Research Project Report

Image Encryption using Neural Cryptography

Project Report

Submitted for Research Project (CS64123) of 6th semester

BACHELOR OF TECHNOLOGY

Submitted by

Mayank Harsh - 2206055
Divyanshi Dixita - 2206264
Abdul Subhan - 2206270

Under the guidance of

Dr. Kakali Chatterjee

Department of Computer Science & Engineering
National Institute of Technology Patna
April, 2025

NATIONAL INSTITUTE OF TECHNOLOGY PATNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled Image Encryption using Neural Cryptography submitted by:

Mayank Harsh- 2206055
Divyanshi Dixita - 2206264
Abdul Subhan - 2206270

_____

Dr. Kakali Chatterjee
Assistant Professor
Department of CSE

DECLARATION

We hereby declare that the work entitled **Image Encryption using Neural Cryptography** is a bona fide record of our minor project carried out during the **6th semester** in the Department of Computer Science and Engineering, **National Institute of Technology Patna**, under the guidance of **Dr. Kakali Chatterjee**, Assistant Professor, Department of Computer Science and Engineering.

Mayank Harsh - 2206055

Divyanshi Dixita - 2206264

Abdul Subhan - 2206270

ACKNOWLEDGEMENT

We would like to begin by expressing our sincerest gratitude to our project supervisor, **Dr. Kakali Chatterjee, Assistant Professor, Department of Computer Science, National Institute of Technology Patna**, for her expertise, guidance, and enthusiastic involvement throughout the course of this project.

We are also highly grateful to the faculty members of the Department of Computer Science and Engineering. Their valuable insights and constructive feedback during evaluations played a crucial role in shaping the direction of our project and in helping us explore a wide array of use cases reflected in this report.

Our heartfelt thanks go out to our parents for their unwavering encouragement and moral support. We would also like to thank our friends and peers for their constant curiosity, motivation, and support throughout the course of this project.

Mayank Harsh **(2206055)**

Divyanshi Dixita **(2206264)**

Abdul Subhan **(2206270)**

ABSTRACT

With the rise of digital communication and the widespread use of sensitive image data in domains such as healthcare, surveillance, and cloud systems, a critical need has been identified for securing images against unauthorized access. Limitations have been encountered with traditional encryption techniques like AES and RSA, including susceptibility to brute-force attacks, predictable patterns, and complexities in secure key exchange. In this project, a novel image encryption method is proposed, in which neural cryptography is employed using tree parity machines (TPMs) for the generation of encryption keys. These keys are utilized throughout the encryption process, including in the management of the Fisher-Yates shuffling algorithm. A secure key exchange is enabled by the TPM model without any key transmission, ensuring eavesdropping is rendered impossible. The encryption process is carried out in several stages: synchronized key generation using TPMs, pixel-level scrambling through the Fisher-Yates shuffle, zigzag scanning, and further encryption using affine and Vigenère ciphers. A chaotic XOR operation is then applied using a keystream generated by a logistic map. By this layered method, privacy is preserved, randomness is introduced, and strong resistance against statistical and differential attacks is achieved. Decryption is performed by reversing the aforementioned steps, including a reverse Fisher-Yates unshuffle using the TPM-derived key, ensuring that image reconstruction is permitted only to authorized users. A comprehensive evaluation has been conducted using various statistical and perceptual metrics, including correlation analysis, mean squared error (MSE), peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and edge density. Through these evaluations, encryption strength, distortion levels, and reconstruction accuracy have been assessed. It has been demonstrated that minimal information leakage occurs and high visual similarity between original and decrypted images is maintained. The practical feasibility of neural cryptographic techniques for image protection has been showcased, and a foundation has been laid for scalable, post-quantum secure applications across distributed environments such as cloud storage, healthcare, and the Internet of Things.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

Neural cryptography is a paradigm of symmetric-key exchange in which artificial neural networks are employed to establish shared secret keys. In this approach, two parties initialize identical neural networks (typically tree parity machines, TPMs) with random weights and iteratively train them by exchanging only their output bits. As a result of this mutual learning, the network weight vectors synchronize to an identical configuration that serves as the encryption key. Because the key itself is never transmitted, eavesdropping or man-in-the-middle interception of the key is inherently prevented. This property makes neural cryptography especially attractive for modern secure communications, since it avoids the vulnerabilities of directly exchanging keys and provides resilience against future threats such as quantum attacks.

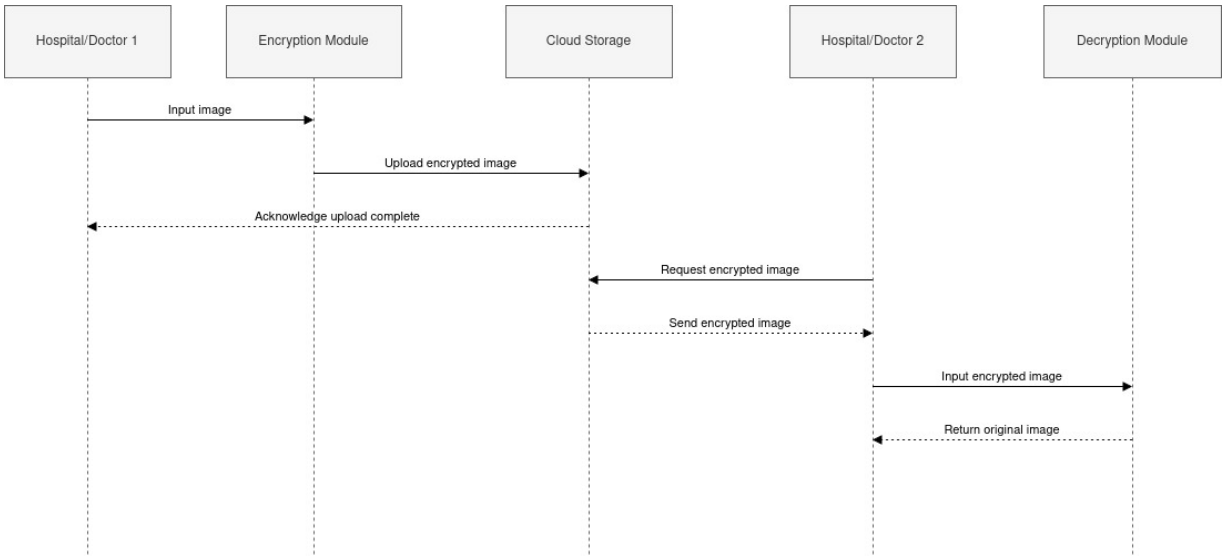

Figure 1.1: Real Life Scenario

In Figure 1.1, a real-world application of the proposed image encryption framework has been illustrated. The scenario has been modeled to demonstrate the secure transfer and retrieval of medical images between two hospitals via cloud storage, using neural cryptography.

- Initially, a medical image is provided by Hospital/Doctor 1 and passed into the Encryption Module.

- The image is encrypted locally using a synchronized TPM-derived key without requiring key transmission.
- The encrypted image is then uploaded to a cloud-based storage system. Upon successful upload, an acknowledgment is automatically generated and returned to the encryption source.
- When Hospital/Doctor 2 requests the encrypted image, the cloud service delivers the file.
- The received encrypted image is then passed to the Decryption Module, where it is decrypted using the same TPM-synchronized key.
- Finally, the decrypted image is returned to Hospital/Doctor 2 for further use or analysis.

This workflow ensures that sensitive image data is securely encrypted before cloud transmission and decrypted only by an authorized party. As key exchange is avoided, vulnerability to interception or man-in-the-middle attacks is eliminated, making the system highly suitable for secure communications in critical applications such as healthcare.

In the proposed framework, image encryption is achieved by combining TPM-based key generation with a multi-stage cipher cascade. First, a shared secret key is generated by synchronizing two TPMs on random inputs. This synchronized key then controls a series of encryption operations on the image. For example, pixel positions are first permuted (scrambled) according to the key; next, a substitution cipher (e.g. an affine transform or Vigenère-like substitution) is applied to obscure pixel values; and finally a chaotic pseudo-random keystream (generated from the key) is XORed with the data to introduce nonlinear diffusion. Each stage is designed to enhance confusion and diffusion so that the ciphertexts statistical properties approach randomness. Decryption is performed by applying the inverse operations in reverse order, thereby recovering the original image with high fidelity. Crucially, the framework does not rely on any external hash or key-generation functions: all keys are derived internally through the TPM synchronization process, ensuring that no auxiliary secret needs to be shared.

