

A Neural Network Enhanced RSA Model Towards a Confidentiality-Integrity-Authenticity Compliant Hybrid



University of Venda

A dissertation submitted to the Department of Mathematics and Applied Mathematics, University of Venda, in partial fulfillment of the requirements for the award of a

Master of Science
(Applied Mathematics)

of the

University of Venda

by

Magoma Promise Tshepiso

Main supervisor : Dr. Colin Chibaya (Sol Plaatje University)
Co-supervisor : Dr. Dephney Mathebula (University of Venda)

Declaration

I, Promise Tshepiso Magoma, student number: 14014751, hereby declare that this research report is my work being submitted in partial fulfilment for the award of a Master of Science degree in Applied Mathematics at the University of Venda. This work has not been submitted for any other degree or examination at any other university or higher learning institution. It is original in design and execution, and all the reference material contained therein has been duly acknowledged.



Main author

19th of March 2021

Date



Main supervisor

19th of March 2021

Date/



Co - supervisor

19th of March 2021

Date

Acknowledgements

Firstly, I would like to thank the All-mighty Mighty God for giving me ideas, enthusiasm, power, and the strength to embark on this research study to completion. He is the One always at work in me, making me willing, making me endure, and propelling me forward. Glory be to God.

I wish to express my special sincere gratitude to my supervisors, Dr. Colin Chibaya and Dr. Dephney Mathebula for their guidance and patience. This research study would not have been a success without their professional engagements. Thank you for your faith in me when, sometimes, I fell into difficult circumstances.

My friends and family cannot go unmentioned. Thank you all for tirelessly listening to my ideas. Thank you for your encouragement when it was needed most. My parents, Cedric Magoma and Florah Rakgakole-Magoma thank you for your parental support.

A special thank you also goes to my funders, the National e-Science Postgraduate Teaching and Training Platform, the University of Venda, the University of Witwatersrand, and the Department of Science and Innovation, for all the financial support. Without you, pursuing these studies would have been close to impossible.

Abstract

Data security is an important aspect in the field of data science where data collection, analysis, interpretation, and sharing are a primary goal. To prevent unauthorized access to data, creative methods to securing data are sought.

Cryptography is about the development of algorithms with which to hide data. The three key objectives of cryptography are to achieve data confidentiality (C), data integrity (I), and data authenticity (A). Algorithms that can achieve all these three objectives at once are said to be CIA compliant. However, there are barely any algorithms out there that can satisfy these three objectives in one goal. However, CIA-compliant cryptosystems are, to the best of our knowledge, rare.

The RSA algorithm is a compelling cryptosystem that was mainly designed to achieve data confidentiality. It demonstrates attractive properties for improvement towards CIA compliancy. Some research has tried to upgrade the RSA algorithm by combining it with the DH model or the El Gamal model. However, still, the outcome would either be CI or CA compliant, leaving out one of the three objectives.

This study investigates the improvement of the RSA algorithm by incorporating a neural network to learn data integrity and data authenticity towards creating a CIA-compliant hybrid RSA model. To the best of our knowledge, this is the first time a neural network has been proposed for improving the RSA model towards CIA compliance.

Experimental results indicate that a neural network can learn data integrity and data authenticity in RSA encrypted messages. Data analysis affirmed that

neural network learning can be generalized. A conclusion that the RSA algorithm can be upgraded towards CIA compliance when a neural network is incorporated was arrived at. These findings have implications for the commercial standing of the RSA algorithm as well as for the body of knowledge in the cryptography domain.

Keywords: RSA algorithm, data confidentiality, data integrity, data authenticity, Neural network, Machine learning

Table of Contents

| | |
|------------------------------------|-----|
| Declaration | ii |
| Acknowledgements | iii |
| Abstract..... | iv |
| Keywords | v |
| List of Tables..... | ix |
| List of Figures..... | x |
| List of Algorithms..... | xi |
| CHAPTER 1 INTRODUCTION | |
| 1.1 Statement of the problem | 6 |
| 1.2 Aim | 7 |
| 1.3 Objectives | 7 |
| 1.4 Questions | 7 |
| 1.5 Hypothesis | 8 |
| 1.6 Motivation..... | 8 |
| 1.7 Contributions | 9 |
| 1.8 Location of the study | 10 |
| 1.9 Limitations | 10 |
| 1.10 Overview | 11 |
| 1.11 Summary..... | 12 |

CHAPTER 2 LITERATURE REVIEW

| | | |
|-----|---|----|
| 2.1 | Components of the RSA algorithm | 15 |
| 2.2 | The RSA algorithm and data confidentiality | 16 |
| 2.3 | The RSA algorithm and data integrity..... | 18 |
| 2.4 | The RSA algorithm and data source authentication | 19 |
| 2.5 | Computational performance of the RSA algorithm | 21 |
| 2.6 | The RSA algorithm and CIA compliance | 22 |
| 2.7 | Interventions in which neural networks are considered | 23 |
| 2.8 | Summary of the gap | 25 |
| 2.9 | Summary..... | 26 |

CHAPTER 3 METHODOLOGY AND THEORETICAL FRAMEWORK

| | | |
|---------|--------------------------------------|----|
| 3.1 | Statement of the problem | 27 |
| 3.2 | Overview of the chapter | 28 |
| 3.3 | Theoretical framework..... | 28 |
| 3.4 | Research design | 31 |
| 3.4.1 | Proposed methods..... | 32 |
| 3.4.1.1 | The RSA algorithm | 33 |
| 3.4.1.2 | A neural network | 36 |
| 3.4.1.3 | The hybrid algorithm..... | 40 |
| 3.4.2 | Samples and sampling techniques..... | 42 |
| 3.4.3 | Data collection..... | 42 |
| 3.4.4 | Tools..... | 43 |
| 3.4.5 | Data analysis..... | 44 |

| | | |
|--|---|----|
| 3.4.5.1 | Descriptive statistics | 44 |
| 3.4.5.2 | Inferential statistics | 46 |
| 3.5 | Summary | 47 |
| CHAPTER 4 EXPERIMENTS, FINDINGS, AND INTERPRETATION | | |
| 4.1 | Statement of the problem | 50 |
| 4.2 | Overview of the chapter | 50 |
| 4.3 | Design of the neural network's learning process | 50 |
| 4.3.1 | Training a neural network | 51 |
| 4.3.2 | Testing of the neural network after training | 67 |
| 4.4 | Discussion of the results | 77 |
| 4.5 | Summary | 78 |
| CHAPTER 5 CONCLUSION | | |
| 5.1 | Summary of the chapters covered | 79 |
| 5.2 | Answers to the research questions | 81 |
| 5.3 | Reflections | 82 |
| 5.4 | Contributions | 82 |
| 5.5 | Future works | 83 |
| | References | 84 |
| | Appendices | 96 |

List of Tables

| | |
|--|----|
| Table 3.1: Python libraries..... | 43 |
| Table 4.1: Average Training Rates..... | 58 |
| Table 4.2: Summary of the neural network's training rates..... | 65 |
| Table 4.3: Analysis of variances in the neural network's training rates..... | 66 |
| Table 4.4: Average Testing Rates..... | 71 |
| Table 4.5: Summary of the neural network's testing rates..... | 76 |
| Table 4.6: Analysis of variances in the neural network's testing rates..... | 77 |

List of Figures

| | |
|---|----|
| Figure 2.1: Overview of the chapter..... | 14 |
| Figure 3.1: Design science research cycle (Source: Hevner, 2007)..... | 31 |
| Figure 3.2: A simple neural network..... | 37 |
| Figure 3.3: Single-layer neural network..... | 38 |
| Figure 3.4: RSA's predictive crypto-system workflow..... | 41 |
| Figure 4.1: Test for normality on learning rates achieved in the first run..... | 61 |
| Figure 4.2: Test for normality on learning rates achieved in the second run.. | 61 |
| Figure 4.3: Test for normality on learning rates achieved in the third run..... | 63 |
| Figure 4.4: Test for normality on learning rates achieved in the fourth run... | 63 |
| Figure 4.5: Test for normality on learning rates achieved in the fifth run..... | 64 |
| Figure 4.6: Test for normality on learning rates achieved in the sixth run..... | 64 |
| Figure 4.7: Correlation between training results..... | 66 |
| Figure 4.8: Normality test for the first column..... | 73 |
| Figure 4.9: Normality test for the second column..... | 73 |
| Figure 4.10: Normality test for the third column..... | 74 |
| Figure 4.11: Normality test for the fourth column..... | 74 |
| Figure 4.12: Normality test for the fifth column..... | 75 |
| Figure 4.13: Normality test for the sixth column..... | 75 |
| Figure 4.14: Correlation between testing results..... | 76 |

List of Algorithms

| | |
|---|----|
| Algorithm 3.1: The RSA algorithm..... | 34 |
| Algorithm 3.2: Rabin-Miller algorithm (Rahim et al, 2017)..... | 35 |
| Algorithm 3.3: Euclid's algorithm (Backhouse & Ferreira, 2011)..... | 36 |
| Algorithm 3.4: Predictive RSA algorithm..... | 40 |

Chapter 1 : Introduction

Cryptography is the study of mathematical techniques for data protection (Galbraith, 2012). In data science terms, it seeks to achieve, at least, one of the five main objectives of cryptosystems, namely, data confidentiality, data integrity, data source authentication, data users' accountability, or data availability (Menezes, Katz, Van Oorschot & Vanstone, 1996). In this context, data confidentiality (C) refers to a set of rules that limit access to data by incorrect parties. This ensures that only authorized parties have access and permission to data (Samonas & Coss, 2014). On the other hand, data integrity (I) refers to the ability to keep data away from unexpected or unwanted modifications. The idea is to maintain the data's original form (Samonas & Coss, 2014). Data integrity provides evidence that data has not been altered or tampered with by unauthorized parties (Samonas & Coss, 2014). Contrary, data source authentication (A) tackles the property that data originated from its purported source (Dworkin, 2010). It tracks data to its originator. Accountability (A) is about users taking responsibility for the losses or misuse of data by unauthorized parties (Li, Lou & Ren, 2010). Data availability (A) then refers to our ability to access data for use by the rightful parties at the right time and place (Li, Lou & Ren, 2010). Precisely, cryptography examines how these five objectives culminate the concepts of secrecy in computing and broadly speaking, data security. A cryptosystem that satisfies data confidentiality, data integrity, and data authenticity is said to be CIA compliant. A cryptosystem that goes beyond CIA (Confidentiality-Integrity-Authenticity) compliance can be CIAA (Confidentiality-Integrity-Authenticity-Accountability) or CIAAA (Confidentiality-Integrity-Authenticity-Accountability-Availability) compliant.

We understand data security as a collection of norms, methods, and techniques intended to protect data against malicious loss, modification, disclosure, or data leakage (Hauer, 2015). Data security can be enforced by using a variety of technologies and strategies, including administrative reviews (Crawford & Schultz, 2014), logical tests (Yavuz, Yazici, Kasapbaşı & Yamaç, 2016), business guidelines (Peltier, 2016), firewalls (Blaisdell & Vuong, 2010), and most importantly, encryption techniques (Galbraith, 2012).

In this context, administrative reviews are about the provision of security breach awareness, and employee training, before granting access to data (Aman & Snekkenes, 2013; Crawford & Schultz, 2014). However, administrative reviews are not clear on which data security objectives, among the five (see the first paragraph of this section for the five objectives), will be achieved.

On the other hand, logical tests involve auditing the cryptosystem using a risk-based approach to checking security breaches (Botella, Legiard, Peureux & Vernotte, 2014; Yavuz, Yazici, Kasapbaşı & Yamaç, 2016). This approach, nonetheless, satisfies more than one data security objective towards CIA compliance. We indicated that a CIA-compliant cryptosystem is one in which emphasis is on data confidentiality, data integrity, and data authenticity.

Contrary, business guidelines refer to written policies and procedures that guide businesses on how to protect their sensitive data (Aman & Snekkenes, 2013; Peltier, 2016). These are more theoretical policies and procedures such as ensuring that all laboratory doors are kept closed. However, business guidelines do not provide clear evidence that they can achieve total data confidentiality. They only focus on the theoretical ways of preventing data security breaches.

Data encryption techniques are more interesting. These mainly limit unauthorized users' access to data (C), checking for data originality (I), while also verifying data sources (A). The properties of encryption techniques are more appealing. Data security through encryption techniques is indispensable in the field of data science. Although data collection, data analyses, data interpretation, and data sharing are the primary goals of data science, data security is paramount. How then can we ensure data security? How can we ensure data confidentiality, data integrity, and data source authentication? How do we develop encryption algorithms that are CIA compliant? In this study, we will defer considerations for user accountability (A) and data availability (A) as future work.

There are many cryptosystems discussed in the literature. The Caesar algorithm (Paar & Pelzl, 2009) is the root of many cryptosystems. The Caesar cipher gave birth to cryptography. However, it is too easy to brute force attack because of its key size and the character set it supports. The Caesar cipher uses a numeric integer key from 2 to 24. It only supports alphabetic characters from A to Z. It is not case sensitive. In this study, the terms; cipher, algorithm, model, and cryptosystem are synonyms.

Improvements to the Caesar cipher gave birth to the Vigenere cipher (Omolara, Oludare & Abdulahi, 2014), and the One Time Pad (Rubin, 1996), among many others. The Vigenere model replaced Caesar's numeric integer key with a keyword. The One Time Pad model replaced Vigenere's user-chosen keyword with a randomly generated keyword. However, the character set supported remained the same (alphabetic letters from A to Z). However, there have always been some flaws noted in these Caesar siblings (commonly linked to the problem of only focusing on one objective, –data confidentiality).

Alternative approaches to the Caesar, Vigenere, and one time pad followed which revised the character set supported. The Vernam ciphers (Dey, 2012) was the first promising model to satisfy this expansion. In this case, all characters in the plain text would be converted to corresponding ASCII codes before encryption. Data would be encrypted in binary form after relevant conversions. Data would also be transmitted between communicators in binary form. Decryption would also take place in binary form. The character set supported was, therefore, upgraded from the 26 letters of the alphabets to all the 256 characters in the ASCII set. Case sensitivity was introduced. Ciphertext was, therefore, further complicated.

Other alternative interventions yielded transposition ciphers such as the rail fence model (Singh, 2000). In these, encryption changed from the substitution of characters by other characters, to shifting the positions of characters within the same text. Although these were plausible improvements to the body of knowledge, transposition models lacked the mathematical complexities extendable to CIA-compliant cryptosystems. These algorithms continued to only focus on data confidentiality.

