A Review of Artificial Intelligence (AI) Applications in Key Generation for Encryption Algorithms

Zahraa Atheer Ali[1], **Hasan Falah Hasan**[2]

[1]Department of Computer Engineering, College of Engineering, Al-Iraqia University, Iraq
Email: zahraa.a.ali@aliraqia.edu.iq

[2]School of Computer Sciences Universiti Sains Malaysia 11800 USM, Penang, Malaysia
Email: Hfhasan@student.usm.my

Article History

Received: Nov. 09, 2024

Revised: Feb. 16, 2025

Accepted: Feb. 27, 2025

Abstract

In the last few years, Artificial Intelligence (AI) has greatly increased. One specific area has seen significant progress: the use of AI in producing key generation algorithms for cryptography. These algorithms are a critical part of the encryption process. By using AI in this way, the cryptographic community hopes to make the process more secure and efficient. This push has led to several research papers and a wide range of different approaches. Here, we survey the scientific literature in this area and discuss the various strategies that have been taken.

Keywords- AI Applications, key generation, encryption algorithms, string, image.

I. INTRODUCTION

In the past, we have often trusted the generation of secret keys to a subsystem called a Pseudo-Random Number Generator (PRNG). A PRNG has a few properties: it is deterministic, it can be implemented as a relatively small and fast computer program, and it can produce a very, very long pseudo-random sequence. Unlike "true random number generators," which use physical processes like radioactive decay or air turbulence, the PRNG starts from a seed and goes through a series of usually simple and usually very fast computations to produce the (large) sequence of numbers that constitute the "output" of the PRNG [1].

Encryption keys are central to contemporary cryptographic methods [2],[3]. These keys are just data or even images that have been created or selected by a secure process. They are then used by a pair of communicating entities in much the same way as an old-fashioned lock and key. One party has the key, and the other has to have it to decrypt the message.

The strength of the whole system is determined by how to create an encryption key. Two significant things make an encryption key effective. One is the key space, which can be imagined as a big set of possible keys to try. Moreover, we are not too good at imagining large numbers of anything. An encryption key is also good because it is truly random. If it is long enough, if half of the possibilities are as good as or better than the other, and if the appearance of any pattern is a rare and random event, then the encryption system is relatively secure. Moreover, this good randomness does not happen when one kind of sequence is added to another.

Hence, massive key space and unadulterated randomness are necessary for an encryption key to be secure. It's really that simple. There is no rocket science here. These two conditions are the sine qua non of strong key generation in the fortresses and palaces of encryption. And they do not make for just any old encryption key; they make for what is hopefully the kind of encryption key that is so strong only our best technological minds can break it. They make for the kind of encryption key mankind can use to effectively uphold the right to private communication in the public square.

Encryption key generation used to be done via traditional algorithms, mostly associated with number theory. However, traditional number-theoretical algorithms are cumbersome and not very adaptable to modern encryption. As seen in the late 1990s, with the much-maligned but still widely used Data Encryption Standard (DES), somewhat simpler (but no less secure) algorithms can be used instead of traditional ones. Since then, many cryptographic algorithms have not only improved on DES but also dispensed with the use of number theory almost entirely. Simple neural networks have also been used to generate keys because they are much simpler to implement (in software, at least) than DES [4].

II. Literature Review

Numerous studies have looked deeply into the topics of neural networks and key generation in cryptography, trying to discern what has already been done and understand this approach's most appealing aspects. For example, one recent conference paper demonstrated using a tree parity machine (TPM) to generate a secret key over a public channel. The study was conducted to provide a key generation with a high degree of security in [1], [3], [4], [5], [6]. Also, in [7], a Vector-Valued Tree Parity Machine (VVTPM), can be applied to generate a flexible length of secret key. Additionally, [8] is combined with a Genetic Algorithm for the key generation process.

