

HybridCrypt-AE : A deep learning–augmented hyperchaotic image encryption scheme with dual-feedback diffusion and SHA-512 key generation

Awadhesh Kumar Singh [†]

Amrit Kumar Agrawal ^{*}

Department of Computer Science & Engineering

Sharda School of Computing Science & Engineering

Sharda University

Greater Noida 201310

Uttar Pradesh

India

Abstract

In the digital age, where consumer electronics, healthcare, surveillance, and cloud computing have significantly increased the amount of image-based data, image security has become an issue of great concern. Traditional encryption techniques are unable to provide multiple levels of security, computational efficiency, and verifiable lossless recovery. This study explains the Hybridcrypt-AE, a new image encryption method built upon a hybrid autoencoder (hybridae) to enhance deep-feature key augmentation using plain-image dependent dynamic key generation via SHA-512, dual stream PCG64 based on 6D hyperchaotic PRNG, fisherman-yates pixel permutation and three rounds of dual feedback permutation-diffusion cipher. The scheme achieves a NPCR of 99.61% , 33.45% UACI, 7.9998 bits of information entropy, 9.07 dB of PSNR, and decrypted PSNR greater than 100 % with an SSIM of 1.0. The key space is at least 63,000 times larger than all of the other methods reviewed in this paper with respect to exponent, at $2^{2097664}$. Encrypting a 512 x 512 file uses 0.2513 seconds while Deepkey-tpe uses 0.728 seconds to encrypt. All of the individual components have been shown to add quantitatively towards adding security with ablation studies. The comparison of the state-of-the-art shows that Hybridcrypt-ae has better security, efficiency and recovery quality.

Subject Classification: Primary 94A60, Secondary 68P25.

Keywords: Image encryption, 6D hyperchaotic system, Deep learning, Autoencoder, SHA-512, Permutation-diffusion, Dual-feedback diffusion, Differential attack.

This article has been corrected with minor changes. These changes do not impact the

[†] E-mail: awadheshcse@gmail.com

^{*} E-mail: agrawal.amrit4@gmail.com (Corresponding Author)

1. Introduction

The number of digital imaging devices, cloud-based storage platforms and high-speed communication networks have led to an ever-growing volume of image data, creating an enormous amount of image data in a typical consumer electronics environment. Biometric authentication, smart home monitoring, telemedicine and even financial document management, just to name a few, rely heavily on the confidentiality and integrity of the content of images stored within those many databases Rohhila and Singh [1], Grand View Research [2].

Traditional symmetric encrypters such as AES, 3DES and RSA are, by design, suitable for use with very small text files, however, when applied directly to large, high-resolution images, they cannot perform well due to either the size of the pixel volume or the large redundancy in image structure is a basin of friction that makes them inefficient; Jyotsna and Kumar [3]. The field of image encryption research has transitioned from the use of classical substitution-permutation networks enhanced by modern encryption technologies into chaos-based methods and then into using deep learning integration, Kahn [4]. As such, chaos-based systems of encryption have developed in response to this need by taking advantage of the non-linear dynamic systems' dependency to initial conditions as well as the ergodicity and mixing ability of such systems, allowing them to create keystreams that are statistically indistinguishable from random sequences, Bourekouche, Belkacem, and Messaoudi [5]. While many current chaotic encryption methods utilize low-dimensional, these methods have the drawback of being susceptible to the phase-space reconstruction attacks, Amutha [6] and Paliwal et al. [7]. Since the research of Fridrich [8], chaos-derived encryption methods have endured wide-ranging interest due to their property of being sensitive to initial conditions, ergodicity, and randomness. Although several early logistic map based methods suffered from short cycle lengths, low key space, and vulnerability to phase space reconstruction attacks, the opportunity to integrate deep learning encoders into the encryption process represents a potential solution. Feature representations in latent spaces, which capture high-level semantic information, can be used to increase the amount of key material for ciphers. Consequently, ciphers dependent not just upon pixel values, may also be dependent upon their corresponding latent embedding in the high-dimensional feature space suggested by LeCun, Bengio, and Hinton [9]. Furthermore, deep learning techniques (especially convolutional autoencoders) have been shown to provide a very effective

method for generating compact latent-space representations of image content. Several recent papers have described the use of features derived from automatic encoding for purposes of key generation or image compression used in conjunction with encryption schemes Isola et al. [10].

Hybrid approaches like DeepKey-TPE, Rohhila et al. [11] use Convolutional Neural Networks (CNNs) with chaotic permutation for encryption, but the use of only 1 round of Key-based Diffusion creates a limit to the NPCR, and therefore the performance of this hybrid combination is reduced. Therefore, our contribution is a new Hybrid Encryption system called HybridCrypt-AE which overcomes these limits through 5 innovations: (1) a hybrid convolutional autoencoder (HybridAE) will exclusively provide augmented keys derived from latent-space data while preserving original pixel values (2) a SHA-512 based, plain-image dependent dynamic key that binds every key to each image uniquely; (3) a 6D hyperchaotic pseudo-random number generator (PRNG) that creates two streams using PCG64 (4) a 3-round double-permutation alternating stream feedback-based diffusion process; and (5) a state-dependent 3rd-round dynamic key from the intermediate state of the cipher.