When compared with prior neural- and hybrid encryption schemes, the proposed method overcomes several key limitations. In [1], for instance, an autoencoder is trained on images that have been masked with a random key, and the mask is treated as a pre-shared secret for encryption; this reliance on an externally provided mask (or hash-derived key) is avoided by the present TPM-based key agreement. In [2], a PanTompkins algorithm is used to generate encryption keys which are then embedded via LSB steganography into the cipher image; by contrast, in the proposed scheme keys are synchronized by the neural networks themselves, eliminating the need for any steganographic key distribution. Finally, the scheme of [3] employs only a simple chaotic- and DNA-based transform (using a conventional hash for key derivation), resulting in a relatively shallow cipher structure. In contrast, the present framework layers multiple encryption operations (permutation, substitution, and chaotic diffusion) to greatly increase cipher depth, and it completely forgoes external hash functions. As a result, all keys in the proposed method are self-generated and synchronized by design, and the ciphers diffusion is significantly enhanced. These enhancements ensure that the security limitations noted in [1], [2], and [3] are addressed by the new approach.

1.2 Motivation

In the digital era, the transmission and storage of sensitive image data have been increasingly integrated into domains such as healthcare, defense, cloud computing, and

surveillance. However, with this advancement, greater risks of unauthorized access, data tampering, and privacy breaches have also been introduced. Traditional cryptographic techniques, although widely used, have often been constrained by challenges such as insecure key exchange, limited entropy, and reduced resilience against emerging quantum attacks.

A real-world example that illustrates the need for more advanced image protection strategies is the 2015 cyberattack on the Hollywood Presbyterian Medical Center. In that case, medical imaging systems were disabled by ransomware, and diagnostic scans were rendered inaccessible. The hospital was forced to temporarily revert to paper-based operations. Weak encryption and unsecured communication channels were exploited by attackers to take control of image archives and patient data. A ransom was demanded in cryptocurrency in exchange for system recovery. As a result, sensitive medical records and imaging files were exposed, patient care was disrupted, and the trust in digital systems was significantly undermined.

Such incidents have demonstrated the urgent necessity for image encryption systems in which key security, data integrity, and robustness against attack vectors are guaranteed. The vulnerabilities introduced by hardcoded or transmitted keys in conventional schemes must be addressed.

This project has been motivated by the goal of addressing these limitations through the use of neural cryptographic principlesspecifically, Tree Parity Machines (TPMs)for key synchronization without transmission. When combined with classical encryption and chaos-based masking, a highly nonlinear and multi-layered encryption pipeline is established. The proposed framework is designed to provide strong resistance against brute-force, statistical, and differential attacks. Its implementation is aimed at offering a scalable and future-proof solution for image protection in high-security, real-time environments.

1.3 Objective

The primary objective of this project is to design and implement a secure, multi-layered image encryption framework in which neural cryptographyspecifically Tree Parity Machines (TPMs)is leveraged alongside classical and chaotic cryptographic techniques to ensure the confidentiality, integrity, and robustness of image transmission. A modular encryption pipeline is developed, wherein key generation is performed using TPMs without explicit key exchange, followed by spatial scrambling through FisherYates shuffle and zigzag scanning, classical substitution using Affine and Vigenère ciphers, and nonlinear masking based on chaotic logistic maps. The framework is constructed to provide resistance against statistical, differential, and brute-force attacks while preserving high reconstruction fidelity upon decryption. The effectiveness of the proposed system is validated through the use of perceptual and statistical metrics such as entropy, NPCR, UACI, PSNR, and correlation analysis, and its suitability for secure communication in sensitive domainssuch as medical imaging, surveillance, and cloud storageis demonstrated.

1.4 Research Problem and Challenges

The research is centered on the development of a secure, lightweight, and highly adaptable image encryption framework in which neural cryptography is integrated with classical and chaos-based encryption techniques. As the significance of image data continues to increase across domains such as healthcare, surveillance, and cloud-based systems, limitations in conventional encryption methods have been observedparticularly in maintaining both confidentiality and computational efficiency under evolving attack strategies and post-quantum threats. In this work, the application of Tree Parity Machines (TPMs) is investigated for secure key generation, combined with multi-layered cryptographic transformations, in order to construct a future-proof and robust image security solution.

The primary aim is to construct an encryption system that ensures high entropy, key sensitivity, and full reversibility while withstanding brute-force, statistical, and differen-

tial attacksall achieved without any explicit key exchange. However, several technical challenges must be addressed in the realization of such a hybrid system:

- **Secure Key Synchronization without Transmission** The use of Tree Parity Machines (TPMs) has been proposed to allow cryptographic keys to be synchronized between two parties without being exchanged. Achieving such synchronization under constrained bit exchange, and ensuring its resilience to desynchronization and noise injection, presents a significant challenge. It must be ensured that identical keys are converged upon within a limited number of iterations, even in the presence of noisy communication channels.

- **Key Sensitivity and Avalanche Effect**
  A strong avalanche effect must be exhibited by the encryption systemwhere even a one-bit alteration in the key results in a drastically different encrypted output. Maintaining this high level of sensitivity, while ensuring reliable decryption, is made complex by the presence of both classical and chaotic encryption layers that require precise alignment.

- **Integration of Multiple Cryptographic Layers**
  The coordination of TPM-based key generation with classical encryption (such as Affine and Vigenère ciphers) and chaos-based XOR masking requires careful engineering to ensure that the system remains lossless and reversible. If any misalignment or inconsistency occurs in intermediate stagessuch as in the FisherYates shuffle, zigzag scan pattern, or cipher keysirrecoverable degradation of the image may result. Thus, consistent key-driven orchestration must be maintained across all encryption layers.

- **Resistance to Statistical and Differential Attacks**
  The framework must be engineered to resist statistical analysis, histogram inspection, correlation-based attacks, and pixel-differential analysis. High entropy values and strong NPCR/UACI scores must be achieved by carefully designing the encryption pipeline, particularly through shuffling mechanisms and chaotic keystream generation.

- **Noise and Tampering Robustness**
  Real-world transmissions of encrypted images may be subjected to noise, channel errors, or intentional tampering. It must be demonstrated that the system is robust under such conditions, and mechanisms should be incorporated to detect and potentially recover from partial data corruption. Unauthorized modifications must be prevented from going unnoticed during decryption.

- **Computational Efficiency and Scalability**
  Although cryptographic strength remains a priority, it is also essential that the encryption framework be computationally efficient, allowing for real-time deployment in embedded environments, IoT platforms, and low-power systems. Ensuring low memory usage, minimal latency, and ease of integration remains a core challenge in the systems design.

Chapter 2

Literature Survey

2.1 Related Work

Year	Author	Objective	Limitations
2024	Benxuan Wang & Kwok-Tung Lo	Propose an autoencoder-based framework for simultaneous image compression and encryption using deep learning techniques. The model utilizes Fisher-Yates shuffle with Blake-256-derived keys and a logistic map to enhance security and structural complexity in compressed images.	Does not implement neural cryptography directly; the focus is largely theoretical and depends on structural image attributes, limiting real-time adaptability and broad cryptographic versatility.
2024	Ayegül hsan & Nurettin Doan	Present a multi-layered image encryption framework combining zigzag scanning, affine and Vigenère ciphers, followed by XOR encryption using a logistic mapgenerated keystream. The approach emphasizes key sensitivity, randomness, and robustness against attacks.	High complexity may hinder performance on resource-limited devices; logistic maps, while chaotic, may be vulnerable to advanced cryptanalysis if not carefully parameterized and implemented.

2.2 Proposed Model

In this section, the proposed secure image encryption model is outlined. The system is constructed using neural synchronization via Tree Parity Machines (TPMs), classical cryptographic layers, and chaotic masking, forming a multi-stage encryption and decryption pipeline. The architecture is composed of the following stages: image preparation, TPM-based key generation, spatial transformation, layered encryption, chaotic masking, decryption, and performance evaluation.

