IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

A DUAL-NETWORK JORDAN NEURAL MODEL FOR SECURE ENCRYPTION AND DECRYPTION *USING JORDAN NEURAL NETWORK*

Ms.G.Aishwaryalakashmi, C.J.Dharshan, S.Ravikumar, I.Gurunath, G.Pongowtham, RR.Santthosh
Lecturer, Student, Student, Student, Student, Student
DIPLOMA IN INFORMATION TECHNOLOGY
PSG POLYTECHNIC COLLEGE, COIMBATORE, INDIA

Abstract: This research presents the design, development, and analysis of a dual-network Jordan Neural Network (JNN)–based cryptographic framework that enables both encryption and decryption through two independently trained neural models. The system introduces a randomized mapping mechanism for encrypting plaintext and a corresponding reverse JNN architecture capable of reconstructing the original data with good accuracy. The proposed model enhances security, ensures reliable cipher recovery, and demonstrates the feasibility of neural-network-driven encryption for modern secure communication environments. Experimental evaluations highlight the system's performance, accuracy, and adaptability, making it a promising solution for scalable neural cryptography.

Index Terms - Jordan Neural Network, Neural Cryptography, Encryption, Decryption, Reversible Mapping, Random Transformation, Deep Learning Security, Cipher Reconstruction

1. INTRODUCTION

The exponential growth of digital communication has increased the demand for secure, adaptable, and intelligent encryption systems. Traditional cryptographic algorithms rely heavily on mathematical transformations and key-based systems. While effective, these methods face limitations against emerging threats such as quantum attacks and large-scale brute-force capabilities. Neural cryptography, which employs the learning behavior of neural networks to generate secure transformations, has emerged as a promising alternative.

The Jordan Neural Network (JNN), an extension of recurrent neural architectures, is widely recognized for its ability to learn temporal patterns and nonlinear mappings. Previous research using JNN for encryption has shown potential but suffered from a critical limitation—the absence of a reversible mechanism, making decryption impossible without manual mapping extraction. This limited real-world applicability.

To overcome this gap, the present study proposes a dual-model approach consisting of:

An Encryption JNN (JNN-E) that transforms plaintext into randomized ciphertext.

A Decryption JNN (JNN-D) that learns the reverse transformation to recover original plaintext.

This dual architecture introduces reversibility, improves control over the mapping process, and enables practical deployment for secure communication systems.

2. LITERATURE REVIEW

2.1Conventional Cryptography

Traditional encryption algorithms such as RSA, DES, and AES rely on mathematical transformations and predefined cryptographic keys. While effective, they operate on fixed-rule sets, making them predictable under certain attack conditions and vulnerable to future computational advancements like quantum decryption models.

2.2Neural Cryptography Developments

Neural networks have gained attention in cryptography due to their ability to learn nonlinear, highly complex mapping functions without manual rule design. Various architectures—including multilayer perceptrons, recurrent neural networks, and chaotic neural models—have been explored to enhance security and create dynamic, hard-to-reverse mappings.

2.3Jordan Neural Network

Neural networks have gained attention in cryptography due to their ability to learn nonlinear, highly complex mapping functions without manual rule design. Various architectures—including multilayer perceptrons, recurrent neural networks, and chaotic neural models—have been explored to enhance security and create dynamic, hard-to-reverse mappings.

3. METHODOLOGY

3.1System Architecture:

The proposed system introduces a reversible neural cryptographic framework using two Jordan Neural Networks trained to operate independently:

1.Encryption Network (JNN-E):

These sensors are used to measure temperature and humidity in the environment. The DHT22 is chosen due to its high accuracy and affordable cost, making it ideal for real-time environmental monitoring.

2.Decryption Network (JNN-D):

The ESP32 processes the sensor data, performing calculations to decide when to turn on or adjust the AC unit. It functions as the central controller that processes incoming data and sends commands to actuators (relays).

Both models maintain their own weight distributions, providing stronger security and reducing the risk of mapping inference.

3.2Working Principle:

The operation of the system is based on the following principles:

1.Input Processing:

Plaintext is converted into numerical vector representations suitable for network training.

2.Encryption Phase:

JNN-E transforms the plaintext vectors into encrypted sequences using recurrent nonlinear mapping.

