

Efficient matrix key homomorphic encryption of medical images

Prabhavathi Krishnegowda, Anandaraju M. Boregowda

Department of Electronics and Communication Engineering, BGS Institute of Technology, Adichunchanagiri University,
B. G Nagara, India

Article Info

Article history:

Received Sep 20, 2022

Revised Mar 2, 2023

Accepted Mar 12, 2023

Keywords:

Double sided encryption

Homomorphic encryption

Integer matrix keys

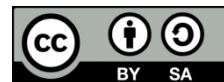
Left null space

Orthogonal matrix in Z_p

ABSTRACT

A sure way of providing privacy to sensitive images is to encrypt them, especially when they are stored in a public cloud server. Homomorphic encryption enables arithmetic operations over encrypted data without access to the secret key. This facility can be well harnessed for secure outsourced image processing by exploiting the available computational capabilities of modern cloud servers. This paper presents a new homomorphic image encryption scheme that uses integer matrix keys. The homomorphic operations are carried out in the finite field Z_p to avail the advantages of integer arithmetic and to limit the cipher text sizes to reasonable levels. Our method does not use any error vector as in learning with errors (LWE) to improve the security.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Prabhavathi Krishnegowda

Department of Electronics and Communication Engineering, BGS Institute of Technology

Adichunchanagiri University

B. G Nagara, India

Email: pdevimtech@gmail.com

1. INTRODUCTION

Images are an eloquent representation of information beyond languages. Globally, a large number of images are generated and disbursed in all walks of life, as in the field of education, E-commerce, engineering and scientific diagrams, geographical maps, health services, and so on. In most cases, these images are stored in public cloud servers (CS) for low-cost storage and distribution. When sensitive images like medical images are stored in CS, privacy and security are the major concerns [1] of the image owners. The standard practice is to use steganography or encryption [2]. For homomorphic operations, encryption is well suited, and hence, in this work, we adopt image encryption for the privacy and security of the images at the CS.

Homomorphic encryption (HE) enables mathematical computations in the cipher domain to return the correct results on decryption [3]. Here, the computational unit itself cannot decode or access the true data. Thus, using HE, privacy-preserving image processing tasks can be outsourced to the cloud servers, which can handle the heavy computational load. Therefore, with homomorphic encryption, the image storage, processing, and selective distribution tasks are delegated to the low-cost yet powerful cloud servers while maintaining adequate privacy.

Let the HE functions be represented by $E(\dots)$. Then, the encryption of a plaintext x , which is a number, gives the corresponding ciphertext, another number c as,

$$c = E(x) \tag{1}$$

let the matching decryption function be denoted by $D(\dots)$. Then the plaintext is recovered by decrypting c as,

$$x = D(c) = D(E(x)) \quad (2)$$

since $D(\dots)$ is the inverse of $E(\dots)$, they get cancelled, and we have,

$$\begin{aligned} D(E(x)) &= x \\ \text{Let } c_1 &= E(x_1) \text{ and } c_2 = E(x_2). \text{ Then, } c_1 + c_2 = E(m_1) + E(m_2) \end{aligned} \quad (3)$$

in homomorphic addition (HA), The encryption function $E(\dots)$ is constructed such that,

$$E(m_1) + E(m_2) = E(m_1 + m_2) \quad (4)$$

such $E(\dots)$ is called Additive Homomorphic Encryption. Now, $(c_1+c_2)=E(m_1+m_2)$ and the decryption of $(c_1+c_2)=D(E(m_1+m_2))$ gives the sum, (m_1+m_2) in plaintext domain. Subtraction and multiplication are similar. In the case of matrix operations, the operands M_1 and M_2 are matrices as,

$$E(M_1) \phi E(M_2) = E(M_1 \phi M_2) \quad (5)$$

here, ϕ stands for +(addition), -(subtraction), *(multiplication) and *(element-wise multiplication).

The realization of (...) that satisfies (5) is the HE for matrices. Since images are represented by matrices, homomorphic encryption of matrices facilitates homomorphic image processing (HIP). Several well-known techniques are available for image encryption [4] that provides the required security and privacy. However, in this paper, we briefly review only those works which provide homomorphic encryption to facilitate image processing in the cipher domain. Dowlin *et al.* [5], have introduced homomorphic encryption for biomedical calculations. An imaginative “manual” has been presented that uses the simple encrypted arithmetic library (SEAL) developed by microsoft research for homomorphic computations. Extensive usage of SEAL for common arithmetic and statistical operations is demonstrated. Fu *et al.* [6], have used SEAL for the homomorphic encryption of images. The authors have applied HE for image resizing using bilinear and bicubic methods. Additionally, privacy-preserving image compression and decompression are carried out using HE. Scale-invariant feature transform (SIFT) operation is outsourced using homomorphic encryption while keeping its key features intact [7]. Sift features are useful for image searching and matching. Here, the image is split into two shares and delegated to two separate cloud servers for implementing the SIFT. Then they are combined back to get the final result. Thus the process is inherently complex and time-consuming. Dowlin *et al.* [8], have presented a technique to transform previously trained neural networks into CryptoNets, which operate on encrypted data. The data owner can send queries and get back the predicted responses, both in encrypted form. Thus, the CS that hosts the CryptoNet does not access any real information about the data. However, the proposed method uses HE based on noise vectors and dual moduli, which results in reduced throughput due to the increased computational overhead when images are used as data inputs.

