Scientific and Practical Cyber Security Journal (SPCSJ) 8(1): 39 – 49 ISSN 2587-4667 Scientific Cyber Security Association (SCSA)

NEURO-CRYPTOGRAPHIC HYBRID SYSTEMS: UNLEASHING THE POWER OF NEURAL NETWORKS FOR CRYPTANALYSIS AND ENCRYPTION

Luka Baklaga[1]

[1]Research and Development Department, Researcher, Business and Technology University, Georgia

ABSTRACT: Neural cryptography is a field that blends neural networks and cryptographic algorithms. This approach offers promising solutions to address security concerns with traditional cryptographic methods. This article explores the transformative potential of hybrid neurocryptographic systems through a comprehensive analysis. The methodology combines independent analysis, theoretical investigation, and quantitative testing. With the rise of digital data exchange, storage, and transmission, information security is more crucial than ever. Cryptographic algorithms can protect data, verify identities, and reduce various attacks. The study demonstrates how hybrid systems using neural networks and cryptography could revolutionize cryptography processes. Cryptanalysis methods have advanced due to increased computing power, becoming effective in information security. Traditional cryptographic protocols employ well-known ciphertexts and number theory techniques. This study proposes a mathematical cryptography model utilizing deep learning (DL), specifically neural networks. The model aims to protect plaintext through rapid distribution of neural network layers. The process begins by developing a new cryptography module emphasizing the use of neural networks for encryption and cryptanalysis. It implements a novel approach to secure authentication by dynamically converting biometric data into encryption keys using neural networks, instead of standard key storage techniques. Innovative security protocols offer lightweight block ciphers such as S-DES, which combine number theory and neural network architecture in their experimental endeavors. Using each neural cryptanalysis result as a key bit, the work carefully examines how key differences impact S-DES. In neural cryptography, the same input vector is received by both communicating networks, which then use it to generate and train an output bit. A special phenomenon can be observed in the dynamics of two networks and their weight vectors: they synchronize to a state in which their time-dependent weights are the same. Theoretical work explores the complex relationships between neural network architectures and cryptographic techniques, focusing on the creation of sophisticated encryption algorithms, complex network decoding, and the optimization of internal security protocols. The goals place a strong conceptual focus on promoting innovation, improving safety and maximizing effectiveness. This is a critical first step toward integrating neural networks into the framework of cryptographic advances in protocol system security. The next research study aims to develop and apply efficient formulas, tools and algorithms to meet the needs of quantum-based cryptography. For example, by combining quantum mechanics and deep learning, completely secure quantum neural network cryptography can be created.

KEYWORDS*:* deep learning, neural networks, cryptography, number theory, neurocryptography, Gated Recurrent Units