Block ciphers were also proposed as better improvements to cryptosystems. The popular block ciphers that made a mark on the markets include the Playfair algorithm (Singh, 2000; Rahim & Ikhwan, 2016), the Hill cipher (Rahman, Abidin, Yusof & Usop, 2013; Singh, 2000), the Feistel cipher (Knudsen, 1993; Singh, 2000), and the popular Data Encryption Standard (DES), together with its many siblings (Singh, 2013). Most of these block ciphers have been very successful. However, similarly, they emphasize on data confidentiality only. They do not satisfy the requirements for CIA compliance.

More sophisticated approaches were proposed which exploited Euclidian algebra. These approaches include the DH model (Kallam, 2015), the RSA

algorithm (Stallings, 2006), and the El Gamal algorithm (Ahmed & Ali, 2011), to mention a few. In these models, modulo arithmetic is paramount. However, to the best of our knowledge, we are not aware of any cryptosystem in this category that satisfies three or more objectives of cryptography. We are not aware of a CIA-compliant model in this domain. Some algorithms only focus on data confidentiality (Alhassan, Ismaila, Waziri & Abdulkadir, 2016), while others isolatedly look at data integrity (Sadikin & Wardhani, 2016) or data authenticity (Thayananthan & Albeshri, 2015) independently.

The demand for cryptosystems that satisfy more objectives at once is apparent. Precisely, how do we develop CIA, CIAA, or even CIAAA compliant cryptosystems to broaden the body of knowledge? How do we strengthen cryptosystems by allowing them multiple focus? The need for such re-focused algorithms is evidently apparent (Cha, Schmidt, Leicher, & Shah, 2014).

The RSA model is a compelling asymmetric cryptographic algorithm (Stallings, 2006) that uses two different key sets, a public key for encryption and a private key for decryption. Communicating parties can share and use the public key (Bangju & Huanguo, 2006). The intended receiver of encrypted data keeps the private key hidden (Stallings, 2006). Encryption is achieved when senders of data use the public key to get cipher data before data transmission. However, as already said, the RSA model achieves only data confidentiality. It lacks any aspects of data integrity and data authenticity. How can the RSA model be improved to consider data integrity and user authentication?

A neural network is an algorithm aimed at understanding the relationship underlying a range of data through a mechanism that imitates the way the human brain works (Agarwal & Agarwal, 2013). It consists of a distributed network of nodes with local memory, performing localized information processing, linked by unidirectional signal channels (Hecht-Nielsen, 1992). The

learning algorithm of a neural network is supervised or unsupervised (Agarwal & Agarwal, 2013). A neural network can adapt to changing inputs without re-designing the output criteria. These features of a neural network are attractive for their integration into the RSA algorithm towards learning data integrity and data authenticity (Meletiou, Tasoulis & Vrahatis, 2002).

This work proposes, as a baseline to further studies and proof of concept, the design, and implementation of a CIA compliant cryptosystem based on the RSA algorithm which incorporates a neural network to learn data integrity and data authenticity. This is a proposal to improve the RSA model to CIA compliance. While the RSA algorithm, on its own, ensures data confidentiality (Goshwe, 2013), it lacks evident aspects for data integrity checks and data authentication. The main contribution of this study will be around those improvements to the RSA algorithm towards the creation of a CIA compliant hybrid RSA model with a neural network embedded.

1.1 Statement of the problem

The statement of the problem addressed by this study is, primarily, an investigation of ways of integrating the RSA algorithm with a neural network system that learns data integrity and data authenticity in RSA encrypted data towards a CIA compliant hybrid RSA model. It is envisioned that the proposed neural network will learn patterns related to the origins of data and the integrity of data encrypted using the RSA algorithm. The neural network is hoped to predictively associate incoming cipher information (on the receiver's end) with its source. The neural network is also expected to predictively verify any chances that data was modified or tampered with. In our understanding, these improvements are substantially novel in the related body of knowledge.

1.2 Aim

This study aims to investigate improvement to the RSA algorithm by incorporating a neural network which will learn data integrity and data authenticity towards a CIA-compliant hybrid RSA model.

1.3 Objectives

Guided by the aim of the study, three objectives are of interest as follows:

- a) To implement the RSA algorithm.
- b) To design and embed a neural network, into the RSA algorithm, which learns data integrity and data authenticity patterns in RSA encrypted data.
- c) To evaluate the performances of the CIA-compliant hybrid RSA model.

1.4 Questions

Steered by the aim and objectives of the study, three questions are asked in this study which we seek to answer upon completion of this study. The three questions are:

- a) How do we design and implement the RSA algorithm? The answer to this question is a piece of code that implements the RSA model.
- b) How do we design and embed a neural network into the RSA model to learn data integrity and data authenticity patterns? The answer to this question is another piece of code showing the steps we follow in designing a neural network and its incorporation into the RSA model.
- c) To what extent does the hybrid RSA model satisfy CIA compliance? The answer to this question is the evaluations administered on the hybrid RSA model. The performance results yielded from these evaluations are statistically analyzed.

1.5 Hypothesis

The main hypothesis driving this study is that:

- *H₀ - incorporation of a neural network into the RSA algorithm will yield an upgraded hybrid RSA model which satisfies three of the five objectives of cryptographic algorithms - data confidentiality, data integrity, and data authentication.*

We test this hypothesis through experimental processes driven by a tailor-designed simulated RSA model integrated with a neural network. The aim is to demonstrate RSA based data encryption, data integrity, and data authentication.

1.6 Motivation

The undertaking of this study was motivated, mainly, by two factors as follows.

- Data hiding through encryption is key in minimizing the risk of attacks by intruders, especially in the field of data science. There are several data hiding methods. However, most methods emphasize only on data confidentiality, ignoring data integrity and source authentication. The RSA algorithm also ensures only data confidentiality (Goshwe, 2013). Data integrity and user authentication are blurred or implicitly inferred. We are inspired by the desire to assess application of a neural network in the improvement of the RSA algorithm towards explicitly achieving data confidentiality, data integrity, and data authenticity towards CIA compliancy.
- The idea of bringing together cryptography and machine learning is creative and inspiring. Cross-disciplinary research is particularly encouraged because it yields outcomes more than the sum of the

contributions of the individual contributing disciplines. We are curious to assess the extent to which bringing these two branches of Computer Science together can add content to the body of knowledge. We are inspired by the desire to assess the performance of a neural network in tracing data integrity and data authenticity patterns in RSA encrypted data. In our view, these are innovative interventions in the fields of cryptography and machine learning.

1.7 Contributions

Three contributions are envisioned to arise from this study as follows:

- The primary contribution of this study is the creation of additional literature to the body of knowledge. New content is created in the fields of cryptography and machine learning. Innovative and creative integration of the RSA algorithm and a neural network is new content that will be of value to future researchers in these fields.
- Another contribution arises from the practical and commercial perspectives. Successful improvement of the RSA algorithm by incorporating a neural network towards achieving a CIA compliant hybrid may, potentially, bring the RSA algorithm closer to organizational demands and many governments' expectations. These improvements may directly benefit entities where CIA features are indispensable, such as the Department of Home Affairs, Department of National Defense, and the Department of State Security. Similarly, the banking sector and the entire e-commerce industry may prefer CIA-compliant cryptosystems.
- The last contribution is born from the notion that hybrid models achieve much more than the parent component units. The idea of bringing together the RSA algorithm with a neural network brings about strength

in the product cipher thereto. Successful completion of this work, thus, directly adds hope to all stakeholders regarding winning the war against hackers (Aiguokhian, 2013). The work, therefore, further contributes goodwill to cryptosystems especially when CIA compliance is mentioned.

1.8 Location of the study

This study solves a problem in the data security domain, particularly looking at aspects related to implementation and achievement of data confidentiality, data integrity, and user authentication in one goal. We hope to be able to design and implement a predictive cryptosystem based on the RSA model and a neural network, where data confidentiality remains the role of the RSA model, while data integrity and data authenticity are handled by a neural network.

This work is part of a study towards a Master of Science degree. All simulations and experimental works will be completed at the University of Venda on a personal laptop. The data generated through simulations will be stored at the University of Venda students' data repository. However, this data is simulated and generated by our system. It does not have any major ethical implications.

The results of this work are software codes that implement the RSA model and incorporation of a neural network into the RSA model. This work may inspire commercial developments in various departments in South Africa. It may inspire policy reviews. However, testing the applicability of the hybrid RSA model in these other practical contexts is outside the scope of this study.

1.9 Limitations

Two limiting factors are noted in this study as follows.

- The computational speed of available hardware is a challenge. Precisely, systems' performance will drop drastically when large numbers

associated with the RSA model are computed. Some programming languages will not even be able to handle such data types. Some programming languages will freeze when computations give numbers out of range. This is a challenge we will face in the implementation of the RSA model. Nevertheless, we intend to exploit the use of the random library in Python in order to tackle this challenge.

- The time allocated for completing this work is a challenge. A lot of sideline investigations are required before this project is completed. We cannot complete all possible aspects to include in the study. Rather, we will concentrate on proving the concept of incorporation of a neural network in the RSA model.

1.10 Overview

The rest of the project proceeds as follows:

- Chapter 2 will present related works, emphasizing on how the RSA algorithm has been improved in the past.
- Chapter 3 will give the methodology we follow towards solving the research questions. It clarifies the design and implementation issues, presenting the key routines of the proposed cryptosystem, unit testing methods, and the proposed integration strategies with which units are put together into one predictive cryptosystem.
- Chapter 4 will present the experimental setup, the research design, the results achieved, and the scientific analyses and discussions of the results. It thoroughly explains the meaning of the results.
- Our observations, reflections, contributions, conclusions, and recommendations are presented last in chapter 5, also highlighting potential future works emanating thereto.

1.11 Summary

This chapter set the ball rolling by introducing the study. It introduced the statement of the problem as an investigation of ways of integrating the RSA algorithm with a neural network system that learns data integrity and data authenticity in RSA encrypted data towards a CIA-compliant hybrid RSA model. The aim of the study was pinpointed as; to investigate improvement to the RSA algorithm by incorporating a neural network that learns data integrity and data authenticity towards a CIA compliant hybrid RSA model.

Three objectives were pinpointed as key. Precisely, the study seeks to implement the RSA algorithm. It also pursues the design and embedment of a neural network into the RSA algorithm to learn data integrity and data authenticity patterns in RSA encrypted data. The work then evaluates the performances of the proposed CIA-compliant hybrid RSA model.

Three questions were asked in posed in this chapter, all aligned to the objectives of the study. The first question seeks to understand how the RSA algorithm is designed and implemented. The second question seeks to understand how a neural network is designed and incorporated into the RSA model to learn data integrity and data authenticity patterns. The final question assesses the extent to which the hybrid RSA model satisfies CIA compliancy.

The chapter stated the null hypothesis that drives the entire study. We believe that: *H₀ - incorporation of a neural network into the RSA algorithm will yield an upgraded hybrid RSA model which satisfies three of the five objectives of cryptographic algorithms - data confidentiality, data integrity, and data authenticity.*

Two motivating factors for undertaking this study were singled out. Precisely, the desire to assess the application of a neural network in the improvement of

the RSA algorithm towards explicitly achieving data confidentiality, data integrity, and data authenticity towards CIA compliance will drive this study. Also, curiosity to assess the extent to which bringing a cryptography model and a machine learning system together further propels this study. That desire to assess the performance of a neural network in tracing data integrity and data authenticity patterns in RSA encrypted data is a motivating factor noted in this chapter.

Three contributions are envisaged. It is hoped that the work will create literature and new content in the respective fields. It is also hoped that successful improvement of the RSA model incorporating a neural network towards CIA compliance will respond to most organizations' needs. Generally, the hybrid model will be stronger. This adds goodwill to cryptosystems.

The location of the study was indicated from a conceptual perspective, along with the limitations of the study. Two limitations were pointed out. The first pointed to computational performance during computation with large numbers. The second limitation has to do with the time allocated for the completion of this research work. The overview of the study culminated the work presented in this chapter.

Chapter 2 : Literature review

This chapter reviews literature related to the field of cryptography where improvements to the RSA algorithm have been connoted towards CIA compliance. Mainly, the RSA model is unpacked, and reviews related to how the RSA model has addressed data confidentiality, data integrity, and data authenticity in the past are presented. The computational challenges noticed in the RSA model are discussed before any attempts to move the RSA model towards CIA compliance are deliberated.

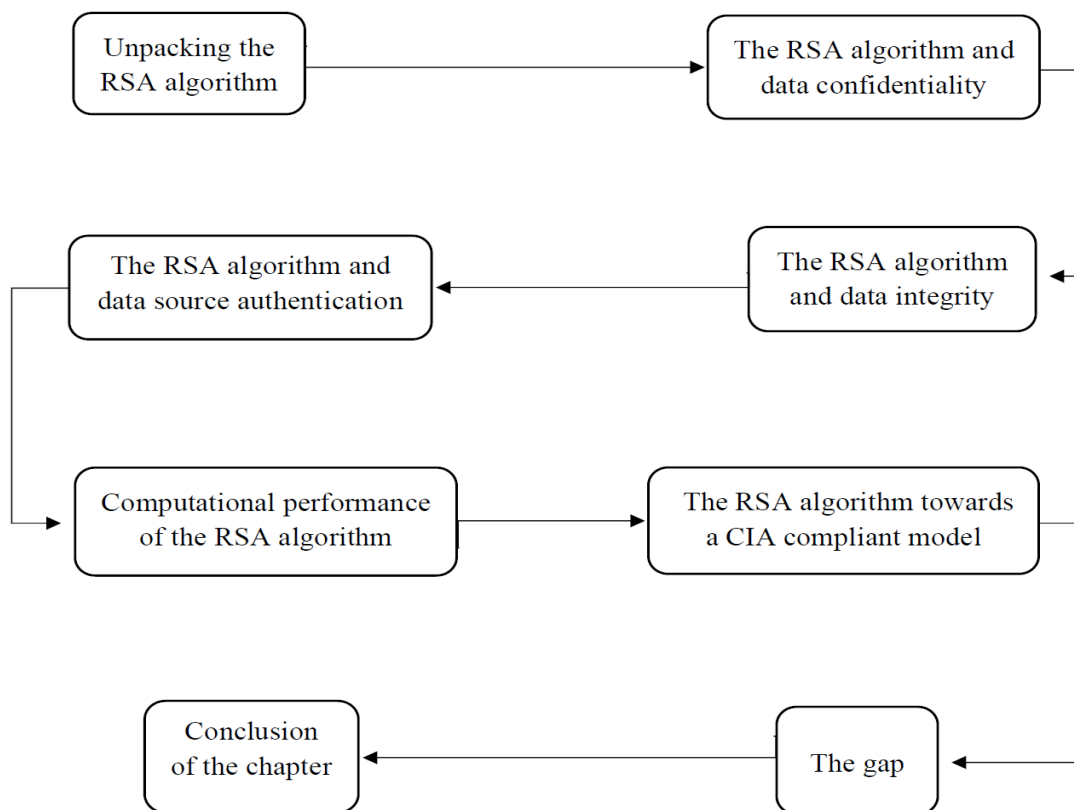


Figure 2.1: Overview of the chapter.