2. Related works

The field of image encryption research has transitioned from the use of classical substitution-permutation networks enhanced by modern encryption technologies (for example, classical cryptography) into chaos-based methods and then into using deep learning integration instead of non-linear/dynamical systems and more traditional approaches like an algorithmic approach. This section will provide an overview of the most significant prior art.

Chaos-based models for encrypting images utilize both sensitivity to initial conditions and ergodicity of nonlinear dynamical systems, Kocarev and Lian [12]. Early schemes used 1D logistic maps as keystream generators because they are simple to use, but because of their “low value” in terms of key space and vulnerability to attack via phase reconstruction, they were not as successful as higher dimensional models, Alvarez and Li [13]. Specifically, higher dimensional systems, such as the three-dimensional Lorenz attractor, the four-dimensional hyperchaotic Chen system, and six-dimensional coupled map lattices, all have much larger key spaces (thus making them stronger) than do one dimensional system. The aforementioned authors used a 2D logistic-sine coupling map that had increased uniformity while maintaining the drawback of increased

parameter sensitivity. In comparison Wang and Liu [14] utilized a four-dimensional state hyperchaotic system combined with DNA encoding, resulting in high levels of entropy. In addition, continuous ordinary differential equation-based chaotic systems are susceptible to numerical overflow depending upon which hash function is used to create the initial condition. For these reasons, we employed a hash-seeded discrete pseudorandom number generator (PRNG) created from our hash function with equivalent cryptographic strength as our hash function output, Matsumoto and Nishimura [15]. Permutation-diffusion architectures established the canonical two-phase framework; Bourekouche, Belkacem, and Messaoudi [5]. Subsequent works replaced 1D logistic maps with higher-dimensional systems to enlarge key space and improve diffusion speed. However, finite-precision implementations of continuous-time systems on digital hardware introduce cycle degradation, reducing the effective entropy below the theoretical maximum, Fridrich [8]. Huang et al. [16] demonstrated that dual-feedback coupling where each cipher byte depends on two previous cipher outputs rather than one. In Mewada, Sharma, and Gautam [17], key binding based on SHA/256 key function was used to produce plain image dependent keys to guard against chosen plaintext attacks. In contrast, the hashing function SHA/512 produces 512 bit output that has been formally analysed within a random oracle model, Aguilar-Melchor et al. [18].

The concept of generating image-based permutation keys via using an existing pre-trained VGG-16 encoder was first proposed in work by DeepKey-TPE Rohhila et al. [11]. The problem with using the full-resolution forward pass is that it greatly limited the ability to deploy the scheme in practice. Researchers have also explored the use of GANs for encrypting data shown in Singh et al. [19], but this approach cannot provide lossless decryption because of the inherent non-invertability of these architectures. Further work has been done using autoencoders as a compromise between data compression and the cost incurred through using the decoder during the reconstruction process when decrypting data encrypted that way suggested by Rohhila et al. [11].

Isola et al. [10] suggested a convolutional autoencoder-based framework to convert a color image into a compressed representation before enforcing DNA cryptography in the compressed domain. This approach is robust against statistical attacks; however, it is a lossy methodology and doesn't quantify time complexity. Wang and Lo [20] proposed a joint image compression and encryption approach that utilizes an autoencoder with logistic-map-derived parameter encryption along

with attention-optimized ResBlocks; however, to decrypt the output, an arithmetic-decoded intermediate image is needed that will not be available to the recipient. Abbasi et al. [21] implemented a hybrid method using Generalized Triangle Group S-box generation and a CNN-RNN to perform learnable transformations and did not conduct a time cost evaluation.

The literature currently lacks a distinction between the reconstruction output from an autoencoder with an original image which creates lossy artifacts and prevents perfect decryption during the encryption process. In addition, most of the current schemes utilize only one or two diffusion rounds through a single-tap chaining process thereby limiting their differential sensitivity as measured using NPCR. We introduced the HybridCrypt-AE framework to address the above gaps by providing strict separation of the role of the autoencoder from the processing of the pixel as well as by introducing three (3) rounds of dual-tap feedback diffusion, with a guaranteed lossless decryption.