Shortell and Shokoufandeh [9], have designed a convolutional neural network (CNN) based classifier in the cipher domain using fully homomorphic encryption (FHE). Thus the classifier operation is outsourced to the homo-classifier housed in powerful CS. The homomorphic operations are carried out using fixed point formats to represent real numbers. Therefore the proposed scheme is computationally expensive for large-sized images. Jiang *et al.* [10], have proposed a method for the homomorphic evaluation of scale-invariant feature transform (SIFT) for encrypted images. It uses NTRU (“N-th degree truncated polynomial ring units”) to implement a full homomorphic encryption that can be scaled up easily. This scheme provides homomorphic image matching apart from SIFT coefficients. However, the SIFT calculations are not exact due to the inaccuracy in the homomorphic division. Basic image morphological operations like erosion, dilation, opening are implemented in the cipher domain using homomorphic multiplication in the finite field Z_2 [11]. For HE, the authors use monoid algebra which is not fully immune to chosen plaintext and ciphertext attacks.

Jiang and Yang [12], have implemented privacy-preserving block truncation coding (BTC) of images after image encryption based on block permutation, diffusion, and bit plane shuffling. The resulting encryption does not alter the BTC parameters, and hence the image compression process is implemented in the cipher domain based on BTC. However, BTC-based compression incurs heavy information loss and may not be suitable for general medical images. Vengadapurvaja *et al.* [13], have implemented HE using dual moduli symmetric encryption scheme using two large prime numbers. When applied to images, this method encrypts/decrypts data element by element. Therefore, for large-sized images, it is computationally costly. Li *et al.* [14], have used the elgamal method based on elliptic curve cryptography (ECC) to provide homomorphic addition of images. The main disadvantage of this method is that it cannot implement homomorphic multiplication, and thus, its applications are limited. Additionally, ECC inherently involves extensive computations and is not suitable for large-sized data, as in the case of images. Biometric images (iris, fingerprints, and faces) are encrypted for homomorphic image recognition and matching [15]. Here biometric

tokens are generated in the cipher domain for the purpose of matching. Therefore, this method is suitable for matching only and not for other image processing operations.

Fully homomorphic encryption based on LWE has been developed in [16]–[18]. But the plaintext message space is a bit in [16], [17], and an integer vector in [18]. Therefore, plaintext matrices cannot be encrypted directly. Additionally, noise vectors are to be added during encryption which limits the flexibility of these encryptions. Chao *et al.* [19], have used compact and resource-efficient CNN (CaRENets) for homomorphic inference on encrypted medical images. Vizitiu *et al.* [20], have used matrix operation for randomization or encryption (MORE) for secured data manipulation in the deep learning environment. However, in [19], [20], before encryption, the plaintext matrices must be packed column or row-wise, which leads to a higher computational cost. Dutil *et al.* [21], have used element-by-element encryption and then extended this to the whole matrix. This process consumes excessive time for encrypting large-sized matrices. Row and column shift operations must be carried out for homomorphic matrix encryption [22] which results in a higher computational cost. Zeriuoh *et al.* [23], have adopted matrix exponentiation to generate the encryption matrix keys. matrix exponentiation is computationally very expensive for image matrices of large sizes. Additionally, the eigenvalues of the plaintext matrix can be derived from the cipher matrix, which is a significant security lapse. Kumar *et al.* [24], have used blockchain and homomorphic encryption to provide privacy while aggregating medical images. However, the ‘gradients encryption’ scheme employed here leads to a substantial computational cost when the image sizes are large. Jain *et al.* [25], have deployed Cheon-Kim-Kim-Song (CKKS) homomorphic scheme for secured image encryption while the images are being used in the deep learning scenarios. The major disadvantage of this method is the complexity involved in converting the integer vectors to the corresponding cyclotomic polynomials and *vice versa*.

The main objective of this work is to mitigate the limitations of the existing HE techniques by developing an efficient matrix-based homomorphic encryption scheme for image processing, especially medical images. The proposed method is designated as matrix encryption for homomorphic image processing (MEHIP).

2. THEORETICAL BASIS

Special matrices whose elements are integers in finite field \mathbb{Z}_p are used in MEHIP for encryption and decryption. The decryption matrix key is designated by H . It is an integer matrix of size $m \times n$ in finite field \mathbb{Z}_p with $m > n$. Thus, $H \in \mathbb{Z}_p^{m \times n}$ and the elements of H are in the range 0 to $p-1$. Matrix H is generated such that,

$$\text{mod}(H^T * H, p) = I_{n \times n} \quad (6)$$

in (6), p is the modulus of the finite field \mathbb{Z}_p , and the scalar n depends on the size of the plain matrix to be encrypted. H^T is the Transpose of H and $I_{n \times n}$ is the identity matrix of size $n \times n$. The generation of H is given in (3.10.)

2.1. Encryption key matrices

Let us denote H^T by E as,

$$E = H^T \quad (7)$$

from (6) and (7),

$$\text{mod}(E * H, p) = I_{n \times n} \quad (8)$$

when there is no ambiguity, (8) can be simply rewritten as,

$$E * H = I_{n \times n} \quad (9)$$

here, the multiplication is modular in the finite field \mathbb{Z}_p , and the size of E is $n \times m$.