In a study by [9], a Bidirectional Associative Memory Neural Network (BAMNN) has been trained using randomized data for key generation. In [10], Extreme Learning Machine (ELM) based Sub-key generation for cryptographic systems is proposed and designed to generate a key that offers a high degree of sensitivity and security for use in cryptographic algorithms. It also guarantees elevated sensitivity and strong security regarding the key space. In [11], a time-invariant cryptographic key generation mechanism based on electroencephalogram (EEG) signals. Also, [12] generated the private key, and training is carried out with a single-layer perceptron.

While others have employed deep learning networks, the generative adversarial network (GAN) is adopted as the learning network to generate the private key in [13], [14], [15], [16]. Additionally, [17] introduced the Least Squares Generative Adversarial Networks(LSGAN)into random number generation. Also, proposed Deep Convolutional Generative Adversarial Networks (DCGANs) to create a random sequence with better randomness and complexity as a key stream. In the study, [18] Deep Convolutional Neural Network (DCNN) and Genetic Algorithm (GA) are used to generate keys. Similarly, in [19], a public key is generated using Convolutional Neural Networks (CNNs). In the study, In [20], [2],[21],[22], an autoencoder is used to encrypt and decrypt digital information. The key lies in the internal structure and general architecture of the network.

Some researchers have combined the Chaotic system with ANN to generate the key. In [23], based on a combination of the Logistic map (LM) and the perceptron neural network (PNN)in a key generation to advantage from their characteristics. Such as a more extensive range of parameters, fewer periodic windows in bifurcations, and a larger key space. Also, [24] used a chaotic pulse-coupled neural network (PCNN) in key generation. In [25], an encryption method based on logistic chaotic systems and a deep autoencoder was proposed.

However, another group’s research has sought to determine if the digital encryption algorithms used to produce our secure communications can be enhanced by using artificial neural networks (ANN). The researchers, based at the Future University in Egypt, have chosen a specific class of ANN, known as the Fractional-Order Memristive Hopfield Neural Networks (6D-FMHNN), and have utilized this for generating secure keys to be used in the encryption of both text and still images [5].

Finally, the proposed 3D CUBE algorithm is designed to generate a secret key for deep learning with neural networks. Its goal is to minimize the shared information between the sending and receiving parties. This, in turn, reduces the potential for data leakage and the associated damage [6]. An innovative attempt to deal with the challenges of key exchange and management in contemporary encryption systems is what the 3D CUBE algorithm is all about. It uses the dynamic and complex states involved in a Rubik's cube to change key components in a way comprehensible to the artificial intelligence (AI) principles used in encryption [7]. Figure (1) shows AI Applications in key generation.

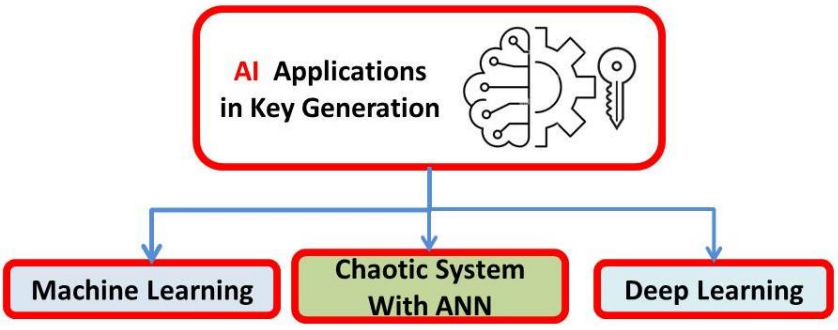

Figure 1: AI Applications in Key Generation

III. STUDIES AND FINDINGS

Table 1 provides a summarized critical evaluation of prior research efforts. The analysis in this table concentrates on seven key aspects: Methodology, initial seed, key space, key sensitivity, encrypted data, Frequency (Monobit) Test and application, which are explained below.

A. Methodology: Intelligence technology is used to build the key.

B. Initial seed: In encryption, "initial seed" refers to a random value or sequence used to generate encryption keys or random values. The initial seed achieves greater diversity and security in encryption operations. The role of the initial seed in encryption protocols is to ensure the generation of unique, random keys for each encryption session. Using a solid and random initial seed makes it difficult for attackers to predict or guess the encryption keys.