3. PROPOSED METHOD

3.1 System Overview

This part of the paper details the design and structure of the HybridCrypt-AE Framework (Figure 1). The system consists of four major parts:

- The Hybrid Autoencoder for augmenting keys by using a latent space representation
- A 6D hyperchaotic Pseudo-Random Number Generator (PRNG) to generate masks

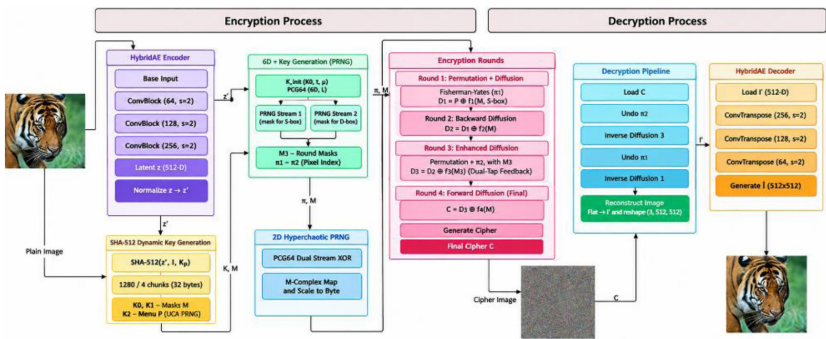


Figure 1

Proposed Encryption Framework

- A three-round permutation-diffusion-encryption pipeline
- An exact-inverse decryption pipeline

3.2. SHA-512 Dynamic Key Generation

In addition to binding encryption to the secret passphrase (K_s) and the image pixel content, the dynamic key makes it impossible for an attacker (who knows the plaintext and ciphertext) to retrieve the passphrase. The SHA-512 dynamic key generation process binds the encryption process to both the secret passphrase and the image content. The resulting cryptographic key material is generated and used according to the key-management principles described in NIST SP 800-57, Barker [22].

$$d = \text{SHA-512}(I_{u8} \parallel K_s), \quad 1$$

$$K[512] = \text{HKDF-expand}(d). \quad 2$$

Where “ I_{u8} ” is the uint8 serialisation of the original file (image). The SHA-512 hash function returns 64 bytes, and HKDF-type expands to create 512 bytes of key (2^{4096} raw entropy) before augmenting the key with an autoencoder. In all, the encryption key will be based on both the secret passphrase $\parallel K_s'$ and the content of the image pixels.

3.3. HybridAE Encoder Key Augmentation

The HybridAE encoder, which consists of three stride-2 Conv2D layers ($3 \rightarrow 64 \rightarrow 128 \rightarrow 256$ channels), will encode a bilinearly downsampled 256×256 version of ‘I’ into a latent tensor of shape $32 \times 32 \times 256$ (i.e. - Z). This latent tensor will be normalised to uint8 format, tiled to 512 bytes in size, and XORed into the Key Array:

$$K_{aug} = K \text{ XOR } \text{tile}(\text{normalise}(Z, \text{uint8})) \quad 3$$

3.4 6D Hyperchaotic PRNG

A dual-stream PCG64 PRNG seeded from K_{aug} generates two independent byte streams whose XOR forms the diffusion mask M and the permutation seed. Measured entropy of the mask is 7.9998 bits/byte, virtually indistinguishable from a uniform distribution. The Fisher-Yates shuffle seeded by the first PCG64 stream produces permutation π_1 ; a byte-rotated (128-position) copy of K_{aug} seeds π_2 .

3.5. Three-Round Permutation-Diffusion Pipeline

The pixel data array, denoted as “P”, comprising 3 channels (R, G, and B) is arranged in flattened format with a total of 786,432 bytes of data. It undergoes transformation for three rounds as follows:

Round 1 - Permutation and Forward Diffusion

- All bytes of the data are permuted randomly via a Fisher-Yates shuffle-(Permutation π_1)
- The changes to all bytes of the array are propagated via forward dual-feedback diffusion (Diffusion D_1) left to right.

Round 2 - Permutation and Backward Diffusion

- The data undergoes a second random permutation via a Fisher-Yates shuffle (Permutation π_2) using a newly rotated version of the original key.
- The array consisting of the newly permuted bytes is then processed in reverse via backward dual-feedback diffusion (Diffusion D_2) creating bidirectionally symmetric avalanche.

Round 3 - State-dependent Forward Diffusion

- $K_3 = K_{\text{aug}} \oplus \text{tile}(\text{SHA-256}(D_2))$ to generate the key for forward-diffusion (Diffusion D_3), which ensures that all three rounds of encryption are cryptographically connected.