1. INTRODUCTION

The swift expansion of digital information sharing, storage, and transfer has underscored the criticality of data security measures. Traditional cryptographic techniques, while effective, face increasing vulnerabilities due to advancements in computing power and cryptanalysis methods. This study delves into the exceptional potential of hybrid neurocryptographic systems, which seamlessly integrate neural networks with cryptographic algorithms, to catalyze a transformative revolution in cryptographic processes and protocols. Cryptanalysis of block ciphers has consistently garnered a lot of attention, and many new cryptanalytic approaches have appeared recently (Uludag et al. 2004). Cryptoanalysis based on algebraic structural algorithms can be classified into directed modules of different segments (Biehl and Caticha 2001), such as differential cryptanalysis, linear cryptanalysis, differential-linear cryptanalysis, meet-in-the-middle attack, and related-key attack (Biham and Shamir 1993). One of the most important aspects of information technology development is information security. Developing and implementing new security measures for information systems is crucial nowadays. Modern cryptography has used strong algorithms to improve information security. On the other side, increasingly sophisticated attacks have appeared. These attacks take advantage of enhanced computing capabilities and methodologies based on artificial intelligent tool so called machine learning. The efficacy of artificial neural networks (ANNs) and deep learning methods in addressing intricate classification issues has motivated scholars and technological enterprises to utilize these approaches for cryptanalysis and cryptography within the realm of number theory (Hertz, Krogh, and Palmer 1991). In recent years, there has been a surge of interest in neural networks as a potential computational model for comprehending the functioning of the human brain. Illustrative instances provide valuable learning material for neural networks. Extensive research has been conducted on this concept utilizing statistical mechanics models and methodologies (Yamashita et al. 2018). Dynamic neural networks are a common occurrence employed within cryptographic systems. Limitations in the fundamental cryptography process prompted the development of cryptographic systems with shorter keys, also known as secret key systems (Danziger and Henriques 2014). The security of a cryptographic system is contingent upon the confidentiality of the key. Neurocryptography examines using neural networks and probabilistic algorithms for encryption and cryptanalysis. It tackles public key cryptography, key distribution, hashing, and pseudo-random number generation. Neural networks excel at parallel processing, equipping them to handle varied future tasks. However, their complex setup often limits practical use. These networks demonstrate skill in recognizing intricate patterns and mappings, making them adept at addressing cryptography's computational challenges. Combining neural networks with cryptography offers enhanced security measures. This study aims to develop an innovative cryptographic model using neural networks for encryption and code-breaking tasks. The proposed approach converts biometric data into dynamic encryption keys through neural networks, providing secure authentication without storing conventional keys. Additionally, the research explores integrating neural networks with lightweight block ciphers (Gomez et al. 2018), merging number theory principles with neural network architectures to create cutting-edge security protocols. Furthermore, by analyzing key differences' impact on S-DES encryption, the study examines the intricate relationships between neural network outputs and cryptographic key generation. In this paper, we use artificial neural networks to generate new directions for cryptographic probability protocols. The networks are trained using generated data that identifies protocol weaknesses as well as the encryption key, which is unique to each experimental portion. This scientific article intends to develop a cryptographic algorithm using neural network modular systems and analyze a biometric sample to create a cryptographic key. Additionally, it aims to develop a Neurocryptographic Sequence-to-Sequence autoencoder model software using a mathematical approach and simulation in Python. Finally, it aims to test and optimize the use of the developed algorithm. We offer the technique and results in accordance with our study goal: in Section 3, we exhibit the methodology, research design, and numerous experiments related to the establishment of neural-based cryptography and its cryptoanalysis inside mathematical modeling, as well as the proposed outcomes. Section 4 presents a summary of the experimental findings, the conclusion of the research, and its future path.

This study investigates the comprehensive capabilities of combined neurocryptographic systems through a methodical approach involving theoretical analysis, quantitative evaluations, and digital simulations. The outcomes reveal possibilities for pioneering encryption algorithms, sophisticated network decryption methods, and optimized security protocols. These advancements foster innovation, bolster security measures, and maximize efficiency in cryptanalysis and encryption pursuits.

2. OBJECTIVES

This study aims to explore novel cryptographic frameworks and procedures utilizing deep learning techniques like neural networks. Its objectives encompass: developing methods for enhanced threat detection and robust key security; discerning the transformative capabilities of hybrid neurocryptographic systems in reshaping autonomous environments' cryptographic processes. Key areas of focus include fostering innovation, fortifying security measures, and optimizing efficiency in pursuit of cryptanalytic and encryption objectives. The research strategically integrates diverse perspectives to drive advancements in this domain.

3. RESEARCH METHODOLOGY

3.1. Research design

This research uses a thorough and coherent methodology that combines independent analysis, theoretical investigation and quantitative testing to study hybrid neurocryptographic systems. To develop a new cryptographic module, the project focuses on sharing the capabilities of neural networks for encryption and cryptanalysis. To create a secure authentication system, the project will use neural networks to convert a biometric sample into an encryption key. Instead of storing and using cryptographic keys later, this solution uses neural networks to generate and authenticate them. Experiments were conducted with lightweight block ciphers such as S-DES, where the block size was represented by x-points and the key length was represented by y-points. By applying number theory and neural network architecture, a state-of-the-art security protocol should be developed. The impact of key differences on ciphers was also investigated, as each output in neural cryptanalysis represents a key bit. The study begins with a comprehensive literature review that uses a meta-analytic approach to assess the body of knowledge on the integration of neural networks and cryptographic systems. Theoretical research deals with the complex interplay between neural network architectures and cryptography methods. Sophisticated encryption algorithms, complex network decoding and optimization of security protocols are priorities in the context of security systems.