C. Keyspace: Analyzing the key spaces, the key space must be sufficiently large to withstand brute-force attacks. Key spaces larger than 2180 have a certain degree of protection against brute-force attacks [29]. Key space can be computed by multiplying the key parameter as in Eq. 1 [30]:

$$Sp = \prod_{i=1}^{n} spi \quad (1)$$

Where spi is the i^{th} secret parameter.

D. Key analysis metric: An effective encryption design should be sensitive to each secret key to resist brute-force attacks. In other words, a minor modification to each encryption key will result in severe image distortions when decrypted [30]. Mean square error (MSE) and peak signal-to-noise ratio (PSNR) is used to quantify this metric as in Eq.2 and Eq.3, respectively [30]:

$$\mathrm{MSC} = \frac{1}{\mathrm{W}\times\mathrm{H}} \sum_{\mathrm{i}=1}^{\mathrm{W}} \sum_{\mathrm{j}=1}^{\mathrm{H}} [\mathrm{D_e}(\mathrm{i},\mathrm{j}) - \mathrm{p}(\mathrm{i},\mathrm{j})]^2 \quad (2)$$

$$\mathrm{PSNR} = 10 \log_{10} \frac{[255]^2}{\mathrm{MSC}} \quad (3)$$

Where *De* is a decrypted image with incorrect secret keys that differ slightly from the correct ones, and *P* is the plain image.

E. Encrypted Data: This is the type of data input into the AI network.

F. Frequency (Monobit) Test: This test aims to determine whether the number of ones and zeros in a sequence is approximately the same as expected for a truly random sequence.

$$S_{abs} = \frac{|S_n|}{\sqrt{n}} \quad (4)$$

Where *n* the length of the bit stream, *Sn* Add the entire sequence

G. Application: Shows the type of application to be encrypted.

Table 1. Artificial Neural Network Applications

No	Research Name	Methodology	Initial Seed	Key Space	Key Sensitivity	Encrypted Data	Frequency (Monobit)	Application
1	J. Sahana and S.Bisalapur [1]	Tree Parity Machine (TPM)	Random weight vector	-	-	Text	X	Cryptography System
2	T. Dong et al.[8]	The complex-valued tree parity machine network (CVTPM)	Complex value	-	-	Text	X	Cryptography System
3	J. Dey[9]	Tree Parity Machine (TPM)	Initialize random weight vectors and random inputs	1024 bits	-	The patient's data	X	Symmetric key encryption system to encrypt data in