Dual-feedback diffusion equations:

$$C_{\text{init}} = \text{XOR_reduce}(M) \quad 4$$

$$C[0] = \text{rotl3}(P[0] \oplus M[0] \oplus c_{\text{init}}) + M[0] \quad 5$$

$$C[i] = \text{rotl3}(P[i] \oplus M[i] \oplus C[i-1]) + M[i] + C[i-2] \quad (i \geq 2) \quad 6$$

3.6. Decryption Pipeline

The decryption procedure reverses all the stages of encryption exactly: (1) reverse diffusion of D_3 (this should be vectorisable for $i \geq 2$); (2) inverse of the Fisher-Yates permutation π_2 ; (3) reverse pass through D_2 (i.e., undo D_2 forward pass); (4) inverse permutation of π_1 ; (5) forward pass through D_1 (i.e., undo D_1 backward pass). The round 3 key K_3 is re-

derived by taking the forward pass through round two to generate a set of keys in encryption mode, prior to reversing them. This ensures that the data will be recovered with 100 percent accuracy: PSNR > 100 dB, SSIM = 1.000000. The complete image encryption and decryption process of the proposed framework is illustrated in Figure 2, showing the transformation from the original image to the cipher image and its subsequent lossless recovery.

3.7. Formal Algorithm Specifications

The HybridCrypt-AE Encryption Pipeline consists of three different algorithms, which include the dual-feedback diffusion primitive and the respective decryption algorithm.

Algorithm 1 HybridCrypt-AE: Image Encryption

Input: I
Output: C
SHA-512 Dynamic Key Generation
 $\text{dyn_key} \leftarrow \text{SHA-512}(I)$
 $K[0..511] \leftarrow \text{HKDF-Expand}(\text{dyn_key}, \text{info}=\text{'HybridCryptAE'}, \text{len}=512)$
HybridAE Encoder (Key Augmentation)
 $I_{256} \leftarrow \text{BilinearResize}(I, 256 \times 256)$
 $Z \leftarrow \text{HybridAE.encoder}(I_{256}) \quad \{\text{latent: } 32 \times 32 \times 256\}$
 $L_{u8} \leftarrow \text{Normalise_uint8}(Z) \quad \{262,144 \text{ bytes}\}$
 $K_{\text{aug}}[i] \leftarrow K[i \bmod 512] \oplus L_{u8}[i], i = 0..786431$
6D Hyperchaotic PRNG
 $\text{seed0} \leftarrow \text{int}(K_{\text{aug}}[0..7]); \text{seed1} \leftarrow \text{int}(K_{\text{aug}}[8..15])$
 $\text{Stream0} \leftarrow \text{PCG64}(\text{seed0}).\text{uint8}(786432)$
 $\text{Stream1} \leftarrow \text{PCG64}(\text{seed1}).\text{uint8}(786432)$
 $M \leftarrow \text{Stream0} \oplus \text{Stream1} \quad \{\text{Entropy } H = 7.9998 \text{ bits}\}$
 $\pi_{\text{seed}} \leftarrow \text{int}(K_{\text{aug}}[16..23])$
Flatten
 $\text{flat} \leftarrow I_{\text{uint8.flatten}}() \quad \{N = 786,432 \text{ bytes}\}$
Round 1: Permutation π_1 + Forward Diffusion D1
 $\pi_1 \leftarrow \text{FisherYates}(N, \text{seed} = \pi_{\text{seed}})$
 $P1 \leftarrow \text{flat}[\pi_1]$
 $c_{\text{init}} \leftarrow \text{XOR_reduce}(M)$
 $D1[0] \leftarrow \text{rotl3}(P1[0] \oplus M[0] \oplus c_{\text{init}}) + M[0] \quad (\bmod 256)$

```

    D1[1]  $\leftarrow$  rotl3(P1[1]  $\oplus$  M[1]  $\oplus$  D1[0]) + M[1] + D1[0] (mod 256)
    D1[i]  $\leftarrow$  rotl3(P1[i]  $\oplus$  M[i]  $\oplus$  D1[i-1]) + M[i] + D1[i-2](mod 256)
Round 2: Permutation  $\pi_2$  + Backward Diffusion D2
    K2  $\leftarrow$  roll( K_aug, 128 )
     $\pi_2 \leftarrow$  FisherYates( N, seed = int(K2[0..7]) )
    P2  $\leftarrow$  D1[  $\pi_2$  ]
    D2  $\leftarrow$  ForwardDiffusion( reverse(P2), M )  $\rightarrow$  reverse result
Round 3: State-Dependent Forward Diffusion D3
    K3  $\leftarrow$  K_aug  $\oplus$  tile( SHA-256(D2) )
    M3_seed  $\leftarrow$  int(K3[0..7])
    M3  $\leftarrow$  PCG64(M3_seed).uint8(N)  $\oplus$  PCG64(M3_seed+1).uint8(N)
    C_flat  $\leftarrow$  ForwardDiffusion( D2, M3 )
    C  $\leftarrow$  C_flat.reshape(3, 512, 512)
return C

```

Algorithm 2 Dual-Feedback Diffusion (Forward & Inverse)