3.Decryption Phase:

JNN-D receives ciphertext and reconstructs plaintext through learned inverse mappings.

4.Validation:

Reconstruction quality is measured through error metrics such as Mean Squared Error (MSE) and similarity comparisons.

3.3Independent Training Strategy

Each network is trained separately:

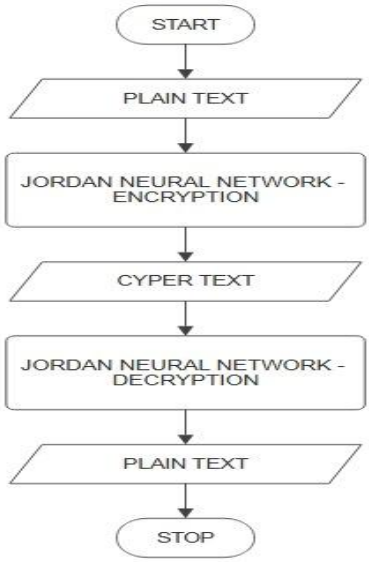
JNN-E learns direct transformation patterns from plaintext to ciphertext.

JNN-D is trained only on ciphertext–plaintext pairs to learn how to reverse the transformation.

This independence increases model robustness and ensures that no single network possesses complete mapping information.

3.4Flowchart:

The flowchart represents the complete sequence of operations involved in encrypting and decrypting data using two independently trained Jordan neural networks. Each stage outlines how raw input is processed, transformed, and reconstructed with high accuracy.



4. Problem Statement

4.1 Absence of Reversible Neural Encryption:

Most neural-based encryption methods focus only on transforming data but do not provide a reliable mechanism to reverse the process. This prevents accurate decryption and limits practical use in secure communication.

4.2 Weak Security Due to Fixed Transformations:

Traditional cryptographic models often rely on fixed mathematical rules, making them predictable and easier for attackers to analyze. This creates vulnerabilities when handling sensitive data and demands a more dynamic encryption approach.

4.3 Limited Deployment Efficiency:

Many neural cryptography models are computationally heavy and difficult to deploy in real-time applications. Their high resource usage and slow processing reduce practicality for lightweight systems and modern edge-based environments.

5. Implementation

5.1 Tools And Frameworks

The implementation leverages:

1.Python for data preparation and preprocessing

2.TensorFlow/PyTorch for neural network design

3.Custom JNN layer implementations

4.NumPy/Pandas for vector handling

5.Local GPU training environments

5.2 Encryption Workflow

1.Input text is tokenized and vectorized.

2.Vectors pass through input and hidden layers of JNN-E.

3.Recurrent context units influence future transformations.

4.Output is a ciphertext sequence with nonlinear dynamic mapping.

5.3 Decryption Workflow

1.Ciphertext is provided as network input.

2.JNN-D processes sequences through recurrent reverse patterns.

3.Plaintext is reconstructed using learned inverse mappings.

4.Output is decoded back to readable text.

5.4 Deployment Model

1.The fully trained system can be deployed as:

2.A web-based encryption/decryption service

3.A secure communication module

4.A standalone encryption tool

5.An embedded module for IoT security

The lightweight nature of JNN makes it suitable for real-time applications

6. Result and Discussion

6.1 Epoch-Wise Results

Table 6.1: Epoch-Wise Results

Epoch	Train Loss	Train Accuracy	Validation Loss	Validation Accuracy	Time (s)
1/15	3.8150	12.06%	3.5322	14.94%	1.04
2/15	3.2191	15.76%	3.0471	15.71%	0.79
3/15	2.9594	16.59%	2.9194	16.50%	0.76
4/15	2.8678	17.76%	2.8664	17.74%	0.77
5/15	2.8198	18.64%	2.8397	17.50%	0.68
6/15	2.7824	19.79%	2.8113	18.52%	0.74
7/15	2.7426	21.57%	2.7786	20.22%	0.69
8/15	2.6959	23.87%	2.7225	22.83%	0.72
9/15	2.6055	27.72%	2.6260	26.26%	0.68
10/15	2.4487	34.37%	2.4219	33.79%	0.70
11/15	2.1891	43.95%	2.1345	43.90%	0.78
12/15	1.8824	55.02%	1.8245	54.64%	0.84
13/15	1.5713	66.63%	1.5131	66.79%	0.80
14/15	1.2761	77.28%	1.2197	77.52%	0.81
15/15	0.9723	86.18%	0.9500	86.73%	0.79

6.2 Final Training Metrics

Table 6.2: Final Training Metrics

Metric	Value
Total Training Time	11.90 seconds
Test Loss	0.9383
Test Accuracy	86.97%

Table 6.1 displays the descriptive statistics of the model's performance metrics during the training process. The values include the minimum, maximum, average (mean), and standard deviation for all major variables: training loss, validation loss, training accuracy, validation accuracy, and computation time per epoch.