We also review literature in which neural networks have been considered for improving cryptosystems. Our basis for making most of the conclusions we arrive at in this research study is grounded on the literature reviewed in this chapter. The gap our work seeks to fill in the body of knowledge is presented at the end of this chapter. Figure 2.1 shows the layout of this chapter in pictures.

2.1 Components of the RSA algorithm

We already mentioned that the RSA algorithm is asymmetric (Bhanot & Hans, 2015). Asymmetric algorithms use two keys, a public key that is used for encryption and a private key that is used for decryption (Stallings, 2006). We also mentioned that communicating parties can share the public key, but they cannot share the private key. In this context, data encryption refers to a mechanism with which to transform a message, known as a plain text, into an alternative form called a ciphertext (Goshwe, 2013). Decryption would mean the reverse process (Goshwe, 2013).

The security of the RSA algorithm depends on its internal mathematical properties (Zhou & Tang, 2011). The same mathematical properties are inferred in the generation of the RSA model's public and private keys. Precisely, two large prime numbers, say p and q , are chosen secretly (Bangju & Huanguo, 2006). These two prime numbers are subsequently used to generate other large numbers of interest, including n , e , $\varphi(n)$, and d . In this case, n is the product of p and q . The totient function, $\varphi(n)$ is the product of $(p - 1)$ and $(q - 1)$. The integer e is such that the $\gcd(\varphi(n), e) = 1$. The public key would be denoted as $K_1 = (n, e)$, while the private key would be denoted as $K_2 = (p, q, \varphi(n), d)$ (Miller & Trbovich, 1982). Note that the value of d is such that $d \times e \bmod \varphi(n) = 1$.

2.2 The RSA algorithm and data confidentiality

We indicated that the RSA algorithm emphasizes on data confidentiality (Alhassan, Ismaila, Waziri & Abdulkadir, 2016). Although the RSA algorithm is powerful in this regard, it lacks aspects to probe data integrity and data authenticity. This is not the first time these flaws and vulnerabilities have been noted (Pfleeger & Pfleeger, 2012). Precisely, the possibilities of factorizing the public key components into picking the prime numbers used to compute the public key elements, then the private key components such as the totient function, is worrying. Hackers may tamper with the data integrity of RSA encrypted messages (Nguyen, 2009) once the key components are identified. Adding data integrity and data authenticity to RSA encrypted data is, thus, an apparent gap in the field.

Other common issues raised about the RSA algorithm have to do with the possibilities of guessing the values of the private key component d - a private value (Al-Hamami & Aldariseh, 2012; Van Tilborg & Jajodia, 2014) used in decoding encrypted messages. Once correctly guessed, encrypted data can be decoded, modified, or masqueraded. Attempts have been made in the literature towards modifying and improving the RSA algorithm to bring in aspects of data integrity and data authenticity. This would qualify the RSA model as a CIA-compliant hybrid model. However, no tangible results are available in the literature.

Another improvement to the RSA algorithm considered the use of algebraic finite fields towards achieving data confidentiality on wireless networks (Frunza & Scripcariu, 2007). Key generation, in their case, considered the use of two keys instead of the use of the traditional one key for encryption. This intervention reduced the chances of brute force attacks on the private key

(Jahan, Asif & Rozario, 2015). However, the algorithm's CPU time and RAM demands escalated.

The strength of the RSA algorithm was also enhanced by increasing or varying the key size (Amalarethinam & Leena, 2017). However, that on its own increased the CPU time the algorithm required to complete encryption and decryption processes (Amalarethinam & Leena, 2017).

Concepts from quantum physics have also been considered and merged with the RSA algorithm to detect third-party eavesdropping in internet communication (Odeh, Elleithy, Alshowkan & Abdelfattah, 2013; Plesa & Mihai, 2018). Although this intervention improved the RSA algorithm with regards to the data confidentiality aspect, data integrity and data authenticity were still ignored, leaving the RSA algorithm still not CIA-compliant.

Other attempts to improve the RSA algorithm considered the use of the cubic congruential generator algorithm which, rather, improved the robustness of the RSA algorithm (Khairina & Harahap, 2019). This still did not touch on aspects related to data integrity and data authenticity. Instead, the claimed robustness remained on data confidentiality.

Also, a hybrid encryption algorithm based on the RSA and the data encryption standard (DES) was considered towards enhancing the security of data during transmission in Bluetooth communication (Frunza & Scripcariu, 2007; Ren & Miao, 2010; Singh, 2013). This, again, enhanced data confidentiality, ignoring data integrity and data authenticity.

Contrary, a convertible authenticated encryption scheme has been proposed and incorporated into the RSA algorithm to add data authenticity and non-repudiation (Wu & Lin, 2009; Lin, Hsu & Huang, 2011). This is inspiring as it directly informs our intention to improve the RSA in similar directions.

Nonetheless, in our case, we do not only consider data authenticity, but also data integrity on top of data confidentiality.

Many other works looked at modifying the RSA algorithm towards strengthening its robustness in terms of binary code format (Aiswarya, Raj, John, Martin & Sreenu, 2016). However, the Diffie-Hellman key exchange algorithm brought more hope and better improvements to the RSA model when it handled the sharing of a public key on an open communicating channel (Bodkhe & Jethani, 2015). This improvement also directly informs the hypothesis we bring into our study. Nonetheless, all the improvements noted in the literature practically re-emphasized improving data confidentiality, a bit of data authenticity, and non-repudiation, completely leaving out data integrity towards a fully CIA-compliant hybrid model. Our work seeks to fill that gap, hence worthwhile.

2.3 The RSA algorithm and data integrity

In responding to calls for improvement of the RSA algorithm to incorporate data integrity, various existing techniques were considered. On top of the list was an identity-based cloud data integrity checking protocol that was used to verify the integrity of the public key generated by the RSA algorithm (Yu, Xue, Au, Susilo, Ni, Zhang, Vasilakos, & Shen, 2016). Instead of emphasizing the integrity of data, the model stressed the integrity of the public key shared (n , e). The rationale of their study was that it is critically essential to verify the RSA algorithm's public key before it is used in data encryption. This was a milestone. However, we are more concerned about the integrity of RSA encrypted data rather than the integrity of the public key.

Issues to do with data integrity in the RSA model also emanated from the design of the message digest hashing algorithm. In this, the RSA partial

homomorphic algorithm was incorporated to maintain data confidentiality and data integrity in cloud computing (Ora & Pal, 2015). In this case, the RSA algorithm handled data confidentiality, while the message digest hashing algorithm kept responsible for the verification of data integrity (Ora & Pal, 2015). Although this intervention directly informs the work we present in this study, these improvements still left out data source authenticity towards a CIA-compliant hybrid RSA model.

Compelling is mediations brought in when the RSA algorithm was combined with the AES (Kuswaha, Waghmare & Choudhary, 2015). In this combination, the RSA model provided data confidentiality, and the AES further encrypted already encrypted outcomes. This intervention neither looked at data integrity nor data authenticity. Rather, a product cipher arose which still emphasized data confidentiality. The inclusion of data integrity and data authenticity in the RSA model is the gap we seek to fill in the body of knowledge.

Interventions through the use of the dynamic Merkle hash tree, at least, took the RSA model one step forward towards CIA compliance when dynamic data integrity was considered (Saranya, Usha & Alex, 2017). This was done for security assurance when recovering lost data blocks in the cloud storage (Saranya, Usha & Alex, 2017). These improvements are relevant to our work. This research study is aimed at bringing together the three aspects of data security into the RSA algorithm by incorporating a neural network that would learn data integrity and data authenticity in RSA encrypted messages.

2.4 The RSA algorithm and data source authentication

Data source authentication has been a challenge in the cryptography domain for a while. How do we avoid denial of action? How do we avoid non-

repudiation? How do we get data digitally signed? These questions have not yet been fully answered, especially where the RSA algorithm is involved.

A hybrid algorithm called the elliptic curve RSA was proposed to deal with the issue of data authentication (Krishnamoorthy & Perumal, 2017). This is a hand-off authentication protocol that was used in the roaming of mobile nodes (Krishnamoorthy & Perumal, 2017). However, this intervention only dealt with data authentication, ignoring data confidentiality and data integrity. The outcome hybrid algorithm was, therefore, not CIA compliant.

A four-layered authentication stack has also been proposed and used to improve the RSA algorithm towards a data authenticating hybrid. This is when and where the techniques of password, external digital certificates, and a third party in data authentication emanated (Bhattacharjya, Zhong & Li, 2019). In this case, the four-layered authentication stack enforces a built-in way of identifying the source of RSA encrypted messages. This is inspiring. However, the works discussed here only emphasized the issue of tracing the source of the data (data authenticity), while on the other hand ignoring, the quality of the data (data integrity) as well as data confidentiality. We want a system in which the three objectives are all tackled at once.

Further attempts to improve the RSA algorithm led to the adaption of the handshaking theorem on an extensible authentication protocol framework to ensure the security of client data in the cloud (Marium, Nazir, Ahmed, Ahthasham & Mirza, 2012). In this, the origin of data was considerably verified, hence the name extensible authentication protocol framework. However, again, only data authentication is considered, ignoring data integrity. We propose an improved RSA model which achieves data confidentiality, data integrity, and data authentication on one goal.

The digital signature technology slotted into the RSA model allowed proving that medical images are authentic or not (Smith, 1995; Gola, Gupta & Iqbal, 2014; Sadikin & Wardhani, 2016). The same technology has been in use in the security circles related to the transfer of funds. Key in these attempts is authentication of the data source, thereby preventing non-repudiation. Nonetheless, the idea of bringing all three aspects of data security at once, towards CIA compliant hybrid models was not tackled.

Tracing the source of data in an RSA model is an ongoing subject for study in cryptography (Khan, Pervez & Abbasi, 2017). Tracing the source of data curbs most possible data security breaches related to non-repudiation (Venkatraman & Overmars, 2019). The introduction of an improved RSA algorithm based on the use of prime number factorization to protect the newly introduced field, the internet of things, from data security breaches, provided a thorough explanation of the vulnerability of the keys generated by the RSA algorithm (Andrea, Chrysostomou & Hadjichristofi, 2015; Venkatraman & Overmars, 2019). However, CIA compliancy still lacked. This research study tries to close that gap by bringing together the RSA model and a neural network.

2.5 Computational performance of the RSA algorithm

Improvements to the RSA model require us to also pay attention to issues around the computational performance of the model. Generally, the RSA algorithm degrades in performances when large integers are used to generate the related keys (Nozaki, Motoyama, Shimbo & Kawamura, 2001). As a result, Montgomery multiplication based on residue number systems has been proposed to remedy such speed issues (Nozaki et al., 2001). However, still, no explicit analysis was made regarding speed issues versus CIA compliancy.

On the other hand, the El-Gamal algorithm has also been proposed in combination with the RSA algorithm to enhance the computational speed of integer factorization during the encryption process (Ahmed & Ali, 2011). A similar merge of the RSA algorithm with the algorithmic OpenMP also came in towards improving execution time when computing large prime numbers in parallel (Saxena, Jain, Singh & Kushwah, 2017; Ayub, Onik & Smith, 2019). Exploration of speed issues continued when attempts became visible when the elliptic curve method hardware engines were used to improve the massive, large prime numbers computation and cost time product of the RSA algorithm's moduli factorization by the general numbers field sieve (Cavallar et al., 2000). However, still, these interpolations did not bring aspects of CIA compliance in the RSA model. Our work seeks to fill this gap.

One other weakness of the RSA algorithm has been identified as time-consuming during modular exponentiation computation when a plaintext undergoes the encryption process. To compact this problem, Kumaravel and Marimuthu (2007) introduced a concept that enhances the RSA algorithm by using the Indian Vedic mathematics. This is a good view to embrace in our study. Great consideration will be made towards adopting some of the good views purported in Indian Vedic mathematics.

2.6 The RSA algorithm and CIA compliance

Prevalently, improvements to the RSA algorithm have not achieved CIA compliance. Fault attacks have, mainly, been considered along with the incorporation of the Chinese remainder theorem in smartcards (Blömer, Otto & Seifert, 2003). Similarly, the hash function, together with the RSA algorithm, has been used to check the correctness of the users' data in mobile cloud computing (Garg & Sharma, 2014). These improvements pointed to the RSA algorithm realizing digital signature schemes, especially in secure electronic

health record applications (Gola, Gupta & Iqbal, 2014; Sadikin & Wardhani, 2016). These improvements connoted embracement of data integrity as well as data authenticity at the same time. However, this was never formalized as an aspect of CIA compliance in the RSA model.

Similar attempts to bring data integrity and data authenticity into the RSA model are also visible in the work of Blömer et al. (2003), where checks have been made regarding whether there were faults when the cryptosystem was tested (Blömer et al, 2003). The RSA algorithm was modified to focus on storage security to assure and check the correctness of the data stored on the cloud service (Venkatesh, Sumalatha & SelvaKumar, 2012). A blowfish algorithm was adapted in cloud computing to merge with the RSA algorithm to perform the modules of authentication and integrity of data files stored on the cloud (Syam & Subramanian, 2012; Yamuna & Anusha, 2015). These attempts inform our study as well. Nevertheless, they did not explicitly connote CIA regulations. In continuation, Mishra, Singh, and Ali (2018) introduced an improved RSA model based on cross-domain secure deduplication to minimize the possibilities of security breaches and making it easier to add new capacity. However, still, data confidentiality, data integrity, and data authenticity aspects were looked at separately, ignoring the need to comply with all these three aspects of data security at once. Our aim in this study is to investigate improvement to the RSA algorithm towards a CIA compliant hybrid model by incorporating a neural network that learns and reports data integrity and data authenticity.