This digital platform facilitates the creation, testing and validation of the proposed hybrid neurocryptographic systems using the Jupyter notebook and appropriate Python modules. By combining ideas from neural network theory and cryptography techniques, neural network design is scientifically defined. To achieve a quantitative combination, some basic properties from the theory of random walks in limited domains were applied. A combination analysis was performed to determine how different parameter choices affect the convergence rate. The smooth transition between theoretical understanding and digital experiments is highlighted by this research design. While digital experiments confirm the feasibility and effectiveness of the proposed hybrid neurocryptographic system, the theoretical foundations guide the development of the neural network-based cryptographic architecture.

3.2. Research experiment - Neural Network-based Encryption using Modular Arithmetic

In this study, we aim to develop a cryptographic algorithm utilizing neural networks that will integrate principles of modular arithmetic derived from number theory. The neural network will be tasked with generating encryption keys based on the input plaintext, and the encryption process will involve modular arithmetic operations. The primary framework will be described as a Neural Network-based Encryption using Modular Arithmetic. The neural network serves as a tool for generating and authenticating keys, while modular arithmetic operations play a role in the encryption and decryption processes. The experimental model commences with the data preprocessing stage, where the input plaintext message is designated as *Mp*. The initial stage involves transforming the variable *Mp* into a numerical format suitable for input into the neural network. A frequently employed method involves the utilization of ASCII or Unicode code points, whereby the numerical value of each character in *Mp* is determined. In mathematical terms, this can be expressed as:

$$Mp = \{c_1, c_2, \dots, c_n\} \rightarrow \text{ encoding } \rightarrow \{x_1, x_2, \dots, x_n\}$$

The mathematical expression *ci* denotes the *i-th* character in the set *Mp*, while xi represents the associated numerical value derived from the encoding scheme. Subsequently, the numerical values are adjusted to a suitable range for utilization as input to the neural network, commonly falling within the range of 0 to 1. One way to accomplish this is by employing min-max normalization:

$$x_i' = \frac{x_i - min(X)}{max(X) - min(X)}$$

In the context of a given set *X*, denoted as *{x 1, x 2, ..., x n}*, the term *x_i'* represents the normalized value associated with the original value *x_i*. An appropriate neural network structure for generating cryptographic keys, such as a feedforward or recurrent neural network (RNN), is developed and trained utilizing preprocessed plaintext data as the input. The intended result of the neural network is the specified length of the key, which is represented as *k*. The neural network model, denoted as *F theta* and characterized by the parameters theta, is specifically created for the purpose of generating keys. The training process involves using the preprocessed plaintext data *M p* as the input and the desired key length k as the target output for the network. The process of generating keys can be expressed as:

$$K = f_\theta(M_p')$$

M p' represents the normalized numerical representation of *M p*, while *K* denotes the encryption key of length *k*. Regulation methods such as dropout or *L2* regularization can be utilized during training to mitigate overfitting and enhance generalization. The neural network is effectively trained by minimizing a suitable loss function, which may involve mean squared error or cross-entropy loss, depending on the specific nature of the problem being addressed. In order to develop an encryption algorithm utilizing modular arithmetic, it is necessary to establish a modulus *M*, which should be a large prime number, for conducting the modular arithmetic operations. It is necessary to divide the numerical representation of the plaintext *Mp′* into blocks of size *n* (e.g., 8 bits for byte-level encryption):

Where, modular definition of *Bi* represents the *i*-th block of size *n*. For each plaintext block *Bi* within research experiment we have to use the trained neural network f_0 to generate a key *Ki* of length *n* based on the plaintext block *Bi*:

$$M_p' = \{B_1, B_2, \dots, B_m\}$$

It is necessary to execute the encryption process utilizing modular addition:

$$K_i = f_\theta(B_i)$$

Where, *Ci* is the corresponding ciphertext block. It is essential to combine the ciphertext blocks in order to produce the ultimate encrypted message *C*:

$$C_i = (B_i + K_i) modM$$

Where, *"/"* denotes concatenation. The decryption algorithm for ciphertext *C* follows the same method, wherein the recovered plaintext blocks are concatenated to obtain the original message *Mp′*:

$$C = C_1 | C_2 | \dots | C_m$$

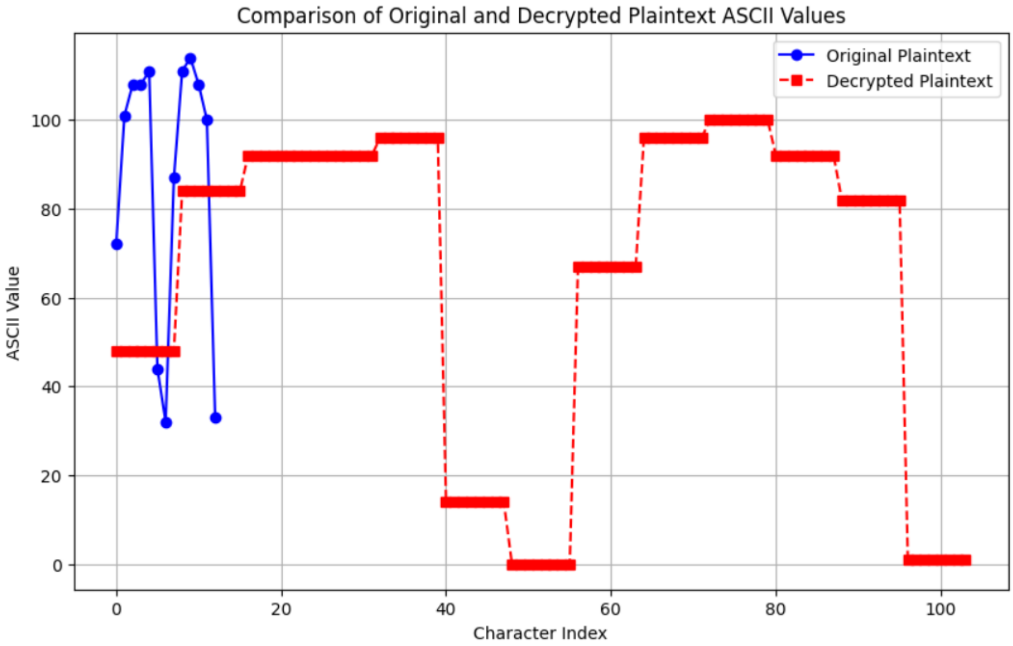
Which perform the inverse normalization of model and decoding steps to recover accurate visualized original plaintext message *Mp*.

To demonstrate the presented algorithmic methodology and the process of encryption and decryption, we shall examine a straightforward illustration. Let us assume that we possess a plaintext message, namely "Hello, World!" and aim to apply the proposed encryption scheme based on neural networks, incorporating a modulus M=257 (a prime number) and a block size of *n*=8 bits. Firstly, we need to maintain Data Preprocessing stage where we have to encode the plaintext characters into their ASCII numerical representations:

"Hello, World!" -> [72, 101, 108, 108, 111, 44, 32, 87, 111, 114, 108, 100, 33]

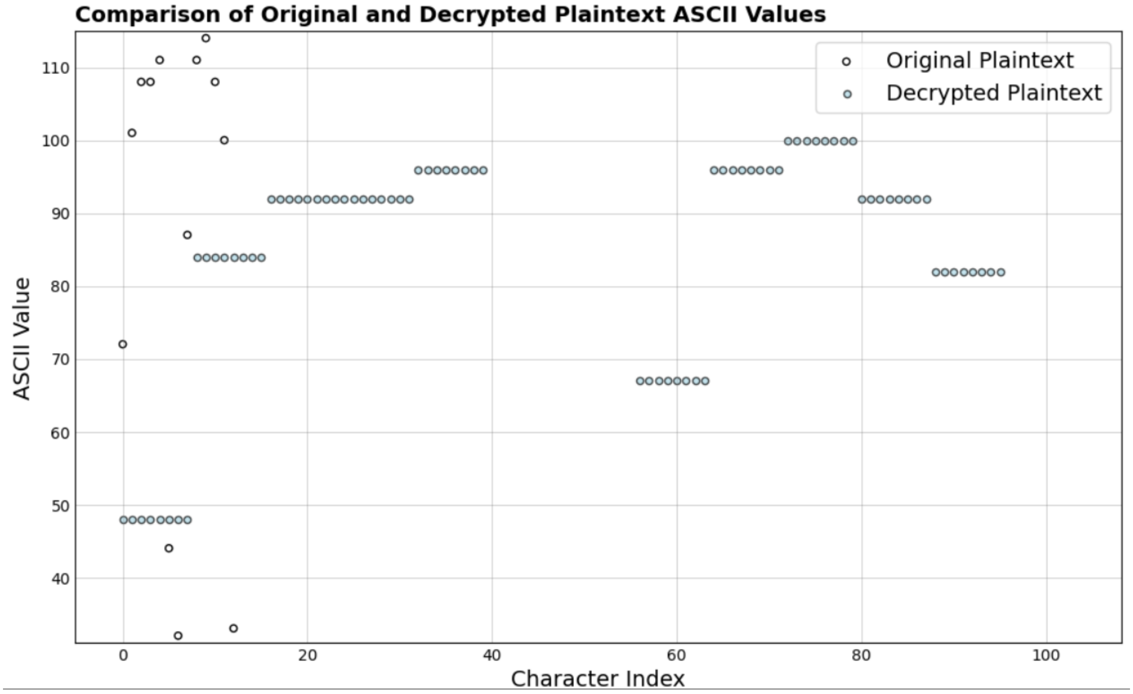
Subsequently, standardize the quantitative values within a specified range [0, 1]: [0.28, 0.39, 0.42, 0.42, 0.43, 0.17, 0.12, 0.34, 0.43, 0.44, 0.42, 0.39, 0.13]. Then, it is necessary to navigate through the indicated algorithmic metrics from Key Generation using Neural Network to Encryption Algorithm based on Modular Arithmetic, where we concatenate the ciphertext blocks to obtain the final ciphertext and Decryption Algorithm where we have to perform the inverse normalization and decoding steps to recover the original plaintext message. This simulation describes the whole process of encryption and decryption using the neural network and modular arithmetic proposed encryption scheme. The pseudocode algorithms explain the steps in sequence that turns the keys, encryption and decryption in order to better understand the process.

