 1. In the digital domain, this is achieved by utilizing various digital carriers, such as digital images, videos, or even audio files. Among these, digital images are the most widely utilized due to their high frequency in online communication and the inherent spatial redundancy within pixel arrays, a property that is especially advantageous in medical data, where X-rays, Computed Tomography (CT) scans, Magnetic Resonance Imaging (MRI), and Ultrasound images are available.

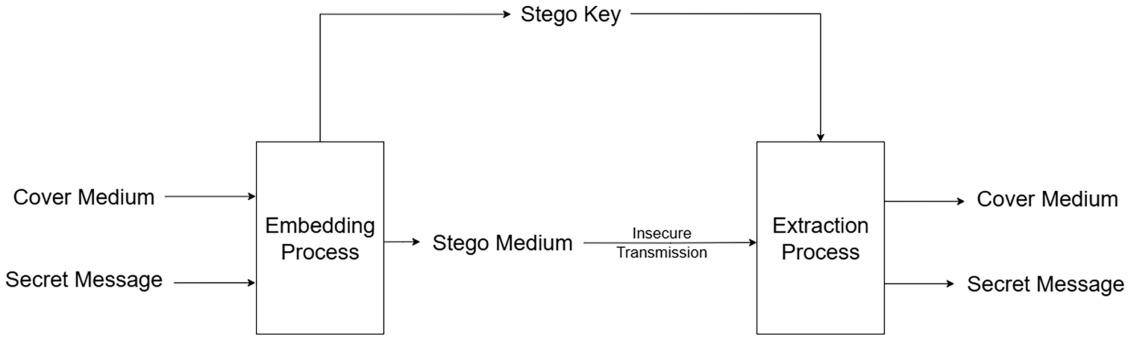

Figure 1: General flow of steganographic methods.

Digital image steganography primarily operates by modifying pixel values of a cover image to embed secret data, typically targeting the Least Significant Bits (LSB) to minimize visual distortion [5]. A significant and persisting challenge lies in balancing payload capacity with imperceptibility [6]. High-capacity embedding without regard for visual fidelity results in noticeable artifacts, particularly in smooth regions where the human eye is highly sensitive to subtle intensity changes [7,8]. Adaptive approaches have been proposed to concentrate payload within high-texture or high-edge-density regions, where pixel modifications are naturally masked [9–11]. Despite that, a critical gap remains in that most spatial-domain methods do not apply pre-embedding optimization to the secret payload itself. Raw, uncompressed data is inserted directly, consuming more pixel capacity than is strictly necessary, which enlarges the statistical footprint and raises the likelihood of steganalytic detection. In medical steganography, this is compounded by the additional requirement for exact reversibility, the ability to fully recover the original cover image after extraction [12] which further constrains the design space and rules out many conventional strategies. These limitations point to a clear need for a unified approach that simultaneously minimizes payload size before embedding and restricts modifications to statistically suitable image regions.

This paper proposes a reversible dual-layered steganographic framework that directly addresses the above limitations through two complementary mechanisms. The first layer applies Huffman entropy coding to compress the secret payload prior to insertion, exploiting statistical redundancy inherent in most data to significantly reduce the total number of bits to be embedded. By minimizing required pixel modifications at the source, this pre-embedding compression directly reduces the statistical footprint and lowers detectability [13,14]. The second layer implements a localized variance-based adaptive embedding strategy. By computing the neighborhood variance (S^2) of each pixel and comparing it to the maximum local variance (S^2_{max}), the system dynamically categorizes any given pixel to be residing in one of three areas: lowly textured regions, moderately textured regions, and highly textured regions. Embedding is performed

exclusively in high-variance regions where modifications are mathematically masked by natural intensity fluctuations, while smooth regions are fully preserved to maintain visual fidelity. This two-layer architecture directly addresses the dual problem identified in the literature: it reduces payload size before embedding and concentrates modifications where they are least detectable, achieving a tighter balance between capacity, imperceptibility, and reversibility than either mechanism alone can provide.

The main contributions of our work are as follows:

1. Implementation of a localized, variance-based adaptive embedding strategy that optimizes the trade-off between payload capacity and visual imperceptibility.
2. Integration of Huffman entropy coding to maximize data density and effective embedding capacity.
3. Introduction of a redundancy-based logical embedding mechanism that utilizes statistical redundancies in medium-variance regions to increase capacity without image degradation.

The rest of this paper is organized as follows: Section 2 reviews the current state of the art and existing literature. Section 3 describes the proposed methodology and its stages in detail. Section 4 provides experimental results from the direct application of the proposed method, and conclusions are drawn in Section 5.

2 Literature Review

Recent advancements have prioritized enhancing the security and efficiency of steganographic methods, aligning with steganography's main concerns. While the Least Significant Bits (LSB) method serves as a foundational technique due to its simplicity, its uniformity in embedding often leads to increased vulnerability to processing attacks and increased visual degradation. Moreover, a portion of these LSB techniques has not implemented reversible image modification yet. These issues are particularly problematic in the medical field, where the integrity of EHR and diagnostic images is paramount. Consequently, the field has evolved toward more adaptive steganographic schemes, surpassing the limitations of basic LSB insertion. These adaptive techniques prioritize recognizing complex textures in cover images to decide which pixels are suitable for modification [15–17]. By focusing on areas with complex features, these methods exploit the human eye's inability to perceive minor changes in noisy regions [18–20]. In order to further refine these adaptive strategies, many researchers have integrated Pixel Value Differencing (PVD) [21–23] to adjust embedding capacity based on the difference between adjacent pixels, hiding more data in high-contrast areas and minimizing distortions in smooth regions. An approach from [24] utilizes PVD alongside LSB substitution and the Censor Elfving algorithm to embed confidential patient data within images, reaching Peak Signal-to-Noise Ratio (PSNR) values of up to 61 dB for long strings.

Recent advancements have also explored frequency-domain techniques, such as those using the Discrete Cosine Transform (DCT) [25,26] or the Discrete Wavelet Transform (DWT) [27,28], which offer significantly higher resilience against compression attacks. The method in [26] utilizes the DCT Residual Modulation (DRM) algorithm to specifically address spatial pixel overflow and applies unit-based shuffling to the encoded message to enhance robustness. A DWT-based method in [29], specifically based on the Cohen-Daubechies-Feauveau Discrete Wavelet Transform (CDF-DWT), uses the Elgamal algorithm and the Speeded-Up Robust Features (SURF) method, achieving good results in the chosen metrics, such as 45.78% in PSNR, 2.09% in Mean Squared Error (MSE), and 4.56% in Bit Error Rate (BER). While the mentioned methods produce satisfactory results, the DCT and DWT methods suffer from high computational complexity and costs. Other methods and strategies have also been studied to push the boundaries of capacity and imperceptibility further, such as multi-image distribution strategies like dual-image and quadristego techniques, which spread the payload across multiple carriers [30,31]. By creating multiple stego images from

a single cover image, the method in [32] is able to maintain the original image's visual integrity while still allowing an effective and uniform data embedding.

Despite these improvements in multi-image and frequency-domain strategies, the challenge of optimizing the secret data itself before it reaches the embedding stage remains. Many spatial domain methods suffer from a lack of pre-embedding data compression, resulting in larger statistical footprints, more visual distortions, and a higher likelihood of detection by modern steganalysis tools. Furthermore, while multi-image strategies effectively distribute data, they often lack a localized mechanism to avoid smooth regions in the embedding process, producing changes in areas the human eye is sensitive to. This is especially risky in medical imaging, where smooth textures must remain comparatively untouched to avoid noticeable visual artifacts and affecting doctors' diagnosis [33]. Additionally, in the medical field and medical imaging, the ability to undo all alterations done to sensitive images is critical, which many current methods lack. This research proposes a reversible dual-layered steganographic method that integrates Huffman entropy coding with a localized variance-based adaptive embedding strategy. By utilizing Huffman coding as a preprocessing step, the secret payload is compressed, therefore reducing the number of pixels required to be modified and minimizing visual distortion. This is then followed by a categorization mechanism that analyzes the neighborhood variance of each pixel to ensure that modifications are done in complex and highly textured areas, all the while employing a key creating algorithm to ensure complete data extraction and full image recovery. By combining these two processes, the proposed method seeks to achieve a superior balance of high capacity and visual imperceptibility.

3 Proposed Method

The domain of digital steganography is fundamentally built and developed based on the general competing constraints of embedding capacity, imperceptibility, and robustness. Previously created spatial domain techniques, such as the simple Least Significant Bit substitution, offer high capacity but often succumb to steganalysis attacks. Conversely, adaptive techniques that restrict embedding to complex regions, such as the edges and textures of the medium, improve imperceptibility significantly but reduce the available payload capacity.

This research proposes a novel dual-layered framework designed to reconcile these conflicting objectives by integrating multiple techniques. First, the framework ingests the incoming secret data through a Huffman encoding scheme, significantly cutting the size of the to-be embedded bits. Next, the proposed framework calculates the local bit variance of each individual pixel, denoted as $S^2_{x,y}$ for any pixel $p(x, y)$. The highest local bit variance S^2_{max} of the medium will be recorded and utilized in the adaptive region selection. Every pixel will be separated into three categories based on the comparison between $S^2_{x,y}$ to S^2_{max}: Low for pixels in Eq. (1), Medium for pixels in Eq. (2), and High for pixels in Eq. (3). Finally, an adaptive redundancy-based embedding algorithm is deployed to insert differing amounts of bits into each cover bit. Furthermore, Fig. 2 illustrates the high-level design of the proposed method.

$$S^2_{x,y} \leq \frac{S^2_{max}}{5} \tag{1}$$

$$\frac{S^2_{max}}{5} \leq S^2_{i,j} \leq \frac{3 \times S^2_{max}}{5} \tag{2}$$

$$S^2_{x,y} > \frac{3 \times S^2_{max}}{5} \tag{3}$$

Considering the importance of image integrity in the medical field, it is important that all altered medical images are recoverable without any risk to their integrity. To address this critical issue, the proposed

method comes with a stego key generation algorithm, where each individual key holds not only the amount of the embedded bits in any given pixel string, but also how they are altered with only a single integer for each pixel. Using this stego key, during the data extraction process, the last phase of the framework, the method recovers the bits embedded into every pixel, all the while restoring the original cover image. The flow diagram in Fig. 3 details an example implementation of the method designed specifically for healthcare, using the proposed method to hide patient data from unauthorized access.

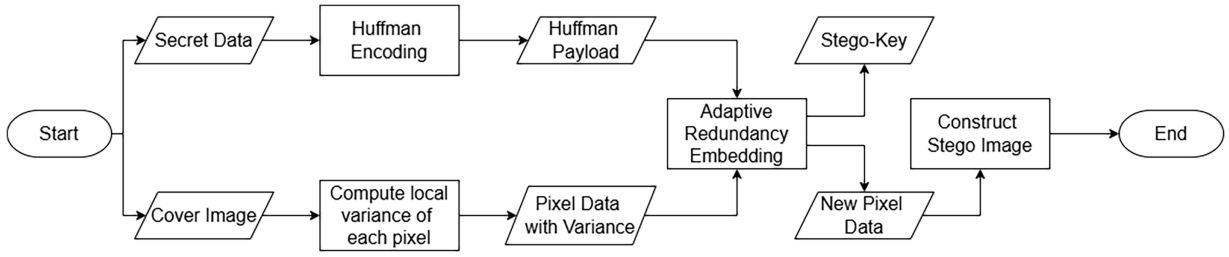

Figure 2: Flow of the proposed embedding method.