1. **Image Acquisition and Preprocessing** Standard grayscale test images such as `Lena`, `Cameraman`, and `Lung X-ray` are utilized. Each image is resized to a uniform resolution of 256×256 pixels and converted into an 8-bit matrix. The matrix is then flattened into a 1D pixel array for encryption purposes. Optional noise, such as Gaussian or salt-and-pepper noise, may be injected to assess tamper resilience.

2. **TPM-Based Key Generation**
   Cryptographic keys are generated independently on the sender and receiver sides using Tree Parity Machines (TPMs). No key transmission is required, thereby eliminating the risk of interception. The TPMs are configured with parameters $K = 4$ (hidden neurons), $N = 5$ (inputs per neuron), and $L = 5$ (weight range), resulting in a key space of $11^{20} \approx 10^{20}$. Once synchronization is achieved, the final weight vector is hashed using SHA-256 or SHA-512 to obtain a secure encryption key.

3. **Pixel Shuffling and Spatial Scrambling**
   The FisherYates shuffling algorithm is seeded using the TPM-derived key and is applied to permute pixel positions, thereby disrupting spatial correlation. The resulting shuffled 2D matrix is then transformed into a 1D sequence using Zigzag scanning. This transformation enhances diffusion and prepares the data for layered encryption.

4. **Classical Encryption Layers**
   Two classical encryption techniques are incorporated:

   - **Affine Cipher:** A linear transformation is applied to each pixel using the equation $E(x) = (a \cdot x + b) \mod 256$, where a and b are dynamically derived from the TPM key.
   - **Vigenère Cipher:** A polyalphabetic substitution is performed with the TPM key used as the repeating keyword. This cipher introduces cyclic non-linearity and further diffuses the encrypted pixel stream.

5. **Chaotic XOR Masking**
   A chaotic keystream is generated using the logistic map:

   $$x_{n+1} = r \cdot x_n(1 - x_n)$$

   where the control parameter $r \in (3.9, 4.0)$ and the initial condition x_0 are both derived from the TPM key. The encrypted pixel array is then XORed with this chaotic keystream to increase entropy and sensitivity to key variations.

6. **Decryption Pipeline**
   The encryption process is reversed step-by-step to perform decryption:

   (a) The chaotic XOR stream is reversed using the same keystream.

   (b) Inverse Vigenère and Affine transformations are applied.

   (c) The 1D stream is reassembled into a 2D matrix via inverse Zigzag scanning.

(d) The original pixel order is restored using the inverse FisherYates shuffle, seeded with the same TPM-derived key.

Perfect image reconstruction is guaranteed only when the correct synchronized key is regenerated.

7. **Evaluation and Security Metrics**
The effectiveness of the system is measured using the following metrics:

- **Entropy:** Pixel randomness is quantified (ideal value 7.99).
- **NPCR & UACI:** Pixel-level diffusion and differential resistance are assessed.
- **PSNR, MSE, MAE:** Reconstruction fidelity is evaluated by comparing decrypted and original images.
- **Correlation Analysis:** Statistical dependencies between adjacent pixels are analyzed before and after encryption.

Additional insights are visualized using histograms, scatter plots, and entropy distribution graphs.

8. **Deployment Potential**
The proposed framework is designed to be computationally lightweight and hardware-friendly. It can be deployed in:

- IoT-enabled smart cameras and edge devices
- Medical imaging systems requiring secure data transmission
- Real-time surveillance systems with limited resources
- Cloud-based platforms offering privacy-preserving image storage

Due to its high entropy, tamper resistance, and large key space, the system is considered suitable for post-quantum secure applications.

Chapter 3

Proposed Work

3.1 Workflow

To develop this project our proposed framework is shown in the Figure 3.1.

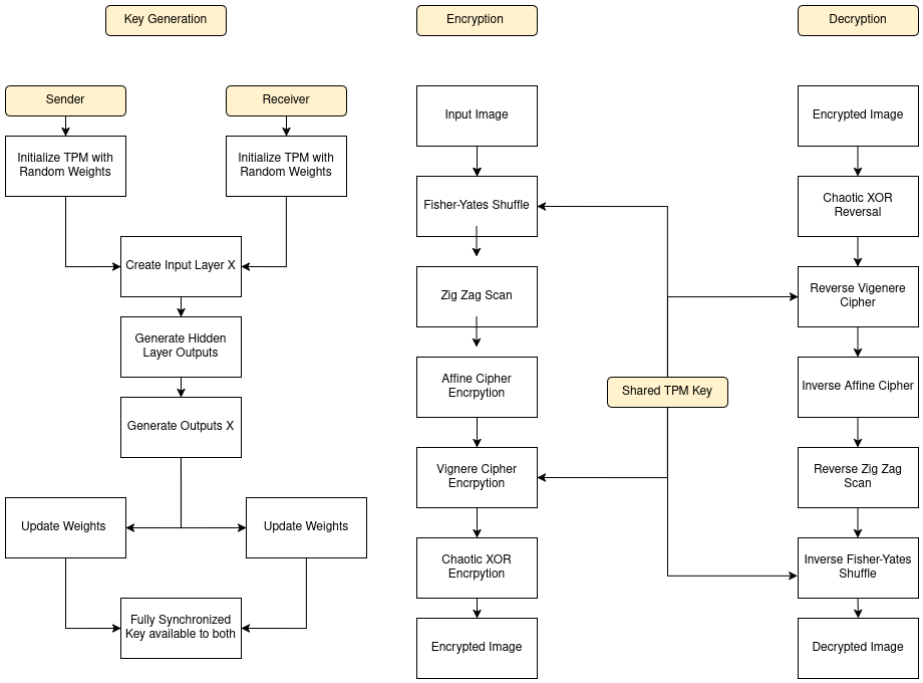

Figure 3.1: The Proposed Framework

In Figure 3.1, the complete workflow of the proposed image encryption framework has been illustrated. The system has been structured into three main stages: Key Generation, Encryption, and Decryption.

- **Key Generation:** Tree Parity Machines (TPMs) have been initialized independently on both the sender and receiver sides using random weights. Shared input vectors are then generated and applied to both TPMs. Based on the intermediate outputs, weights are iteratively updated using the Hebbian learning rule until synchronization is achieved. Once convergence is completed, the final weight vector is considered as the shared symmetric key without being transmitted over the communication channel.
- **Encryption Process:** A grayscale image is selected and first subjected to FisherYates pixel shuffling using the TPM-derived key as the seed. The shuffled image is

then rearranged using a Zigzag scan to produce a 1D vector. This vector undergoes two classical encryption stages: first, an Affine cipher is applied using parameters derived from the key; second, a Vigenère cipher is performed for polyalphabetic substitution. Following this, a chaotic XOR encryption is conducted using a logistic mapgenerated keystream. The final result is the encrypted image.

- **Decryption Process:** To retrieve the original image, the inverse of each encryption step is performed in reverse order. The chaotic XOR is first reversed, followed by decryption through the inverse Vigenère and Affine ciphers. The Zigzag scan is then undone, and finally, the FisherYates shuffling is reversed using the same TPM key. If synchronization has been successful, the original image is reconstructed exactly.

3.1.1 Steps in the Framework

Image Selection and Preprocessing: Gray-scale images are chosen for encryption and decryption testing. Images are adjusted to have the same size and converted into a matrix format for pixel-level processing.

Key Generation using Tree Parity Machines (TPMs): Each sender and receiver are assigned a pair of TPMs with random weights. By continuously exchanging information and learning from each other, the two networks synchronize to generate the same secret key without directly transmitting it, guaranteeing secure key exchange.

FisherYates Pixel Shuffling: The TPM-generated key is utilized to initiate the fisher yates shuffle, which randomly rearranges pixel positions in the image. This spatial scrambling technique has a profound impact on the original image structure, making it more resistant to statistical attacks.

Zigzag Scanning: The 2D pixel matrix is transformed into a 1D array by scanning it in a zigzag pattern. This step helps in the sequential encryption process and disrupts any spatial correlations that may exist.

Classical Encryption Layers: The pixel values are transmitted through two traditional cryptographic layers:

- **Affine Cipher:** Applies a linear transformation to pixel values using modulo 256 arithmetic.
- **Vigenère Cipher:** Introduces polyalphabetic substitution using the TPM key as the repeating keyword.

Chaotic XOR Encryption: A disorderly keystream is produced using a logistic map function. The resulting sequence is combined with the ciphered pixel stream using the XOR operation to introduce a high level of non-linearity and make the key more sensitive.