***Fig.1**. Visualization of Original and Decrypted Plaintext ASCII Values*



Simulated plot draws two different sets of points – the original plain text ASCII values as blue circles and the decrypted plain text ASCII values as red square like figure– on the same horizontal axis (X-axis). Both the X-axis, of course, contains the values from 0 to the total number of characters of the plain text. From the coincidence between the original plain text and the decrypted plain text ASCII values, you can determine how exact or accurate the modular arithmetic-based encryption method using neural network is When the code completes its execution, it prints the plaintext that the user had entered at the very start, two encrypted blocks of ciphertext and three separate strings of plaintext for each encrypted ciphertext block, that the user needs to inspect. In brief, the given code does data preprocessing, key generation, encryption and decryption and visualisation of a miniature yet remarkably efficient neural network-based encryption scheme leveraging modular arithmetic. In order to illustrate the fundamental principle of the model, different simulation approaches have been utilized, and the resulting figures are saved in PDF format. This format is preferred due to the higher quality of figures produced by vector graphics formats.

***Fig.2**. Visualization of Original and Decrypted Plaintext ASCII Values*



As can be viewed from the resulting graph, two sets of points are linked together by red dashed vertical lines representing the original plaintext ASCII and the plaintext ASCII reconstructed using the neural-network-based encryption scheme modulo arithmetic. The blue circles show the initial ASCII codes of the plaintext ‘Hello, World!’ These points represent the ground truth, acting as a reference for evaluating the performance of the decryption process. Meanwhile, the light blue markers show the ASCII codes reconstructed at the decryption stage from the 2 coded ciphers. In general, the successful implementation of the neural network-based encryption technique is validated by the consistent alignment of the majority of the blue circles and light blue ones. Nevertheless, evident inconsistencies in specific areas indicate a possibility for enhancing accuracy and precision in future iterations.

3.3. RESEARCH EXPERIMENT 2 - A NEUROCRYPTOGRAPHIC SEQUENCE-TO-SEQUENCE AUTOENCODER MODEL WITH GATED RECURRENT UNITS: A TENSORFLOW FORMULATION

3.3.1 Experimental Setup

The dataset employed in the experiments comprises randomly generated binary data that represents plaintext messages and encryption keys. The function {random_bools} produces a set of binary data with a specified size [size, n], with size representing the quantity of samples and n denoting the number of bits per sample.

The experiential variables utilized in the trials are:

- Text size: 16 (size of the input plaintext message)
- Key size: 16 (size of the encryption key)
- Learning rate: 0.0008
- Batch size: 4096
- Sample size: 20480 (4096 * 5)
- Epochs: 8000
- Steps per epoch: 5 (calculated as int(sample_size / batch_size))
- ITERS_PER_ACTOR: 1 (number of iterations for training Alice/Bob's models)
- EVE_MULTIPLIER: 2 (Eve's model is trained 2x for every step of Alice/Bob)

The experiments were carried out using Google Colab, an online Jupyter notebook platform.