								telemedicine
4	C. BISWAS et al. [5]	Tree Parity Machine TPM)	Random weight	128-bit	Verified	Text	X	symmetric Cryptography System
5	J. Dey and A. Bhowmik[10]	Neural Tree Parity Machine (TPM)	Random weight vector	$[2^{128}, 3^{192}]$	-	The patient's data	X	The patient's datain Telecare Health System
6	S. Jeong et al.[11]	Vector-Valued Tree Parity Machine (VVTPM)	Random integer	-	-	Text	X	Asymmetric public key system
7	S. Jhajharia et al. [12]	Tree Parity Machine (TPM)	Genetic Algorithm (GA) with Pseudo Random Number Generator (PRNG)	2^{192}	-	Text	X	Public Key Cryptography (PKC)
8	M. Indrasena et al.[13]	A Bidirectional Associative Memory Neural Network (BAMNN)	Random binary	-	-	String	X	Cryptography System
9	H. Abdulzahra et al.[14]	Extreme learning Machine (ELM)	Weights are generated randomly by pseudo-random number generation	2^{120}	Verified	String	X	For Cryptography System
10	M. Indrasena et al.[15]	EEG	Employed Discrete Wavelet Transform and autoencoders to extract the biometric features from the EEG signals	2^{196}	-	Personal data	X	Cryptography System
11	Y. Fatma et al.[16]	Perceptron	Image	-	-	Text	X	Asymmetric key encryption system (cryptography)
12	Y. Ding et al.[17]	DeepKeyGen, the generative adversarial network (GAN)	Image	$(2^8)^{196\,608}$	Verified	Image	X	Stream cipher to safeguard medical images
13	Y. Ding et al.[18]	he cycle-generative adversarial network (Cycle-GAN)	Randomly initialized	$(2^{32})^{2757936}$	Verified	Medical image	X	Image Encryption and Decryption Network for Internet of Medical Things
14	X. Chai et al.[19]	CycleGan GAN	Binary string the initial parameters of network are randomly generated	256-bit	Verified	Image	X	Thumbnail preserving encryption
15	K. Panwar et al.[20]	A cycle-consistent generative adversarial network (CycleGAN)	The parameters are randomly initialized	$2^{Parameters_* 32}$	Verified	Medical image	X	Cryptography of medical images
16	Z. Man et al.[21]	Six chaotic systems+ Least Squares Generative Adversarial Networks(LSGAN)	Random numbers	2^{192}	-	Image	X	Cryptography
17	S. Venkatesan et al.[22]	Deep Convolutional Neural Network (DCNN)	Weights of DCNN	-	-	Massage	X	Asymmetric key encryption system to encrypt Secure Data Communication Over Wireless Network
18	E. A. Hagras et al.[23]	Elliptic Curve Based on Deep Convolutional Neural Network (APK-EC-DCNN)	-	2^{924}	Verified	Image	X	Public Key Cryptography
19	F. Quinga et al.[24]	Autoencoder	-	-	-	String	X	Stream cipher cryptosystems
20	L. Zhinin And O. Chang [4]	autoencoder	Pseudo random sequence, a seed generator based on the secret password	10^{761} and 10^{324}	-	Text	X	Symmetric Key Cryptography System
21	S. Ramesh and T. van	Deep Neural Network and	Image	-	-	Image	X	Image security with an aid of deep neural

	[25]	Chaotic Map						network
22	M. Dridi et al.[26]	Chaotic map and chaotic–neural network	Randomly initialized	2^{325}	Verified	Medical image	X	Block ciphering algorithm Cryptography of medical images
23	J. Ye et al.[27]	Pulse-Coupled Neural Network (PCNN)+ chaotic system	-	2^{430}	Verified	Image	X	Symmetric Key Cryptography
24	Y. Sang et al.[28]	logistic chaotic systems +deep autoencoder	The precision of floating-point data	2^{99}	Verified	Image	X	Cryptography System
25	F. Yu et al.[29]	6D fractional-order memristive Hopfield neural network (6D-FMHNN)	Pseudo random sequence	10^{6}	Verified	Image	X	Cryptography System
26	J. JIN and K.KIM[5]	Deep Cube algorithm	-	256-bit	Verified	Image	X	Symmetric key encryption system (cryptography)
27	J. Jin and S.Park[6]	Deep Cube algorithm	-	2^{256}	-	Image	X	Symmetric key encryption system (cryptography)

A range of studies is concentrated on generating keys for encryption systems. They are quite diverse and employ various methods in their work. Among them are Tree Parity Machines (TPMs), Generative Adversarial Networks (GAN), and some not-so-simple neural network architectures. What should catch the eye, though—and what to stress here—is this: There is a complete lack of uniformity in the metrics used to judge the methods across the different studies.

Several researchers, including those working with Tree Parity Machines and some neural network models, did not carry out this test, which raises concerns over the robustness of their key generation processes. Also, a few studies did not manage to achieve sensitivity testing or provide sufficient key size specifications—both of which are vital to understanding the relationship between input and output. For example, some methods reported key sizes anywhere from 128 to 2,192 bits, while others did not even mention key size, leaving us with serious holes in our understanding of their methods' security.