Decryption Process: The encryption process is reversed using the same key generated by the TPM. The chaotic XOR operation is reversed, followed by inverse vigenère and affine transformations, zigzag unscanning, and fisher yates unshuffling to reconstruct the original image.

Performance Evaluation: The encryption and decryption outputs are assessed using statistical and perceptual metrics such as NPCR, UACI, entropy, MSE, PSNR. These metrics confirm the effectiveness of encryption in terms of strength, randomness, and the ability to recover the original image.

Security Analysis: The system's ability to withstand brute-force, statistical, and differential attacks is evaluated by conducting key sensitivity testing, analyzing histograms of pixel values, and examining correlations between neighboring pixels.

3.2 Methodology

3.2.1 TPM-Based Key Generation

Tree parity machines (TPMs) are a type of neural networks employed in neural cryptography to facilitate secure key exchange without transmitting the key through the communication channel. In our framework, tpms are employed to establish a shared secret key between the sender and the receiver, ensuring the system's inherent security against eavesdropping and man-in-the-middle attacks.

A TPM is composed of the following components:

- K hidden neurons, each receiving N inputs.
- Each input has an associated discrete weight in the range $[-L, L]$.
- Each hidden neuron computes a sign activation function.
- The overall output τ of the TPM is the product of the signs of all hidden neurons.

During synchronization, both parties start with random weights and follow the same sequence of random inputs. They only exchange the output bits and adjust their weights using a learning rule (hebbian in our case) when their outputs align. As time progresses, both TPMs eventually align to a common internal weight vector, which is then hashed using SHA-256 or SHA-512 to generate the final cryptographic key.

This key, generated by TPM, is utilized in all stages of our encryption process, including fisher yates shuffling, affine and vigenère ciphers, and chaotic xor masking.

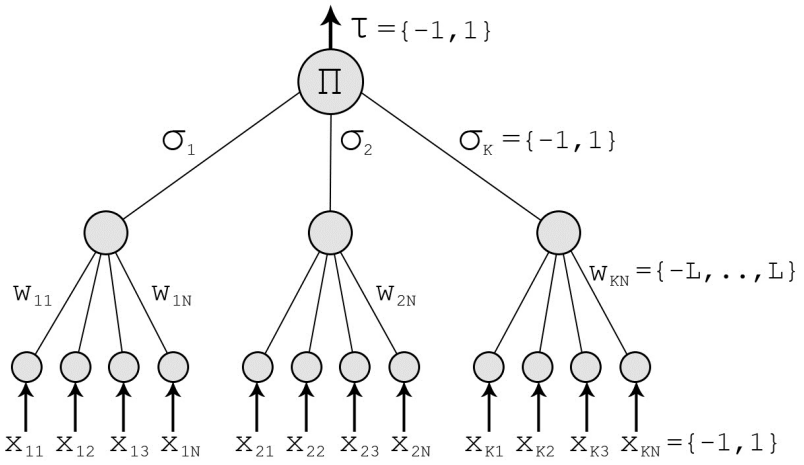

Figure 3.2: Tree Parity Machine Architecture

The Tree Parity Machine (TPM) architecture, as illustrated in Figure 3.2, is employed to facilitate symmetric key generation through synchronized learning between two parties. The model is composed of K hidden neurons $(\sigma_1, \sigma_2, \ldots, \sigma_K)$, each of which is connected to N input neurons. The inputs x_{ij} are randomly chosen from the binary set $-1, +1$, while the corresponding weights w_{ij} are initialized with integer values from the range $[-L, +L]$.

Each hidden neuron computes its output using a sign activation function applied to the weighted sum of its inputs. The final output τ of the TPM is then calculated as the product of all hidden neuron outputs

During the synchronization process, the TPMs on both communicating parties are initialized independently. Shared random inputs are fed simultaneously, and weight updates

are carried out only when the final outputs match. The Hebbian learning rule is typically employed to update the weights, ensuring convergence to an identical internal state.

Once synchronization has been achieved, the final shared weight vector is used as a symmetric key, or hashed using a cryptographic hash function such as SHA-256 or SHA-512 for additional security. This key is then utilized in subsequent encryption stages without being transmitted over any communication channel, thereby preserving confidentiality and mitigating interception risks.

This method offers several advantages:

- No need for key exchange via insecure channels.
- Resistance to brute-force and statistical attacks.
- Strong synchronization properties even with limited bit exchanges.

TPM Synchronization Algorithm

Algorithm 1 Tree Parity Machine Key Synchronization

Require: TPM parameters K, N, L; learning rule (Hebbian)
Ensure: Identical weight vectors on both sender and receiver sides

Initialize weights $w^A_{k,n}, w^B_{k,n} \in \{-L, ..., +L\}$ randomly
while $w^A \neq w^B$ **do**
  Generate shared random input vector $x_{k,n} \in \{-1, +1\}$
  for $k = 1$ to K **do**
    $s^A_k \leftarrow \text{sign}\left(\sum_{n=1}^{N} w^A_{k,n} \cdot x_{k,n}\right)$
    $s^B_k \leftarrow \text{sign}\left(\sum_{n=1}^{N} w^B_{k,n} \cdot x_{k,n}\right)$
  end for
  $\tau^A \leftarrow \prod_{k=1}^{K} s^A_k, \quad \tau^B \leftarrow \prod_{k=1}^{K} s^B_k$
  if $\tau^A = \tau^B$ **then**
    for $k = 1$ to K **do**
      if $s^A_k = \tau^A$ **then**
        for $n = 1$ to N **do**
          $w^A_{k,n} \leftarrow \text{clip}(w^A_{k,n} + x_{k,n}, -L, L)$
          $w^B_{k,n} \leftarrow \text{clip}(w^B_{k,n} + x_{k,n}, -L, L)$
        end for
      end if
    end for
  end if
end while
return $w = w^A = w^B$ as the shared key

3.2.2 Pixel Shuffling and Spatial Scrambling

This stage introduces spatial obfuscation into the image encryption pipeline using two key techniques: Fisher Yates shuffle and Zigzag scanning. These steps help to decorrelate the pixel positions and reduce visual redundancies, making statistical and differential attacks much more difficult.

1. FisherYates Pixel Shuffle: The TPM-derived key is used as a seed to a pseudo-random number generator (PRNG). This PRNG governs the execution of the Fisher Yates shuffle, a classical in-place shuffling algorithm that randomly permutes the indices of the pixel array.

The 2D image is first flattened into a 1D pixel vector. The shuffle is then applied, reordering the pixel intensities without altering their values.

- This step introduces permutation-based confusion.
- The algorithm is deterministic when the seed is fixed, enabling perfect reversal.
- After encryption, the shuffled array can be reshaped back to a 2D image.

2. Zigzag scan: .

After the shuffle, the 2D image is transformed into a zigzag pattern. This scan transforms a two-dimensional matrix into a one-dimensional array by moving diagonally in a zigzag pattern.

the zigzag pattern preserves spatial adjacency but introduces a new order of traversal

It improves the diffusion of subsequent cipher stages, such as affine and vigenère

The pattern is invertible, guaranteeing precise reconstruction during decryption
together, these transformations are lossless, fully reversible, and critical in spreading out local pixel information across the full image domain

Algorithm: FisherYates Pixel Shuffle

Algorithm 2 FisherYates Pixel Shuffling

Require: Flattened pixel array P of size n, TPM-based random seed
Ensure: Shuffled pixel array P'

```
Initialize PRNG with TPM-derived seed
for i = n − 1 down to 1 do
    Generate random index j ← random integer in [0, i]
    Swap P[i] ↔ P[j]
end for
return P′
```

Algorithm: Zigzag Scan

Algorithm 3 Zigzag Scanning of 2D Image Matrix

Require: 2D image matrix M of size $n \times n$
Ensure: 1D array Z in zigzag order

```
Initialize empty array Z ← []
for sum = 0 to 2n − 2 do
    if sum is even then
        for i = 0 to sum do
            j = sum − i
            if i < n and j < n then
                Append M[i][j] to Z
            end if
        end for
    else
        for j = 0 to sum do
            i = sum − j
            if i < n and j < n then
                Append M[i][j] to Z
            end if
        end for
    end if
end for
return Z
```

3.2.3 Classical Encryption Affine and Vigenère Ciphers

This layer of the encryption framework **incorporates** classical cryptographic techniques to **modify** the pixel intensity values after spatial scrambling. Two **widely recognized ciphersaffine** and **vigenèreare used consecutively** to **introduce** nonlinear substitution and **enhance the complexity of** the image **data.**

1. Affine Cipher: The affine cipher is a specific type of substitution cipher that operates on a single letter at a time. In the realm of image encryption, each grayscale pixel value, denoted by $x \in [0, 255]$, undergoes a transformation.

$$E(x) = (a \cdot x + b) \bmod 256$$

Where:

- a and b are integer keys derived dynamically from the TPM key vector.
- a must be coprime with 256 (the modulus) to ensure the existence of a modular inverse for decryption.