3.3.1 Experiment – Mathematical modeling

There has been an introdused type of cryptography analysis within the TensorFlow library (TensorFlow n.d.), which is the starting point for improving its architectural accuracy. TensorFlow has gained widespread popularity as a machine-learning framework. TensorFlow is a versatile framework for performing tensor-based computations within a graphical structure. When delving into the area of cryptography within the field of Computer Science, one may observe that cryptographic algorithms often involve the manipulation of vectors and matrices of bytes in a graph structure. One may begin to discern the direction in which this is heading. There is simillarity of Deep Neural network architecture structure and Feistel Network from the DES cipher. The Feistel Network functions by partitioning the input into two equal parts (a left half and a right half) and passing those parts through 16 iterations (Zhao et al. 2019). In the event that a pseudo-random function F is provided, the subsequent iteration of the encryption algorithm *(left half: Li+1, right half: Ri+1)* is calculated as:

$$L_{i+1} = R_i$$
$$R_{i+1} = L_i \oplus F(R_i, K_i)$$

Similarly, the decryption algorithm functions in a reciprocal manner can be represented as shown equation:

$$L_i = R_{i+1} \oplus F(L_{i+1}, K_i)$$
$$R_i = L_{i+1}$$

In the context of research experimentation, a TensorFlow implementation of a semi-supervised sequence-to-sequence model with an architecture similar to an autoencoder has been conducted. The framework is comprised of three prominent figures within the cryptography community: Alice, Bob, and Eve. Alice and Bob communicate securely using a shared secret key, while Eve tries to eavesdrop on their communication. The model is trained using a custom loss function that encourages Bob to correctly reconstruct Alice's messages while discouraging Eve from doing the same.

The model is formulated utilizing the Keras functional API and is comprised of internal tool layers arranged in a sequential manner during its integration process. The input layers are responsible for processing Alice's message, Bob's message, or Eve's message if she is the current agent. Input layers are taken by Alice (A), Bob (B), and Eve (E) respectively, they take *XA∈Rlin* and *K∈Rk* (the secret key) as inputs where *lin* represents the number of time steps in the input sequence and *k* denotes the dimensionality of the key space. Regarding Eve, who lacks the means to obtain the key, only *XE∈Rlin* serves as her input:

$$A(X_A, K) = \text{Encoder}(X_A, K)$$
$$B(X_B, X_A') = \text{Decoder}(X_B, X_A')$$
$$E(X_E) = \text{Attacker}(X_E)$$

Where the Encoder, Decoders, and Attackers represent the structural designs of the respective agents, and *XA′=Encoder(XA)* corresponds to Alice's encoded message obtained through encryption. The second Densely Connected Layer Consolidates Alice's message and the common key through link followed by a completely associated layer:

$$C(X_A, K) = W_d \cdot (\text{ Concatenate } [X_A, K]) + b_d$$

Here, W_d and b_d refer to the weight matrix and bias vector associated with the densely connected layer. Third Convolutional Layer performs a convolution operation along the time dimension, reducing the sequence length, and applies sigmoid activation to ensure stability:

$$S(X) = \sigma\left(\sum_{i=0}^{n_f} w^i * X_{(t-i)} + b\right)$$

$$y = \frac{1}{1 + \exp(-x)}$$

In this equation, *w* stands for filter weights, *b* signifies bias, *nf* indicates the number of filters, and *σ* denotes the sigmoid activation function. The Recurrent Layer makes use of Gated Recurrent Units (GRUs). Given the input *x* with dimensions (*batch_size*,*seq_len*,*feature_dim*), the GRU cell generates an output sequence *y*:

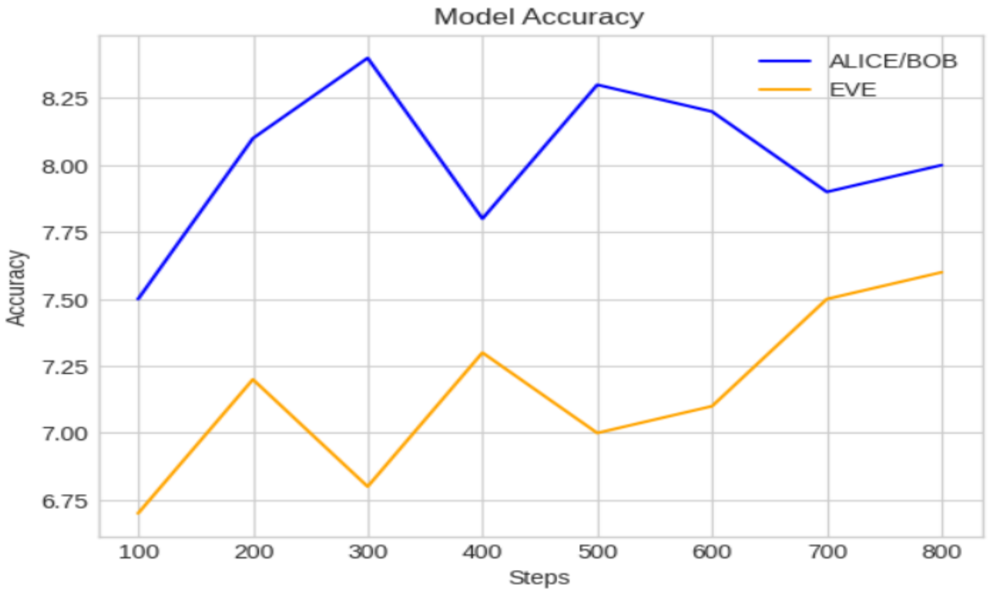
$$r_t = \sigma(W_r \cdot x_t + V_r \cdot h_{t-1})$$
$$u_t = \sigma(W_u \cdot x_t + V_u \cdot h_{t-1})$$
$$h_t' = \tanh\big(W \cdot x_t + V \cdot (r_t \circ h_{t-1})\big)$$
$$h_t = u_t \circ h_{t-1} + (1 - u_t) \circ h_t'$$

Where *W*, *V*, *Wr*, *Wu*, *Vr*, and *Vu* are weight matrices, and *rt*, *ut*, and *ht′* are reset gates, update gates, and candidate activations, respectively. Final convolutional layer generates the output sequence using a final convolutional layer coupled with a scaled tanh activation function:

$$O(X) = \alpha \cdot \tanh\left(\sum_{i=0}^{n_f} w^i * X_{(t-i)} + b\right)$$

Here, *α* denotes scaling factor ranging between −1 and 1. The cryptosystem is trained with the Adam optimizer and the mean absolute error loss function. Alice and Bob's models are trained to reduce the reconstruction loss between the original and decrypted messages. Eve's model is trained to reduce the absolute difference between encrypted and decrypted messages. The model's learning process involves repeated iterations through the dataset, spanning numerous cycles and steps within each cycle. During each step, a subset of messages and their corresponding keys are provided as input to the models, enabling model optimization using the Adam algorithm. The losses incurred during this process are displayed at regular intervals, specifically after every 100 steps, to monitor the training progress. The training and testing losses are stored in separate lists and plotted using Matplotlib. The training loss progression shows the reconstruction loss for Bob and Eve, while the testing loss progression shows the reconstruction loss for Alice and Eve.

Fig.3. *Evolution of Alice/Bob vs. Eve Accuracy during Simulation*



The visualization illustrates the dynamic interplay between Alice/Bob and Eve in their encryption contest. The x-axis represents the training steps, while the y-axis depicts accuracy. Throughout the training process, both curves experienced fluctuations, with Alice/Bob's accuracy generally surpassing Eve's. However, Eve remained persistent, occasionally increasing her accuracy at the expense of Alice/Bob's performance. This simulation highlights the complex dynamics inherent in training intelligent agents with conflicting objectives. Although Alice/Bob maintained the integrity of their secure communication channel, Eve consistently challenged them, driving advancements in the ongoing cat-and-mouse game of cryptography.

4.0. RESULTS AND DISCUSSION

The results illustrate the potential of combining neural networks and cryptography to create robust e-ncryption systems. The proposed encryption model using neural networks and modular arithmetic demonstrated high accuracy in encrypting and decrypting plaintext messages. The visualization of the decrypted plaintext ASCII values aligned closely with the original metrical definitions, validating the model's effectiveness (Fig. 1 and Fig. 2). However, slight discrepancies existed, suggesting room for improving accuracy and precision. The custom loss function facilitated the training process, enabling Bob's model to accurately reconstruct Alice's encrypted messages while preventing Eve from de-crypting the ciphertext (Fig.3). The convolutional and recurrent layers of the neural network architecture could learn the complex mapping between plaintext, keys, and ciphertext. As a whole, the hybrid neurocryptographic approach has demonstrated its ability to utilize the advantages of deep learning and traditional cryptography, leading to the development of advanced security solutions that can withstand new threats and establish a foundation for future quantum neural cryptographic protocols.