2.7 Interventions in which neural networks are considered

The inclusion of a neural network in the design of a cryptosystem is a powerful idea. There is various research that looked at an angle of incorporating a neural network into cryptographic algorithms. Cryptographic algorithms are aimed at

avoiding the occurrence of cryptanalysis. Alallayah et al. (2012) have adopted the idea of bringing a neural network closer to cryptosystems. For example, a neural network has been incorporated in the data encryption standard (DES) by combining a mathematical black-box model and a system identification technique with an adaptive system technique, to create the Neuro-Identifier that would combat the problem of cryptanalysis in this cryptographic algorithm. This would be achieved by first revealing the encryption algorithm and its key from the given plaintext and ciphertext pair. However, this work dwelt too much on the violation of the three cryptographic algorithm objectives which are data confidentiality, data integrity, and data authenticity. Nevertheless, our work aims at preserving these three objectives into one cryptographic algorithm, the RSA algorithm.

Attempts have been made to consider the training of a neural network model over encrypted data by using the emerging functional encryption scheme instead of homomorphic encryption or secure multi-party computation for data security (Xu, Joshi & Li, 2019). Practically and conceptually, their work aimed at tackling the problem of data confidentiality and data integrity, leaving out the aspect of data authenticity. The idea here was similar to ours. However, their work ignored other objectives of cryptographic algorithms. Our work considers tackling the inclusion of all three objectives at once in a single cryptographic algorithm.

In general, we indicated that the RSA algorithm degrades in performances when large integers are used to generate the related keys (Nozaki, Motoyama, Shimbo & Kawamura, 2001). To tackle the problem of computational performance in the RSA algorithm, Chakraborty et al. (2018) proposed a model that analyzed the performance of an artificial neural network based on the RSA technique towards execution time of the RSA algorithm's large integers for key

generation. However, these interventions ignored the inclusion of the aspects of CIA compliance into the RSA model or any other cryptosystem. Nevertheless, our work seeks to fill this gap, mainly, to create a CIA-compliant hybrid model built on the incorporation of a neural network into the RSA model.

2.8 The gap

We noted that the RSA algorithm alone responds to the data confidentiality problem. We also noted that attempts to improve the RSA algorithm are a niche research area. More interestingly, considerations to aspects of data integrity and data authentication are higher on the list of desired improvements to the RSA model. To the best of our knowledge, these attempts have not, as yet, arrived at the prescription of a new version of the RSA model that addresses all three aspects of data security (data confidentiality, data integrity, and data authenticity) towards a CIA compliant RSA hybrid. This is the apparent gap we intend to fill in the field of cryptography's body of knowledge.

A neural network is considered. It is brought forward to learn data integrity and data authenticity in RSA encrypted data. Precisely, the neural network is meant to learn the sources of data, as well as the data's original form before it was encrypted. The aim of learning the source and the form of data before encryption is to allow the neural network to verify such data integrity and data authenticity when the data arrives at the receiver's end. The neural network verifies an RSA encrypted piece of data's integrity and authenticity at the receiver's end. Once this verification is completed, RSA encrypted data would comprise data confidentiality, while also satisfying data integrity and data authenticity. The RSA model would then be regarded as CIA compliant. This is the precise gap our research work intends to fill in the body of knowledge.

2.9 Summary

This chapter commenced with an introduction and a flowchart describing an overview of what we reviewed. A detailed explanation of the RSA algorithm came first in section 2.2, emphasizing the component units of the RSA algorithm and mainly key generation.

The link between the RSA algorithm and aspects of data confidentiality was dwelt on in section 2.3, emphasizing on literature that attempted to improve the RSA model along with the data confidentiality angles.

Section 2.4 worked on the improvements connoted from the literature around aspects of data integrity on the RSA model. Conclusions were derived which pointed to a lack of RSA models which combined data confidentiality, data integrity, and data authenticity. This review gave pointers to the gap this study seeks to fill in the body of knowledge.

Section 2.5 then reviewed works in which attempts were made to bring aspects of data authenticity into the RSA model. Similarly, conclusions were arrived at which indicated a lack of literature which brought together the three objectives.

In section 2.6, we dwelt on literature that looked at the computational flaws of the RSA model. Attempts to bring data confidentiality, data integrity, and data authenticity together are shared in section 2.7. This literature was found inspiring and informing our study.

On the contract, section 2.8 reviewed interventions towards bringing in neural networks in cryptography. It was noted that this was a novel idea in the context of cryptology. The gaps we seek to fill were elucidated in section 2.9 before the chapter closed in this section 2.10. The next chapter presents the methodology and theoretical framework we embrace in this study.

Chapter 3 : Methodology and theoretical framework

The choice of incorporating a neural network into the RSA model is mainly motivated by the desire to improve the RSA algorithm to include, on top of data confidentiality, also data integrity and data authenticity towards a CIA compliant hybrid RSA model. The purported improvements require us to code the RSA algorithms separately before the proposed integration with a neural network is undertaken.

This chapter explains the procedures we followed in implementing the RSA algorithm and the integration subsystem. It explains, consequently, how the RSA algorithm is merged with a neural network to create the proposed hybrid RSA model. The explanation of the theoretical framework on which this research is underpinned is also included in this chapter.

3.1 Statement of the problem

The statement of the problem addressed in this chapter can be summarized into three sub-questions as follows:

- a) What is the methodology that informs the development of this project?
- b) What is the theoretical framework that underpins the argumentation and reasoning followed in this study?
- c) What are the methods used for data collection, reporting, and data analysis?

In our views, answers to these three questions would summarize the methodology followed and the theoretical ground for the reasoning presented in the study.

3.2 Overview of the chapter

Key in this chapter is the description of the software tools employed in the development and integration of the RSA algorithm and a neural network towards a CIA-compliant predictive cryptosystem. The procedures and software tools described in this chapter are all aimed at yielding a hybrid RSA model that satisfies data confidentiality, data integrity, and data authenticity.

We will first explain the theoretical framework on which the reasoning presented in this study is grounded. Requirement elicitation and how each algorithm works will be explained hereafter. Precisely, we look at the research design, emphasizing how the RSA model was implemented, how the neural network was implemented, and how the integration problem was tackled. We then look at the data samples, sampling procedures, as well as the tools employed for arriving at data-based conclusions. The data analyses of interest and the key statistics we extract are also presented in this chapter.

3.3 Theoretical framework

Design science research is the theoretical framework on which this study is built. Design science research is a set of synthetic and analytical techniques and perspectives for performing research in information technology (Hevner & Chatterjee, 2010). It presents a large opportunity to increase the relevance of research in the field of Computer Science (Nunamaker, Dennis, Valacich, Vogel & George, 1991; Hevner & Chatterjee, 2010). Mainly, design science research addresses unsolved and important problems in unique and innovative ways (Hevner & Chatterjee, 2010), solving problems in more efficient ways as

opposed to routine design. Achievement of knowledge when design science research is connoted is based on the foundations and strategies used (mainly inspired by the Boehm spiral model). In these strategies, positivism is dominant, requiring scientific evidence to argumentations and reasoning.

Adoption of the design science research methodology is mainly motivated by the desire to improve the RSA algorithm. Such improvements are based on the incorporation of a neural network into the RSA towards adding data integrity and data authenticity aspects. The introduction of new and innovative artifacts and the processes for building these artifacts are the key activities of this study (Simon, 1996). There are three cycles associated with design science research namely, relevance, design, and rigour.

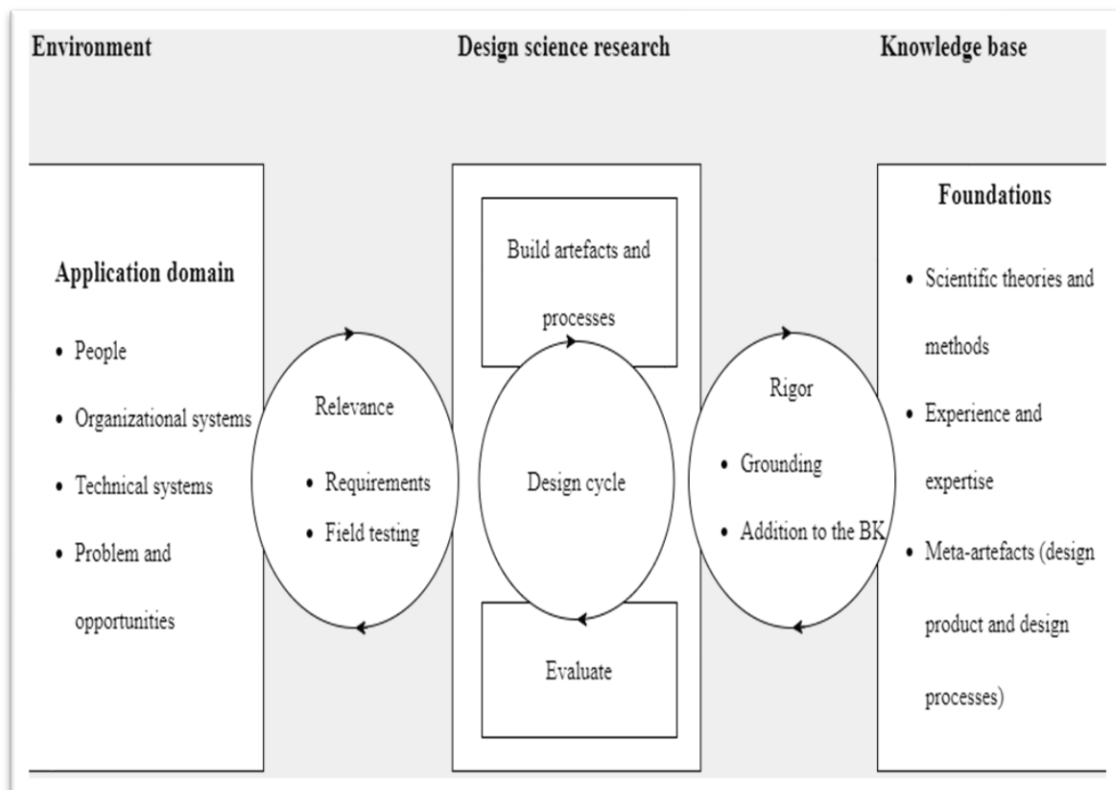


Figure 3.1: Design science research cycle (Source: Hevner, 2007).

The relevance cycle initiates design science research with an application context that not only provides the requirements for the research as inputs but also defines acceptance criteria for the ultimate evaluation of the research results (Hevner, 2007). The central design cycle then follows. This is the core and heart of every design science research study. Simon (1996) describes the nature of this cycle as generating design alternatives and evaluating these alternatives against the requirements until a satisfactory design is achieved. The rigour cycle then provides grounding theories and methods along with domain experience and expertise from the foundational knowledge base into the research (garnered from literature in the community or body of knowledge). It adds the new knowledge generated by the research to the growing knowledge base of a research community (Hevner, 2007). Figure 3.1 illustrates the processes of each cycle in the design science research paradigm.

The key activities in the design science research framework are theory building and evolution. It involves solution technology invention activities that bring about information technology artifacts involving information system development methods, techniques, tools, algorithms for computer processing, and planning methods among others (Venable, 2006). The design science research framework is not necessarily concerned with methods of testing theories. Rather, it is more concerned with bringing more knowledge in the field. Such new knowledge refers to constructs, models, methods, instantiations, and more improved theories towards the literature. The knowledge that the design science research framework brings forward can provide vision and complete guidelines to newcomers in the field. The findings should provide a thorough and complete statement of the outcomes. These outcomes should be tested for validity. They should potentially be improved by other researchers. Design science research is a provable theory for checking whether the solution is consistent. It verifies the performance of the solution

discovered. A framework for theory and theorizing (needed for evaluation) is thus apparent. The theory thereto will be the link between researchers and different research activities over time. This is a central activity that ties in various areas of research (Venable, 2006).

The theory that guides the building or formulating of this research study corresponds to that of prescriptive information system design theory which makes use of utility theory (Venable, 2006). There are three major concepts in the utility theory, namely, the problem space, the solution space, and utility theories. The problem space channels the researcher for a thorough understanding of the problem domain. We do the same. The solution space gives more details on the solution techniques used by the researcher in addressing the problem. The utility theories deal with the implementation of the proposed solution method. Our study embraces all these spaces.

3.4 Research design

Research design describes the overall approach used to conduct a research study. It defines a concise and logical plan to tackle the research question (s). Precisely, procedures for data collection, data interpretation, data analysis, and the discussions thereof are presented (Yoshikawa, Weisner, Kalil & Way, 2008). In this case, the research design explains the procedure used in the development of the predictive cryptosystem built towards improving the RSA algorithm through the incorporation of a neural network to learn data integrity and data authentication. The algorithmic and mathematical procedures followed in the development of these two algorithms, the original RSA and a neural network are the key routines we discuss.

This is a quantitative research study in which simulated data (related to the performance of a neural network in predicting data integrity and data

authenticity) is collected. Quantitative research is mainly concerned with objective, measurable, and repeatable processes (Yoshikawa et al, 2008). Precisely, we investigate the performances of the proposed hybrid model.

Quantitative research tests hypotheses. The null hypothesis of this research is that no significant changes would be observed in the RSA model even when a neural network is incorporated to learn data integrity and data authenticity in RSA encrypted data. Two directional alternate hypotheses emanate, one suggesting significant upgrade, and another pointing to degraded performance.

A positivist philosophy is apparent, where knowledge, reasoning, and argumentations are grounded in deductive phenomena (Bechtel, 2013). Precisely, positivism and deductive reasoning are, fundamentally, methods aimed at arriving at conclusions grounded in scientific theories, laws, and proofs (Bechtel, 2013). The purported positivist beliefs and deductive argumentation and the reasoning thereto will be supported by experiments. Experiments are scientific approaches for creating observable proofs, investigating the validity of theories, or testing scientific facts (Windschitl, Thompson & Braaten, 2008). Experiments will be used for generating data with which to test the hypothesis and its alternates. Precisely, experiments will provide data upon which conclusions will be derived.

3.4.1 Proposed methods

The development of two algorithms is the fulcrum of this study. The first algorithm to develop is the RSA model. The other one is a neural network to be incorporated into the RSA model. We discuss the methods followed in developing these two algorithms before we look at how they were combined into the proposed hybrid RSA model. A discussion around the implementation of each of these two algorithms would address the problem posed.