2.1.1. Left null space of H

Since H is a tall matrix ($m > n$), it has the left null space [26]. Let matrix F be the modular left null space of H (Determination of F is given in 3.10.). Then,

$$F * H = 0_{(m-n) \times n} \quad (10)$$

here, the size of F is $(m-n) * m$. On pre-multiplying both sides of (10) by an arbitrary random integer matrix $W_{nx(m-n)}$ that belongs to Z_p , we have,

$$W_{nx(m-n)} * (F * H)_{(m-n)xn} = W_{nx(m-n)} * 0_{(m-n)xn} = 0_{n \times n}$$

on deleting the dimension subscripts, we have,

$$W * F * H = 0 \tag{11}$$

let us denote the product $W * F$ by R as,

$$R = W * F \tag{12}$$

2.1.2. Multiple versions of R

In (12), W is an arbitrary random matrix in Z_p . Hence, it can take different values as $W \{1\}, W \{2\}, \dots, W \{i\}, \dots$, and so on. Correspondingly, R also can take different values as, $R \{1\} = W \{1\} * F, R \{2\} = W \{2\} * F, \dots$, and so on as,

$$R \{i\} = W \{i\} * F \tag{13}$$

for $i=1, 2, \dots$ and so on. $R \{i\}$'s are the multiple versions of R . In (13), the size of $W \{i\}$ is $n \times (m-n)$, that of F is $(m-n) \times m$, and that of R is $n \times m$. From (13) and (10) for $i=1, 2, \dots$,

$$R \{i\} * H = 0_{n \times n} \tag{14}$$

in (14), the size of $R \{i\}$ is $n \times m$, and that of H is $m \times n$.

Encryption matrices $G \{1\}, G \{2\}, \dots, G \{i\}$ are derived from E as, $G \{1\} = E + R \{1\}, G \{2\} = E + R \{2\}, \dots$, and so on. That is,

$$G \{i\} = E + R \{i\} \tag{15}$$

here, $G \{i\}$ is the i^{th} version of the encryption matrix. The size of $G \{i\}$ is $n \times m$. On post multiplying both sides of (15) by H , we have

$$G \{i\} * H = E * H + R \{i\} * H \tag{16}$$

from (9), $E * H = I_{n \times n}$ and from (10), $R \{i\} * H = 0_{n \times n}$. Therefore, for $i=1, 2, \dots$,

$$G \{i\} * H = I_{n \times n} \tag{17}$$

multiple versions $G \{1\}, G \{2\}, \dots$, are used for successive encryptions to prevent chosen plaintext attack (CPA) [27].

3. HOMOMORPHIC ENCRYPTION AND DESCRIPTION OF AN IMAGE MATRIX

The homomorphic encryption of an image matrix generates an entirely different matrix whose pixel elements appear to be random. Therefore, the encrypted matrix does not reveal any information about the original image. The decryption process recovers the original image matrix. The homomorphic encryption and decryption methods used in MEHIP are described in the following sections.

3.1. Homomorphic encryption

In MEHIP, an image matrix is homomorphically encrypted using $G \{i\}$'s and the H matrix as the secret keys. The double-sided encryption of the image matrix is carried out as,

$$C = \text{mod} (H * A * G \{i\}, p)$$

for convenient writing, the mod prefix and p are avoided, and C is expressed as,

$$C = H * A * G \{i\} \tag{18}$$

In (18), the multiplication operations are carried out in Z_p , using modular algebra. Further, in (18), A is the image matrix of size $n \times n$. The elements of A belong to type uint8 (8-bit unsigned integer). The size of H is

$m \times n$, and that of $G\{i\}$ is $n \times m$. Thus, the size of C is $m \times m$. The version index i of $G\{i\}$ can be any suitable value for which $G\{i\}$ has already been calculated and kept ready. Cipher matrix C is sent to the CS for secured storage and consequent access by the authorized users.

3.2. Decryption of C

On receiving matrix C from the CS, the end-user decrypts it (using modular algebra) to get B as,

$$B = H^T * C * H \quad (19)$$

substituting for C from (18) gives,

$$B = H^T * H * A * G\{i\} * H \quad (20)$$

from (6), $H^T * H = I_{n \times n}$ and from (17), $\{i\} * H = I_{n \times n}$. Using these two properties in (20) yields, $B = A$. Thus, the correctness of the decryption formula (19) is proved.

3.3. Homomorphic addition of images

In general, images are added to alter the contents of the images. The purpose of image addition could be to,

- Insert some text, timestamp, or an icon.
- Blur a region of an image for privacy.
- Modify the background or make image collages.
- Overlay the contours or edges of an image on itself.

With MEHIP, these operations can be carried out in the cipher domain with ensured privacy. Since matrix addition is used to add the respective images, the sizes of the images must be same for addition. Basically, it is pixel-wise addition. In MEHIP, the addition in the cipher domain takes place at the CS, and the cipher sum is accessed by the end user, who decrypts it to get the actual sum in the plaintext domain.