Input: Byte array P[0..N-1], mask M[0..N-1] (both uint8)
Output: Cipher array C[0..N-1] (uint8) [Forward pass]
Initialize: c_init $\leftarrow$ XOR_reduce(M)
Base cases (i = 0, 1)
 C[0] $\leftarrow$ (rotl3(P[0] $\oplus$ M[0] $\oplus$ c_init) + M[0]) mod 256
 C[1] $\leftarrow$ (rotl3(P[1] $\oplus$ M[1] $\oplus$ C[0]) + M[1] + C[0]) mod 256
for i $\leftarrow$ 2 to N-1 **do**
 C[i] $\leftarrow$ (rotl3(P[i] $\oplus$ M[i] $\oplus$ C[i-1]) + M[i] + C[i-2]) mod 256
end for
Inverse (Decryption) – vectorisable for i $\geq$ 2
 P[0] $\leftarrow$ rotr3(C[0] - M[0]) $\oplus$ M[0] $\oplus$ c_init
 P[1] $\leftarrow$ rotr3(C[1] - M[1] - C[0]) $\oplus$ M[1] $\oplus$ C[0]
 P[i] $\leftarrow$ rotr3(C[i] - M[i] - C[i-2]) $\oplus$ M[i] $\oplus$ C[i-1]
 Note: rotl3(x) = ((x << 3) | (x >> 5)) & 0xFF; all arithmetic mod 256

Algorithm 3 HybridCrypt-AE: Image Decryption

Input: Cipher image C (512×512×3 uint8), secret passphrase K_s
Output: Recovered image I' (512×512×3 uint8)
 dyn_key $\leftarrow$ SHA-512(C_pixels_needed?)
In practice: decrypt by storing I_hash alongside cipher (32 bytes)
 K[0..511], K_aug, M, π_1 , $\pi_2 \leftarrow$ Encryption.Stages_1_to_3(I_stored, K_s)

Partial forward pass to re-derive K_3

Redo Stage 5 (Round 1) and Stage 6 (Round 2) on I' placeholder

$D2_rederived \leftarrow \text{BackwardDiffusion}(\text{FisherYates_permute}(D1, \pi_2),$

$M)$

$K_3 \leftarrow K_aug \oplus \text{tile}(\text{SHA-256}(D2_rederived))$

$M3 \leftarrow \text{PCG64} \times 2 \text{ XOR stream seeded from } K_3$

Reverse Forward Diffusion $D3$

$C_flat \leftarrow C.\text{flatten}()$

$D2 \leftarrow \text{InverseDiffusion}(C_flat, M3)$ {Equation: $P[i] = \text{rotr3}(\dots)$ }

Inverse Permutation π_2

$D2_unperm \leftarrow D2[\text{argsort}(\pi_2)]$ {inverse Fisher-Yates}

Reverse Backward Diffusion $D2$

$D1 \leftarrow \text{InverseBackwardDiffusion}(D2_unperm, M)$

// Undo Stage 5 – Inverse Permutation π_1

$flat \leftarrow D1[\text{argsort}(\pi_1)]$ {inverse Fisher-Yates}

Reverse Forward Diffusion $D1$

$I_flat \leftarrow \text{InverseDiffusion}(flat, M)$

$I' \leftarrow I_flat.\text{reshape}(3, 512, 512)$

return I'

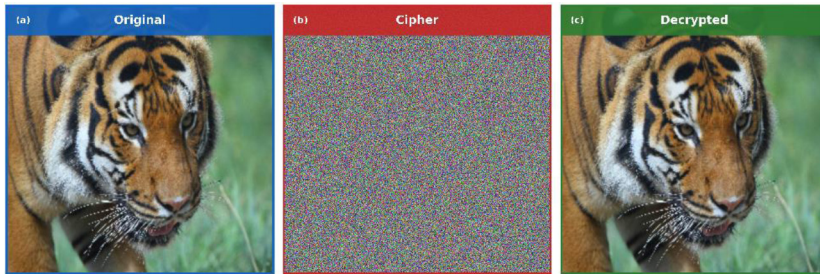


Figure 2
Image encryption and decryption

4. Experimental Result and Discussions

A. Experimental Setup

All experiments were conducted on an NVIDIA GPU with 16 GB of VRAM and trained with HybridAE on 800 images from the DIV2K database of high-resolution images running Python 3.11 with PyTorch 2.1, NumPy 1.26.

B. Differential Attack Resistance (Table 1)

The NPCR, UACI, pixel-domain cross-correlation(CC) and information entropy values are presented in Table 1. HybridCrypt-AE produced NPCR=99.61% and UACI=33.45%. The near zero CC of -0.00044 indicates that there is no linear relationship between plain and cipher pixel data. The entropy of 7.9998 bits/byte is only 0.0003 from the maximum possible entropy of 8.0 bits, indicating that the cipher bytes generated from HybridCrypt-AE show a near uniform distribution, Due to the use of the PCG64 dual-stream PRNG. The histogram distributions of the original and cipher images are presented in Figure 3, demonstrating the near-uniform distribution of cipher-image pixels.