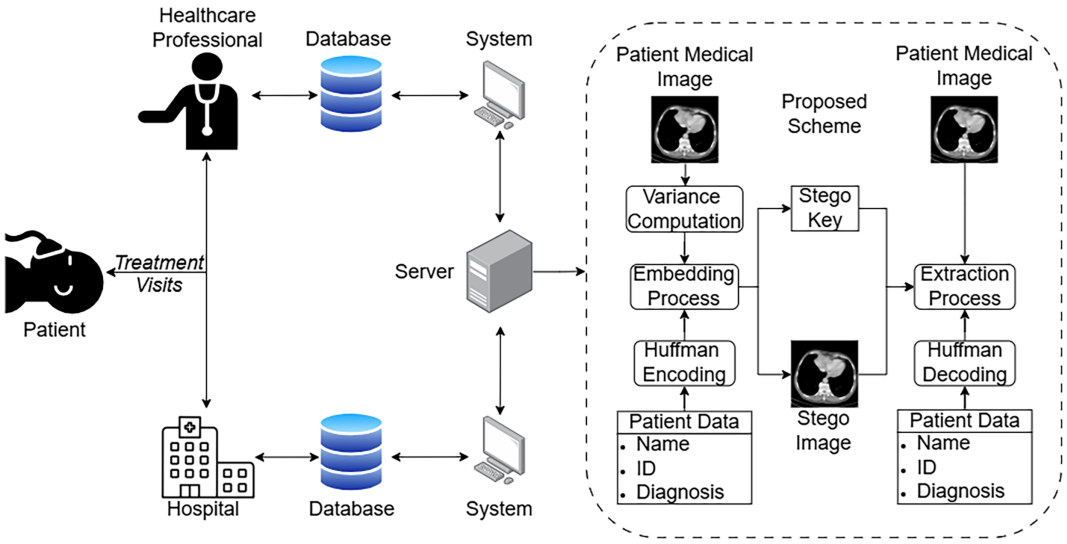

Figure 3: Steps taken to implement the proposed method.

3.1 Secret Data Preprocessing

Prior to the embedding process, the secret message undergoes a preprocessing phase designed to minimize the total payload through lossless compression. By reducing the number of bits required for concealment, the method effectively lowers the degradation of the stego image's visual integrity. The proposed method utilizes a Huffman encoding scheme for this goal, creating a Huffman Payload containing both the compressed bitstring and the necessary metadata required for reconstruction. If the secret data is in the form of binary, the data is converted into a hex string beforehand. This compression procedure is executed through the following steps:

1. Step 1: Tree Construction

The initial stage involves a statistical analysis of the secret data, counting the frequency of each unique character. Then, a priority queue is inserted with all unique character data, represented by a leaf node

specifying the character itself and its frequency in the secret data. A binary tree is constructed iteratively by merging the two nodes with the lowest frequencies into a parent node, whose frequency is the sum of its children's frequencies. This parent node is then reinserted into the queue. Repeating this process until no node is left in the queue results in a Huffman tree necessary for the next step.

2. Step 2: Codebook Generation

Using the created tree from Step 1, a Huffman codebook is derived by traversing the tree from its root. Starting from the root, binary values are assigned based on directional movement: a "0" is appended every time a left-branch traversal occurs, while a "1" is appended every time a right-branch traversal occurs. Upon reaching a leaf node that holds a unique character in the secret data, the accumulated binary sequence is assigned as the unique codeword for that specific character.

3. Step 3: Data Encoding

Using the created Huffman codebook, the original secret data is completely encoded into a compressed bitstream by replacing each character with its corresponding unique codeword.

4. Step 4: Final Payload Encapsulation

To ensure complete extraction and decoding, the created Huffman codebook and compressed bitstring are integrated into a single Huffman payload. A distinct delimiter is employed to separate these components, allowing the decoder to correctly parse the metadata. This Huffman scheme is completed once a Huffman Payload is fully generated, then it is passed onto the next phase.

The detailed logic of this Huffman scheme is elaborated further in the provided Algorithm 1.

Algorithm 1: Huffman preprocessing

```
INPUT: Secret Data msg
freq = frequency of unique characters in msg
Q = Min-heap using unique characters and their freq
WHILE Q.length > 1:
    node_left = Q.min
    node_right = Q.min
    node_parent = NEW_NODE(node_left.freq + node_right.freq, children: node_left, node_right)
    INSERT(Q, node_parent)
root = Q.min
C = {}
GENERATE_CODEBOOK(root, prefix=""):
    node = root
    IF node is leaf node: C[node.char] = prefix
    ELSE:
    GENERATE_CODEBOOK(node.left, prefix + '0'):
    GENERATE_CODEBOOK(node.right, prefix + '1'):
B = CONCATENATE(C[d] for each d ∈ msg)
M = ASCII_TO_BITS(SERIALIZE(C))
H = M + 0*32 + B
return H
```

3.2 Local Variance-Based Image Preprocessing

The cover image is preemptively analyzed before the embedding process in order to determine how many bits could be embedded in each pixel. This is done using a local bit variance calculation, where each pixel's variance is computed with up to 8 of its direct neighboring pixels.

3.2.1 Neighborhood Variance Calculation

For a pixel at coordinates (x, y) with intensity $I(x, y)$, let N be defined as the set of pixels within a 3×3 window centered at (x, y) and n be the total count of pixels in that set (typically $n = 9$). If coordinates (x, y) are at the edge or corner of the image, any empty space in the 3×3 window is ignored, reducing n. The local mean $\mu_{x,y}$ and local variance $S^2_{x,y}$ are calculated using the formulas in Eqs. (4) and (5), respectively.

$$\mu_{x,y} = \frac{1}{n} \sum_{(i,j)\in N} I(i, j) \tag{4}$$

$$S^2_{x,y} = \frac{1}{n-1} \sum_{(i,j)\in N} (I(i, j) - \mu_{x,y})^2 \tag{5}$$

This calculation serves as the method of spatial selection for embeddable pixels and as a proxy for local texture complexity, where higher variance indicates regions of high contrast or noise, which are more robust to human detection of bit-level modifications.

3.2.2 Adaptive Capacity Allocation

To maintain imperceptibility, the proposed method classifies pixels into three distinct categories based on the global peak variance S^2_{max} found within the image after calculating every neighborhood's sample variance S^2. The embedding capacity k of each pixel is then determined according to these categories, following the next threshold logic:

- Low Variance Region: These pixels are located in smooth regions of the image. To preserve visual integrity, the method only makes use of these pixels to hold a single bit of data at a time.
- Medium Variance Region: These pixels are located in moderately textured areas of the image and have an embedding capacity of one by default. To better utilize these pixels without producing overly disruptive changes, the proposed method utilizes its redundancy-based logical embedding algorithm. This technique works simply by checking for the two LSBs of a medium variance pixel; if at least one of the LSBs matches the corresponding next two Huffman bits, that particular pixel will hold two bits of data. This allows for extra bit embedding without producing any significant threat to structural integrity, as one bit doesn't change yet still holds embedded data.
- High Variance Region: These pixels reside in highly textured regions of the image. Making use of the wildly varying intensities of the pixels in these domains, the algorithm allows for a 2-bit LSB substitution, significantly increasing the total payload throughput without introducing perceptible distortions.

3.3 Data Concealment Process

The proposed data concealment mechanism serves as the core integration phase of the method, merging the compressed Huffman Payload with the spatial variance data of the cover image. This procedure is designed to maximize payload capacity while ensuring lossless restoration through a pixel-wise iterative cycle. As outlined in Algorithm 2, the embedding protocol is generally divided into three distinct stages: adaptive capacity determination, bitwise LSB substitution, and reversibility preservation via stego key

generation. Additionally, this embedding follows a checkered pattern to spread the payload evenly across the cover image, decreasing the amount of significant visual artifacts and ensuring the use of higher variance pixels. This would result in generally higher SSIM values and increased resistance against steganalysis methods. However, it is important to note that using the checkered pattern will reduce the overall embedding capacity of any images used. Therefore, this checkered pattern could be omitted and replaced with a sequential embedding pattern when prioritising embedding capacity.

Algorithm 2: Data embedding

INPUT: Huffman Payload H, Pixel Data P
idx = 0, keys = [], $p' = P$
FOR each pixel $p(x, y) \in P$:
    IF idx >= H.length:
        Append 7 to keys
        Break
    IF (p.x + p.y) % 2 ! = 0:
        continue
    IF $S^2(x, y) <= \lambda_1 * S^2max$:
        $k = 1$
    ELSE IF $S^2(x, y) <= \lambda_2 * S^2max$:
        pixel_bits = $LSB_2(p(x, y))$, huff_bits = H[idx:idx+2]
        k = 2 IF pixel_bits [0] == huff_bits [0] OR pixel_bits [1] == huff_bits [1] ELSE 1
    ELSE:
        $k = 2$
    mask = 2^k − 1
    orig = $p(x, y)$ AND mask
    data = INT(H[idx: idx+k])
    $p'(x, y)$ = ($p(x, y)$ AND NOT mask) OR data
    $\delta(x, y)$ = (orig XOR data) + mask
    Append $\delta(x, y)$ to keys
    idx = idx + k
stego_image = construct image from collection of p'
return stego_image, keys

1. Adaptive Capacity Determination

The algorithm evaluates the preanalyzed properties of the target pixel $p(x, y)$, determining its embedding capacity $k(x, y)$ based off of the variance threshold as seen in Eq. (6) where λ_1 and λ_2 are threshold coefficients satisfying $0 < \lambda_1 < \lambda_2 < 1$. In this work, these coefficients are empirically determined and will be discussed in Section 4. Furthermore, this calculation incorporates a control function $\alpha(x, y)$, defined in Eq. (7), to represent the redundancy-based embedding algorithm. By dynamically assigning the capacity value of every pixel, the proposed method balances high payload volume with perceptual imperceptibility. This step ensures that no pixel conceals more data than it is capable of to avoid introducing significant visual distortions.

2. Bitwise Substitution

Once the capacity $k(x, y)$ is established, the method executes its embedding operation. This operation is an LSB substitution, where $k(x, y)$ LSBs of $p(x, y)$ are replaced by the $k(x, y)$ following bits from the

Huffman Payload, represented as H_{next} and its decimal value as M. This direct bitwise manipulation, detailed in Eq. (8), embeds an appropriate amount of the compressed data into the pixel, ensuring that the alterations are not easily detected by the human visual system.