3.4.1.1 The RSA algorithm

We indicated that the RSA algorithm is widely used to provide data confidentiality. The algorithm is quite visible in the commercial space (Boneh, 1999). It is a globally renowned public-key encryption algorithm. Security protocols such as the TLS/SSL, transport data security (web), PGP email security, IPSEC/IKE IP data security, SILC conferencing service security, and SSH terminal connection security are all based on the RSA model. Its importance and the need to improve is quite apparent in the body of knowledge.

Implementation of the RSA algorithm is based on the use of two prime numbers (Bakhtiari & Maarof, 2012), say p and q . The two prime numbers are multiplied together to give one key component of the public key, which we here denote as n . This n , the product of p and q , is a very important part of the RSA algorithm both on the public and private sides. Encryption is completed in modulo n . Decryption also uses the same modulo n . As a result, the main threat against the RSA algorithm is around decoding of the two prime numbers through factorization from n . Once p and q are decoded, the RSA algorithm is broken into because the private key can be found.

The public key emanating is (n, e) . The private key would be $(p, q, \phi(n), d)$. Note that the public key is broadcast to everyone who wants to communicate hidden data. The private key remains secret to the receiver of hidden data.

The Rabin-Miller algorithm is used during the RSA algorithm to determine if the provided p and q are prime numbers. Preferably, p and q are likely odd numbers unless one of these two parameters is a 2. However, the use of 2 as one of the two prime numbers p or q is discouraged because of the simplicity of factorizing the n that arises thereto.

Algorithm 3.1: The RSA algorithm.

Input: Data set D .

Output: Encrypted data set E .

Internal computation:

1. Generates two prime numbers p and q ,
 - a) Apply algorithm 3.2 on p ,
 - b) Apply algorithm 3.2 on q
2. Calculates $n = p \times q$,
3. Calculates $\varphi(n) = (p - 1) (q - 1)$,
4. Choose e , such that e is a co-prime to $\varphi(n)$ and $\gcd(\varphi(n), e) = 1$,
 - a) Apply algorithm 3.3 on e and $\varphi(n)$
5. Find d , such that $d \times e \bmod \varphi(n) = 1$,
6. The public key would be (n, e) ,
7. The private key would be $(p, q, \varphi(n), d)$,
8. Ciphertext is computed as:
$$E = D^e \bmod n, \text{ and,}$$
9. Plaintext is recovered as: $D = E^d \bmod n$.

Algorithm 3.2: Rabin-Miller algorithm (Rahim et al, 2017).

Input: Any integer, preferably odd.

Output: Primality.

Internal computation:

1. Select a random number p ,
2. Calculate b , where b is the number $(p - 1)$ divided by 2,
 b is, therefore, the largest power of 2, such that $2b$ is a factor of $(p - 1)$,
3. Calculate m , such that $p = 1 + 2^b m$,
4. Choose a random number a such that a is smaller than p ,
5. Set $j = 0$ and set $z = a \times m \bmod p$,
6. If $z = 1$ or if $z = p - 1$, then p passes the test and may be a prime number,
7. If $j > 0$ and $z = 1$, then p is not a prime number,
8. Set $j = j + 1$. If $j < b$ and $z \neq p - 1$, set $z = z^2 \bmod p$ and return to step 4,
If $z = p - 1$, then p passes the test and may be prime, and,
9. If $j = b$ and $z \neq p - 1$, then p is not a prime number.

Algorithm 3.3: Euclid's algorithm (Backhouse & Ferreira, 2011).

Input: Two positive integers, $\varphi(n)$, and e .

Output: The greatest common divisor, $gcd(\varphi(n), e)$.

Internal computation:

1. If $\varphi(n) < e$ exchange $\varphi(n)$ and e ,
2. Divide $\varphi(n)$ by e and get the remainder r ,
3. If $r = 0$, report b as the $gcd(\varphi(n), e)$, Replace $\varphi(n)$ by e and replace e by r , and,
4. Return to step 2.

The Euclidean algorithm is a way to find the greatest common divisor of two positive integers (Shantz, 2001). In the context of this research, Euclid's algorithm is used by the RSA algorithm in the implementation stage, to determine the greatest common divisor of the totient function and the value, e , such that e is a co-prime to $\varphi(n)$ (Zhou & Tang, 2011). The totient function, $\varphi(n)$, is the product of $(p-1)$ and $(q-1)$. On the other hand, the integer e is a primitive root of the totient function such that $gcd(\varphi(n), e) = 1$. There are four algorithmic steps involved in the computation of the greatest common divisor of the RSA's totient function, $\varphi(n)$, and the value of e in Euclid's algorithm.

3.4.1.2 A neural network

We indicated that the RSA algorithm will be enhanced by a neural network. The incorporation of a neural network into the RSA model will enable data integrity and data authenticity towards a CIA-compliant hybrid RSA model. The goal is

to come up with an algorithm that tackles three of the five objectives of cryptography at once.

A neural network is a machine learning algorithm that tries to learn the underlying properties of a certain phenomenon, through a mechanism that imitates the way the human brain works (Agarwal & Agarwal, 2013). In this context, the phenomenon learned by the neural network is data integrity and data authenticity. Successful incorporation of a neural network to learn the RSA's encrypted data integrity and data authenticity will take us closer to a CIA-compliant hybrid RSA model. That will be a creative innovation in the data security domain. Below is the structure of a neural network comprising the input layer, hidden layer, and output layer.

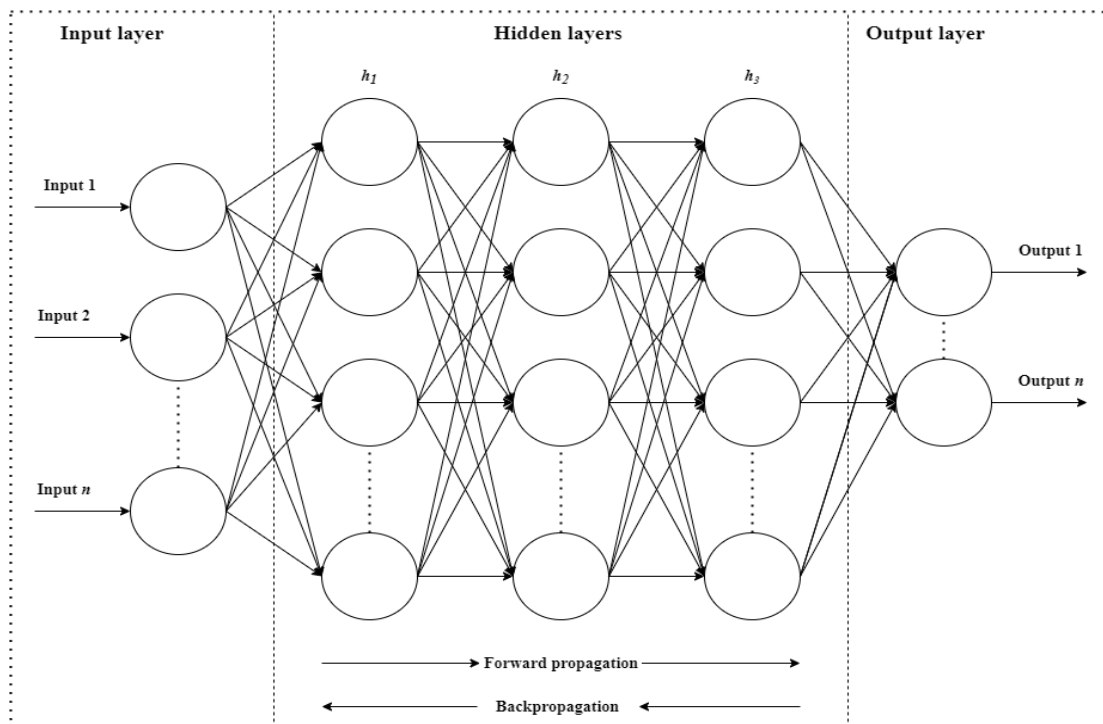


Figure 3.2: A simple neural network.

The input layer is responsible for receiving data into the neural network (Agarwal & Agarwal, 2013). It is the starting point of the neural network. The hidden layer is the subsequent layer after the input layer where input data relating to the RSA's encrypted data is transformed through mathematical procedures and thereafter directed through an activation function to the output layer (Guliyev & Ismailov, 2016). The output layer is responsible for producing the results after learning from the workflow of the neural network (Guliyev & Ismailov, 2016). There are two-layer perceptron involved in a neural network, namely single-layer perceptron and multi-layer perceptron. We discuss each perceptron below.

A single-layer perceptron is the simplest form of a neural network. It consists of only one layer of the input layer that directs the computed inputs to the subsequent layers, the hidden layer, and the output layer (Stengel, 2017). The structure of a single-layer neural network can be visualized as follows.

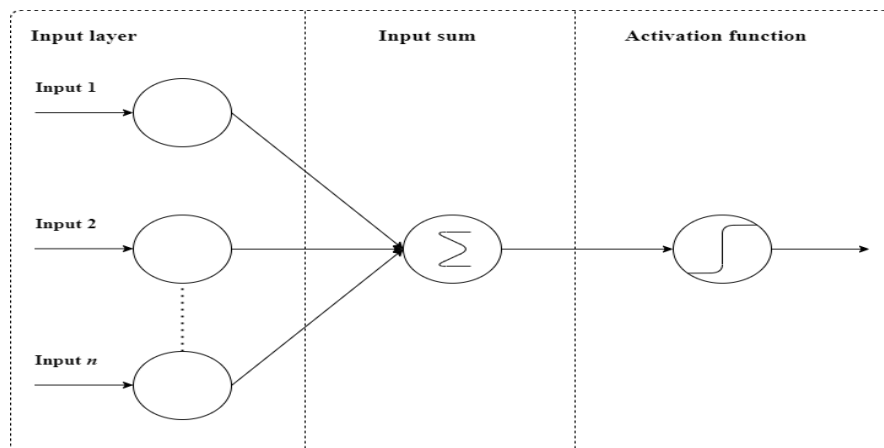


Figure 3.3: Single-layer neural network.

The mathematical expression which describes a single-layer neural network can be written as follows:

$$y_k = g \left(\sum_{i=0}^D w_i x_i \right)$$

where y_k is the output, and $g(.)$ is an activation function. In this function, x_i is the input and w_i represents the corresponding weight of x_i . Although this type of neural network may not be practical, it helps us in understanding the basics of neural networks (Stengel, 2017).

A multi-layer perceptron consists of multiple layers of computational units, usually interconnected in a feed-forward way (Lee & Choeh, 2014). Mathematically, a multi-layer neural network with one hidden layer can be expressed as follows:

$$y_k = h \left(\sum_{j=0}^M w_{kj}^{(2)} g(a_j) \right)$$

where,

$$g(a_j) = g \left(\sum_{i=0}^D w_{ij}^{(1)} x_j \right)$$

A multi-layer neural network is very similar to a single-layer neural network except that a multi-layer neural network's output is again multiplied by a new weight vector and wrapped in an activation function as input to the next layer. This research work uses a multi-layer perceptron in the learning of data integrity and data authenticity of the RSA algorithm's encrypted data. It uses the feed-forward perception where the learning of the data integrity and data authenticity is supervised.

3.4.1.3 The hybrid algorithm

We hope to combine the RSA model and a neural network to improve the RSA algorithm towards a CIA compliant model that achieves, on top of data confidentiality, data integrity, and data authenticity. The idea is to realize CIA compliance in the hybrid model. Algorithm 3.4 summarizes the proposed predictive RSA cryptosystem, supported by Figure 3.4.

Algorithm 3.4: Predictive RSA algorithm.

Input: data set D .

Output: Decrypted data set D .

Internal computation:

1. D is encrypted using the RSA model to form data set E ,
2. Data set E is split into test and training data,
3. Training and testing data are input into a neural network,
4. A neural network's learning process on the hidden layer occurs on the training and testing data, and,
5. Tests are conducted on the training and testing results to check CIA compliance.

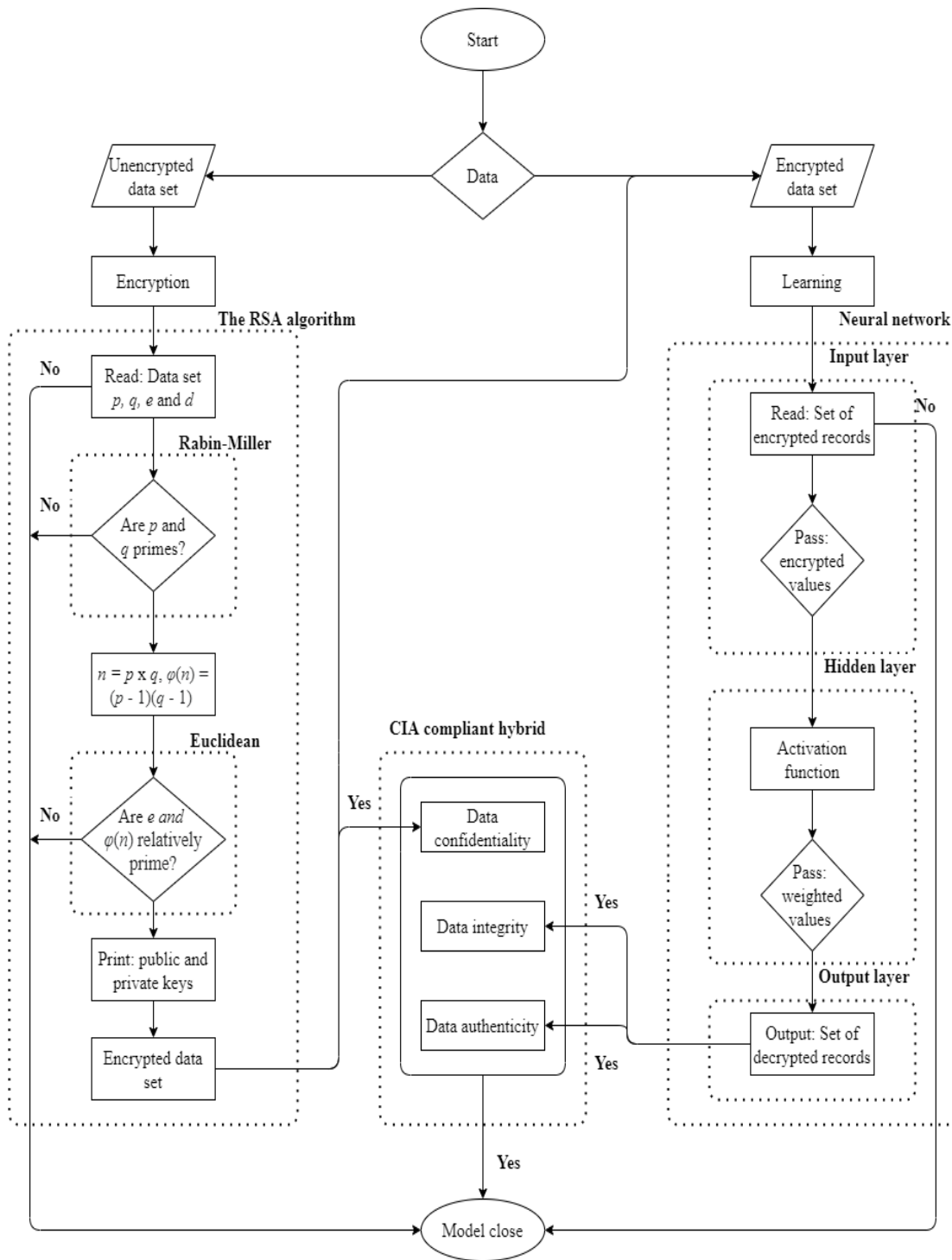


Figure 3.4: RSA's predictive crypto-system workflow.