Let A_1 and A_2 be two matrices to be added whose sizes are $k \times n$. They are encrypted as,

$$\begin{aligned} C_1 &= A_1 * G\{1\} \\ C_2 &= A_2 * G\{2\} \end{aligned} \quad (21)$$

in (21), the multiplication is in the finite field Z_p . Cipher matrices C_1 and C_2 are sent to the CS. The CS adds (homomorphic addition) C_1 and C_2 (in finite field Z_p) to get C_3 as,

$$C_3 = C_1 + C_2 \quad (22)$$

C_3 is sent to the end user, who owns the decryption key H , and C_3 is decrypted as,

$$B_3 = C_3 * H \quad (23)$$

3.3.1. Correctness of decryption

From (21)-(23),

$$B_3 = (A_1 * G\{1\} + A_2 * G\{2\}) * H$$

using the property $G\{i\} * H = I_{n \times n}$, matrix B_3 is given by,

$$B_3 = A_1 + A_2$$

thus, addition in cipher domain (22) results in getting the correct result in the plaintext domain. The maximum value of an element, when two uint8 image matrices are added, can go up to $255+255=510$. Hence, for the correct result of addition (addition in Z_p should give the same result), the modulus of the finite field p has to be greater than 510. Hence the immediately succeeding prime number 521 is chosen for p .

3.3.2. Homomorphic subtraction

Homomorphic subtraction (HS) uses $C_3 = C_1 - C_2$. In this case, assuming that the result of the subtraction $(A_1 - A_2)$ is non-negative and within Z_p , the resulting decrypted B_3 would be correctly equal to $(A_1 - A_2)$. The HS can be used to recover one of the addends from the sum or to erase the target regions from an image.

3.4. Homomorphic transpose with double side encryption

Homomorphic transpose (HT) is the transpose operation on a matrix in the cipher domain. HT is used as an intermediate operand in many image processing algorithms. Consider the encryption of the plain matrix A of size $n \times n$ as,

$$C = H * A * G \{1\} \tag{24}$$

let the transpose of C (cipher domain) be carried out at the CS as,

$$C^T = (H * A * G \{1\})^T = G \{1\}^T * A^T * H^T \tag{25}$$

decrypt C^T to get B as,

$$B = H^T * C^T * H \tag{26}$$

from (25) and (26),

$$\begin{aligned} B &= H^T * (G \{1\}^T * A^T * H^T) * H = H^T * G \{1\}^T * A^T * H^T * H \\ &= (H^T * G \{1\}^T) * A^T * (H^T * H) \end{aligned} \tag{27}$$

from (6), $H^T * H = I_{n \times n}$ and from (17), $G \{i\} * H = I_{n \times n}$, which, on taking the transpose, is the same as $H^T * G \{1\}^T = I_{n \times n}$. Substitution of these relations in (27) results in,

$$B = A^T \tag{28}$$

3.5. Homomorphic multiplication

Homomorphic multiplication (HM) of plaintext square matrices A_1 and A_2 whose sizes are $n \times n$, is carried out in Z_p as follows. Initially, A_1 and A_2 are encrypted to get their cipher matrices C_1 and C_2 as,

$$C_1 = H * A_1 * G \{1\} \tag{29}$$

$$C_2 = H * A_2 * G \{2\} \tag{30}$$

sizes of C_1 and C_2 are $(m \times m)$. Now, C_1 and C_2 are sent to the CS, which calculates the product C_3 as,

$$C_3 = C_1 * C_2 \tag{31}$$

3.5.1. Decryption of C_3

Matrix C_3 , whose size is $(m \times m)$, is decrypted by the user as,

$$B_3 = H^T * C_3 * H \tag{32}$$

on substituting for C_3 from (31) and further substituting for C_1 and C_2 from (29) and (30),

$$B_3 = H^T * (H * A_1 * G \{1\}) * (H * A_2 * G \{2\}) * H \tag{33}$$

that is,

$$\begin{aligned} B_3 &= H^T * H * A_1 * G \{1\} * H * A_2 * G \{2\} * H \\ &= (H^T * H) * A_1 * (G \{1\} * H) * A_2 * (G \{2\} * H) \end{aligned}$$

from (6), $H^T * H = I_{n \times n}$, from (17), $G \{1\} * H = G \{2\} * H = I_{n \times n}$. Therefore, in (33) reduces to,

$$B_3 = A_1 * A_2 \tag{34}$$

thus, homomorphic multiplication is achieved. All the above operations are carried out using modular arithmetic in Z_p .

3.6. Discussion on homomorphic multiplication

Consider the matrix product: $B_3 = A_1 * A_2$, which is obtained after decrypting the cipher product C_3 as in (32). Since encryption, multiplication, and decryption are carried out in Z_p , the result B_3 is also in Z_p . That is, $B_3 = \text{mod}(A_1 * A_2, p)$. Hence, if the maximum element of $A_1 * A_2$ is less than p , then $\text{mod}(A_1 * A_2, p)$ is equal to the exact numerical product $A_1 * A_2$. Thus, for the correct working of the homomorphic multiplication, the constraint to be satisfied is,

$$p > \max(A_1 * A_2) \quad (35)$$

In the case of normal images, the $\max(A_1)$ is $255 = 28 - 1 \approx 28$. Therefore, the maximum individual product of 2 elements $\approx 2^{16}$. Consider the product $A_1 * A_2$, where the size of each matrix is $n \times n$. Then, in matrix multiplication, each product element is formed by the summation of n individual product terms. Therefore, the maximum element of the product can go up to $n * 2^{16}$. Hence according to (35), for the correct result:

$$p > n * 2^{16} \quad (36)$$

Under this constraint, the computational complexity will be very large. Therefore, HM is not generally feasible for the product of two regular image matrices. However, HM is useful when one matrix is a binary matrix, and the other is a regular image matrix. Then, the product represents some elementary operations on the image matrix, like row swap, and column swap. Therefore, in these cases, the HM will provide privacy-preserving image processing. Homomorphic multiplication can be used for flipping a matrix up-down or left-right.