3. Stego Key Construction

To facilitate exact recovery of the original cover image and Huffman Payload, the method simultaneously constructs a stego key during the embedding process. This key records specific metadata required to extract data from every pixel and allows undoing of any alterations made. Each entry in the key, denoted by $\delta(x, y)$, is the result of a XOR operation between $k(x, y)$ amount of LSB from the cover pixel and $k(x, y)$ amount of bits from the Huffman Payload. This will result in an integer, as specified in Eq. (9), which can be used to recover the original LSB as long as the chosen data bits are available. Furthermore, this $\delta(x, y)$ will be incremented by $2^{k(x,y)}-1$ to differentiate between different amount of embedded bits. This results in the stego key entries ranging from the integers 0 to 7 (3 bits) to fully enable reversibility. This would mean that the stego key could grow beyond the original payload size, shifting the focus from steganography to cryptography. To mitigate this limitation, the proposed method takes advantage of the stego key's characteristics of repeating numbers, in this case the numbers from 0 to 7, and processes it through the Huffman encoding scheme in Algorithm 1. Through this pipeline, the ratio between the size of the stego key and the size of the payload shrinks as the payload grows larger.

$$k(x,y)\begin{cases}1 & if\ S_{i,j}^2 \le \lambda_1 \times S_{max}^2 \\ \alpha\,(x,y) & if\ \lambda_1 \times S_{max}^2 \le S_{i,j}^2 \le \lambda_2 \times S_{max}^2 \\ 2 & if\ S_{i,j}^2 > \lambda_2 \times S_{max}^2\end{cases} \tag{6}$$

$$\alpha\,(x,y)\begin{cases}2 & if\ LSB_2(p\,(x,y)) \oplus H_{next\,2} \neq 3 \\ 1 & otherwise\end{cases} \tag{7}$$

$$p'\,(x,y) = \left(p\,(x,y) - \left(p\,(x,y)\ mod\ 2^{k(x,y)}\right)\right) + M \tag{8}$$

$$\delta\,(x,y) = \left(LSB_{k(x,y)}\,(p\,(x,y)) \oplus H_{k(x,y)}\right) + (2^{k(x,y)} - 1) \tag{9}$$

3.4 Data Extraction Process

The data extraction mechanism serves as the direct inverse of the embedding phase. Unlike the embedding algorithm, which relies on the calculation of spatial variance, the extraction algorithm primarily utilizes the stego key $\delta(x, y)$ to guide and guarantee the complete retrieval of data, as well as the full recovery of the cover image. This design is critical, as the embedding process results in differing pixel intensities, leaving a high chance of different results when recalculating variance on the stego image. By relying on the stego key, the method ensures robust and synchronized extraction without needing the original cover image. Similar to the embedding mechanism, the extraction process follows an iterative cycle through every key, as can be seen in Algorithm 3.

Algorithm 3: Data extraction

```
INPUT: Stego Image I, Stego Key keys
idx = 0, bits = []
FOR each pixel p'(x, y) ∈ I:
    IF (p'.x + p'.y) % 2 ! = 0:
        p(x, y) = p'(x, y)
        continue
```

(Continued)

Algorithm 3 (continued)

```
            IF idx >= keys.length:
                p(x, y) = p′(x, y)
                Continue
        δ = keys[idx], idx = idx + 1
        IF δ = 7:
            break
        k = 1 IF δ < 3 ELSE 2
        mask = 2^k − 1
        H_k = p′(x, y) AND mask
        orig_k = (δ − mask) XOR H_k
        p(x, y) = (p′(x, y) AND NOT mask) OR orig_k
        Append H_k to bits
    H = CONCATENATE(bits)
    R = construct image from collection of p
    return H, R
```

The decoder first inspects the stego key entry $\delta(x, y)$, determining the corresponding pixel's capacity $k(x, y)$. As the key generation applies a distinct offset based on the capacity, the embedding rate of that specific pixel can be retrieved by checking the numerical range of $\delta(x, y)$ in Eq. (10). Once the capacity $k(x, y)$ is determined, the hidden Huffman data are directly extracted from the stego pixel $p'(x, y)$ through the modulo operation in Eq. (11). To restore the original pixel $p(x, y)$, the method reconstructs the $k(x, y)$ amount of bits that were overwritten. By rearranging the XOR-based key defined in Eq. (9) used in the embedding phase, the original LSBs are calculated using Eq. (12). Finally, the original pixel is reconstructed by replacing the current $k(x, y)$ LSBs from $p'(x, y)$ with the recovered LSBs, as seen in Eq. (13), ensuring that the restored image is mathematically identical to the cover image. This entire process is completely repeated, until a $\delta(x, y)$ with the value of 6 is found, indicating the last of the stego keys, or if there is no more $\delta(x, y)$ to be inspected.

$$k(x, y) \begin{cases} 1 & if\ \delta(x, y) < 2 \\ 2 & if\ \delta(x, y) < 6 \end{cases} \tag{10}$$

$$H_{k(x,y)} = p'(x, y)\ mod\ 2^{k(x,y)} \tag{11}$$

$$LSB_{k(x,y)}(p(x, y)) = (\delta(x, y) - \left(2^{k(x,y)} - 1\right) \oplus H_{k(x,y)} \tag{12}$$

$$p(x, y) = \left(p'(x, y) - H_{k(x,y)}\right) + LSB_{k(x,y)}(p(x, y)) \tag{13}$$

4 Results and Discussion

This section, separated into eight subsections, presents a comprehensive evaluation of the proposed steganographic method, analyzing its performance in terms of imperceptibility, embedding capacity, and reversibility. The first subsection details the dataset used to evaluate the method, namely the cover images and payload used. The second subsection briefly describes the metrics used to evaluate the proposed method. Following that, the third subsection presents the experimental results, while sharing a comprehensive analysis based on observation and the chosen evaluation metrics, the fourth subsection provides an analysis on the generated stego key, the fifth subsection provides a comparative analysis against existing state-of-the-art schemes, the sixth subsection discusses the method's performance under several steganalysis methods,

the seventh is an ablation study, comparing the full version of the method against several incomplete versions. Finally, the eighth subsection discusses VARStego's results in depth.

4.1 Experimental Dataset

The experimental results are derived from the application of the complete method pipeline to a dataset of medical images from two separate datasets. The first source is the DICOM Library dataset [34], containing DICOM-formatted files. Five images, namely Abdominal, Head, Hand, Leg and Chest were chosen from DICOM library as cover images. All DICOM files were converted to Tagged Image File Format (TIFF) format using a lossless bit-depth preserving conversion. In particular, the resulting TIFF file preserved the original pixel depth of each image, meaning that there was no quantization or dynamic range truncation in the translation process. TIFF was chosen because it natively supports high bit-depth grayscale images with lossless storage, unlike JPEG or PNG with limited bit depths, making it the appropriate intermediary format. The second source of the experiment is the Kaggle dataset from [35], providing CT medical images. From this dataset, ten images were randomly selected for the experiment, as specified in Table 1. The cover images from these two sources have the same resolution of 512 × 512 pixels. These images are then individually embedded with dummy payload data from the Lorem Ipsum story [36] with varying sizes and types. The specifics of these payloads are: ASCII payload data of sizes 1, 5, 10, 20, 30, 40, 50 kilobyte and bitstring payload data of sizes 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 kilobit. Finally, the experimentation also uses a secondary dataset consisting of 5000 medical cover images with a resolution of 1024 × 1024 pixels for a dedicated security analysis. These images, taken from [37], will also similarly be processed by the proposed VARStego scheme, creating corresponding stego images.

Table 1: List of the filenames used in the experiments and their corresponding images from [35].

Filename	Original Filename
image-01	ID_0001_AGE_0069_CONTRAST_1_CT
image-02	ID_0002_AGE_0074_CONTRAST_1_CT
image-03	ID_0003_AGE_0075_CONTRAST_1_CT
image-04	ID_0004_AGE_0056_CONTRAST_1_CT
image-05	ID_0005_AGE_0048_CONTRAST_1_CT
image-06	ID_0006_AGE_0075_CONTRAST_1_CT
image-07	ID_0007_AGE_0061_CONTRAST_1_CT
image-08	ID_0008_AGE_0051_CONTRAST_1_CT
image-09	ID_0009_AGE_0048_CONTRAST_1_CT
image-10	ID_0010_AGE_0060_CONTRAST_1_CT

4.2 Evaluation Metrics

To objectively assess the experimental results, two standard metrics commonly used to assess steganography methods are employed: the Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index Measure (SSIM). PSNR is utilized to measure the statistical difference between the cover images and corresponding stego images. Specified in Eq. (14), this metric quantifies the magnitude of the distortion introduced by the embedding process. It is important to note that MAX_I is the maximum number possible of a pixel intensity, while H and W are the height and width of the images, respectively. Generally, the more bits that are changed, the lower the value of the PSNR between two images is, ranging from 0 to 100 dB. While

PSNR shows the statistical difference of the cover image and the stego image, SSIM offers a more perception-based assessment, evaluating structural changes to determine how similar the mentioned two images are to the human eye. The value of SSIM ranges from 0 to 1, where 1 means complete structural fidelity between the cover image and stego image. This metric is defined in Eq. (15), where p denotes the cover image and p' denotes the stego image. The μ_p and $\mu_{p'}$ holds the mean pixel intensities of the respective images, while σ_p and $\sigma_{p'}$ holds the variance of the images. Lastly, C_1 and C_2 define constants used to stabilize the division.

$$PSNR = 10\log_{10}\left(\frac{{MAX_I}^2}{\frac{1}{HW}\sum_{i=0}^{H-1}\sum_{j=0}^{W-1}\left[p\left(i,j\right)-p'(i,j)\right]^2}\right) \tag{14}$$

$$SSIM = \frac{(2\mu_p\mu_{p'}+C_1)(2\sigma_{pp'}+C_2)}{(\mu_p^2+\mu_{p'}^2+C_1)(\sigma_p^2+\sigma_{p'}^2+C_2)} \tag{15}$$

4.3 Experimental Results

The proposed method was directly applied to the 15 cover images, embedding and extracting payloads with sizes ranging from 1 to 100 kb for bitstring payload and 1 to 50 KB for ASCII payload. In this experiment, the threshold coefficients λ_1 and λ_2 were selected via a sensitivity analysis across combinations of $\lambda_1 \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ and $\lambda_2 \in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, averaged across the 15 cover images embedded with 50 kb of bitstring payload. As shown in Table 2, the difference of PSNR across all valid configurations is less than 0.1 dB, demonstrating strong robustness to threshold selection. The values $\lambda_1 = 0.2$ and $\lambda_2 = 0.6$ were retained as they represent a stable and consistent configuration. After the completed embedding process, the resulting stego images are put through the extraction process, fully retrieving the payload data without any loss. Additionally, using the stego images, the original cover images have been reconstructed perfectly without fault. The resulting stego images are then compared to the original cover images using the calculation from Eq. (14) for PSNR, and the calculation from Eq. (15) for the SSIM. The PSNR between the cover images and the reconstructed cover images (from the extraction phase) is also computed.

Table 2: Average PSNR of all variance threshold combinations.

PSNR in dB							
		High Threshold					
		0.4	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
Low Threshold	0.5	–	–	58.195	58.199	58.201	58.204
	0.4	–	58.189	58.197	58.197	58.195	58.198
	0.3	58.168	58.178	58.183	58.188	58.188	58.189
	0.2	58.151	58.155	58.160	58.163	58.166	58.168
	0.1	58.099	58.112	58.116	58.118	58.116	58.120

The data in Table 3 presents the achieved PSNR scores of all images across all bitstring payload data. At the lowest payload size (1 kb of bitstring), the PSNR hovers around 70 to 72 dB, which is considered very satisfactory and proves that the distortions introduced by the method are minimal. As the payload size increases, the PSNR score naturally declines, with the lowest hovering around 55 dB at the highest payload size of 100 kb. Nevertheless, this data shows that the proposed method is able to produce relatively good results in terms of statistical distortions. It also indicates that the method is adaptive and performs consistently in various situations, as shown by the small differences between the average PSNR of each payload.