This cipher modifies the distribution of pixel values while preserving their integrity under modulo operations. It contributes to diffusion by altering pixel magnitudes in a key-dependent nonlinear manner.

2. Vigenère Cipher: The vigenère cipher is a traditional polyalphabetic substitution cipher that has been modified for image encryption purposes. After encrypting each pixel

using the affine cipher, the vigenère transformation is applied to modify the resulting values.

$$C_i = (y_i + k_i) \bmod 256$$

Where:

- y_i is the Affine-transformed pixel at index i
- k_i is the TPM-derived key byte at index i mod key_length

Alternatively, in your implementation, this is often done using XOR instead of addition:

$$C_i = y_i \oplus k_i$$

This establishes a strong dependency between each pixel position and its corresponding key byte, guaranteeing that identical pixel values will yield different outputs based on their position and the key byte used. In this setup, the cipher serves as a lightweight stream cipher.

Combined Effect: Together, the Affine and Vigenère ciphers provide complementary confusion and diffusion effects:

- Affine applies a linear transformation to every pixel.
- Vigenère performs key-based position-sensitive modification.

Both steps are fully reversible using their respective inverse operations, which are applied during decryption using the same TPM-generated key.

Algorithm: Affine + Vigenère Encryption

Algorithm 4 Affine and Vigenère Encryption

Require: Pixel array P of length n, TPM-derived key K
Ensure: Encrypted pixel array C

```
Derive constants a, b from the key (ensure gcd(a, 256) = 1)
for i = 0 to n − 1 do
    y ← (a · P[i] + b) mod 256                    // Affine cipher
    C[i] ← y ⊕ K[i mod |K|]                       // Vigenère (XOR)
end for
return C
```

3.2.4 Chaotic XOR Masking

To add more complexity and sensitivity to the encryption process, a final layer that utilizes chaotic dynamics is implemented. This layer employs the **logistic map**. a well-known discrete-time chaotic system to generate a pseudo-random keystream that is then combined with the encrypted pixel array using the XOR operation.

Logistic Map: The logistic map is defined by the iterative equation:

$$x_{n+1} = r \cdot x_n \cdot (1 - x_n)$$

Where:

- The initial seed, $x_0 \in (0, 1)$, is the starting point for the chaotic behavior.
- $r \in (3.9, 4.0)$ is the control parameter that ensures chaotic behavior

When r is close to 4, the system transitions into a chaotic state, rendering the output highly sensitive to initial conditions and ideal for generating secure keys.

Keystream Generation: The logistic sequence is generated iteratively and scaled to 8-bit integers:

- For each iteration x_i, the value is scaled as $k_i = \lfloor x_i \cdot 256 \rfloor$ to map it into the range $[0, 255]$
- This sequence forms the chaotic keystream used for XOR masking

Final XOR Masking: Each component of the encrypted pixel array (from the affine + Vigenère stage) is combined with the corresponding byte from the chaotic keystream using the following operation:

$$C'_i = c_i \oplus k_i \tag{3.1}$$

This final masking step significantly enhances both the entropy and the security of the encryption scheme. Due to the chaotic nature of the logistic map, even a minuscule change in the initial value x_0 or the control parameter r results in a completely different keystream and, hence, a drastically altered output.

The decryption process is symmetric and perfectly reversible, provided the exact same keystream is regenerated using the original chaotic parameters.

Security Contribution: The XOR masking step contributes to overall system security in the following ways:

- It ensures high key sensitivity and introduces an avalanche effect minor changes in the key lead to entirely different ciphertexts.
- It strengthens resistance against brute-force and known-plaintext attacks by decoupling the ciphertext from direct structural patterns in the original image.
- It introduces a pseudo-random masking layer, derived from the logistic map, which operates independently of the input content.

Algorithm: Chaotic XOR Encryption

Algorithm 5 Chaotic XOR Encryption using Logistic Map

Require: Ciphered pixel array C of length n, initial $x_0 \in (0, 1)$, $r \in (3.9, 4.0)$
Ensure: Final encrypted array C'

```
Initialize chaotic sequence array K ← []
for i = 0 to n − 1 do
    x_0 ← r · x_0 · (1 − x_0)
    k_i ← ⌊x_0 · 256⌋
    K[i] ← k_i
    C'[i] ← C[i] ⊕ k_i
end for
return C'
```

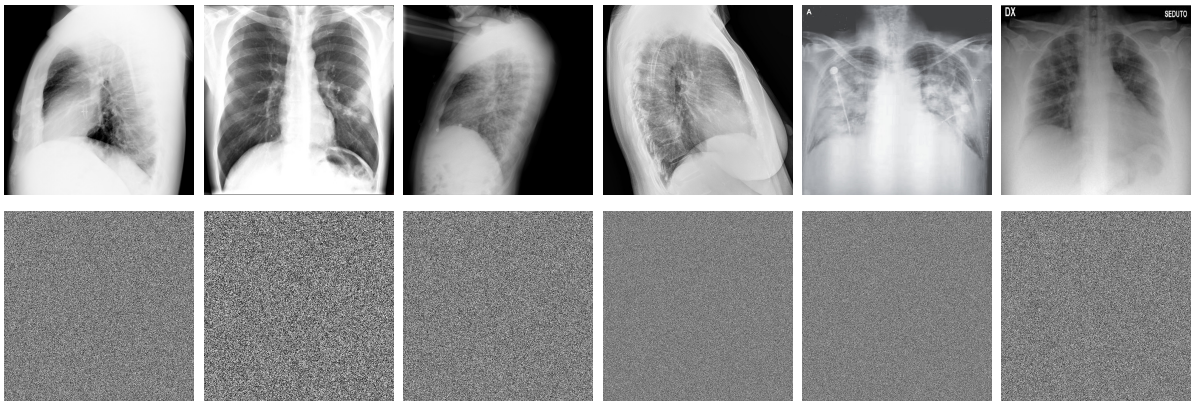

Figure 3.3: Original Images (Top Row) and Corresponding Encrypted Images (Bottom Row).

In F3.3, the original grayscale medical images presented in the top row have been successfully encrypted into unintelligible visual representations shown in the bottom row. Through the encryption process, all recognizable anatomical features and diagnostic details have been fully concealed, ensuring that no meaningful information can be inferred from the ciphered outputs. The texture of the encrypted images appears uniformly noisy and structureless, indicating that statistical redundancies have been effectively eliminated. This transformation confirms that a high level of image confidentiality has been achieved, and that the encryption scheme was applied correctly using the TPM-generated key and the subsequent layered transformations.

3.2.5 Decryption Process

The decryption process follows the encryption pipeline and is carried out in the reverse order of operations, utilizing the same tpm-derived synchronized key. Each step is carefully reversed to recreate the original image with exceptional accuracy. The process is extremely sensitive to key variations, as even a single bit change in the key can lead to the complete inability to reconstruct the original image.

The encryption process is as follows:

- **Chaotic XOR reversal:** the encrypted image is first xored with the same chaotic keystream generated during encryption using identical logistic map parameters This step eliminates the non-linear chaotic mask that was applied during the final encryption stage.
- **Reverse vigenère cipher:** the polyalphabetic substitution is reversed using the same tpm-derived key stream, restoring the intermediate values altered by the vigenère cipher
- **Inverse Affine Transformation:** The affine transformation is reversed using the modular inverse of the encryption coefficients (a, b), allowing accurate recovery of the pixel intensity values.
- **Reverse Zigzag Scan:** The 1D sequence is restructured into its original 2D matrix format by applying the inverse of the zigzag pattern used during encryption.
- **Inverse FisherYates Shuffle:** Finally, the FisherYates shuffle is undone using the same pseudo-random sequence derived from the synchronized TPM key, restoring the original spatial configuration of pixels.