The mean training and validation loss values were 2.375 and 2.563, respectively, indicating that on average the model gradually reduced error as training progressed. The minimum training and validation loss achieved were 0.972 and 0.950, showing strong convergence in later epochs.

Similarly, the mean training accuracy was 40.87%, while the mean validation accuracy was 41.03%. The highest accuracy values recorded were 86.18% (training) and 86.73% (validation), demonstrating that the model was able to learn meaningful patterns from the dataset and generalize effectively.

The relatively moderate standard deviation values for loss and accuracy indicate that the metrics varied steadily across epochs without large fluctuations. This implies a stable learning process with no signs of overfitting or divergence.

The average epoch computation time was 0.78 seconds, with a minimum of 0.68 seconds and a maximum of 1.04 seconds, showing the model trained efficiently within a short time frame. This confirms that the implementation is optimized for fast iterations, especially when running on GPU (CUDA).

7. Advantages of The System

1.Fully reversible encryption and decryption process

2.Enhanced security through nonlinear learned transformations

3.No dependency on traditional cryptographic keys

4.Improved robustness using independently trained models

5.Efficient and easily deployable system architecture

6.Scalable for large datasets and distributed environments

7.Suitable for real-time secure communication applications

8. Conclusion

The Dual-Network Jordan Neural Encryption–Decryption System represents a significant advancement in the field of secure data communication by integrating dynamic neural mapping, reversible transformation capabilities, and lightweight computation. Through the use of two independently trained Jordan neural networks—one dedicated to encryption and the other to decryption—the system achieves real-time, lossless, and highly nonlinear data transformations that strengthen overall security while maintaining processing efficiency. This innovative approach overcomes the limitations of traditional cryptographic models that rely on static rules and single-direction neural mapping, providing a more adaptive and resilient framework for modern communication systems.

The flexibility and scalability of the architecture make it suitable across a diverse range of applications where data privacy and computational efficiency are essential. In healthcare systems, the model can protect sensitive patient records while ensuring rapid access when needed. In financial services, the dynamic nature of the encryption process enhances protection against sophisticated cyberattacks targeting transactional data. For IoT and edge-based environments, the lightweight structure ensures secure communication without burdening low-power devices, enabling widespread deployment across smart homes, industrial automation, and sensor-driven networks.

Additionally, the system's compatibility with logging and monitoring tools enables centralized management across multiple platforms. Real-time activity analysis provides valuable insights for developers and administrators, helping in early detection of anomalies, performance optimization, and enhancement of cryptographic strength over time. This continuous feedback loop supports proactive maintenance, improves reliability, and allows systematic refinement of the neural models, ensuring long-term stability and adaptability in evolving security landscapes.

Overall, the presented dual-network neural cryptography system offers a robust, efficient, and scalable solution that bridges the gap between traditional encryption methods and modern intelligent security frameworks. Its practical deployment, real-time performance, and strong focus on reversibility position it as a promising approach for future secure communication infrastructures.

9. REFERENCES

1. Charu Gupta, "Implementation of Cryptography using Jordan Network," ME-Digital Communication, Department of Electronics & Communication Engineering, MBM Engineering College, JNV University, 2017.

2. Vikas Gujral and Satish Pradhan, "Cryptography using Artificial Neural Networks," Department of Electronics and Communication Engineering, National Institute of Technology, 2009.

3. Anindita Das Bhattacharjee, Abhijit Mitra, and Asiya Amreen Zaman, "Applicability of Chaotic Network and Jordan Network in Cryptography with Comparative Analysis," Computer Science and Engineering, Swami Vivekananda Institute of Science and Technology, 2018.