Table 3: PSNR (in dB) across all images from [34,35] and all bitstring payload sizes using the proposed method.

Cover Image	Bitstring Payload Size in Kilobit										
	1	10	20	30	40	50	60	70	80	90	100
Abdominal	70.658	64.424	61.808	60.175	59.031	58.148	57.382	56.717	56.149	55.666	55.214
Chest	71.147	64.608	61.888	60.271	59.078	58.174	57.384	56.741	56.164	55.686	55.242
Hand	70.955	64.391	61.827	60.241	59.071	58.171	57.382	56.733	56.170	55.672	55.175
Head	71.354	64.427	61.845	60.192	59.024	58.052	57.224	56.560	55.977	55.474	55.045
Leg	71.042	64.692	61.983	60.254	59.096	58.115	57.379	56.725	56.119	55.686	55.227
image-01	72.192	64.805	62.020	60.366	59.150	58.208	57.427	56.727	56.155	55.667	55.218
image-02	72.192	64.805	62.032	60.370	59.182	58.246	57.433	56.767	56.156	55.669	55.215
image-03	72.192	64.805	61.998	60.307	59.147	58.172	57.352	56.732	56.149	55.642	55.212
image-04	72.192	64.805	62.003	60.352	59.164	58.197	57.420	56.743	56.155	55.664	55.206
image-05	72.192	64.805	62.017	60.330	59.131	58.161	57.381	56.705	56.111	55.614	55.146
image-06	72.192	64.805	61.998	60.307	59.147	58.172	57.352	56.732	56.149	55.642	55.212
image-07	72.192	64.811	62.060	60.384	59.145	58.208	57.424	56.746	56.193	55.679	55.203
image-08	72.192	64.805	62.017	60.334	59.146	58.187	57.425	56.729	56.145	55.656	55.175
image-09	72.192	64.805	61.984	60.341	59.110	58.153	57.400	56.722	56.128	55.646	55.148
image-10	72.192	64.805	62.004	60.322	59.123	58.157	57.400	56.748	56.168	55.650	55.213

The calculated SSIM scores of the proposed method across all images and bitstring payload are shown in Table 4. Throughout all images and payload sizes, the method consistently achieves near-perfect scores, with the highest being around 0.9999 when embedding 1 kb of data. As the payload size increases, the SSIM gradually degrades in value. However, this decline is relatively minimal, as the lowest SSIM in the experiment is still very satisfactory, with a value of 0.9934. It is important to note that SSIM values go from 0.0 to 1.0, with scores closer to 1 meaning less structural differences. This proves the proposed method's effectiveness in preserving structural details after the embedding process, aligning with its initial goal of using a local variance-based algorithm to better suit the human visual system.

Table 4: SSIM across all images from [34,35] and all bitstring payload sizes using the proposed method.

Cover Image	Payload Size in kb										
	1	10	20	30	40	50	60	70	80	90	100
Abdominal	0.9999	0.9997	0.9993	0.9989	0.9985	0.9980	0.9976	0.9971	0.9967	0.9962	0.9959
Chest	1.0	0.9996	0.9990	0.9985	0.9981	0.9977	0.9972	0.9967	0.9962	0.9957	0.9952
Hand	1.0	0.9998	0.9995	0.9992	0.9990	0.9987	0.9985	0.9983	0.9981	0.9979	0.9977
Head	0.9999	0.9996	0.9991	0.9987	0.9983	0.9979	0.9975	0.9971	0.9967	0.9963	0.9959
Leg	1.0	0.9998	0.9995	0.9993	0.9990	0.9987	0.9985	0.9982	0.9979	0.9977	0.9975
image-01	0.9999	0.9989	0.9979	0.9975	0.9971	0.9965	0.9960	0.9956	0.9951	0.9947	0.9944
image-02	0.9999	0.9989	0.9981	0.9979	0.9977	0.9972	0.9967	0.9962	0.9957	0.9951	0.9948
image-03	0.9999	0.9989	0.9980	0.9975	0.9971	0.9966	0.9960	0.9954	0.9946	0.9940	0.9934
image-04	0.9999	0.9989	0.9979	0.9975	0.9973	0.9969	0.9965	0.9961	0.9957	0.9952	0.9949
image-05	0.9999	0.9989	0.9979	0.9972	0.9966	0.9960	0.9954	0.9948	0.9943	0.9938	0.9935
image-06	0.9999	0.9989	0.9980	0.9975	0.9971	0.9966	0.9960	0.9954	0.9946	0.9940	0.9934
image-07	0.9999	0.9992	0.9990	0.9989	0.9985	0.9980	0.9975	0.9969	0.9963	0.9957	0.9950
image-08	0.9999	0.9989	0.9979	0.9969	0.9959	0.9955	0.9954	0.9949	0.9945	0.9943	0.9937
image-09	0.9999	0.9989	0.9983	0.9980	0.9973	0.9967	0.9962	0.9956	0.9949	0.9942	0.9936
image-10	0.9999	0.9989	0.9980	0.9974	0.9969	0.9966	0.9964	0.9962	0.9961	0.9958	0.9954

Next, Table 5 provides the PSNR scores achieved by the method across all images and all ASCII payload data with the sequential pattern to showcase the maximum embedding capacity of the method. Using ASCII character payload data, the highest received PSNR is 66.589 with the lowest payload size of 1 KB (kilobyte). Similar to the case of bitstring payload data, the PSNR scores decrease as the payload size increases, with the lowest hovering around 51–52 dB at the highest payload of 50 KB. The SSIM scores of the experiment using ASCII payload can be found in Table 6. Across all images, the method manages to achieve near perfect scores, with the highest being 0.9999, similar to the bitstring results. However, at 10 kilobyte (80 kilobit) of data, the ASCII sequential pattern experiment received several SSIM in the 0.992–0.993 range, lower than the bitstring experiment with the checkered pattern which went as low as 0.9933 while using a higher payload size (100 kilobit). This can be attributed to the difference of embedding pattern used between the two experiments and will be discussed further in the ablation study.

Table 5: PSNR (in dB) across all images from [34,35] and all ASCII payload sizes using the proposed method with sequential pattern.

Cover Image	ASCII Payload Size in Kilobyte						
	1	5	10	20	30	40	50
Abdominal	65.535	60.924	58.333	55.589	53.904	52.710	51.798
Chest	65.755	61.081	58.362	55.612	53.944	52.725	51.773
Hand	65.632	60.894	58.393	55.698	54.019	52.798	51.847
Head	65.928	61.002	58.320	55.522	53.775	52.532	51.628
Leg	65.814	61.118	58.466	55.594	54.006	52.787	51.858
image-01	66.589	61.188	58.399	55.597	53.894	52.656	51.714
image-02	66.589	61.188	58.432	55.633	53.916	52.672	51.751
image-03	66.589	61.188	58.377	55.552	53.858	52.620	51.679
image-04	66.589	61.188	58.401	55.616	53.921	52.672	51.735
image-05	66.589	61.188	58.385	55.551	53.825	52.593	51.653
image-06	66.589	61.188	58.377	55.552	53.858	52.620	51.679
image-07	66.589	61.261	58.596	55.682	53.956	52.721	51.765
image-08	66.589	61.188	58.385	55.521	53.859	52.640	51.671
image-09	66.589	61.191	58.408	55.541	53.857	52.634	51.671
image-10	66.589	61.188	58.400	55.543	53.903	52.716	51.738

Table 6: SSIM across all images from [34,35] and all ASCII payload sizes using the proposed method with sequential pattern.

Cover Image	ASCII Payload Size in Kilobyte						
	1	5	10	20	30	40	50
Abdominal	0.9998	0.9990	0.9977	0.9950	0.9918	0.9889	0.9866
Chest	0.9998	0.9984	0.9966	0.9935	0.9904	0.9869	0.9834
Hand	0.9999	0.9995	0.9989	0.9979	0.9970	0.9962	0.9954
Head	0.9997	0.9987	0.9972	0.9943	0.9916	0.9888	0.9855
Leg	0.9991	0.9960	0.9920	0.9861	0.9807	0.9760	0.9723
image-01	0.9991	0.9960	0.9925	0.9891	0.9856	0.9819	0.9793

(Continued)

Table 6 (continued)

Cover Image	ASCII Payload Size in Kilobyte						
	1	5	10	20	30	40	50
image-02	0.9991	0.9960	0.9931	0.9914	0.9878	0.9838	0.9818
image-03	0.9991	0.9960	0.9928	0.9895	0.9852	0.9798	0.9755
image-04	0.9991	0.9960	0.9926	0.9901	0.9873	0.9839	0.9818
image-05	0.9991	0.9960	0.9925	0.9892	0.9857	0.9821	0.9796
image-06	0.9991	0.9960	0.9932	0.9916	0.9880	0.9841	0.9820
image-07	0.9991	0.9960	0.9928	0.9894	0.9853	0.9799	0.9757
image-08	0.9991	0.9960	0.9926	0.9901	0.9873	0.9840	0.9819
image-09	0.9991	0.9960	0.9920	0.9874	0.9832	0.9791	0.9761
image-10	0.9991	0.9960	0.9928	0.9894	0.9853	0.9799	0.9757

Additionally, to further validate the generalizability of the proposed method, VARStego was applied to a separate dataset of 5000 images taken from [37], embedded with 10 KB of ASCII data. This larger set of images yielded an average PSNR score of 64.441 dB and an average SSIM of 0.9995. Notably, these scores represent a significant improvement compared to the primary experimental dataset from [34,35], where the highest recorded PSNR and SSIM at the same 10 KB of ASCII data is 58.596 dB and 0.9989, respectively. This difference is primarily attributed to the difference in image resolution between the datasets, as the secondary dataset consists of 1024 × 1024 images, with each image having exactly 4 times the amount of pixels compared to the 512 × 512 primary dataset. Since PSNR is derived from the Mean Squared Error (MSE), which is computed as the average squared pixel-wise difference over the total number of pixels, embedding a fixed payload size of 10 KB into a larger image would dilute the same embedding-induced error across a proportionally larger pixel count, directly reducing the MSE and increasing the resulting PSNR. The remaining difference can additionally be attributed to the difference of characteristics of the textured areas in the chest X-ray images used in the secondary dataset.

To further evaluate the statistical imperceptibility of the proposed method, Fig. 4 illustrates the histogram comparison of the cover image 'Hand' against the stego image created at the highest bitstring payload size in the experiment (100 kilobits). As observed, the stego image's histogram almost perfectly overlaps with the cover image's histogram, maintaining a high accuracy of the general shape and distribution of pixel intensities. This proves that the proposed method does not introduce significant statistical anomalies and visual artifacts, offering an almost 1-to-1 comparison to the original image to the naked eye. Additionally, Fig. 5 visualizes a difference map between the 'Head' cover image, the created stego image, and the recovered image at 100 kilobits of bitstring payload. To clearly illustrate the adaptive capacity of the algorithm, the 'Cover vs. Stego Diff' map applies a threshold to display only absolute pixel changes greater than 1, while the 'Cover vs. Restored Diff' completely show every change. As shown in Fig. 5, the map indicates that the local variance embedding algorithm successfully focuses on highly rough textures of the image, as represented using warm and contrasting colors, such as red, yellow, and white. The second map, showing a purely black color with no disruptions proves the method's capability in recovering the cover image without loss.