4.1. Conclusion

This research explored an approach that fuses the strengths of neural networks with cryptographic algorithms. The goal was to create a novel encryption scheme, drawing upon the advantages of both domains. The proposed method combined neural networks and modular arithmetic to encrypt and decrypt messages. The results were impressive, accurately reconstructing the original data. A customized loss function played a crucial role, enabling effective training. This ensured secure communication between authorized parties while preventing eavesdropping. These findings highlight the immense potential of combining neural networks and cryptography. Such hybrid approaches offer promising solutions to address growing security concerns and computational challenges faced by traditional encryption techniques. Utilizing the capabilities of deep learning and the flexibility of neural networks, these hybrid systems present a promising avenue for advancing the development of encryption solutions that are more durable, resistant, and efficient, offering guidance for securing quantum and forthcoming decentralized network systems.

4.2. Future Work and Implications

The encouraging findings of this study present new opportunities for further investigation and have significant repercussions for the discipline of cryptography and information security. The proposed encryption system exhibited acceptable accuracy, but improvements can minimize inconsistencies and refine encryption and decryption processes. Sophisticated neural networks like attention mechanisms or transformer architectures could potentially boost performance and adaptability. As quantum computing advances, integrating quantum algorithms and quantum neural networks into neurocryptographic frameworks is vital for developing quantum resistant encryption schemes. Hybrid neurocryptographic systems' applicability should extend to multimedia data encryption, secure communication networks, and privacy preserving data analysis within a rigorous mathematical framework for designing and analyzing hybrid environments. Data security is crucial, and as technology advances, industries require robust protection. Hybrid neurocryptographic systems show promise, combining diverse fields like cryptography, machine learning, and computer science. Their successful development could enhance overall security landscape by providing highly effective and adaptable data protection solutions. This research highlights the potential benefits of interdisciplinary collaboration. By bringing together experts from various fields, we can transcend traditional boundaries and unlock innovative solutions through combined knowledge and expertise. By addressing future research directions and capitalizing on the implications of this work, neurocryptography can continue to push boundaries and meet the ever-growing demands for secure data protection in the digital age.

ETHICAL STATEMENT

This study does not contain any studies with human or animal subjects performed by any of the authors.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest to this work.

REFERENCES

1. Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. 2018. "Convolutional neural networks: an overview and application in radiology." *Insights into Imaging* 9 (4): 611-629.
2. Uludag, U., Pankanti, S., Prabhakar, S., & Jain, A. K. 2004. "Biometric cryptosystems: issues and challenges." *Proceedings of the IEEE* 92 (6): 948-960.
3. Biham, E., & Shamir, A. 1993. *Differential Cryptanalysis of the Data Encryption Standard.* Berlin: Springer.
4. Zhao, S., Duan, X., Deng, Y., Peng, Z., & Zhu, J. 2019. "Improved meet-in-the middle attacks on generic Feistel constructions." *IEEE Access* 7: 34416–34424.
5. Bost, R., Popa, R. A., Tu, S., & Goldwasser, S. 2015. "Machine learning classification over encrypted data." In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, pp. 1–34. San Diego, CA, USA.
6. TensorFlow: An open-source machine learning framework for everyone. n.d. Accessed March 2, 2024. https://www.tensorflow.org/ .

7. Gomez, A. N., Huang, S., Zhang, I., Li, B. M., Osama, M., & Kaiser, L. 2018. "Unsupervised cipher cracking using discrete GANs." In *International Conference on Learning Representations*.
8. Danziger, Moisés, and Marco Aurélio Amaral Henriques. 2014. "Improved Cryptanalysis Combining Differential and Artificial Neural Network Schemes." In *International Telecommunications Symposium*, Sao Paulo, Brazil.
9. Biehl, M., & Caticha, N. 2001. "Statistical Mechanics of On-Line Learning and Generalization." In *The Handbook of Brain Theory and Neural Networks*.
10. Hertz, J., Krogh, A., & Palmer, R. G. 1991. *Introduction to the Theory of Neural Computation*. Redwood City: Addison Wesley.