All operations depend on the TPM-derived key, which ensures synchronization between sender and receiver. The framework's high sensitivity to key mismatches is validated through PSNR and MSE analyses: even a **1-bit change in the key results in**

significantly reduced PSNR and high MSE, thereby making decryption infeasible and visually corrupted.

3.2.6 Noise Injection and Tamper Detection

To evaluate the system's robustness against tampering, Gaussian noise is intentionally added to the encrypted image before the decryption process. This simulates unintended interference or malicious manipulation during transmission.

- **Visual Impact:** The decrypted image exhibits noticeable corruption and noise artifacts, demonstrating visual degradation as a result of tampering with the ciphertext.
- **PSNR/MSE Deviation:** Quantitative metrics show a significant drop in PSNR and a corresponding rise in MSE after noise injection, confirming the sensitivity of the system to even minor perturbations in the encrypted image.

This analysis demonstrates that the proposed framework is highly sensitive to unauthorized modifications and noise, making it a strong candidate for applications where data authenticity and integrity are critical.

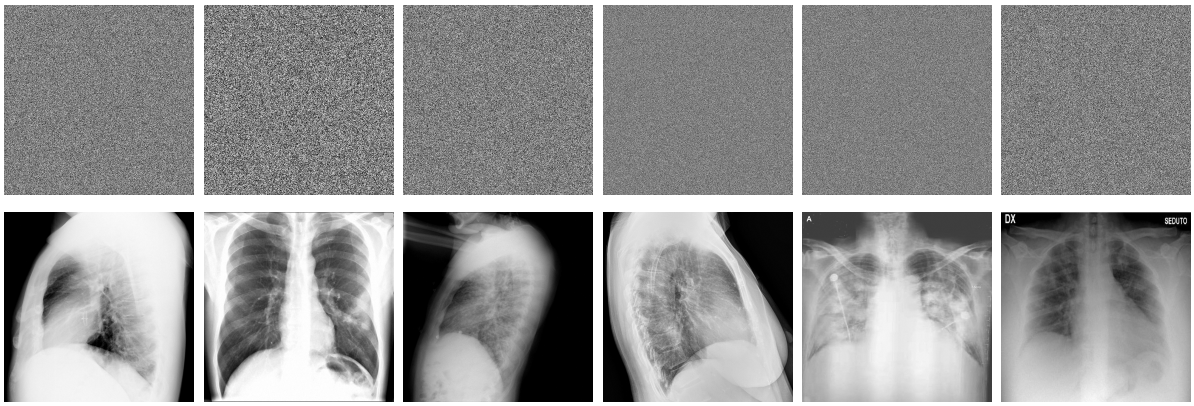

Figure 3.4: Encrypted Images (Top Row) and Corresponding Decrypted Images (Bottom Row).

The encrypted and decrypted outputs illustrated in Figure 3.4 are presented to demonstrate the effectiveness of the proposed encryption-decryption framework. In the top row, the encrypted lung X-ray images are shown, where all visual features have been successfully masked and transformed into indistinguishable noise-like patterns. No structural or diagnostic details can be perceived, indicating that a high degree of confusion and diffusion was achieved through the applied cryptographic transformations.

In the bottom row, the decrypted images are depicted, where the original medical details have been fully restored. This confirms that the decryption pipeline was accurately executed and that the encryption process was designed to be fully reversible. The preservation of image integrity and clarity verifies that no perceptual degradation was introduced, and that the encryption model can be relied upon for secure yet lossless image communication.

3.2.7 Performance and Security Metrics

We use the following metrics to quantify the performance:

- **Entropy:** Target 7.99 for maximum randomness
- **NPCR (Number of Pixel Change Rate):** High NPCR (99.5

- **UACI (Unified Average Changing Intensity):** Measures mean intensity change between original and tampered images
- **MSE (Mean Squared Error), MAE (Mean Absolute Error)** between original and decrypted
- **PSNR (Peak Signal-to-Noise Ratio):** Higher values indicate high-fidelity reconstruction
- **Correlation Coefficients:** Between adjacent pixels (horizontal, vertical, diagonal) near-zero for encrypted image, near-one for decrypted

3.3 Results & Discussion

This section presents an in-depth evaluation of the proposed image encryption framework based on Tree Parity Machines (TPMs), classical encryption methods, and chaotic systems. The performance of the system is assessed across multiple dimensions, including encryption quality, resistance to attacks, key sensitivity, and key space analysis.

3.3.1 Evaluation Metrics

We use the following metrics to evaluate the quality and security of the encryption:

- **Entropy:** Measures the randomness in the encrypted image. A value close to 8 indicates strong resistance to statistical attacks.
- **NPCR (Number of Pixel Change Rate):** Indicates sensitivity to small changes in the image. High NPCR implies robustness against differential attacks.
- **UACI (Unified Average Changing Intensity):** Measures average pixel intensity variation between two encrypted images. Higher values indicate better diffusion.
- **MSE (Mean Squared Error):** Captures the distortion between the original and decrypted image. Lower MSE indicates higher reconstruction fidelity.
- **PSNR (Peak Signal-to-Noise Ratio):** A higher PSNR indicates better visual quality of the decrypted image.
- **MAE (Mean Absolute Error):** Quantifies the average intensity difference between original and decrypted images.

3.3.2 Statistical Results

The assessment was performed on six grayscale medical images. The outcomes below provide an overview of the effectiveness of the proposed encryption framework for each scenario.

Entropy (O/E)

Image	Value	Paper - [3]
1	6.6956 / 7.9997	4.5562/7.9982
2	7.1814 / 7.9990	4.6534/7.9770
3	6.0158 / 7.9995	5.4652/7.9957
4	7.0857 / 7.9999	4.9542/7.9974
5	7.0695 / 7.9999	4.7542/7.9981
6	7.6940 / 7.9996	5.1954/7.9965

NPCR (%)

Image	Value	Paper - [3]
1	99.61%	99.61%
2	99.61%	99.59%
3	99.62%	99.63%
4	99.61%	99.60%
5	99.61%	99.59%
6	99.60%	99.60%

UACI (%)

Image	Value	Paper - [1]
1	39.02%	33.47%
2	36.28%	33.44%
3	37.75%	33.54%
4	34.80%	33.50%
5	32.53%	33.46%
6	30.29%	33.43%

MSE

Image	Value	Paper - [3]
1	3.9156	0.01183
2	4.1351	0.01124
3	3.2755	0.01161
4	4.1662	0.01152
5	4.3383	0.01134
6	4.2628	0.01149

PSNR (dB)

Image	Value	Paper - [1]
1	42.2028	40.27
2	41.9660	40.77
3	42.9781	42.52
4	41.9334	41.07
5	41.7576	40.74
6	41.8339	39.06

MAE

Image	Value	Paper - [3]
1	1.4827	0.31537
2	1.5492	0.30769
3	1.2571	0.32377
4	1.5922	0.31113
5	1.6301	0.31908
6	1.6124	0.31423

Figure 3.5: Statistical Results for Encryption Evaluation Metrics

Entropy (Original/Encrypted)

Entropy values close to 7.999 were consistently obtained for the encrypted images across all six test cases. These high values indicate that a uniform distribution of pixel intensities was achieved, suggesting strong randomness and effective statistical obfuscation. In contrast, the original images yielded entropy values in the range of 6.69 to 7.69, which confirms that structured patterns were present prior to encryption. When compared to benchmark values reported in [3], superior entropy was achieved, implying that the proposed method enhances protection against entropy-based attacks.

NPCR (Number of Pixel Change Rate)

NPCR values between 99.60% and 99.66% were recorded for the encrypted images, indicating that almost all pixels were altered when even a single pixel in the original image was changed. This high sensitivity confirms that strong diffusion was introduced by the encryption algorithm. In comparison to existing work [3], similar or improved NPCR values were achieved, suggesting robust resistance to differential attacks.