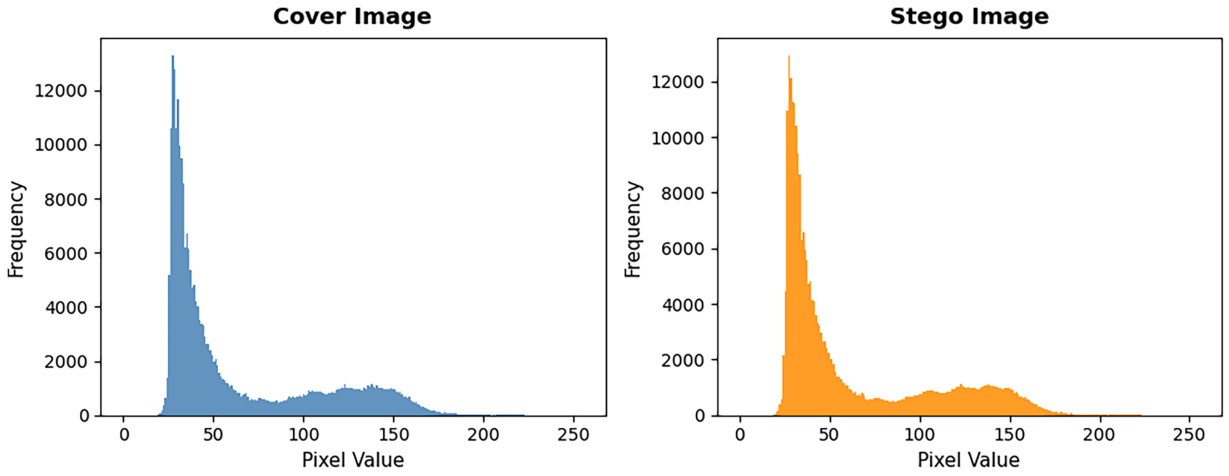

Figure 4: Pixel intensity histogram between cover and stego image.

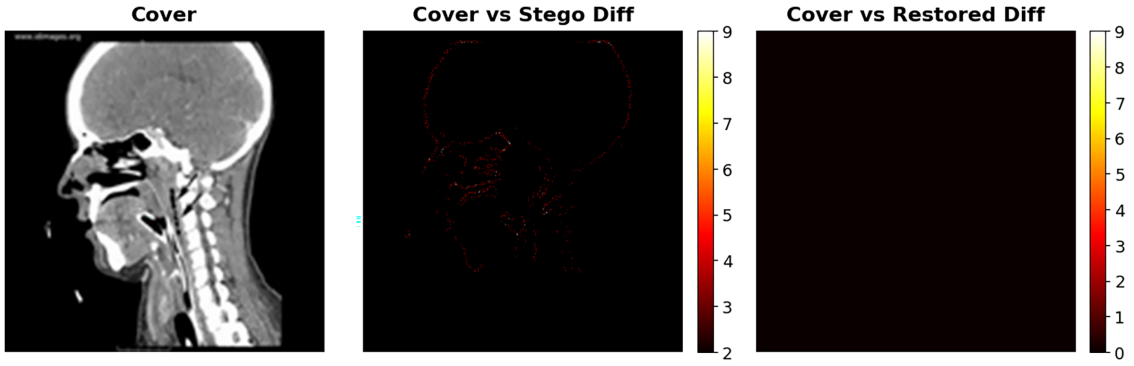

Figure 5: Difference map between cover, stego, and recovered images.

Furthermore, Table 7 delineates the computational performance of the proposed method across varying payload sizes. The results indicate a consistent and stable embedding process, where the execution time exhibits minimal increases relative to the payload size. The times recorded range from 0.699 s for a 1 kb payload to 0.759 s for the maximum 100 kb payload. A similar trend is observed in the extraction phase, which exhibits a similar linear progression in execution time relative to data size. The extraction phase lasts for 0.030 s for a 1 kb payload at the shortest, and 0.055 s for a 90 kb payload at the longest. Notably, the record time for extraction of the 100 kb payload is 0.053 s, a negligible deviation from the upward trend that is attributable to standard operating system scheduling rather than algorithmic instability. With all operations completed in under one second, the proposed method demonstrates significant viability for real-time medical applications. In the context of medical imaging, where rapid diagnosis is critical, such efficiency is paramount. Hence, the proposed method provides a viable solution for the secure transmission of Electronic Health Records and diagnostic imagery within high-throughput hospital networks.

The computational complexity of the proposed VARStego framework is analyzed per phase to assess its scalability according to image size and payload length. Let N be the total number of pixels in the cover image, L be the length of secret data, and M denote the number of unique characters in the secret data.

The secret data preprocessing phase applies Huffman encoding, which requires constructing a frequency table and a priority-queue-based binary tree. Generally, this construction operates in $O(M \log M)$, where

M is bounded by the alphabet size of the input data. For ASCII payload data, M is bounded by the ASCII alphabet of 256 characters ($M \leq 256$), while for bitstring payload data M is confined to at most 16 as bitstring payload is converted to hexstring. This renders the tree construction step effectively as O(1) in practice while the following encoding step passes over the payload, operating in O(L) and making the full compression phase O(L) overall.

Table 7: Computational time of the embedding and extraction phases using the image "Hand".

Payload (kb)	Embedding (s)	Extraction (s)
1	0.699	0.030
10	0.720	0.032
20	0.728	0.037
30	0.736	0.040
40	0.808	0.042
50	0.722	0.044
60	0.668	0.048
70	0.773	0.048
80	0.720	0.051
90	0.751	0.055
100	0.759	0.053

The local variance-based image analysis phase computes the neighborhood variance for each pixel within a fixed 3 × 3 window, resulting in a strictly linear O(N) complexity regardless of image content or payload size. The data concealment phase iterates over the pixel data in a single pass, performing constant-time operations per pixel, which are capacity determination, LSB substitution, and stego key generation leading to a complexity of O(N) overall. The data extraction phase mirrors the embedding phase in structure, also operating in O(N) as stego key entry would at most be equal to N and decoding in O(L), simliar to the encoding phase. Finally, the evaluation metrics PSNR, SSIM, and the pixel-wise identity check for reversibility verification are each O(N) computations. The overall complexity of the complete VARStego pipeline is therefore O($N + L$) in the general case.

In terms of space complexity, using the same notation as above, the VARStego pipeline scales in a manner that is similar to its time complexity. The secret data preprocessing phase requires O(M) space for the frequency table, priority queue, and Huffman tree, where M is bounded by the alphabet size of the input (at most 256 for ASCII payloads and 16 for hexstring-converted bitstring payloads), in addition to O(L) space for the resulting compressed bitstream. The local variance-based image analysis phase maintains multiple variables based off of the cover image, such as the local mean, the local mean-of-squares, and the variance map, as full-size arrays, yielding an overall O(N) space requirement. The data concealment phase operates on a duplicated copy of the full pixel list to preserve the original cover data, contributing a further O(N) term, while the accompanying stego key grows with the number of modified pixels, bounded by the number of pixels N, giving another O(N) in the worst case prior to Huffman compression. Next, the data extraction phase requires O(N) for the restored-pixel list and O(N) in the worst case for the decoded stego key, with the recovered payload occupying O(L). Finally, each evaluation metric (PSNR, SSIM, and pixel-wise identity check) requires a further O(N) space to hold the cover, stego, and recovered images. The overall space complexity of the complete VARStego pipeline is therefore O($N + L$) in the general case, similar to the time complexity.

4.4 Stego Key Analysis

The proposed VARStego method generates a stego key during the embedding process to enable complete data extraction and lossless image recovery. Each key entry $\delta(x, y)$ encodes the XOR difference between the original and modified pixel bits, producing one integer value per embedded pixel. In its raw form, this key grows proportionally to the number of modified pixels rather than the payload size itself, potentially exceeding the payload in size and undermining the steganographic efficiency of the method. To address this, the stego key is processed through the same Huffman encoding scheme described in Algorithm 1, exploiting the highly repetitive nature of the key's symbol set (integers from 0–7).

Table 8 presents the Huffman compressed stego key sizes for both ASCII and bitstring payload types across all tested payload sizes with the Hand cover image. The ratio between key size and payload size is calculated through Eq. (16), where both key size and payload size must be in the same unit, such as in bit. For ASCII payload data, the key-to-payload ratio begins at 1.811 at 1 KB, rapidly converging toward and crossing below 1.0 at approximately 5 KB. Beyond this point, the ratio between stego key size and payload size continues to shrink, reaching 0.830 at the maximum tested payload at 50 KB with a smaller stego key size by roughly 17% compared to the original payload size. For bitstring payload data, the key-to-payload ratio follows a similar downward trend, beginning at 4.689 at 1 kilobit and converging asymptotically toward approximately 1.5 at higher payload sizes. While the bitstring key does not achieve a sub-1.0 ratio within the tested range as it is less compatible with the utilized Huffman encoding scheme, the consistent convergence demonstrates that the Huffman compression becomes increasingly effective as the payload and embedded pixel count grow. Through this data, it is clear that the generated stego keys will have smaller ratios compared to payload size the larger the latter grows, especially when the data type is non-random ASCII characters, as most medical records are.

$$Ratio = \frac{Key\ Size}{Payload\ Size} \tag{16}$$

Table 8: Stego key across all payload data.

Size	Payload Type	Key Size (bit)	Ratio
1 KB	ASCII	14,485	1.811
5 KB	ASCII	41,166	1.029
10 KB	ASCII	73,957	0.924
20 KB	ASCII	138,331	0.865
30 KB	ASCII	202,816	0.845
40 KB	ASCII	267,357	0.835
50 KB	ASCII	332,098	0.830
1 kb	Bitstring	4689	4.689
10 kb	Bitstring	18,606	1.861
20 kb	Bitstring	33,607	1.680
30 kb	Bitstring	48,550	1.618
40 kb	Bitstring	63,521	1.588
50 kb	Bitstring	78,466	1.569
60 kb	Bitstring	93,628	1.560
70 kb	Bitstring	108,636	1.552
80 kb	Bitstring	123,581	1.545
90 kb	Bitstring	138,854	1.543
100 kb	Bitstring	154,065	1.541

In critical transmission the stego key does not need to be treated as a secret with the same high level of security as the payload. This key only tells us which pixels are changed and by how much, but not the data that is embedded. Even if the stego key were to be compromised in transmission, it remains effectively useless on its own. Extracting the secret message requires not only the stego key, but also the corresponding stego image and the complete extraction algorithm used, which an attacker would have no straightforward means of obtaining or replicating. This is further reinforced by the fact that the stego key itself is Huffman-compressed prior to transmission, the same scheme used for the payload, meaning a compromised key would first need to be correctly decompressed before it could even be interpreted as pixel-modification data. Without the stego image and the extraction algorithm used, the key does not reveal the secret message. Thus, the key can be passed over typical secure channels, like Hypertext Transfer Protocol (HTTP) over Transport Layer Security (TLS), or even as part of the metadata of an accompanying file, which does not need to be as secure as the payload. This method does not suffer from circular dependency, that is, a situation where the key would need to be protected with the same efforts as encrypting or hiding the original message during transmission.