3.7. Homomorphic flip up-down

Let matrix FI represent the flipud ($I_{n \times n}$). Then FI is an anti-diagonal matrix. An example with $n=5$ appears as,

$$FI_{5 \times 5} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

the elementary operation $FI * A$ gives flipud (A) [28]. Homomorphic flip-up-down is realized using the HM as follows. Cipher matrices C_1 and C_2 are obtained as,

$$C_1 = H * FI * G\{1\} \quad (37)$$

$$C_2 = H * A * G\{2\} \quad (38)$$

the product C_3 in the cipher domain is calculated as,

$$C_3 = C_1 * C_2 \quad (39)$$

the decryption of C_3 is carried out as,

$$B_3 = H^T * C_3 * H \quad (40)$$

after substituting for C_3 from (39), we have,

$$B_3 = H^T * C_1 * C_2 * H$$

further substitution for C_1 and C_2 from (37) and (38) gives,

$$B_3 = H^T * H * FI * G\{1\} * H * A * G\{2\} * H$$

using the properties $H^T * H = I_{n \times n}$, and $G \{1\} * H = G \{2\} * H = I_{n \times n}$, we get $B_3 = FI * A$, which is same as flipud (A), and thus, flip up-down is realized by the homomorphic multiplication. Similarly, it can be verified that $C_3 = C_2 * C_1$ realizes the homomorphic flip left-right operation.

3.8. Ciphertext expansion ratio

Ciphertext expansion ratio, designated by CER is the ratio of the size of the cipher matrix to that of its plain matrix. A large CER means the size of the cipher matrix is large, so the communication and computational costs are relatively high. A lower CER indicates higher encryption efficiency. CER is expressed as, $CER = \frac{\text{Size of Ciphermatrix in bits}}{\text{Size of Plain matrix in bits}}$. In MEHIP, the maximum possible value of an element of the cipher matrix is $(p-1)$. Therefore, the number of bits required to represent an element of the cipher matrix is, $\text{ceil}(\log_2(p-1))$ bits. With $p=521$, the value of $\log_2(p-1)=9.0224$ and $\text{ceil}(\log_2(p-1))=\text{very nearly } 9$ bits. Now, for the plain text, the bit depth is 8 bits. Therefore, for the double side encryption, from (19), the size of the cipher matrix is $m \times m$, and that of the plain matrix is $n \times n$. Hence,

$$CER = (m * m * 9)/(n * n * 8) = (9/8) * (m/n)^2 = 1.125 * (m/n)^2 \approx 1.125$$

since $m = n + 2$, for large values of n , the value of m can be approximated by n . Then, $CER=1.125$. Thus, in MEHIP, this low constant value of CER is a major advantage.

3.9. Security of MEHIP

Some of the security aspects of MEHIP are discussed in this section. Brute force guessing of secret keys is almost impossible as each element of a secret key is a 9-bit integer in Z_p . The probability of correct guessing it is 2^{-9} . Additionally, the size of each key is $m \times n$. Hence, the overall probability of correct guessing is $2^{-9 * m * n}$ which is extremely low. The security level can be increased further by taking $m > (n+2)$ at the cost of increased cipher matrix size. In MEHIP, successive encryptions use randomized encryption keys, namely $G \{i\}$'s. Therefore, the knowledge of cipher matrices corresponding to the given plain matrices cannot reveal the encryption keys. Prevention of chosen ciphertext attack will be implemented in the next version of MEHIP.

3.10. Generation of decryption and encryption key matrices E, F and H

Initially, an orthogonal integer matrix Q is generated in Z_p , by house holder construction [29] as,

$$Q = I_{m \times m} - \frac{2 * V * V^T}{V^T * V} \tag{41}$$

here, V is a non-zero column vector of size $m \times 1$. In the finite field Z_p , division by the scalar $(V^T * V)$ is replaced by multiplication by its modular inverse with respect to p . Therefore, in (41) is expressed as, (all calculations in modular algebra),

$$Q = I_{m \times m} - 2 * V * V^T * \text{ModInv}(V^T * V, p) \tag{42}$$

here, the size of Q is $m \times m$ and it is orthogonal. Thus,

$$Q^T * Q = I_{m \times m} \tag{43}$$

now, matrix Q^T is partitioned row-wise into two sub-matrices, $E_{n \times m}$, $F_{(m-n) \times m}$ and Q is partitioned column-wise into two sub-matrices, $H_{m \times n}$ and $H_{m \times (m-n)}$ as shown in (44),