The failure to apply consistent, rigorous evaluation methods—such as the monobit frequency test and the sensitivity analysis—overlooks a key factor in establishing the reliable security of the cryptographic keys these systems generate. This is an area that future research should not neglect. Instead, they should carry on the necessary credibility checks so that their proposed encryption methodologies can be considered effective.

This section presents case studies and experimental results that demonstrate the effectiveness of the DeepKeyGen system in generating secure encryption keys for medical image encryption [17]. By providing concrete examples and data, we aim to substantiate the advantages of using Generative Adversarial Networks (GANs) in this context.

Case Study 1: Encryption of Chest X-rays

The DeepKeyGen system was applied to a dataset of chest X-ray images. The model was trained using unpaired data, allowing private keys to be generated from initial images without needing labeled pairs. The generated keys were evaluated for their security and efficiency.

Results:

- **Key Space and Randomness:** The generated keys exhibited a key space of $(2^8)^{196\,608}$, indicating many possible keys. Statistical tests confirmed the high randomness of the keys, making them resistant to brute-force attacks.
- **Encryption Time:** The average time taken to encrypt a chest X-ray image was approximately 0.5 seconds, demonstrating the system's efficiency in processing medical images.

Case Study 2: Sensitivity Analysis

To assess the sensitivity of the generated keys, we conducted experiments where small perturbations were introduced to the initial images. The results showed that even minor changes in the input image led to significantly different keys, akin to the behavior of a one-time pad.

Results:

- **Key Variability:** The sensitivity analysis revealed that the keys generated from perturbed images differed by an average of 95% from the original keys, underscoring the system's robustness against small changes in input.
- **Security Implications:** This high sensitivity enhances security, ensuring that even slight alterations in the initial image result in entirely different encryption keys.

Practical Applications

The DeepKeyGen system can be implemented in clinical settings to safeguard patient data. Hospitals can use this technology to encrypt medical images, ensuring compliance with privacy regulations and enhancing the security of sensitive information.

IV. DISCUSSION

The discussion surrounding the application of artificial intelligence in key generation for cryptographic algorithms has highlighted the potential benefits and inherent challenges. I came up with two aspects by using AI to generate the key. The first is to generate the key through machine learning or genetic algorithms. These methods can produce a key, but this key is of a fixed length and eventually becomes repeatable. Although it provides randomness, it only processes short-length data with sensitive and intermediate characteristics. However, I do not recommend it for more extensive data such as images.

Conversely, deep learning techniques, incredibly generative adversarial networks (GANs), can generate adaptive keys capable of encrypting images. Therefore, I recommend using these methods. Moreover, the efficiency of the keys generated by neural networks depends not only on the network structure but also on the network's initial seed and trained weights. This is the rationale behind researchers using a chaos system, which results in increased adaptability and enhanced resistance to attacks on generated keys.

However, it is crucial to note that many researchers have not adequately addressed the randomness of their key generation processes. For instance, the reliance on fixed or insufficiently random initial seeds often leads to limited key space and sensitivity issues.

Future research and development efforts should concentrate on advancements in key generation techniques. Leveraging deep neural network architectures and robust key management protocols represents a promising avenue to address emerging cybersecurity threats. Using deep learning, scientists can create key generation systems that are more random and more resistant to attacks. Figure (2) shows the general method of key generation by AI.