4.5 Comparison against Existing Methods

This section presents a comparison of the proposed variance-based adaptive scheme against several state-of-the-art steganographic methods. Specifically, the proposed method is benchmarked against approaches related to the field [38–41]. This comparison is done using the overall PSNR average of each method with a 1 kb secret bitstring payload. The bar chart in Fig. 6 presents the achieved PSNR values across all steganographic methods reviewed. The proposed method achieves the highest average PSNR score of 71.8 dB, significantly outperforming other approaches. The SHA-256-based approach by Hameed et al. [38] exhibits competitive performance with an average PSNR value of 66.3 dB. Meanwhile, the remaining methods cluster in the 56–60 dB range. The method proposed by Peng et al. [39] and the method by Akhtarkavan et al. [41] achieve commendable results, with an average of 59.3 and 58.8 dB, respectively. Finally, the neural network-based method by Ramapriya and Kalpana [40] records an average score of 56.9 dB.

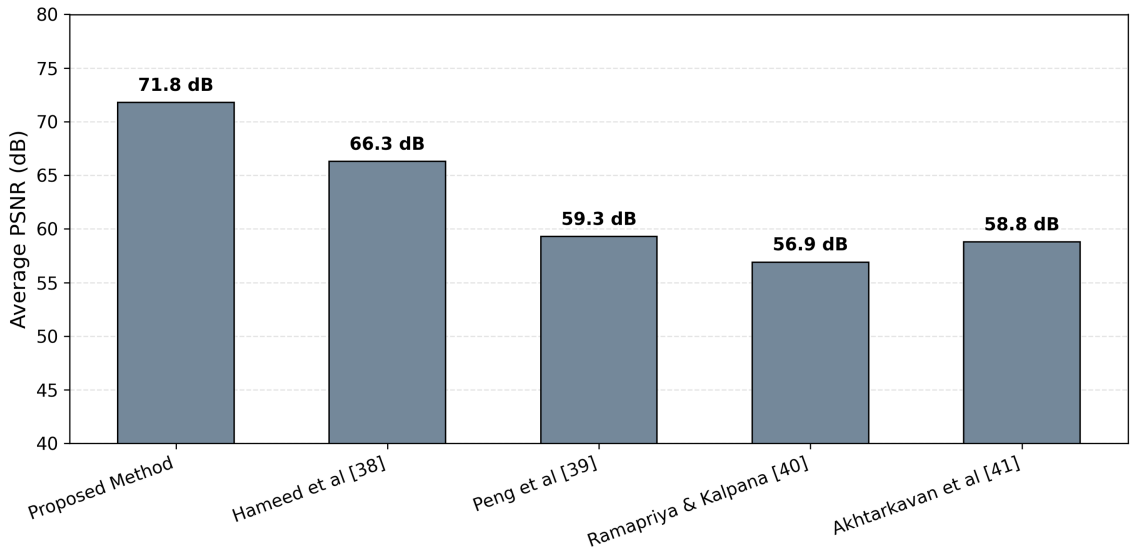

Figure 6: Comparison of the average PSNR against the state-of-the-art in [38–41].

This data substantiates the proposed method's superiority in maintaining a high visual integrity against the established benchmark. While more complex methods, such as the SHA-256 hashing in [38], offer

robustness, they often induce broader modifications and more noticeable visual changes across the cover image. In contrast, the proposed method's variance-based selection algorithm effectively restricts these modifications to only suitable regions, minimizing statistical distortions. Furthermore, the newly introduced redundancy-based algorithm allows for a larger embedding capacity. This solidifies the method's adaptability and suitability for complete real-world applications, as high visual integrity is paramount in the medical field.

A further comparison was conducted against the method proposed in [42] to demonstrate VARStego's competitiveness at higher payload sizes, using the five images labeled image-01 through image-05 as the common evaluation dataset. At payload sizes of 40, 80, and 100 kb, the method in [42] achieved average PSNR scores of 59.29, 56.33, and 55.32 dB, respectively as shown in Fig. 7. In comparison, VARStego achieves an average PSNR of 61.188 dB at an ASCII payload of 5 KB, which is equal to 40 kb, demonstrating its superiority by nearly 2 dB. Furthermore, at 10 and 12.5 KB of ASCII payload sizes, equal to 80 and 100 kb, the proposed method achieves a score of 58.399 and 57.502 dB, showing a consistently higher performance compared to [42] at the same payload size by around 2 dB as well. More notably, even at an ASCII payload of 20 KB, equivalent to 160 kilobits, a payload size 60% larger than the maximum tested by [42], VARStego achieves an average PSNR of 55.590 dB as shown in Table 5. This result outperforms the score achieved by [42] at its maximum tested payload of 100 kilobits. This demonstrates that VARStego produces higher quality stego images at equivalent payload sizes and also degrades at a substantially slower rate as the payload increases, a property that can be directly attributed to the Huffman pre-compression stage and the variance-based adaptive embedding process. While this comparison involves differing payload types, ASCII text represents the most contextually appropriate format for medical steganography, as sensitive electronic health records and clinical reports are predominantly text-based.

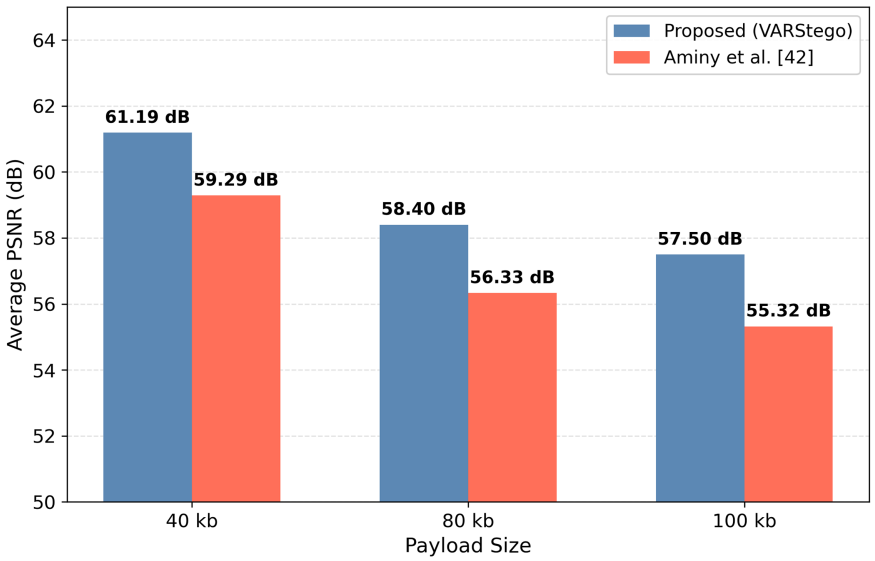

Figure 7: PSNR comparison of the proposed method against [42].

4.6 Steganalysis Results

To evaluate the resistance of the proposed VARStego method against steganalysis attacks, a security evaluation was conducted using the dataset of medical images from [37]. This dataset consists of 5000 grayscale medical images, each embedded with 10 KB of ASCII payload data using the proposed method to produce the corresponding stego images. Three state-of-the-art deep learning-based steganalysis detectors were employed for this evaluation: SRNet [43], Zhu-Net [44], and SiaStegNet [45]. This evaluation utilises

pretrained models of the three detectors available on Kaggle [46], with each model having been trained with a separate dataset consisting of 6000 cover images and their respective 6000 stego images. In this context, an accuracy rate of 50% represents random guessing, which is optimal from a security standpoint, as it suggests that the detector cannot distinguish between stego images and clean ones. Among the three detectors, SRNet [43] achieved the highest accuracy, recording 52% on medical images. Although these values surpass the chance baseline, they remain significantly below the threshold deemed reliable for practical steganalysis, indicating that SRNet detects only weak statistical traces and cannot make confident classifications. Both Zhu-Net and SiaStegNet performed at or below the random-chance level across both datasets. Zhu-Net [44] recorded an accuracy of 50.00%, while SiaStegNet [45] achieved 49.99%. Accuracies at or below 50% suggest that the detectors were effectively unable to identify any embedding signal, performing no better than a coin toss. These findings demonstrate that the proposed scheme is resistant to deep learning-based steganalysis.

4.7 Ablation Study

To further evaluate the proposed method, an ablation study was conducted comparing the performance of the full implementation against three other versions; a version of the method lacking the introduced redundancy-based algorithm, a version lacking the checkered pattern embedding scheme, and a version lacking both. This comparison utilized bitstring data payloads with sizes ranging from 60 to 100 kb to analyze the performance under higher payload capacities, with results measured via PSNR and SSIM. As illustrated in Fig. 8, all configurations maintain stable and consistently high PSNR across all tested payload sizes. However, the full implementation consistently outperforms the ablated versions. At a payload size of 60 kb, the full proposed method achieves a PSNR of 57.384 dB. In contrast, the versions without the redundancy algorithm yields a lower PSNR score of 57.313 dB and a 57.328, while the one lacking only the checkered implementation achieved a similar result to the full implementation with 57.393. The difference of PSNR between the versions using the redundancy algorithm and versions without using it continue the same trend as the payload increases, showcasing that the redundancy-based algorithm positively influences the achieved PSNR of the method. At the highest payload, the difference of PSNR becomes more apparent, with a difference of around 0.1 dB.

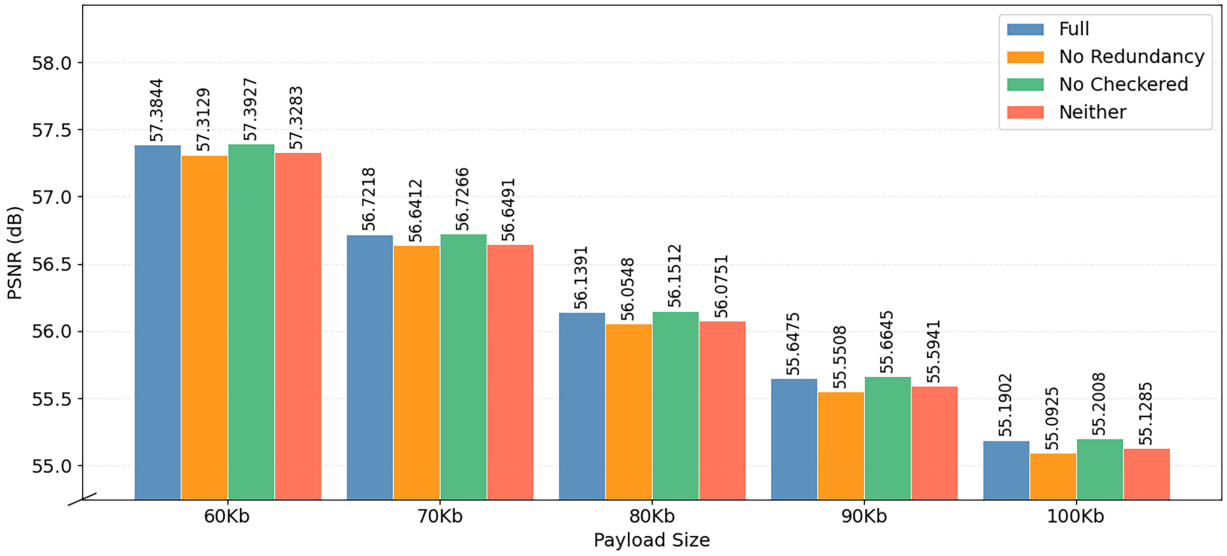

Figure 8: Ablation study PSNR results.