$$(Q_{m \times m})^T = \begin{bmatrix} E_{n \times m} \\ - - \\ F_{(m-n) \times m} \end{bmatrix} \text{ and } Q_{m \times m} = [H_{m \times n} \mid H_{m \times (m-n)}] \tag{44}$$

now (43), $Q^T * Q = I_{m \times m}$ can be rewritten in terms of the sub-matrices as,

$$\begin{bmatrix} E_{n \times m} \\ - - \\ F_{(m-n) \times m} \end{bmatrix} * [H_{m \times n} \mid H_{m \times (m-n)}] = \begin{bmatrix} I_{n \times n} & \mid & 0_{n \times (m-n)} \\ - - & \mid & - - \\ 0_{(m-n) \times n} & \mid & I_{(m-n) \times (m-n)} \end{bmatrix} \tag{45}$$

from (45), it can be seen that,

$$E_{n \times m} * H_{m \times n} = I_{n \times n} = E * H \quad (46)$$

$$F_{(m-n) \times m} * H_{m \times n} = 0_{(m-n) \times n} = F * H \quad (47)$$

from the partition scheme of (44), it can be seen that $E_{n \times m} = H_{m \times n}^T$.

4. EXPERIMENTAL RESULTS AND DISCUSSION

4.1. Homomorphic addition of a logo to an image

In this example, the two images A_1 , an axial view of an MRI and a logo image (character 'A') represented by A_2 , are added after HE. Figure 1 shows the original images and resulting images after HE and addition. Figures 1(a) and 1(c) show A_1 and A_2 respectively. The corresponding encrypted images C_1 and C_2 are shown in Figures 1(b) and 1(d). With HE, the encrypted images are highly random and do not leak any information about the original images. The result of addition, C_3 in the cipher domain, is shown in Figure 1(e) and its decrypted version B_3 is shown in Figure 1(f).

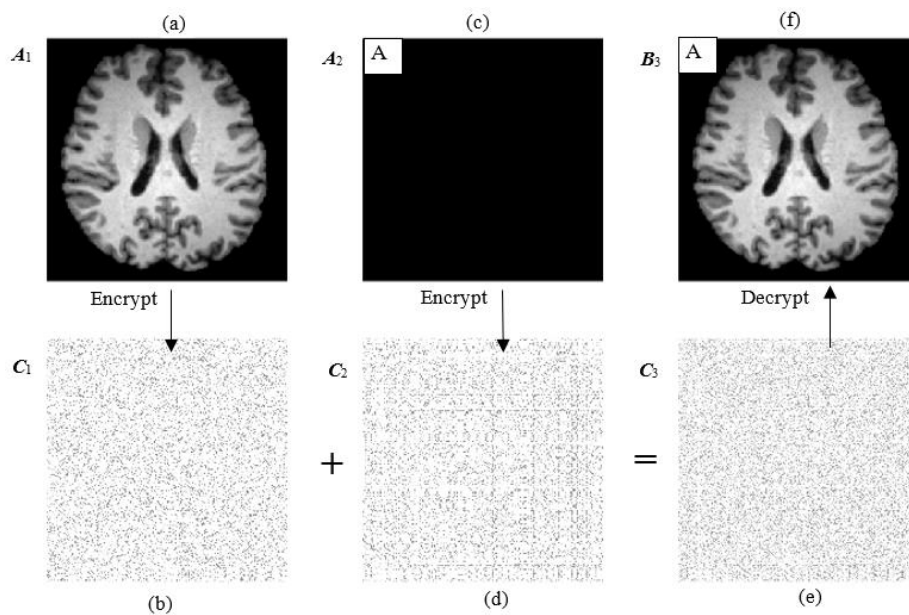


Figure 1. Homomorphic image addition show (a) A_1 , (b) C_1 , (c) A_2 , (d) C_2 , (e) C_3 , and (f) B_3

4.2. Performance analysis

In MEHIP, homomorphic encryption and decryption are carried out in \mathbb{Z}_p using integer matrix keys. The execution time of encryption is discussed in this section. In MEHIP, the double-side homomorphic encryption involves two matrix multiplications as $H * A * G \{i\}$. The sizes of H , A , and $G \{i\}$ are $(m \times n)$, $(n \times n)$, and $(n \times m)$, respectively. Therefore, the number of byte multiplications is $(m * n * n + n * n * m)$. Since $m = n + 2$, we can approximate m by n , and consequently, the number of byte multiplications would be $(2 * n^3)$.

Therefore, the execution time for HE is of the order $O(n^3)$. Since the execution time is machine dependent, we compare the execution time (ET) of MEHIP with Brakerski and Vaikuntanathan (BV) [16] method and the modified gentry, sahai, waters (GSW) [17] method. These two methods also support homomorphic multiplication. Here, starting with an image of size 32×32 , the size is progressively varied by increasing both the height and width by 32 each until the final size of 512×512 is reached. The square matrices are taken for easy calculation only. The execution times for encryption are experimentally determined for MEHIP, BV, and GSW methods. The result is shown in Figure 2. The BV method takes a slightly higher ET compared to MEHIP, as the former method must calculate the cipher text with two components. The modified GSW method takes relatively more time as the original GSW method uses messages in the form of bits, and additional manipulations and multiplications are required to handle integers, compared to MEHIP and BV methods. Since the MEHIP method directly operates on matrices, it uses the built-in matrix multiplication library so that the execution time is faster compared to the other two methods.