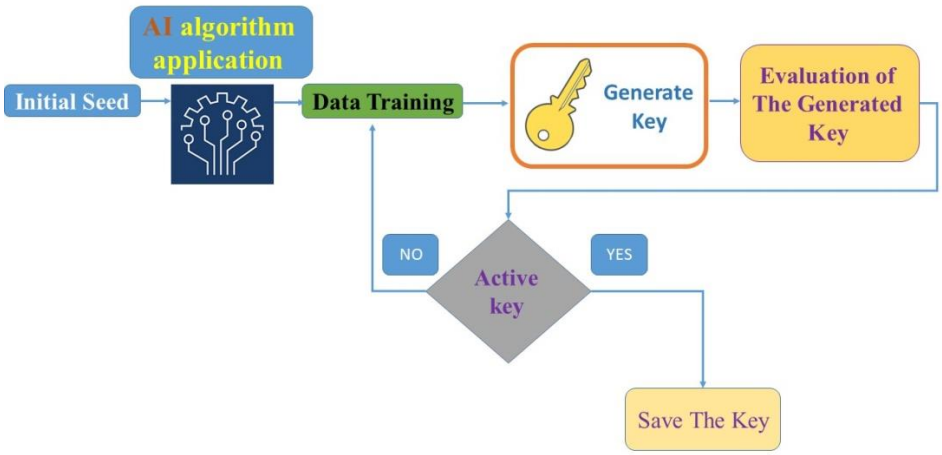

Figure 1: General method of key generation by AI

A general process for generating keys through artificial intelligence can be seen in Figure 2. The first step is to have an "Initial Seed" that will serve as the input to an "AI algorithm application". The algorithm performs the function of a key. It is the most crucial step in the whole key generation process. If you have a good and sound key, then whatever you do with the rest of the process will give you a reliable and well-functioning system.

The produced "Active key" is subsequently assessed to gauge its quality and security features. This assessment phase looks at the key to see if it fulfills the assigned tasks, like being random and, for some variations, being "sensitive" instead of "robust." It also looks at the key space and the number of different keys the model can produce to see if it is large enough. If the "Active key" makes it through this assessment and is pronounced fit for use, it's saved to see action later in time. If the key does not pass, we go back to the top and try again.

The generation of secure keys is such a key aspect of the infrastructure that it necessitates an approach like the one depicted in the diagram—an iterative one that allows every part of every key to improve what's essentially a quality assurance process continuously. Also, as the Figure makes clear, it's not just about the keys themselves; it's also about testing how well they work under a variety of conditions. Particularly cryptography-relevant ones.

This chart generally encapsulates a systematic framework for harnessing artificial intelligence techniques—like machine learning or neural networks—to automate the production of safe, secure cryptographic keys. This is depicted in the next section of the diagram. The framework is set up to allow for both data-driven, training-based enhancement of key generation and rigorous evaluation of the keys' secure properties. Key management, which is crucial for maintaining the security of various applications, stands to benefit from this AI-based approach.

V. CONCLUSION

In this work, we have investigated how artificial intelligence (AI) can be applied to generate encryption keys. We have considered three specific AI methods: The Tree Parity Machine, neural networks, and neural networks that work in conjunction with chaos. We have taken a close, critical look at each AI method, examining the "what, how, and why" of its appearance in generating cipher keys.

The research shows that the Tree Parity Machine (TPM) method gives keys of unvarying length that, at some point, become predictable, rendering this method barely suitable for safeguarding personal data. Nevertheless, suppose we must use the TPM for encryption. In that case, it is best not to wish for too much security and to encrypt small things that can be kept to a consistent length, like names or other personal identifiers that we might want to protect from prying eyes. A graphing calculator's memory can store a TPM, and a calculator can be hacked, but at least we can count on it to give out the same numbers.

To ensure that the creation technique is resilient against cybersecurity threats, it is essential to carefully control and monitor the key management processes and initial seed values.

The researchers suggest that future work should concentrate on using artificial intelligence to make the adaptability of key generation systems even better against new threats. There is a clear need to push this beneficial AI-adjacent technology to the next level. If this sounds like dangerous science fiction, it is not. Cryptographic concepts and protocols already exist that are not too far away from our "simpler" AI-aided key generation system of tomorrow.

REFERENCES

[1] G. Eneration, “D ESIGN O F A N EFFICIENT N EURAL K EY,” vol. 2, no. 1, pp. 60–69, 2011.

[2] A. J. Kadhim and T. S. Atia, “Strengthening Security and Confidentiality in E-Health Systems through Quantum Encryption of Healthcare,” *Al-Iraqia J. Sci. Eng. Res.*, vol. 2, no. 3, 2023, doi: 10.58564/ijser.2.3.2023.83.