Notably, the data in Fig. 9 shows a similar trend for SSIM between the four versions. All versions maintain high and respectable SSIM scores hovering around 0.99. However, the full implementation remains on top with the highest SSIM of 0.9968. The version without the redundancy algorithm achieves the exact same scores as the full implementation across all payloads, while the two versions without the checkered pattern (following the sequential pattern) received a SSIM score of 0.9941 at the highest, demonstrating their inferiority in maintaining visual integrity. The SSIM difference between the versions with and without the checkered pattern widens as the payload increases, further cementing the checkered embedding pattern as an important part of increasing image integrity. At the lowest, the two versions with the checkered pattern achieve the same SSIM score of 0.9950, while the sequential pattern versions achieved 0.9915, revealing a higher difference of 0.0035 in SSIM compared to the 0.0027 difference at the highest SSIM scores. Through these results, it is clear that the full implementation of the proposed method consistently obtains the best and most stable results. Without the redundancy-based algorithm and the checkered embedding pattern, the stego image suffers from a measurable decrease in both PSNR and SSIM. This drop in quality would only increase as even larger payloads are introduced.

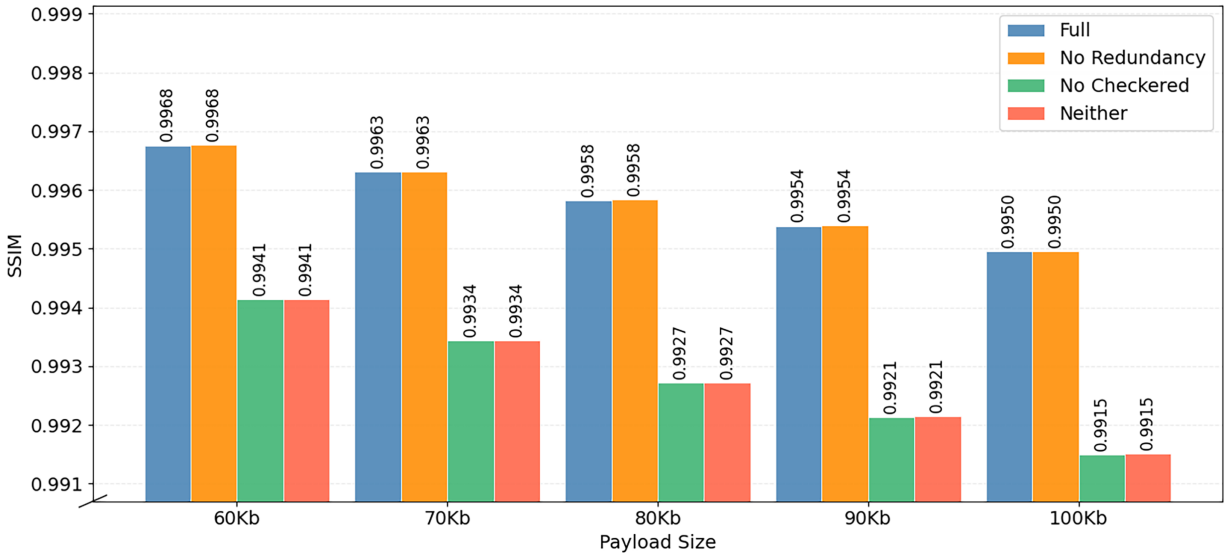

Figure 9: Ablation study SSIM results.

4.8 Discussion

The experimental results validate the efficacy of the proposed method in addressing the fundamental trade-off between embedding capacity and visual imperceptibility in medical image steganography. The method has proven that it is capable of achieving exceptional results. As evidenced in Tables 3 and 4, VARStego achieved a PSNR of 72.192 dB and near-perfect SSIM scores with bitstring payload data. Furthermore, Tables 5 and 6 present the method's performance using a more suitable ASCII payload data, with the highest PSNR of 66.589 dB using 10 kilobyte of data and similarly near-perfect SSIM scores across all payload sizes. This demonstrates the method's ability to maintain high visual imperceptibility and structural details. While these values degrade as the payload size increases, the decrease is minimal and only natural. A more in-depth analysis reveals that the statistical changes produced by the framework shown in Fig. 4 are negligible, displaying the proposed method's high structural integrity. Additionally, the method successfully chooses highly textured regions for embedding as proven through Fig. 5. These results are directly attributed to the novel dual-layered framework design.

Through the use of a Huffman Encoding scheme, any secret data is effectively compressed, reducing the number of bits required for embedding. Combined with the redundancy algorithm, the pair helps in decreasing statistical anomalies. Next, the newly introduced localized variance-based algorithm gives the framework the ability to recognize highly complicated textures, granting the ability to find suitable embeddable pixels. This fine-tuned selection directly contributes to the high visual imperceptibility and integrity that the method produces. Furthermore, the computational time taken to perform the proposed method is satisfactory, with embedding durations not exceeding 1 s and extraction durations not exceeding 0.1 s. This demonstrates the framework's viability for real-world applications, where high quality and efficiency are required. In the medical field, these points are critical, where there is a need to transmit medical images and Electronic Patient Records quickly. A thorough security test utilizing several steganalysis methods has also been conducted, revealing that the selected detection mechanisms only have, at the highest a, 52% accuracy rate of determining stego images. This means that the steganalysis methods perform an almost near 50–50 guess, performing almost no better than a coin toss at finding stego images.

Despite these strengths, the proposed method has notable limitations that warrants a discussion. First, the variance-based adaptive mechanism relies heavily on the presence of significantly textured regions within cover images to function optimally. In extremely smooth medical images the global S^2_{max} of the image approaches zero, threatening to cause nearly all pixels as low-variance. In this scenario, while the method would still have a minimum number of embeddable bits according to image size, the core advantage of the variance-based adaptive selection becomes less meaningful. Therefore, the proposed VARStego scheme is best suited for medical images with sufficient textural complexity, such as CT scans, MRI images, and chest X-rays, and may require additional safeguards or a different embedding strategy when applied to inherently smooth images.

To summarize, this experiment highlights the proposed method's advantages in securing sensitive medical data within medical images. This assumption is based on the high structural integrity and visual imperceptibility that the method is capable of, as proven through the PSNR and SSIM metrics. The decline of these key values is minimal, as it is shown that the degradation rate slows as the payload size increases, attributed to the method's Huffman Encoding scheme. The proposed method is also evaluated as superior to several methods in the state-of-the-art, where every method is given the same dataset to ensure fairness. The results in Fig. 6 illustrate the proposed method's competitiveness in creating high-quality stego images, maintaining excellent scores in X-ray and CT images. This further highlights the proposed dual-layered framework's competitiveness in real-world applications, especially in healthcare environments.

5 Conclusion

This study introduces a dual-layered steganographic framework, VARStego, engineered to hide medical patient records securely within medical images while maintaining their diagnostic value and providing the added option of reversibility. The framework utilizes a Huffman encoding scheme to preprocess patient data, reducing the load of embedding required, combined with a novel localized variance-based algorithm to accurately select suitable regions for pixel modification. This algorithm is further supported by a redundancy-based embedding algorithm to utilize the repeating patterns of pixel values to effectively make use of every space. The VARStego achieves competitive, often superior, results compared to the state-of-the-art, with PSNR values up to 72.192 dB, an average of 59.741 dB across various bitstring payload sizes, as well as an average of 57.104 dB across all used ASCII payload data. Furthermore, built to work around the human visual system, the proposed method consistently creates stego images with near-perfect SSIM scores, demonstrating its ability to preserve visual integrity. The VARStego's high structural fidelity, coupled with its XOR-based

complete image recovery algorithm, presents its suitability for practical applications in the medical field, allowing for efficient data hiding.

To further improve the proposed method, future research could investigate the method's compatibility with other adaptive steganographic techniques, such as PVD or multi-image embedding strategies. Integrating the method with alternative adaptive approaches may enhance embedding capacity while maintaining imperceptibility and robustness against statistical detection. In addition, extending the framework to multi-image scenarios could provide greater flexibility, furthering its main goal of utilizing the human visual system's weakness in perceiving small changes in highly textured areas by spreading them throughout multiple media. Although the VARStego is geared for image steganography, its core principles could be used in other media, namely, the audio and video media.

Acknowledgement: The authors express their sincere gratitude to all members of the Cyber Security Research Group, Net-Centric Computing (NCC) Laboratory, Department of Informatics, ITS, for their continuous support and insightful discussions.

Funding Statement: This research is funded by Institut Teknologi Sepuluh Nopember (ITS) and managed under the Partnership Research Grant IRN Type B Scheme (Penelitian Kemitraan IRN Tipe B), Contract No. 2654/PKS/ITS/2026.

Author Contributions: The authors confirm contribution to the paper as follows: conceptualization, Basten Andika Salim, Adifa Widyadhani Chanda D'Layla, Ntivuguruzwa Jean De La Croix, Tohari Ahmad, Kambombo Mtonga; methodology, Basten Andika Salim, Adifa Widyadhani Chanda D'Layla, Ntivuguruzwa Jean De La Croix, Tohari Ahmad, Kambombo Mtonga; software, Basten Andika Salim; validation, Basten Andika Salim, Adifa Widyadhani Chanda D'Layla; formal analysis, Basten Andika Salim, Adifa Widyadhani Chanda D'Layla, Ntivuguruzwa Jean De La Croix; investigation, Basten Andika Salim; resources, Tohari Ahmad; data curation, Basten Andika Salim; writing—original draft preparation, Basten Andika Salim; writing—review and editing, Adifa Widyadhani Chanda D'Layla, Ntivuguruzwa Jean De La Croix, Tohari Ahmad, Kambombo Mtonga; visualization, Basten Andika Salim; supervision, Tohari Ahmad; project administration, Tohari Ahmad; funding acquisition, Tohari Ahmad. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Data is available on request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Glossary

CT	Computed Tomography
dB	Decibel
DCT	Discrete Cosine Transform
DRM	DCT Residual Modulation
DWT	Discrete Wavelet Transform
HER	Electronic Health Records
$k(x, y)$	Embedding Capacity of Pixel $p(x, y)$
LSB	Least Significant Bit
MRI	Magnetic Resonance Imaging
PSNR	Peak Signal-to-Noise Ratio
PVD	Pixel Value Differencing
$p(x, y)$	Pixel of Cover Image at Coordinates (x, y)
$p'(x, y)$	Pixel of Stego Image at Coordinates (x, y)
SSIM	Structural Similarity Index Measure

$S^2_{x,y}$	Local Variance of a $p\,(x, y)$
S^2_{max}	Maximum Local Variance of Cover Image
$\delta(x, y)$	Stego Key of Pixel $p\,(x, y)$