The flowchart depicted in Figure 3.4 shows how activities in the hybrid algorithm occur concerning how the problem statement stated in chapter one is addressed. The parent RSA algorithm together with its internal algorithms, Rabin-Miller and Euclid's algorithm, is responsible for preserving data confidentiality wherein a neural network is responsible for preserving data integrity and data authenticity. A CIA-compliant model is yielded.

3.4.2 Samples and sampling techniques

The sample space used by this research study is simulated data. A collection of 200 sampled plain data is provided as input. This data set is transformed from plain format into a cipher data set using the RSA algorithm. Once encrypted, further random sampling is handled by the neural network itself. This is done by splitting the encrypted data set into two equal subsets comprising 100 test (50%) and 100 training (50%) data items in each data set. The results related to the performance of the neural network on these data sets are the ones we report and analyze in the next chapter.

3.4.3 Data collection

We indicated that the study extracts data from the simulations administered during the experimental runs. The simulator is given, as input, a public data set in the health domain. This data set is freely available on the internet, comprising two hundred records, with each record built up of five attributes. This is the data set that is encrypted using the RSA model before it is split into the test and the training data sets. This is sensitive data about patients' medical information where data hiding is of paramount priority. The key attributes of interest in this data include patients' first name, last name, email address, gender, and medical policy number. These are the fields that are first hidden and, as proof of concept, whose data integrity and data authenticity are learned by a neural network. All other sensitive attributes of the data have been filtered out.

3.4.4 Tools

The experiments we administer are conducted on a Dell XPS laptop with an Intel(R) Core (TM) i7-8500Y CPU, 16 GB RAM, running Windows 10 Home operating system. The pieces of code that form the units of the proposed hybrid model are all designed and developed using Python. We understand Python as an interpreter, a high-level programming language, which comprises several libraries (Karssenbergh, de Jong & Van Der Kwast, 2007). We use the following libraries in developing the proposed RSA predictive cryptosystem.

| Library | Use in this project |
|-----------|---|
| Random | Generating random prime numbers p and q . |
| Math | Provides access to functions such as power, sqrt, and log. |
| Pandas | Creates data frames out of the RSA's encrypted values. |
| NumPy | Manipulates prime numbers p and q depending on the size of the data set, as well as the learning by the neural network. |
| randrange | Used by Rabin-Miller algorithm to test the primality of p and q . |
| CSV | Used for reading data in the data sets. |
| itertools | Used to iterate into the data set during the encryption process. |

Table 3.1: Python libraries.

3.4.5 Data analysis

The work is a mono-method type of study. This is a choice used when only one research design method is embraced (Onwuegbuzie & Leech, 2005), using a single data collection technique, and following a single data analysis procedure (Saunders, Glenn & Kohn, 2010). In this case, we solely rely on simulated data collected from the experiments administered. We extract descriptive (measures of central tendencies and measures of variability) and inferential statistics (tests for data normality, T-tests, and F-tests) from the collected data.

3.4.5.1 Descriptive statistics

Descriptive statistics explain the internal properties of the data collected from the experiments. They give a concise summary of the sample data and its behavior. Measures of central tendencies assess for commonalities in the data. On the other hand, measures of variability (Khalfan, 2004), also known as measures of dispersion, assess the spread in the data. In this context, the measures of central tendencies we focus on are the mean, mode, and median. We extract these central tendencies from data related to the performances of the neural network in predicting data integrity and data authenticity in RSA encrypted data. On the other hand, the measures of variability we focus on are the standard deviations and kurtosis. The mean, as a measure of central tendency, is defined as the average value of the data set. It is mathematically expressed as follows:

$$X = \frac{1}{n} \sum_{i=1}^n x_i$$

where x_i denotes elements in the data set D , with indexes $i = \{1, 2, \dots, n\}$, and n denotes the sample size. The equation, \sum denotes a sigma notation for sum.

The mode of a data set is k , where $k \in D$ is the most frequent value in the data set. On the other hand, the median is defined as the middle number of the data set, often computed as follows: where n denotes the sample size.

$$M = D \left[\frac{n + 1}{2} \right]$$

The standard deviation describes how the data is spread around the mean. It is mathematically expressed as follows:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}$$

where N denotes the sample size of the data set. Then x_i denotes the elements in the data set, with index $i = \{1, 2, \dots, n\}$. In this, μ is the computed mean of the data set, while σ denoted the standard deviation we are looking for.

The kurtosis is a statistical measure that explains how tails of a statistical distribution differ from the tails of a normal distribution. Sometimes, kurtosis investigates the peak of a statistical distribution relative to the peak of a normal curve. It can be mathematically expressed as follows:

$$Kurtosis = \frac{\sum_{i=1}^N \frac{(X_i - \mu)^4}{N}}{\sigma^4}$$

where N denotes the sample size of the data set, and μ denotes the mean of the same data set. The σ denotes the standard deviation of data in the data set. Our work extracts all these measures of central tendencies from the data collected as performances of a neural network tasked to learn data integrity and authenticity in RSA encrypted data.

3.4.5.2 Inferential statistics

This section commences by describing tests for the normality of the data collected based on Kolmogorov – Smirnov tests. It then discusses inferential statistics related to the administration of T-tests and F-tests towards gathering evidence for possibilities of mapping the observed outcomes to generalized views.

Normality tests are based on Kolmogorov – Smirnov tests. This is a test used to decide if a sample comes from a population with a specific distribution. The Kolmogorov – Smirnov test is based on the empirical distribution function. Given N ordered data points Y_1, Y_2, \dots, Y_N the empirical distribution function is defined as:

$$E_N = \frac{n(i)}{N}$$

where $n(i)$ is the number of points less than Y_i and the Y_i are ordered from smallest to largest value. This is a step function that increases by $1/N$ at the value of each ordered data point. Its main aim is to map and compare the data points to that of a normal distribution.

Once normality is confirmed, we seek evidence for generalization through T-test and F-tests. T-tests decide whether the mean values of two data sets have a meaningful difference which can be related to some features (Marshall & Jonker, 2011). It adopts the use of T-distribution and degrees of freedom to determine the statistical relevance of the differences noted between the two means. The T-test helps us to compare whether the two data sets emanate from the same hypothetical population of data or not. The required outcome when we perform T-tests is a conclusion whether the two means are sampled from the same hypothetical population or not (Marshall & Jonker, 2011). In this

context, the two data sets we refer to would be the outcomes recorded from different runs of the simulations. Mathematically a T-test can be defined as:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{(\sigma^2(\frac{1}{n_1} + \frac{1}{n_2}))}}$$

where \bar{x}_1, \bar{x}_2 and n_1, n_2 denote the means and sample sizes of the two data sets respectively. The σ^2 used in the formula denotes the square of the standard deviation of both data sets. The t we compute is the T-test value.

The F-test, on the other hand, is a regression test in an F-distribution (Marshall & Jonker, 2011). In most cases, it is used to compare statistical models that are used on data sets to identify the model that best fits. In the context of this research study, F-tests are used to compare the variability of data from two data sets. It assesses whether the variations we see in data sets occur by mere chance or whether they represent significant variations. The mathematical model of the F-test can be expressed as:

$$F = \frac{\sigma_1^2}{\sigma_2^2}$$

where σ_1^2 and σ_2^2 describes the variances of the two samples under study. The F we compute is the required F-test value.

3.5 Summary

This chapter commenced with providing the statement of the problem addressed. The statement of the problem connoted interests in presenting the methodology we embrace, as well as the theoretical framework underpinning the study.

The theoretical framework grounding the development of this research study followed after the overview of the chapter. Precisely, design science research, grounded in positivism and deductive reasoning, is the theoretical framework we follow.

A description of the research design followed, emphasizing explaining how the two algorithms, the RSA algorithm, and a neural network, were implemented. Important is the implementation procedure of the integration of the two algorithms. The programming language of choice remained Python, whose libraries of interest were pointed out.

Sampling, data collection, and data analysis techniques were also discussed in this chapter. The emphasis in these analyses was pointed out to be on the descriptive and inferential statistics. The descriptive statistics comprised measures of central tendencies as well as measures of dispersion. Inferential statistics comprised tests for data normality, T-tests, and F- tests. The next chapter administers the experiments, collects data, and presents the data analyses thereto.

Chapter 4 : Experiments, Findings, and Interpretation

This chapter discusses the procedure through which results are generated and reported, as well as the actual results yielded from the simulated execution of the experiments, and the analyses thereto. Results relate to the performances of the proposed RSA predictive cryptosystem built on the notion that a neural network can learn data integrity and data authenticity are reported and analyzed. We mainly pinpoint, record, and interpret the neural network's learning rates to arrive at generalizable views about the hybrid RSA model. These results will be scrutinized concerning centrality, variability, normality, and inferential issues. Evidence is sought with which to accept or reject the null hypothesis that: H_0 : *A neural network has no significant effect on the improvement of the RSA algorithm towards CIA compliancy.* Two alternate directional hypotheses arise. The first alternate hypothesis states that H_1 : *inclusion of a neural network to learn data integrity and data authenticity makes some significant improvement to the RSA algorithm towards a CIA compliant hybrid.* Alternatively, H_2 : *inclusion of a neural network to learn data integrity and data authenticity degrades the RSA algorithm.*

In our view, the outcomes envisioned would answer the questions posed and allow the selection of one of the three alternative hypotheses. The benefits of the study will likely be more than the sum of the contributions of the builder component units of the study.

4.1 Statement of the problem

The key question answered in this chapter is whether the inclusion of a neural network into the RSA algorithm upgrades the RSA model towards CIA compliance or not. To respond to this question, we administer three experiments aimed at investigating and demonstrating possibilities of incorporating the neural network towards assessing capabilities to predict the underlying properties of the RSA algorithm for data integrity and data authenticity. The results yielded will be statistically verified for centrality, variability, normality, and correlations. Useful inferences are then drawn from these analyses towards acceptance or rebuttal of the null hypothesis.

4.2 Overview of the chapter

The rest of the chapter continues to describe the learning process, training, and testing processes the proposed neural network goes through. Precisely, section 4.3 covers the design of the learning process undergone by the neural network. It also touches on the testing and training procedures we embrace. The results yielded from the training and testing processes are reported and analyzed in section 4.4, mainly emphasizing the statistical measures of central tendencies, variability, correlations, normality, and the inferential statistics thereto. Section 4.4 also presents data visualization. In the end, statistically supported conclusions are drawn at a specified level of confidence, in this case, 95% level of confidence.

4.3 Design of the neural network's learning process

In this section, we outline the learning process undergone by the neural network for it to be able to learn data integrity and data authenticity in RSA encrypted data. First, the neural network is trained. Thereafter, it is tested for abilities to learn data integrity and data authenticity. The average number of

epochs and the neural network's learning rate are the key metrics measured. These metrics quantify the cost of using a neural network in this scenario. In this context, an epoch is a measure of how many times all the training vectors have been used to change weights in the hidden layer of a neural network. On the other hand, a neural network's learning rate refers to the configuration of the hyperparameter used by the neural network, often giving small positive values that range between 0 and 1. Data visualization through graphs is presented to augment the meanings of the results reported.

4.3.1 Training a neural network

The focus here is more on the process and the results extracted to validate the neural network meant to learn data integrity and data authenticity in RSA encrypted data towards CIA compliancy. In this case, our neural network was presented with a supervised learning algorithm. The supervised training algorithm used the RSA algorithm's encrypted data set as the input vector to the training process. The desired output formed the output vector.

In this case, the training process required the use of two data sets, the training data set and the testing data set. The training data set comprised data used to train the neural network so that it can accurately predict future outcomes. Contrary, the testing data was used to describe the evaluation of a neural network's outcomes towards model validation.

Supervised learning took advantage of both the testing and the training data sets. The testing and the training data sets were acquired splitting the main data set into two equal data sets 50 - 50. In this case, the main data set comprised the RSA algorithm's encrypted data. As proof of concept, the RSA encrypted data items formed the main data set that was split into training and

testing data sets. It follows that both the testing and the training data sets comprised encrypted data.

Splitting of the main data set was achieved through a Python code that randomly picked data items into each subset. The data in question relates to health records retrieved from a free source on the internet. Precisely, the *sklearn.model_selection* library in Python was used to perform the splitting of this main data set.

While the training data set served as the input vector to the neural network, for learning to occur, an activation function was required. In this case, we used logistic regression as the activation function of choice. Logistic regression is used to find the updated parameters by minimizing the cost function of the predictive cryptosystem model's neural network. The derivative of the activation function was also apparent for minimizing the cost function towards the updated parameters. Logistic regression, in this case, was designed by a Python code's NumPy library where the manipulation of the input data set in terms of numbers was possible. The one hot function designed to handle the labels of the data sets was also of interest. The function took in n numerical labels and created an array of two-dimensional data where each row contained a 'one hot' vector.

A neural network is a layered machine learning algorithm with at least three layers, including the input layer, hidden layer, and output layer. The input layer is responsible for receiving the training data set into the neural network for the learning process. The hidden layer is where the training data set is transformed through mathematical procedures before it is directed to the activation function (logistic regression) as output. The output layer is responsible for disseminating results after the learning process is completed by a neural network.