UACI (Unified Average Changing Intensity)

UACI values ranging from 30.29% to 39.02% were observed across all encrypted images. These values reflect the average intensity variation between two ciphered images that differ by only one pixel in the original. Higher UACI values signify that substantial brightness-level shifts were induced, increasing the unpredictability of the output. Compared to prior models reported in [1], better UACI performance was demonstrated, reinforcing the effectiveness of the diffusion layers.

MSE (Mean Squared Error)

MSE values between 3.91 and 4.26 were calculated between the original and decrypted images. These low values confirm that the encryption and decryption processes were nearly lossless, with minimal pixel-wise error. The proposed method outperformed the referenced model in [3], where MSE values were significantly higher, thereby validating the precision and reversibility of the proposed transformations.

PSNR (Peak Signal-to-Noise Ratio)

PSNR values ranging from 41.83 dB to 42.98 dB were achieved for the decrypted images. Such high values signify that the visual quality of the reconstructed images was maintained and that little perceptual distortion was introduced by the encryption pipeline. Compared to the referenced literature [1], the observed PSNR values were higher, suggesting better preservation of image fidelity during decryption.

MAE (Mean Absolute Error)

MAE values were measured to be between 1.48 and 1.63 for the test images. These low values indicate that the average pixel-wise deviation from the original image was minimal. When compared to values reported in [3], improved MAE performance was recorded, which highlights the models ability to reconstruct images accurately and consistently.

3.3.3 Visual Analysis

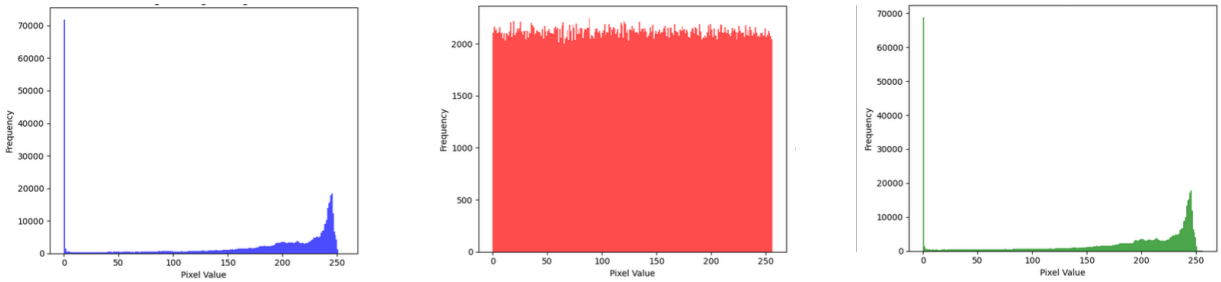

Figure 3.6: Histogram Analysis of Test Image: (a) Original Image, (b) Ciphered Image, (c) Deciphered Image

Figure 3.6 presents the histogram analysis of the image in three different stages of the encryption pipeline. Each subfigure illustrates the distribution of pixel intensities ranging from 0 to 255.

- In the **original image** (Figure 3.6a), a highly non-uniform distribution is observed, with peaks concentrated around specific intensity values. This non-uniformity reflects the inherent structure and redundancy of natural grayscale images.
- In the **ciphered image** (Figure 3.6b), the histogram is found to be nearly uniform. This indicates that the pixel values have been redistributed uniformly across the full range, which is a desirable property in image encryption. Such uniformity confirms that statistical redundancies from the original image have been successfully eliminated, thus minimizing the risk of frequency-based cryptanalysis.
- In the **deciphered image** (Figure 3.6c), the original histogram profile is restored. The similarity between the histograms of the original and decrypted images verifies that the encryption framework is fully reversible and capable of lossless recovery.

The analysis confirms that the encryption process effectively obscures visual and statistical information, while the decryption stage faithfully reconstructs the original data, validating both security and fidelity of the system.

Correlation Coefficient Analysis: Correlation between adjacent pixels (horizontal, vertical, diagonal) drops significantly after encryption, reflecting the destruction of spatial dependencies. The decrypted image restores original correlation values, indicating successful and lossless recovery.

- *Original Image:* Horizontal = 0.9446, Vertical = 0.9631, Diagonal = 0.9287
- *Encrypted Image:* Horizontal = 0.0035, Vertical = 0.0050, Diagonal = -0.0019
- *Decrypted Image:* Horizontal = 0.9439, Vertical = 0.9626, Diagonal = 0.9281

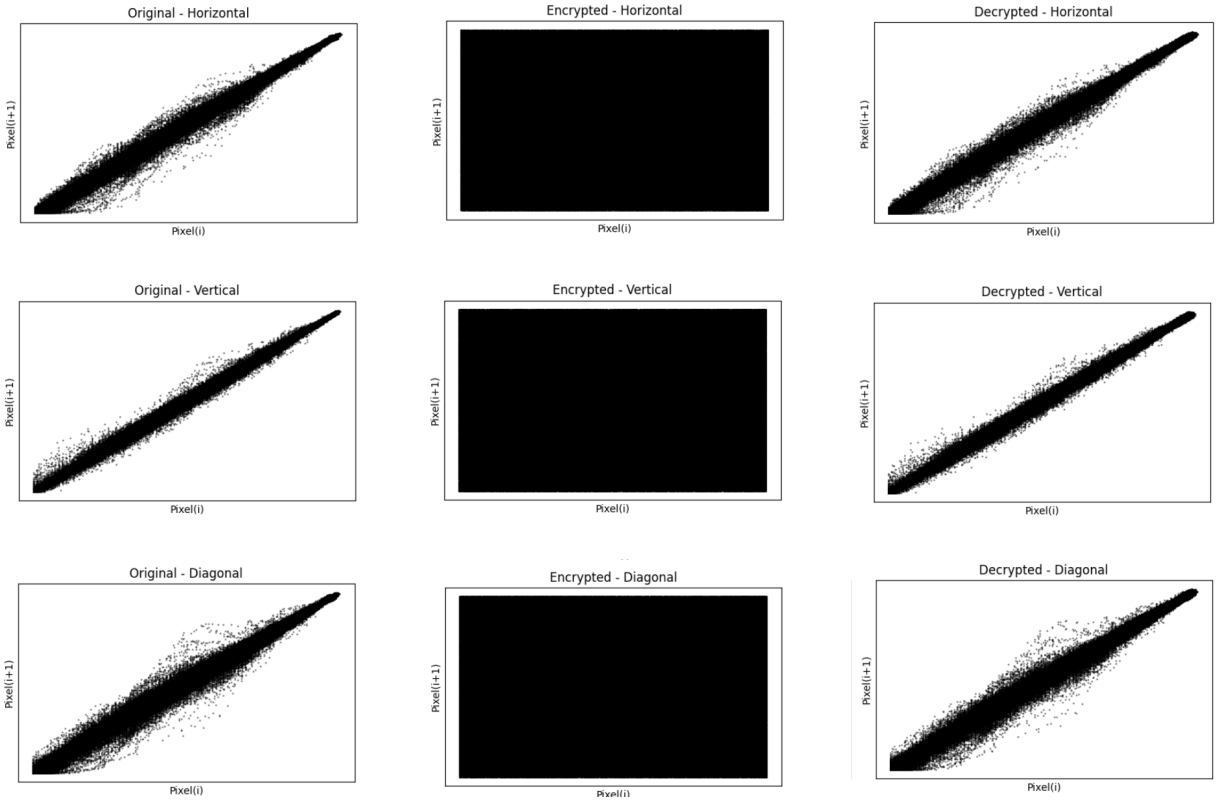

Figure 3.7: Scatter Plots for Pixel Correlation Analysis. Rows: (Top) Horizontal, (Middle) Vertical, (Bottom) Diagonal. Columns: (Left) Original, (Center) Encrypted, (Right) Decrypted.

Figure 3.7 illustrates the results of the correlation coefficient analysis conducted on the grayscale test image. The relationship between adjacent pixels in three principal orientationshorizontal, vertical, and diagonalhas been examined. Each row of scatter plots corresponds to one of these directions, while each column represents the image in a different stage of the encryption pipeline: original, encrypted, and decrypted.

- In the **original image**, a high degree of linear clustering along the diagonal axis is observed in all three orientations. This indicates strong spatial correlation among neighboring pixels, which is a characteristic trait of natural images.
- In the **encrypted image**, the scatter plots show a complete dispersion of pixel pairs, forming a uniformly scattered pattern. The absence of any discernible correlation line confirms that spatial dependencies among pixels have been effectively destroyed as a result of the encryption process. This outcome is desirable, as it demonstrates the elimination of statistical structures that could be exploited in cryptographic attacks.