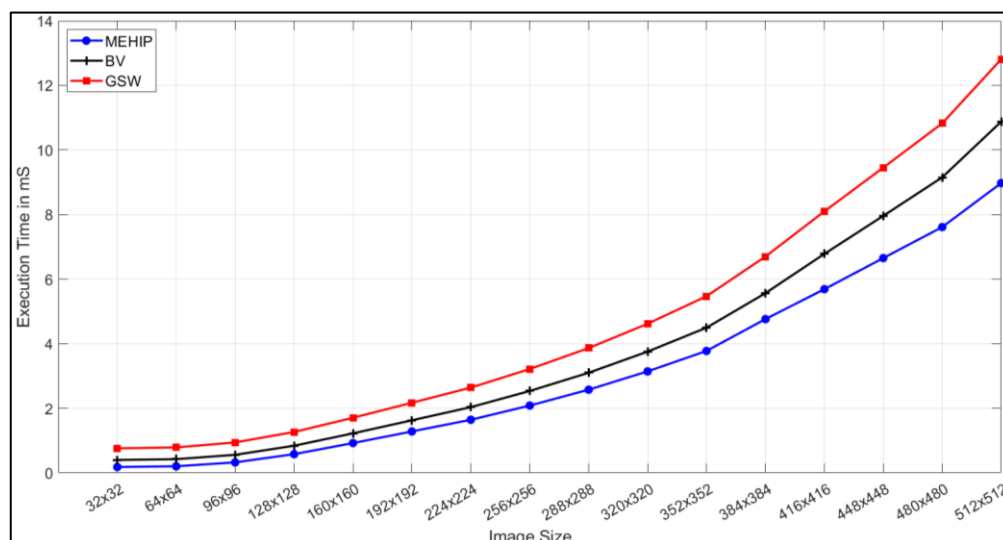


Figure 2. Comparison of execution times for homomorphic encryption of image matrices

5. CONCLUSION

A new method of homomorphic encryption based on integer matrix keys is developed for efficient image processing in the cipher domain. Apart from homomorphic addition and multiplication, the proposed method provides image flipping. The proposed method avoids the use of error vectors and re-linearization. Image addition, multiplication, flipping, and related operations are carried out in the cipher domain. It can take care of black and white, grayscale, and color images. Since the key size is relatively large (almost equal to that of the image to be encrypted, say 256×256 bytes for an image of the same size), it is immune to brute force attack. The adoption of randomized encryption for successive trials prevents chosen plaintext attacks. The ciphertext expansion ratio is slightly greater than one, which leads to lower communication overhead. The basic principle can be extended to high dynamic range images also.




REFERENCES

- [1] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Computing*, vol. 16, no. 1, pp. 69–73, Jan. 2012, doi: 10.1109/MIC.2012.14.
- [2] H. Ghanbari-Ghalehjoughi, M. Eslami, S. Ahmadi-Kandjani, M. Ghanbari-Ghalehjoughi, and Z. Yu, "Multiple layer encryption and steganography via multi-channel ghost imaging," *Optics and Lasers in Engineering*, vol. 134, p. 106227, Nov. 2020, doi: 10.1016/j.optlaseng.2020.106227.
- [3] S. Kumar, B. K. Singh, Akshita, S. Pundir, S. Batra, and R. Joshi, "A survey on symmetric and asymmetric key based image encryption," in *2nd International Conference on Data, Engineering and Applications, IDEA 2020*, Feb. 2020, pp. 1–5, doi: 10.1109/IDEA49133.2020.9170703.
- [4] K. Munjal and R. Bhatia, "A systematic review of homomorphic encryption and its contributions in healthcare industry," *Complex and Intelligent Systems*, May 2022, doi: 10.1007/s40747-022-00756-z.
- [5] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Manual for using homomorphic encryption for bioinformatics: This paper provides a new homomorphic encryption algorithm and associated software for bioinformatics to enhance the security and privacy associated with computing on human genomes," *Proceedings of the IEEE*, vol. 105, no. 3, pp. 552–567, 2017, doi: 10.1109/JPROC.2016.2622218.
- [6] W. Fu, R. Lin, and D. Inge, "Fully homomorphic imageprocessing," 2018, *arXiv: 1810.03249*.
- [7] Q. Wang, S. Hu, K. Ren, J. Wang, Z. Wang, and M. Du, "Catch me in the dark: Effective privacy-preserving outsourcing of feature extractions over image data," in *Proceedings - IEEE INFOCOM*, Apr. 2016, vol. 2016-July, pp. 1–9, doi: 10.1109/INFOCOM.2016.7524460.
- [8] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *33rd International Conference on Machine Learning, ICML 2016*, 2016, vol. 1, pp. 342–351.
- [9] T. Shortell and A. Shokoufandeh, "Secure convolutional neural networks using FHE," 2018, *arXiv: 1808.03819*.
- [10] L. Jiang, C. Xu, X. Wang, B. Luo, and H. Wang, "Secure outsourcing SIFT: Efficient and privacy-preserving image feature extraction in the encrypted domain," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 1, pp. 179–193, Jan. 2020, doi: 10.1109/TDSC.2017.2751476.
- [11] C. Lupascu, C. Plesca, and M. Togan, "Privacy preserving morphological operations for digital images," in *2020 13th International Conference on Communications, COMM 2020 - Proceedings*, Jun. 2020, pp. 183–188, doi: 10.1109/COMM48946.2020.9141997.
- [12] M. Jiang and H. Yang, "Secure outsourcing algorithm of BTC feature extraction in cloud computing," *IEEE Access*, vol. 8, pp. 106958–106967, 2020, doi: 10.1109/ACCESS.2020.3000683.
- [13] A. M. Vengadapurvaja, G. Nisha, R. Aarthy, and N. Sasikaladevi, "An efficient homomorphic medical Image encryption algorithm for cloud storage security," *Procedia Computer Science*, vol. 115, pp. 643–650, 2017, doi: 10.1016/j.procs.2017.09.150.