A class of layers has been designed to handle this important aspect of the predictive cryptosystem model. A class of layers, in this case, consisted of weights, the weighted sum of inputs into the hidden layer, derivatives, or deltas, and the activations. The class function was more concentrated on the hidden layer where most of the learning activities occurred. There was a reserved method used for random initialization of weights in the hidden layer with four parameters. One of them was an activation function explained earlier which activates the weights. The number of nodes per layer was also initialized in the reserved method. Lastly, in the class function, there was an activate function designed to activate the input data with the weights inside the hidden layer for learning purposes.

The main logic for predicting output and updating the weight values in the hidden layer of the neural network has been designed through the Python code which instantiates the NumPy library for easy computations of numbers. A neural network class contained the logic behind the model's predictions. It was used to call the initialized layers in the layer's class. This was done by the reserved method with four parameters, where one of those parameters contains information about the layers. Through a neural network's reserved method, the hidden layer was created by considering the number of nodes initialized. There was an activation function developed inside the neural network class to activate each layer of the model through the input data in the form of a vector. As such, there was a backpropagation algorithm responsible for calculating derivatives for the hidden and output layer. Besides, for the context of this research study, the backpropagation algorithm was also responsible for updating the weights for the hidden and output layers with the derivatives of the main activation function (logistic regression) and weights for the initial input data.

To determine the data integrity and data authenticity of the RSA algorithm's encrypted data set, we designed a method or function called training epochs which has four parameters. Amongst those parameters were, data, targets, and learning rates. In this context, data refers to the batch of training data inputs (half of the encrypted data items), stored in rows or records. Targets would mean the batch of training target outputs (test data set), stored in rows. The learning rate refers to the configuration of the hyperparameter used in the model, with small positive values that range between 0 and 1. The output of this function is defined as the average cost of the neural network which maps the input data to the target outputs after the learning process has occurred for both data sets. This is the function of interest. This is where the data is fed into the model with the desire to answer the research question stipulated in chapter one.

The problem of overfitting may occur during the training process of a neural network. The regularization function has been developed to tackle the problem of overfitting during the learning process. This is a technique used by the neural network to tune the training results by adding a penalty term in the error function, that is the backpropagation function. The additional term controls the excessively fluctuating function such that the coefficients do not take extreme values.

There was a method called cost in the built model in which when given a set of input data, it calculates the average costs or the error output of the neural network. The method was provided for use with advanced optimization routines.

The number of epochs, the neural network's learning rate, data used, node per layer, regularization, and regularization parameter were specified as 10000, 0.008, train data set, [10, 10], True, and 0.01 respectively. This specification

took place in the function named train. One hot function contains the target values, while on the other hand, the average cost values are contained by the training epoch function which is then referenced as a parameter in the training method. The results obtained thereof are appended to a single data frame using Python code's pandas' library for a better understanding of the outcomes.

The developed model used an input layer of five nodes, a hidden layer of ten nodes, and an output layer of two nodes, except the standardized number of nodes per layer which is [10, 10]. This implies that a simple neural network technique was used. The training results for the training data set are presented in table 4.1 below. The results are acquired after the model was run six sequential times, over the number of epochs that are sampled to be equal to 10000 in an experimental setup. Each value in the table below represents the average value of the first and last 100 average costs of the learned training data in terms of a neural network's learning rate. Going down, the numbers represent increased target values, that is, predictions per run.

| Training rate averages | | | | | | |
|------------------------|----------|----------|----------|----------|----------|----------|
| | First | Second | Third | Fourth | Fifth | Sixth |
| 1 | 0.541442 | 0.528595 | 0.277996 | 0.237715 | 0.898377 | 0.48046 |
| 2 | 0.542724 | 0.528646 | 0.278721 | 0.239689 | 0.898393 | 0.480516 |
| 3 | 0.544015 | 0.528748 | 0.279448 | 0.241719 | 0.898425 | 0.480628 |
| 4 | 0.545314 | 0.5289 | 0.280177 | 0.243807 | 0.898473 | 0.480795 |
| 5 | 0.546622 | 0.529101 | 0.280908 | 0.245955 | 0.898536 | 0.481015 |
| 6 | 0.547938 | 0.529353 | 0.281641 | 0.248164 | 0.898614 | 0.481285 |
| 7 | 0.549262 | 0.529652 | 0.282376 | 0.250436 | 0.898708 | 0.481603 |
| 8 | 0.550596 | 0.53 | 0.283113 | 0.252772 | 0.898817 | 0.481967 |
| 9 | 0.551938 | 0.530394 | 0.283851 | 0.255173 | 0.898941 | 0.482372 |
| 10 | 0.553289 | 0.530834 | 0.284592 | 0.257642 | 0.899079 | 0.482817 |
| 11 | 0.554649 | 0.531318 | 0.285334 | 0.260179 | 0.899231 | 0.483297 |
| 12 | 0.556018 | 0.531846 | 0.286078 | 0.262784 | 0.899397 | 0.483811 |
| 13 | 0.557396 | 0.532416 | 0.286824 | 0.26546 | 0.899577 | 0.484354 |

| | | | | | | |
|----|----------|----------|----------|----------|----------|----------|
| 14 | 0.558783 | 0.533026 | 0.287571 | 0.268206 | 0.89977 | 0.484923 |
| 15 | 0.560179 | 0.533676 | 0.288321 | 0.271024 | 0.899975 | 0.485517 |
| 16 | 0.561585 | 0.534363 | 0.289073 | 0.273913 | 0.900193 | 0.486132 |
| 17 | 0.563 | 0.535087 | 0.289826 | 0.276873 | 0.900422 | 0.486766 |
| 18 | 0.564424 | 0.535846 | 0.290581 | 0.279905 | 0.900663 | 0.487417 |
| 19 | 0.565859 | 0.536638 | 0.291338 | 0.283009 | 0.900914 | 0.488082 |
| 20 | 0.567302 | 0.537468 | 0.292097 | 0.286183 | 0.901176 | 0.48876 |
| 21 | 0.568755 | 0.538625 | 0.292857 | 0.289426 | 0.901448 | 0.489449 |
| 22 | 0.570218 | 0.539992 | 0.29362 | 0.292739 | 0.90173 | 0.490148 |
| 23 | 0.571691 | 0.541405 | 0.294384 | 0.296119 | 0.90202 | 0.490854 |
| 24 | 0.573173 | 0.542857 | 0.29515 | 0.299565 | 0.902319 | 0.491567 |
| 25 | 0.574666 | 0.544345 | 0.295918 | 0.303076 | 0.902626 | 0.492286 |
| 26 | 0.576168 | 0.545866 | 0.296688 | 0.306648 | 0.902941 | 0.493008 |
| 27 | 0.577679 | 0.547415 | 0.297459 | 0.31028 | 0.903263 | 0.493734 |
| 28 | 0.579201 | 0.548989 | 0.298232 | 0.31397 | 0.903592 | 0.494462 |
| 29 | 0.580733 | 0.550587 | 0.299007 | 0.317715 | 0.903927 | 0.495192 |
| 30 | 0.582274 | 0.552205 | 0.299784 | 0.321511 | 0.904268 | 0.495922 |
| 31 | 0.583826 | 0.553842 | 0.300562 | 0.325357 | 0.904614 | 0.496653 |
| 32 | 0.585387 | 0.555494 | 0.301342 | 0.329248 | 0.904966 | 0.497383 |
| 33 | 0.586958 | 0.557162 | 0.302124 | 0.333181 | 0.905322 | 0.498112 |
| 34 | 0.588539 | 0.558841 | 0.302907 | 0.337154 | 0.905683 | 0.498839 |
| 35 | 0.59013 | 0.560532 | 0.303692 | 0.341163 | 0.906047 | 0.499564 |
| 36 | 0.591731 | 0.562233 | 0.304478 | 0.345204 | 0.906416 | 0.500286 |
| 37 | 0.593341 | 0.563942 | 0.305266 | 0.349274 | 0.906787 | 0.501006 |
| 38 | 0.594961 | 0.565658 | 0.306056 | 0.353368 | 0.907162 | 0.501723 |
| 39 | 0.596591 | 0.56738 | 0.306847 | 0.357485 | 0.907539 | 0.502436 |
| 40 | 0.598231 | 0.569106 | 0.30764 | 0.36162 | 0.907919 | 0.503145 |
| 41 | 0.59988 | 0.570836 | 0.308434 | 0.36577 | 0.908301 | 0.50385 |
| 42 | 0.601538 | 0.572568 | 0.309229 | 0.369931 | 0.908684 | 0.504551 |
| 43 | 0.603206 | 0.574302 | 0.310026 | 0.374101 | 0.90907 | 0.505248 |
| 44 | 0.604883 | 0.576036 | 0.310824 | 0.378277 | 0.909456 | 0.50594 |
| 45 | 0.606569 | 0.577769 | 0.311623 | 0.382455 | 0.909844 | 0.506628 |
| 46 | 0.608264 | 0.579501 | 0.312423 | 0.386632 | 0.910232 | 0.507311 |
| 47 | 0.609968 | 0.58123 | 0.313225 | 0.390806 | 0.910621 | 0.507988 |
| 48 | 0.611681 | 0.582955 | 0.314027 | 0.394974 | 0.911011 | 0.508661 |
| 49 | 0.613403 | 0.584675 | 0.314831 | 0.399135 | 0.911401 | 0.509329 |
| 50 | 0.615132 | 0.58639 | 0.315635 | 0.403285 | 0.911791 | 0.509991 |
| 51 | 0.616871 | 0.588099 | 0.31644 | 0.407423 | 0.912181 | 0.510648 |
| 52 | 0.618617 | 0.589799 | 0.317246 | 0.411546 | 0.91257 | 0.5113 |

| | | | | | | |
|----|----------|----------|----------|----------|----------|----------|
| 53 | 0.620371 | 0.591491 | 0.318053 | 0.415653 | 0.912959 | 0.511947 |
| 54 | 0.622133 | 0.593173 | 0.31886 | 0.419743 | 0.913348 | 0.512588 |
| 55 | 0.623902 | 0.594844 | 0.319667 | 0.423813 | 0.913736 | 0.513224 |
| 56 | 0.625679 | 0.596503 | 0.320475 | 0.427863 | 0.914123 | 0.513854 |
| 57 | 0.627463 | 0.598149 | 0.321284 | 0.43189 | 0.914509 | 0.514479 |
| 58 | 0.629253 | 0.599781 | 0.322092 | 0.435895 | 0.914894 | 0.515099 |
| 59 | 0.63105 | 0.601397 | 0.322901 | 0.439875 | 0.915278 | 0.515713 |
| 60 | 0.632854 | 0.602998 | 0.32371 | 0.443831 | 0.91566 | 0.516322 |
| 61 | 0.634663 | 0.604581 | 0.32452 | 0.44776 | 0.916041 | 0.516925 |
| 62 | 0.636478 | 0.606145 | 0.32533 | 0.451662 | 0.916421 | 0.517523 |
| 63 | 0.638298 | 0.60769 | 0.326141 | 0.455536 | 0.916799 | 0.518116 |
| 64 | 0.640124 | 0.609214 | 0.326953 | 0.459382 | 0.917175 | 0.518703 |
| 65 | 0.641954 | 0.610717 | 0.327767 | 0.463198 | 0.91755 | 0.519285 |
| 66 | 0.643789 | 0.612197 | 0.328582 | 0.466984 | 0.917923 | 0.519862 |
| 67 | 0.645627 | 0.613653 | 0.3294 | 0.470738 | 0.918294 | 0.520434 |
| 68 | 0.64747 | 0.615084 | 0.330222 | 0.47446 | 0.918663 | 0.521 |
| 69 | 0.649316 | 0.61649 | 0.331049 | 0.478149 | 0.91903 | 0.521562 |
| 70 | 0.651164 | 0.617869 | 0.331883 | 0.481802 | 0.919395 | 0.522118 |
| 71 | 0.653015 | 0.619221 | 0.332725 | 0.485419 | 0.919758 | 0.522669 |
| 72 | 0.654869 | 0.620545 | 0.33358 | 0.488999 | 0.920119 | 0.523215 |
| 73 | 0.656724 | 0.62184 | 0.33445 | 0.492539 | 0.920478 | 0.523756 |
| 74 | 0.65858 | 0.623107 | 0.335341 | 0.496037 | 0.920834 | 0.524292 |
| 75 | 0.660437 | 0.624343 | 0.336257 | 0.499491 | 0.921188 | 0.524823 |
| 76 | 0.662295 | 0.625549 | 0.337205 | 0.502899 | 0.92154 | 0.52535 |
| 77 | 0.664152 | 0.626725 | 0.338195 | 0.506257 | 0.92189 | 0.525871 |
| 78 | 0.666009 | 0.62787 | 0.339236 | 0.509564 | 0.922237 | 0.526388 |
| 79 | 0.667865 | 0.628985 | 0.340343 | 0.512816 | 0.922582 | 0.5269 |
| 80 | 0.66972 | 0.630068 | 0.341531 | 0.516011 | 0.922925 | 0.527408 |
| 81 | 0.671572 | 0.631121 | 0.34282 | 0.519147 | 0.923265 | 0.527911 |
| 82 | 0.673422 | 0.632142 | 0.344234 | 0.522222 | 0.923603 | 0.528409 |
| 83 | 0.675269 | 0.633133 | 0.345801 | 0.525237 | 0.923938 | 0.528903 |
| 84 | 0.677112 | 0.634094 | 0.347554 | 0.528194 | 0.924271 | 0.529393 |
| 85 | 0.678951 | 0.635025 | 0.349535 | 0.531099 | 0.924602 | 0.529878 |
| 86 | 0.680785 | 0.635927 | 0.351791 | 0.533964 | 0.92493 | 0.530358 |
| 87 | 0.682614 | 0.6368 | 0.354376 | 0.536809 | 0.925255 | 0.530835 |
| 88 | 0.684437 | 0.637644 | 0.357353 | 0.539663 | 0.925578 | 0.531307 |
| 89 | 0.686253 | 0.63846 | 0.360793 | 0.542573 | 0.925899 | 0.531775 |
| 90 | 0.688063 | 0.63925 | 0.364776 | 0.545603 | 0.926217 | 0.532239 |
| 91 | 0.689865 | 0.640013 | 0.36939 | 0.548849 | 0.926533 | 0.532699 |

| | | | | | | |
|-----|----------|----------|----------|----------|----------|----------|
| 92 | 0.691658 | 0.64075 | 0.374733 | 0.552436 | 0.926846 | 0.533154 |
| 93 | 0.693442 | 0.641462 | 0.380914 | 0.556534 | 0.927157 | 0.533606 |
| 94 | 0.695217 | 0.642151 | 0.388057 | 0.561354 | 0.927465 | 0.534236 |
| 95 | 0.696982 | 0.642816 | 0.396312 | 0.567142 | 0.927774 | 0.543011 |
| 96 | 0.698736 | 0.643458 | 0.405869 | 0.574164 | 0.928134 | 0.556836 |
| 97 | 0.700478 | 0.644079 | 0.416976 | 0.582663 | 0.928507 | 0.571129 |
| 98 | 0.702208 | 0.644679 | 0.429958 | 0.592804 | 0.928876 | 0.585356 |
| 99 | 0.703926 | 0.645259 | 0.445195 | 0.604609 | 0.92924 | 0.599073 |
| 100 | 0.705629 | 0.64582 | 0.463045 | 0.617916 | 0.929601 | 0.611974 |

Table 4.1: Average Training Rates.