[3] M. H. Abood and S. W. Abdulmajeed, “High Security Image Cryptographic Algorithm Using Chaotic Encryption Algorithm with Hash-LSB Steganography,” *Al-Iraqia J. Sci. Eng. Res.*, vol. 1, no. 2, pp. 65–74, 2023, doi: 10.33193/ijser.2.1.2022.53.

[4] F. Quinga-Socasi, L. Zhinin-Vera, and O. Chang, “A Deep Learning Approach for Symmetric-Key Cryptography System,” *Adv. Intell. Syst. Comput.*, vol. 1288, no. October, pp. 539–552, 2021, doi: 10.1007/978-3-030-63128-4_41.

[5] F. Yu *et al.*, “A 6D Fractional-Order Memristive Hopfield Neural Network and its Application in Image Encryption,” *Front. Phys.*, vol. 10, no. March, pp. 1–14, 2022, doi: 10.3389/fphy.2022.847385.

[6] J. Jin and K. Kim, “3D CUBE Algorithm for the Key Generation Method: Applying Deep Neural Network Learning-Based,” *IEEE Access*, vol. 8, pp. 33689–33702, 2020, doi: 10.1109/ACCESS.2020.2973695.

[7] J. Jin and S. S. Park, “Key generation and management method using AI generated Rubik’s cube states,” *Int. Conf. Inf. Netw.*, vol. 2023-Janua, no. 2020, pp. 719–724, 2023, doi: 10.1109/ICOIN56518.2023.10048916.

[8] T. Dong and T. Huang, “Neural Cryptography Based on Complex-Valued Neural Network,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, no. 11, pp. 4999–5004, 2020, doi: 10.1109/TNNLS.2019.2955165.

[9] J. Dey, “State-of-the-art session key generation on priority-based adaptive neural machine (PANM) in telemedicine,” *Neural Comput. Appl.*, vol. 35, no. 13, pp. 9517–9533, 2023, doi: 10.1007/s00521-022-08169-2.

[10] J. Dey and A. Bhowmik, "Concurring of Neural Machines for Robust Session Key Generation and Validation in Telecare Health System During COVID-19 Pandemic," *Wirel. Pers. Commun.*, vol. 130, no. 3, pp. 1885–1904, 2023, doi: 10.1007/s11277-023-10362-y.

[11] S. Jeong, C. Park, D. Hong, C. Seo, and N. Jho, "Neural Cryptography Based on Generalized Tree Parity Machine for Real-Life Systems," *Secur. Commun. Networks*, vol. 2021, 2021, doi: 10.1155/2021/6680782.

[12] S. Jhajharia, S. Mishra, and S. Bali, "Public key cryptography using neural networks and genetic algorithms," *2013 6th Int. Conf. Contemp. Comput. IC3 2013*, pp. 137–142, 2013, doi: 10.1109/IC3.2013.6612177.

[13] M. Indrasena Reddy, A. P. Siva Kumar, and K. Subba Reddy, "A secured cryptographic system based on DNA and a hybrid key generation approach," *BioSystems*, vol. 197, no. September 2019, p. 104207, 2020, doi: 10.1016/j.biosystems.2020.104207.

[14] H. A. Atee, R. Ahmad, and N. M. Noor, "Extreme Learning Machine Based Sub-key Generation for Cryptography system," no. January 2015, 2018.

[15] L. Hernández-Álvarez *et al.*, "KeyEncoder: A secure and usable EEG-based cryptographic key generation mechanism," *Pattern Recognit. Lett.*, vol. 173, no. June, pp. 1–9, 2023, doi: 10.1016/j.patrec.2023.07.008.

[16] Y. Fatma, R. Wardoyo, and H. Mukhtar, "An Approach to Cryptography Based on Neural Network Using Image for Public Key Generation," *AIP Conf. Proc.*, vol. 2601, 2023, doi: 10.1063/5.0130464.