- [14] L. Li, A. A. El-Latif, and X. Niu, "Elliptic curve ElGamal based homomorphic image encryption scheme for sharing secret images," *Signal Processing*, vol. 92, no. 4, pp. 1069–1078, Apr. 2012, doi: 10.1016/j.sigpro.2011.10.020.
- [15] G. Pradel and C. Mitchell, "Privacy-preserving biometric matching using homomorphic encryption," in *Proceedings - 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2021*, Oct. 2021, pp. 494–505, doi: 10.1109/TrustCom53373.2021.00079.
- [16] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS*, Oct. 2011, pp. 97–106, doi: 10.1109/FOCS.2011.12.
- [17] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10624 LNCS, 2017, pp. 409–437.
- [18] J. Zhao, R. Huang, and B. Yang, "Efficient GSW-style fully homomorphic encryption over the integers," *Security and Communication Networks*, vol. 2021, pp. 1–13, Mar. 2021, doi: 10.1155/2021/8823787.
- [19] J. Chao *et al.*, "CaRENet: Compact and resource-efficient CNN for homomorphic inference on encrypted medical images," *Cryptography and Security (cs.CR)*, pp. 1–11, 2019.
- [20] A. Vizitiu, C. I. Niã, A. Puiu, C. Suci, and L. M. Itu, "Applying deep neural networks over homomorphic encrypted medical data," *Computational and Mathematical Methods in Medicine*, vol. 2020, pp. 1–26, Apr. 2020, doi: 10.1155/2020/3910250.
- [21] F. Dutil, A. See, L. D. Jorio, and F. Chandelier, "Application of homomorphic encryption in medical imaging," *Electrical Engineering and Systems Science, Image and Video Processing (eess.IV)*, pp. 1–20, 2021.
- [22] J. Chiang, "A novel matrix-encoding method for privacy-preserving neural networks (inference)," *CoRR*, pp. 1–16, 2022.
- [23] M. Zeriuoh, A. Chillali, and A. Boua, "Cryptography based on the matrices," *Boletim da Sociedade Paranaense de Matematica*, vol. 37, no. 3, pp. 75–83, Sep. 2019, doi: 10.5269/bspm.v37i3.34542.
- [24] R. Kumar *et al.*, "Blockchain and homomorphic encryption based privacy-preserving model aggregation for medical images," *Computerized Medical Imaging and Graphics*, vol. 102, p. 102139, Dec. 2022, doi: 10.1016/j.compmedimag.2022.102139.
- [25] N. Jain, K. Nandakumar, N. Ratha, S. Pankanti, and U. Kumar, "Optimizing homomorphic encryption based secure image analytics," in *IEEE 23rd International Workshop on Multimedia Signal Processing, MMSP 2021*, Oct. 2021, pp. 1–6, doi: 10.1109/MMSP53017.2021.9733620.
- [26] R. Macausland, "The moore-penrose inverse and least squares," *Advanced Topics in Linear Algebra*, pp. 1-10, 2014.

BIOGRAPHIES OF AUTHORS



Prabhavathi Krishnegowda    received B.E. in electronics and communication engineering from VTU, Karnataka, India. She holds M.Tech. degree in networking and internet engineering from VTU, Karnataka, India. Currently she is working as an assistant professor in the electronics and communication engineering department at BGS institute of Technology, Adi Chunchanagiri University (ACU), India. She has 16 years of experience in teaching and research. She has authored more than 20 publications. Her research areas are image processing, medical image analysis and pattern recognition. She has supervised and co-supervised more than 10 master's degree projects. She is an active member of professional bodies such as ISCA, IETE and ISTE. She can be contacted at email: pdevimtech@gmail.com.



Dr. Anandaraju M. Boregowda    received B.E. in electrical and electronics from University of Mysuru, India. He received M.E. in power electronics from Bangalore University, India and Ph.D. in electronics from University of Mysuru, India. Currently he is working as the professor and head of ECE department at BGS institute of Technology, Adi Chunchanagiri University (ACU), and India. He has more than 25 years of experience in teaching and research. He has authored more than 30 publications. His research areas are image processing, medical image analysis and power electronics. He has supervised and co-supervised more than 10 master's degree projects and has guided 5 Ph.D. candidates. He is a senior member in IEEE. He is a fellow member of IE(I) and an active member of professional bodies such as ISCA, IETE, and ISTE. He can be contacted at email: mb.anandaraju@gmail.com.