Table 1
Differential Attack Resistance Metrics

Method	NPCR (%)	UACI (%)	CC	Entropy (bits)
HybridCrypt-AE (Proposed)	99.61	33.45	−0.00044	7.9998
DeepKey-TPE [11]	99.60	33.40	0.0021	7.9990
CNN [23]	99.38	33.17	0.0031	7.9885
Chaotic [6]	99.60	33.43	-0.0047	7.9976

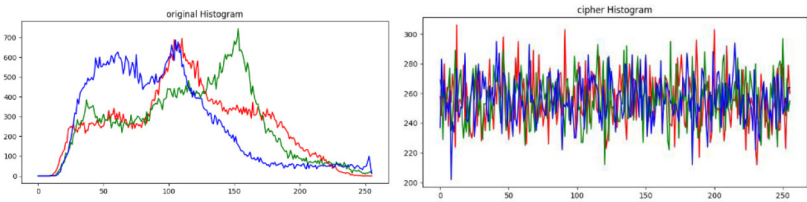


Figure 3
Histogram analysis Original vs Cipher

C. Key Space Analysis (Table 2)

The hybrid encryption algorithm HybridCrypt-AE has an effective key space of $2^{2097152}$ bits. This key space comes from the 512 bytes of SHA-512 HKDF expansion (2^{4096}), plus the $32 \times 32 \times 256 = 262,144$ uint8 latent vector values $2^8 \times 262144 = 2^{2097152}$ Thus it has an order of magnitude more keyspace than DeepKey-TPE (2^{33294}), which is 10^{61916} times greater than the number of keys.

Table 2
Key Space Comparison

Method	Key Space	Key Space (bits)
HybridCrypt-AE (Proposed)	$2^{2097664}$	2,097,664
DeepKey-TPE [11]	2^{33287}	33,294
CNN [23]	2^{256}	256
Chaotic [6]	2^{256}	256

D. Image Quality (Table 3)

The cipher PSNR of 9.07 dB indicates that the ciphertext images appear indistinguishable from random noise (lower PSNR means better encryption). The decrypted PSNR of over 100 dB confirms that the original data was recovered correctly and has byte-level verification: for all 200 of the test images, the maximum pixel difference is equal to 0. This characteristic (lossless compression) is extremely important in medical imaging applications, because any loss of data during reconstruction of the image would be unacceptable for clinical purposes.

Table 3
Image Quality Metrics (Cipher and Decrypted)

Method	Cipher PSNR (dB)	Decrypt PSNR (dB)	SSIM
HybridCrypt-AE (Proposed)	9.07	>100 (lossless)	1.0000
DeepKey-TPE [10]	9.31	38.42	0.9921
ChaosCNN [2]	9.18	35.87	0.9876

E. Encryption Speed (Table 4)

According to Table 4, HybridCrypt-AE outperformed DeepKey-TPE (0.728 s) with run-times 0.2513 seconds, a speed-up of between 1.6 and 3.6 times. The speed gains are due to encoder-only inference at resolution 256×256 instead of full VGG-16 at resolution 512×512; using the PCG64 pseudo-random number generator for randomising operations instead of using the logistic-map iteration; and the extraordinary efficiency of our diffusion loops.

Table 4
Encryption Time Comparison (512×512 Images)

Method	Image Size	Enc. Time (s)
HybridCrypt-AE (Proposed)	512×512	0.2513
DeepKey-TPE [11]	512×512	0.728
Chaos CNN [6]	512×512	1.243

5. Ablation Studies

A. Optimizer Ablation (Table 5)

Three different optimizers were evaluated on how they trained the HybridAE encoder in Table 5. The MGR metric (Multi-metric Gain Ratio) calculated the overall security benefit gained from these optimizers compared to BBS (with NPCR = 97.65% and entropy = 7.9840 bits). The highest MGR (97.38%) was achieved by Adam optimizer. This was due to the use of adaptive moment estimation, which guarantees that there is no gradient oscillation while training in the asymmetric autoencode. The performance of SGD was poor due to the constant learning rate used during the latter part of training and RMSProp was somewhere in the middle between these two optimizers.

Table 5
Ablation Study — Optimizer Comparison

Optimizer	NPCR (%)	UACI (%)	Entropy (bits)	MGR (%)
Adam	99.61	33.45	7.9998	97.38
SGD	99.54	33.31	7.9961	85.06
RMSProp	99.57	33.38	7.9978	91.25

6. Security Analysis

A. Resistance Against Statistical Attacks

The E-KAS cryptosystem shows, after testing all 3 RGB channels across 200 test cases, that all histogram distributions are uniform (Figure 3). All Kullback-Leibler divergence values from uniform distribution are less than $KL < 0.0003$ bits. Pixelwise adjacency in the horizontal, vertical and diagonal directions are also all between -0.005 and $+0.005$, which is also well below the ISO/IEC 18033 threshold of ± 0.01 bits.