[17] Y. Ding, F. Tan, Z. Qin, M. Cao, K. K. R. Choo, and Z. Qin, "DeepKeyGen: A Deep Learning-Based Stream Cipher Generator for Medical Image Encryption and Decryption," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 9, pp. 4915–4929, 2022, doi: 10.1109/TNNLS.2021.3062754.

[18] Y. Ding *et al.*, "DeepEDN: A Deep-Learning-Based Image Encryption and Decryption Network for Internet of Medical Things," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1504–1518, 2021, doi: 10.1109/JIOT.2020.3012452.

[19] X. Chai, Y. Wang, X. Chen, Z. Gan, and Y. Zhang, "TPE-GAN: Thumbnail Preserving Encryption Based on GAN With Key," *IEEE Signal Process. Lett.*, vol. 29, pp. 972–976, 2022, doi: 10.1109/LSP.2022.3163685.

[20] K. Panwar, A. Singh, S. Kukreja, K. K. Singh, N. Shakhovska, and A. Boichuk, "Encipher GAN: An End-to-End Color Image Encryption System Using a Deep Generative Model," *Systems*, vol. 11, no. 1, pp. 1–15, 2023, doi: 10.3390/systems11010036.

[21] Z. Man *et al.*, "A novel image encryption algorithm based on least squares generative adversarial network random number generator," *Multimed. Tools Appl.*, vol. 80, no. 18, pp. 27445–27469, 2021, doi: 10.1007/s11042-021-10979-w.

[22] S. Venkatesan, M. Ramakrishnan, and M. Archana, "Unique and Random Key Generation Using Deep Convolutional Neural Network and Genetic Algorithm for Secure Data Communication Over Wireless Network," *Artif. Intell. Sustain. Appl.*, pp. 249–263, 2023, doi: 10.1002/9781394175253.ch15.

[23] E. A. A. Hagras, S. Aldosary, H. Khaled, and T. M. Hassan, "Authenticated Public Key Elliptic Curve Based on Deep Convolutional Neural Network for Cybersecurity Image Encryption Application," *Sensors*, vol. 23, no. 14, 2023, doi: 10.3390/s23146589.

[24] F. Q. Socasi, R. Velastegui, L. Zhinin-Vera, R. Valencia-Ramos, F. Ortega-Zamorano, and O. Chang, "Digital cryptography implementation using neurocomputational model with autoencoder architecture," *ICAART 2020 - Proc. 12th Int. Conf. Agents Artif. Intell.*, vol. 2, no. Icaart 2020, pp. 865–872, 2020, doi: 10.5220/0009154908650872.

[25] R. Valencia-Ramos, L. Zhinin-Vera, G. E. Pilliza, and O. Chang, "An Asymetric-key Cryptosystem based on Artificial Neural Network," *Int. Conf. Agents Artif. Intell.*, vol. 3, no. Icaart, pp. 540–547, 2022, doi: 10.5220/0010857700003116.

[26] S. R. Maniyath and T. V, "An efficient image encryption using deep neural network and chaotic map," *Microprocess. Microsyst.*, vol. 77, p. 103134, 2020, doi: 10.1016/j.micpro.2020.103134.

[27] M. Dridi, M. A. Hajjaji, B. Bouallegue, and A. Mtibaa, "Cryptography of medical images based on a combination between chaotic and neural network", doi: 10.1049/iet-ipr.2015.0868.

[28] J. Ye, X. Deng, A. Zhang, and H. Yu, "A Novel Image Encryption Algorithm Based on Improved Arnold Transform and Chaotic Pulse-Coupled Neural Network," *Entropy*, vol. 24, no. 8, 2022, doi: 10.3390/e24081103.

[29] Y. Sang, J. Sang, and M. S. Alam, "Image encryption based on logistic chaotic systems and deep autoencoder," *Pattern Recognit. Lett.*, vol. 153, pp. 59–66, 2022, doi: 10.1016/j.patrec.2021.11.025.