References

1. Shojaei P, Vlahu-Gjorgievska E, Chow YW. Security and privacy of technologies in health information systems: a systematic literature review. Computers. 2024;13(2):41. doi:10.3390/computers13020041.
2. Firat Kilincer I, Ertam F, Sengur A, Tan RS, Rajendra Acharya U. Automated detection of cybersecurity attacks in healthcare systems with recursive feature elimination and multilayer perceptron optimization. Biocybern Biomed Eng. 2023;43(1):30–41. doi:10.1016/j.bbe.2022.11.005.
3. Lata K, Gupta C, Cenkeramaddi LR. A cryptographic framework for secure medical imaging in smart healthcare environments. Results Eng. 2025;27(2):106780. doi:10.1016/j.rineng.2025.106780.
4. Tang Y, Zhang M, Lai P, Yue Y, Di F. Fixed neural network image steganography based on secure diffusion models. Comput Mater Contin. 2025;84(3):5733–50. doi:10.32604/cmc.2025.064901.
5. Noor Azam MH, Ridzuan F, Mohd Sayuti MNS, Azni AH, Zakaria NH, Potdar V. A systematic review on cover selection methods for steganography: trend analysis, novel classification and analysis of the elements. Comput Sci Rev. 2025;56(6):100726. doi:10.1016/j.cosrev.2025.100726.
6. Zhang YQ, Zhong K, Wang XY. High-capacity image steganography based on discrete hadamard transform. IEEE Access. 2022;10:65141–55. doi:10.1109/ACCESS.2022.3181179.
7. Li B. Secure reversible data hiding in images with scalable capacity. In: Proceedings of the 2023 Asia Symposium on Image Processing (ASIP); 2023 Jun 15–17; Tianjin, China. p. 74–8.
8. Setiadi DRIM, Rustad S, Andono PN, Shidik GF. Digital image steganography survey and investigation (goal, assessment, method, development, and dataset). Signal Process. 2023;206(3):108908. doi:10.1016/j.sigpro.2022.108908.
9. Chen M, He P, Liu J. HLTD-CSA: cover selection algorithm based on hybrid local texture descriptor for color image steganography. J Visual Commun Image Represent. 2022;89(3):103646. doi:10.1016/j.jvcir.2022.103646.
10. Wang T, Cheng H, Liu X, Xu Y, Chen F, Wang M, et al. Lossless image steganography: Regard steganography as super-resolution. Inf Process Manag. 2024;61(4):103719. doi:10.1016/j.ipm.2024.103719.
11. Pan Y, Ni J, Liu Q, Su W, Huang J. Efficient JPEG image steganography using pairwise conditional random field model. Signal Process. 2024;221(4):109493. doi:10.1016/j.sigpro.2024.109493.
12. Mukherjee S, Mukhopadhyay S, Sarkar S. A reversible steganography algorithm using shell matrix and fake DNA sequence to secure healthcare information in IoMT cloud systems. In: Proceedings of the 2025 IEEE Madhya Pradesh Section Conference (MPCON); 2025 Aug 29–30; Jabalpur, India. p. 801–5.
13. Babu AR, Al-Fatlawy RR, Veeranjaneyulu K, Kumar KS. Canonical Huffman coding (CHC)-discrete wavelet transform (DWT) method for image steganography. In: Proceedings of the 2024 International Conference on Integrated Circuits and Communication Systems (ICICACS); 2024 Feb 23–24; Raichur, India. p. 1–4.
14. Sanjalawe Y, Al-E'mari S, Fraihat S, Abualhaj M, Alzubi E. A deep learning-driven multi-layered steganographic approach for enhanced data security. Sci Rep. 2025;15(1):4761. doi:10.1038/s41598-025-89189-5.
15. Zhou J, Lu Y, Lu G. Individualized image steganography method with dynamic separable key and adaptive redundancy anchor. Knowl Based Syst. 2025;309(3):112729. doi:10.1016/j.knosys.2024.112729.
16. Rustad S, Setiadi DRIM, Syukur A, Andono PN. Inverted LSB image steganography using adaptive pattern to improve imperceptibility. J King Saud Univ Comput Inf Sci. 2022;34(6):3559–68. doi:10.1016/j.jksuci.2020.12.017.
17. Nguyen TD, Le HQ. A secure image steganography based on modified matrix encoding using the adaptive region selection technique. Multimed Tools Appl. 2022;81(18):25251–81. doi:10.1007/s11042-022-12677-7.
18. Younis YM, Mstafa RJ, AL-Dohuki S. AttenHideNet: a novel deep learning-based image steganography method using a lightweight U-Net with soft attention. Appl Soft Comput. 2025;182(3):113583. doi:10.1016/j.asoc.2025.113583.
19. Niu L, Zhang J. An image steganography method based on texture perception. In: Proceedings of the 2022 IEEE 2nd International Conference on Data Science and Computer Application (ICDSCA); 2022 Oct 28–30; Dalian, China. p. 625–8.

20. Kadhim IJ, Premaratne P, Vial PJ. High capacity adaptive image steganography with cover region selection using dual-tree complex wavelet transform. Cogn Syst Res. 2020;60(2):20–32. doi:10.1016/j.cogsys.2019.11.002.
21. Wu DC, Shih ZN. Image steganography by pixel-value differencing using general quantization ranges. Comput Model Eng Sci. 2024;141(1):353–83. doi:10.32604/cmes.2024.050813.
22. Fahim A, Raslan Y. Optimized steganography techniques based on PVDS and genetic algorithm. Alex Eng J. 2023;85(3):245–60. doi:10.1016/j.aej.2023.11.013.
23. Sahu AK, Swain G, Sahu M, Hemalatha J. Multi-directional block based PVD and modulus function image steganography to avoid FOBP and IEP. J Inf Secur Appl. 2021;58(3):102808. doi:10.1016/j.jisa.2021.102808.
24. Gayathri R, Vandhana NS, Akshaya V, Tamizarasi S. A CQ algorithmic approach to stealthy steganography for concealing patient's medical data. In: Proceedings of the 2025 International Conference on Emerging Technologies in Engineering Applications (ICETEA); 2025 Jun 5–6; Puducherry, India. p. 1–5.
25. Geetha R, Rani DJ. Improving healthcare data transmission by combining LSB and DCT in image steganography. In: Proceedings of the 2025 International Conference on Knowledge Engineering and Communication Systems (ICKECS); 2025 Apr 28–29; Chickballapur, India. p. 1–5.
26. Huang Y, Liu Z, Wu Q, Liu X. Robust image steganography against JPEG compression based on DCT residual modulation. Signal Process. 2024;219(3):109431. doi:10.1016/j.sigpro.2024.109431.
27. Lin CY, Chiu SH. Robust coverless image steganography based on ring features and DWT sequence mapping. Signal Process Image Commun. 2026;142(28):117482. doi:10.1016/j.image.2026.117482.
28. Yao Y, Wang J, Chang Q, Ren Y, Meng W. High invisibility image steganography with wavelet transform and generative adversarial network. Expert Syst Appl. 2024;249(4):123540. doi:10.1016/j.eswa.2024.123540.
29. Rekha KS, Joe Amali M, Swathy M, Raghini M, Darshini BP. A steganography embedding method based on CDF-DWT technique for data hiding application using Elgamal algorithm. Biomed Signal Process Control. 2023;80(2):104212. doi:10.1016/j.bspc.2022.104212.
30. Liao X, Yin J, Chen M, Qin Z. Adaptive payload distribution in multiple images steganography based on image texture features. IEEE Trans Dependable Secure Comput. 2022;19(2):897–911. doi:10.1109/TDSC.2020.3004708.
31. Guan Z, Jing J, Deng X, Xu M, Jiang L, Zhang Z, et al. DeepMIH: deep invertible network for multiple image hiding. IEEE Trans Pattern Anal Mach Intell. 2023;45(1):372–90. doi:10.1109/TPAMI.2022.3141725.
32. Ramadhan IF, De La Croix NJ, Ahmad T. IRJT-Secure: open-source image steganography with Quadristego embedding and Huffman compression. Softw Impacts. 2025;26(7):100801. doi:10.1016/j.simpa.2025.100801.
33. Ping P, Wei P, Fu D, Guo B, Bloh OT, Xu F. IMIH: imperceptible medical image hiding for secure healthcare. IEEE Trans Dependable Secure Comput. 2024;21(5):4652–67. doi:10.1109/TDSC.2024.3355165.
34. DICOM Library. DICOMLibrary. Free online medical knowledge exchange portal—DICOM Library. [cited 2025 Jan 1]. Available from: https://www.dicomlibrary.com.
35. Mader KS. CT medical images. [cited 2025 Jan 1]. Available from: https://www.kaggle.com/datasets/kmader/siim-medical-images.
36. Lorem Ipsum. The standard lorem ipsum passage. [cited 2025 Jan 1]. Available from: https://www.lipsum.com.
37. Wang X, Peng Y, Lu L, Lu Z, Bagheri M, Summers RM. ChestX-Ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017 Jul 21–26; Honolulu, HI, USA. p. 3462–71.
38. Hameed MA, Hassaballah M, Abdelazim R, Sahu AK. A novel medical steganography technique based on adversarial neural cryptography and digital signature using least significant bit replacement. Int J Cogn Comput Eng. 2024;5:379–97. doi:10.1016/j.ijcce.2024.08.002.
39. Peng Y, Fu C, Zheng Y, Tian Y, Cao G, Chen J. Medical steganography: enhanced security and image quality, and new S-Q assessment. Signal Process. 2024;223(4):109546. doi:10.1016/j.sigpro.2024.109546.
40. Ramapriya B, Kalpana DY. A dual secured medical image steganography model to enhance network security based on deep learning techniques. In: Proceedings of the 2023 3rd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA); 2023 Dec 21–23; Bengaluru, India. p. 1331–8.

41. Akhtarkavan E, Majidi B, Mandegari A. Secure medical image communication using fragile data hiding based on discrete wavelet transform and A_5 lattice vector quantization. IEEE Access. 2023;11(2):9701–15. doi:10.1109/ACCESS.2023.3238575.
42. Aminy MRH, De La Croix NJ, Ahmad T, Bugingo E, Rugema FX. MedicalFuzzySec: A novel steganography technique using fuzzy logic to secure electronic patient data (EPD) concealment in medical images. Cyber Secur Appl. 2025;3:100113. doi:10.1016/j.csa.2025.100113.
43. Boroumand M, Chen M, Fridrich J. Deep residual network for steganalysis of digital images. IEEE Trans Inf Forensics Secur. 2019;14(5):1181–93. doi:10.1109/tifs.2018.2871749.
44. Zhang R, Zhu F, Liu J, Liu G. Depth-wise separable convolutions and multi-level pooling for an efficient spatial CNN-based steganalysis. IEEE Trans Inf Forensics Secur. 2020;15:1138–50. doi:10.1109/TIFS.2019.2936913.
45. You W, Zhang H, Zhao X. A Siamese CNN for image steganalysis. IEEE Trans Inform Forensic Secur. 2021;16:291–306. doi:10.1109/tifs.2020.3013204.
46. Lan DTP. Đỗ Thị Phương Lan | Kaggle. [cited 2025 Jan 1]. Available from: https://www.kaggle.com/thphnglan/code.