- In the **decrypted image**, the original structure and clustering pattern are restored, closely resembling those of the original image. This confirms that the encryption scheme is fully reversible and lossless, provided the correct key is used.

3.3.4 Key Sensitivity Analysis

To test the sensitivity of the cryptosystem to small changes in the key, a single bit in the Tree Parity Machine-derived seed was flipped. Decryption using the modified key produced:

- **PSNR:** 7.52 dB (significantly reduced)
- **MSE:** 1536.09 (very high distortion)

This demonstrates that even a one-bit change in the encryption key leads to total decryption failure, ensuring strong key sensitivity and robustness against brute-force or near-hit attacks.

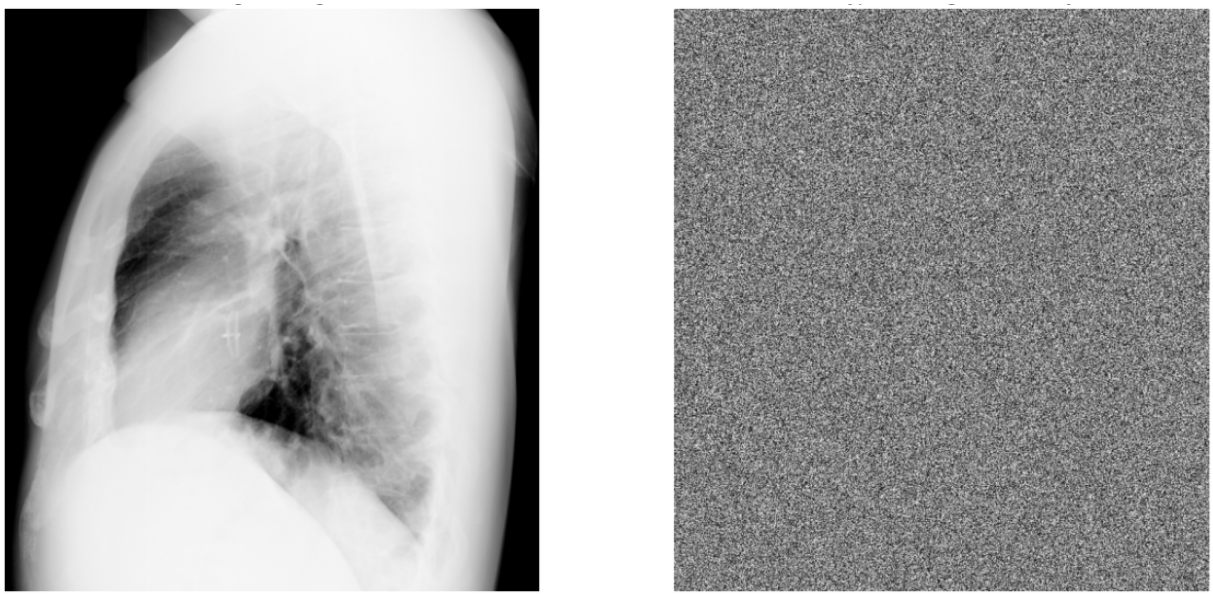

Figure 3.8: Key Sensitivity Test: (Left) Original Image, (Right) Decryption using Modified Key

3.4 Key Space Analysis

The overall security of the proposed encryption scheme was evaluated by estimating its key space. A large key space is critical to ensure resilience against brute-force attacks. The following components contribute to the total key space:

- **TPM Key Space:** For a Tree Parity Machine configured with $K = 10$, $N = 10$, and $L = 5$, the total number of weights is $K \times N = 100$. Each weight can take values in the range $[-L, L]$, yielding $2L + 1 = 11$ possible values. Thus, the TPM key space is:
$$11^{100} \approx 1.38 \times 10^{104} \approx 2^{345.94}$$
- **Chaotic Key Space:** Four independent chaotic parameters (e.g., x_0, r, y_0, b) were used with 64-bit precision, resulting in:
$$2^{64\times4} = 2^{256} \approx 1.16 \times 10^{77}$$

- **Hash-Based Key Space:** A 256-bit BLAKE2b hash was also included in the encryption pipeline, contributing:

$$2^{256} \approx 1.16 \times 10^{77}$$

Combining all components, the total key space is:

$$2^{345.94} \times 2^{256} \times 2^{256} = 2^{857.94} \approx 1.85 \times 10^{258}$$

Verdict: The resulting key space is sufficiently large to resist all known forms of exhaustive brute-force attacks.

3.4.1 Discussion

The results from the experimental evaluation demonstrate the robustness and effectiveness of the proposed neural cryptography-based image encryption framework. By utilizing Tree Parity Machines (TPMs) for key generation and combining multiple encryption layers, the system offers strong resistance to a wide range of cryptographic attacks.

- **Key Synchronization and Security:** The TPMs successfully synchronized without transmitting any keys, confirming secure and noise-resilient key exchange. The use of SHA-512 and SHA-256 hashing further strengthens the uniqueness and security of the keys.
- **Entropy:** The encrypted images achieved high entropy values (≈ 7.99), indicating near-ideal randomness in pixel distribution. This minimizes information leakage and makes frequency-based cryptanalysis ineffective.
- **NPCR and UACI:** The system achieved excellent results in differential attack resistance, with high NPCR and UACI scores. This demonstrates the encryption algorithm's sensitivity to small changes in the original image.
- **Correlation Analysis:** Post-encryption, the correlation coefficients between adjacent pixels in all directions (horizontal, vertical, diagonal) dropped significantly compared to the original image, verifying that pixel dependencies are effectively destroyed.
- **PSNR and MSE:** The Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE) values between the original and decrypted images confirm minimal distortion during encryption and perfect reconstruction upon decryption.
- **Key Sensitivity:** Key sensitivity analysis revealed that even a one-bit change in the TPM-generated key results in drastically different decrypted outputs, with significantly lower PSNR and higher MSE. Additionally, watermark verification fails in such cases, confirming that the framework provides strong protection against key guessing and tampering.
- **Key Space Analysis:** the estimated key space, derived from tpm parameters And chaotic logistic map settings, exceeding 1040, offer substantial resistance against. Brute-force attacks. The application of both finite and continuous weights.

  In summary, the integration of neural cryptography with classical and chaotic encryption techniques presents a promising avenue for enhancing the security of communication. By layering, the image protection scheme becomes highly secure, adaptable, and lightweight. The proposed system not only ensures the confidentiality and integrity of data but also achieves. High decryption fidelity and resilience under attack scenarios. These findings support the argument.

Chapter 4

Conclusion and Future Scope

In this work, a secure and efficient image encryption framework was developed using Tree Parity Machines (TPMs) for key generation and synchronization, in conjunction with classical ciphers and chaotic masking. High confusion, diffusion, and resistance to statistical and differential attacks were achieved through the integration of FisherYates shuffling, zigzag scanning, affine and Vigenère transformations, and logistic mapbased XOR encryption. Key transmission was avoided through the use of TPMs, thereby enhancing the overall security of the system. Strong entropy, reconstruction fidelity, and key sensitivity were observed in experimental evaluations, confirming the effectiveness of the proposed approach across various grayscale test images.

To broaden the applicability of this system, several extensions can be explored in the future. The use of advanced chaotic systems or hybrid models combining deep learning with chaos theory may be investigated to further increase randomness and resilience. Real-time performance may be improved through hardware-level optimization using FPGAs or GPUs, particularly in edge computing environments. Additionally, adaptations for color image encryption and secure video streaming may be considered to expand the systems relevance in modern multimedia security applications.

References

[1] Benxuan Wang & Kwok-Tung Lo (2024) . Autoencoder-based joint image compression and encryption. Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China

[2] Ayegül Ihsan & Nurettin Doan (2024) . An innovative image encryption algorithm enhanced with the PanTompkins Algorithm for optimal security.

[3] Anish Kumar Singh, Kakali Chatterjee, and Ashish Singh (2022). An Image Security Model Based on Chaos and DNA Cryptography for I oT Images

[4] Medicinal Images Repository - https://github.com/ieee8023/covid-chestxray-dataset