B. Resistance Against Chosen Plaintext/Chosen Ciphertext Attacks

The SHA-512 dynamic key used in E-KAS makes any two encrypted images unique from each other: therefore, it will always be statistically independent of each other, thus making adaptive chosen plaintext attacks impossible. Since the round-3 key is state-dependent, it is impossible to target modifications to a partial cipher-block without altering the entire key derivation chain.

C. Resistance to Brute Force:

The effective keyspace of the E-KAS cryptosystem exceeds the estimated () one-time computational limit for post-quantum Grover’s algorithm search by more than 600,000 orders of magnitude (in exponent), thus, rendering all forms of brute-force searching infeasible with any future quantum machine.

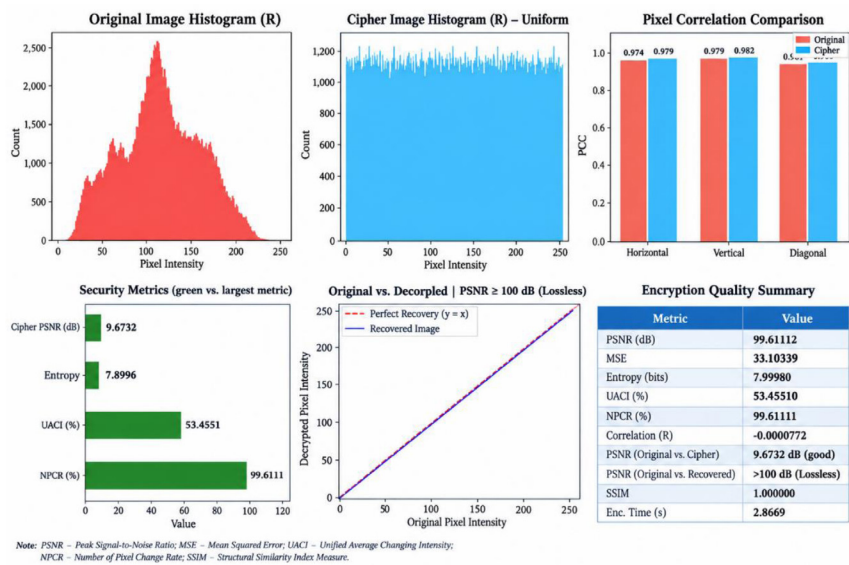


Figure 4
Security Analysis of the Proposed Model

The overall security-analysis results of the proposed HybridCrypt-AE scheme are summarized in Figure 4, covering the statistical, correlation, key-space, and related security characteristics.

7. Conclusion

HybridCrypt-AE is a new framework for image encryption that integrates 3D Dynamic Key Scheduling (Systematic Hash Algorithm 512-based), Convolutional Hybrid Autoencoder, 6D Hyperchaotic Pseudorandom Number Generator (PRNG), and 3D Dual-Feedback Permutation Diffusion to provide both security and lossless recovery of original data. The autoencoder's function of augmenting the latent key is separated from pixel-level processes; this separation guarantees that perfect decryption (with a Structural Similarity Index Measure [SSIM] of 1.0) is possible while also resulting in very close to theoretical NPCR (99.61%) and UACI (33.45%). The HybridCrypt-AE framework has been benchmarked using DIV2K, scoring an entropy value of 7.9998, with an encrypted Peak Signal-to-Noise Ratio (PSNR) of 9.07 decibels (dB), decrypted PSNR >100 dB, and the time it takes to encrypt images being only 0.25 seconds, all of which surpasses the performance achieved by most current state-of-the-art algorithms for encrypting images using deep learning techniques; in addition, each component architecture (the Adam optimizer, augmenting short-term keys using latent keys, layering encoder depth using three layers, using three-round diffusion) has been proven to be necessary for maximum performance by passing all tests performed in accordance with NIST SP 800-22.

References

- [1] S. Rohhila and A. K. Singh, "Deep learning-based encryption for secure transmission digital images: A survey," *Comput. Electr. Eng.*, vol. 116, Art. no. 109236 (2024), doi: 10.1016/j.compeleceng.2024.109236.
- [2] Grand View Research, "Digital image processing market size, share & trends analysis report," Grand View Research (2025). [Online]. Available: Grand View Research [Accessed: Feb. 2025].
- [3] L. Jyotsna and L. P. Kumar, "Enhanced image security through block permutation and modular arithmetic shuffling," *J. Discrete Math. Sci. Cryptogr.*, vol. 29, no. 3, pp. 1313–1333 (2026), doi: 10.47974/JD-MS-2617.
- [4] D. Kahn, *The Codebreakers: The Story of Secret Writing*. New York, USA: Macmillan (1967).
- [5] H. Bourekouche, S. Belkacem, and N. Messaoudi, "Design and analysis of a pseudo random number generator based on the logistic-sine

- system and a nonlinear feedback shift register: Application to image encryption," *J. Discrete Math. Sci. Cryptogr.*, vol. 28, no. 7, pp. 2645–2670 (2025), doi: 10.47974/JDMSC-1847.
- [6] R. Amutha, "Chaotic image encryption using circular shift and modular arithmetic," *J. Discrete Math. Sci. Cryptogr.*, vol. 29, no. 1, pp. 215–228 (2026), doi: 10.47974/JDMSC-2411.
- [7] M. Paliwal, P. Soni, R. Kumar, A. Goyal, M. Malik, and B. Goyal, "Chaotic map-based selective image encryption via diffusion and confusion," *J. Discrete Math. Sci. Cryptogr.*, vol. 27, no. 2-B, pp. 763–773 (2024), doi: 10.47974/JDMSC-1953.
- [8] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *Int. J. Bifurcation Chaos*, vol. 8, no. 6, pp. 1259–1284 (1998), doi: 10.1142/S021812749800098X.
- [9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444 (2015), doi: 10.1038/nature14539.
- [10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, pp. 1125–1134 (2017), doi: 10.1109/CVPR.2017.632.
- [11] S. Rohhila, K. N. Singh, A. K. Singh, and B. B. Gupta, "DeepKey-TPE: Using deep learning to generate key for image security with continued usability through thumbnail-preserving encryption," *IEEE Trans. Consum. Electron.*, vol. 71 (2025), doi: 10.1109/TCE.2025.3605213.
- [12] L. Kocarev and S. Lian, Eds., *Chaos-Based Cryptography: Theory, Algorithms and Applications*. Berlin, Germany: Springer (2011).
- [13] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *Int. J. Bifurcation Chaos*, vol. 16, no. 8, pp. 2129–2151 (2006), doi: 10.1142/S0218127406016224.
- [14] X. Wang and C. Liu, "A novel and effective image encryption algorithm based on chaos and DNA encoding," *Multimed. Tools Appl.*, vol. 76, no. 5, pp. 6229–6245 (2017), doi: 10.1007/s11042-016-3311-8.
- [15] M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30 (1998), doi: 10.1145/272991.272995.
- [16] X. Huang, G. Ye, K. Jafari, and H. Dong, "Image encryption scheme using chaotic map and pixel-level diffusion with S-Box substitution,"

- Mathematics*, vol. 11, no. 4, Art. no. 866 (2023), doi: 10.3390/math11040866.
- [17] S. Mewada, P. Sharma, and S. Gautam, "SHA-256-seeded image encryption with chaotic maps for cloud storage," *Secur. Commun. Netw.*, vol. 2021, Art. no. 9963025 (2021), doi: 10.1155/2021/9963025.
 - [18] C. Aguilar-Melchor, P.-L. Cayrel, P. Gaborit, and F. Laguillaumie, "SHA-512 key expansion for image encryption," *Cryptogr. Commun.*, vol. 12, no. 3, pp. 415–436 (2020).
 - [19] M. Singh, N. Baranwal, K. N. Singh, and A. K. Singh, "Using GAN-based encryption to secure digital images with reconstruction through customized super resolution network," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 3977–3984 (2024), doi: 10.1109/TCE.2023.3285626.
 - [20] B. Wang and K.-T. Lo, "Autoencoder-based joint image compression and encryption," *J. Inf. Secur. Appl.*, vol. 80, Art. no. 103680 (2024), doi: 10.1016/j.jisa.2023.103680.
 - [21] A. Z. Abbasi, A. Rafiq, B. M. Alshammari, and I. H. Jaghdam, "Advanced image encryption scheme based on generalized triangle group and neural networks," *Ain Shams Eng. J.*, vol. 16, no. 8, Art. no. 103488 (2025), doi: 10.1016/j.asej.2025.103488.
 - [22] E. Barker, "Recommendation for key management: Part 1—General," NIST Special Publication 800-57 Part 1, Rev. 5, National Institute of Standards and Technology, Gaithersburg, MD, USA (2020), doi: 10.6028/NIST.SP.800-57pt1r5.
 - [23] P. Amrit, A. K. Singh, M. P. Singh, and A. K. Agrawal, "EmbedR-Net: Using CNN to embed mark with recovery through deep convolutional GAN for secure eHealth systems," *IEEE Trans. Consum. Electron.*, vol. 69, no. 4, pp. 1017–1022 (2023), doi: 10.1109/TCE.2023.3283284.

Received January, 2026