We extract central tendencies, variability, correlations, and test this data for normality before drawing any inferences thereto. In this context, measures of central tendencies refer to the mean, mode, and median of the training rates. Measures of variability refer to standard deviations and kurtosis. Correlation establishes the degree of association between results extracted from different runs. Normality tests decide whether the training rates are normally distributed or not. Thereafter, T-test and F-test would culminate the promised inferential statistics.

We exploit the use of an online tool for determining normality in a data set. The tool also reports the mean, median, mode, standard deviation, skewness, and kurtosis of a distribution. In this case, the Kolmogorov – Smirnov test for data normality is calculated for each set of the training rates yielded in each run as presented in Table 4.1. This Kolmogorov – Smirnov test allows us to decide whether a sample distribution matches the characteristics of a normal distribution. It is important to know this since we intend to extract inferences at the end. The higher the Kolmogorov - Smirnov test value, the less probable it is that the data is normally distributed. The p-value reported quantifies this probability, with a low probability indicating that the sample deviates from a

normal distribution to an extent unlikely to arise merely by chance. Put simply, a high Kolmogorov – Smirnov test value and a low p-value are evidence that data is not normally distributed. The reverse would hold for normally distributed data sets.

Figure 4.1 shows a Kolmogorov – Smirnov test conducted on the first set of training rates reported in the first column in Table 4.1. A value of 0.07353 is yielded, with a p-value of 0.62518. These outcomes favour a conclusion that this set of results does not differ significantly from that which is normally distributed. Precisely, these results are normally distributed.

The mean, mode, and median are the measures of central tendency we use as descriptive statistics. These three measures represent the central tendencies we seek. In this case, the training rates yielded in this first run produced the mean performance of 0.61888, the modal performance of 0.541442, and a median of 0.616002. Interesting is how close to each other these central tendencies are, connoting commonality in the distribution of the data, thus consistently pointing to possibilities that the normality we observe does not occur by chance.

On the other hand, standard deviation and kurtosis describe measures of variability. Standard deviation checks how dispersed the values in the distribution are from the central tendencies. In this case, a standard deviation of 0.049103 is reported. Kurtosis then measures how peaked the normal curve thereto would be, how far the pick of the data set is from the peak of the Gaussian curve. Positive Kurtosis indicates distributions highly peaked than the normal curve. Negative kurtosis would mean curves below the normal curve. In this case, a kurtosis value of -1.228544 is reported from the first set of the neural network's training rates. This is not very far below the normal curve, also then connoting commonality in this data set.

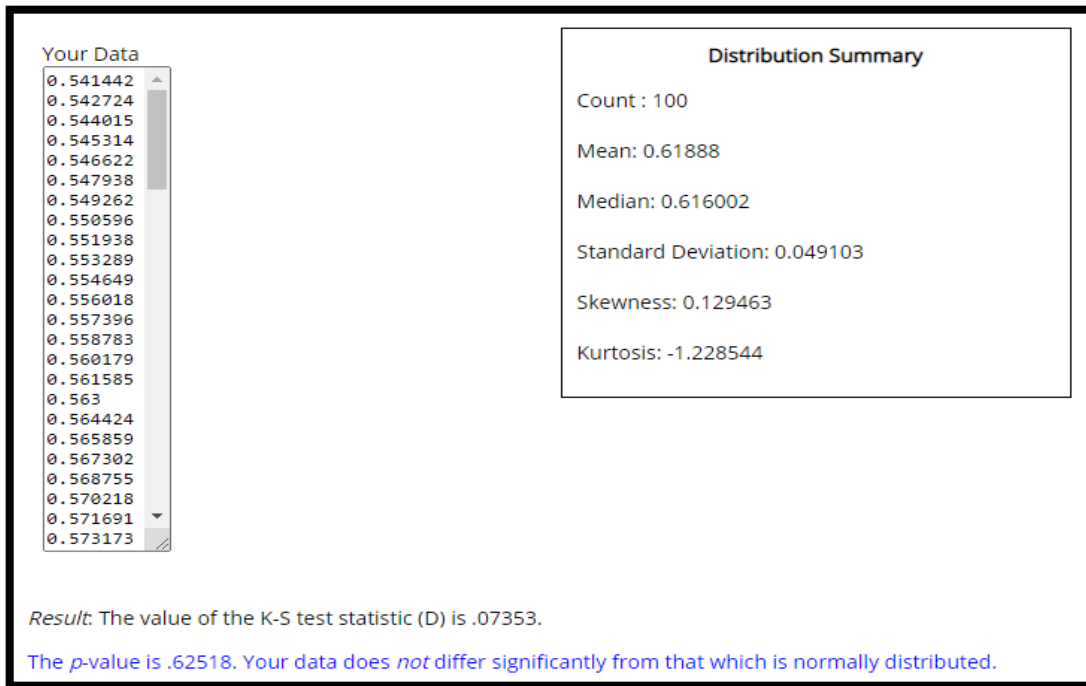


Figure 4.1: Test for normality on learning rates achieved in the first run.

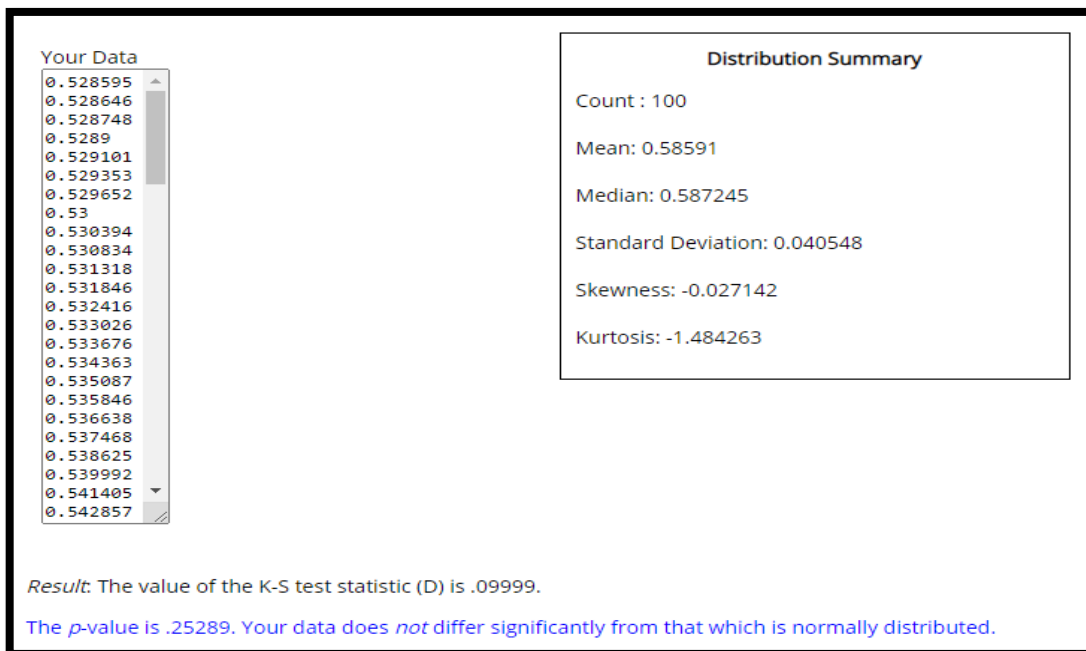


Figure 4.2: Test for normality on learning rates achieved in the second run.

Figure 4.2 reports analyses for the training rates achieved in the second run of the experiment, reported in the second column of Table 4.1. Consistent outcomes are observed, where a Kolmogorov – Smirnov value of 0.09999 and a p-value of 0.25289 are achieved. The probability of a match with the normal curve is high, providing good evidence that the data reported in this run is normally distributed. The measures of central tendency confirm this observation, where a mean of 0.58591, mode of 0.52860, and median of 0.58745 are observed. Compelling is also how these three measures approximate one another. Similarly, measures of variability support an insignificant variation of scores from centrality, where a standard deviation of 0.040548 and kurtosis of -1.484263 are reported. These outcomes purport common trends in the two different experiment runs discussed so far.

Figure 4.3 provides analyses of the training rates achieved when the experiment was run for the third time to ensure rigour and validity of the outcomes of the study. Interestingly, these results do not generally significantly differ from that which is normally distributed. A Kolmogorov – Smirnov test value of 0.10959, with a p-value of 0.16804 is encouraging. It is likely true that the performances of the neural network can be mapped to a generalized opinion regarding normality based on Kolmogorov – Smirnov tests, measures of central tendency, and variability. In this case, a mean value of 0.32265, a mode of 0.27810, and a median of 0.316038 arise, also very close to each other. The standard deviation of 0.036526 is quite low and the kurtosis of 2.969665 is fairly good. These results similarly demonstrate normality patterns.

Similar trends are noticed in Figure 4.4, Figure 4.5, and Figure 4.6 in which the performances of the neural network in the fourth, fifth, and sixth run are analysed. In all cases, good Kolmogorov – Smirnov test values are

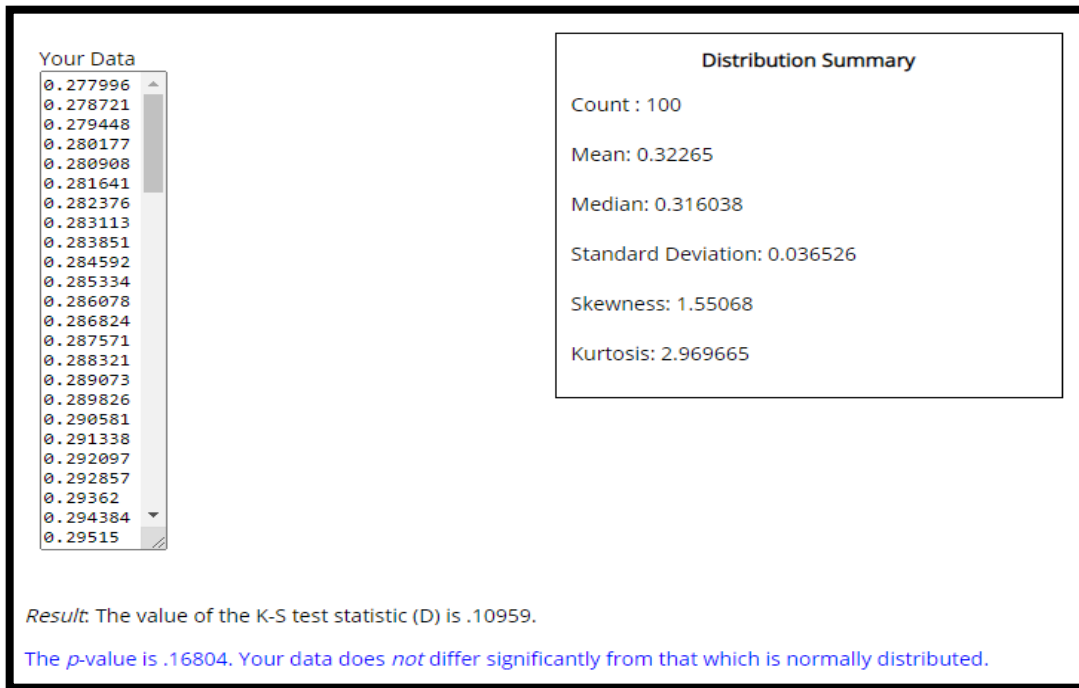


Figure 4.3: Test for normality on learning rates achieved in the third run.

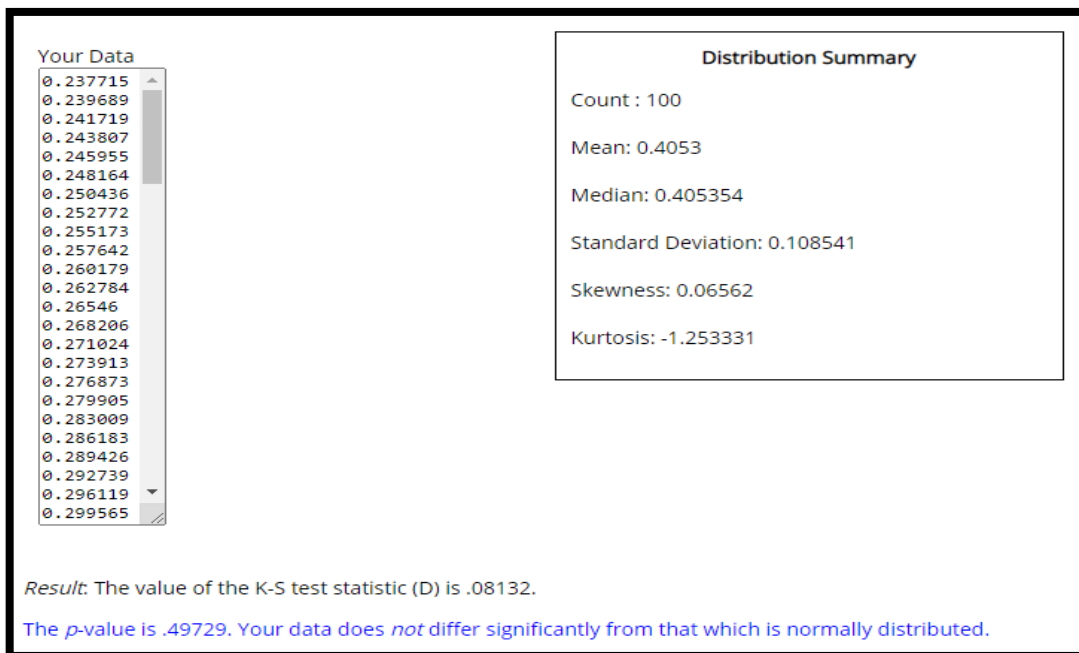


Figure 4.4: Test for normality on learning rates achieved in the fourth run.

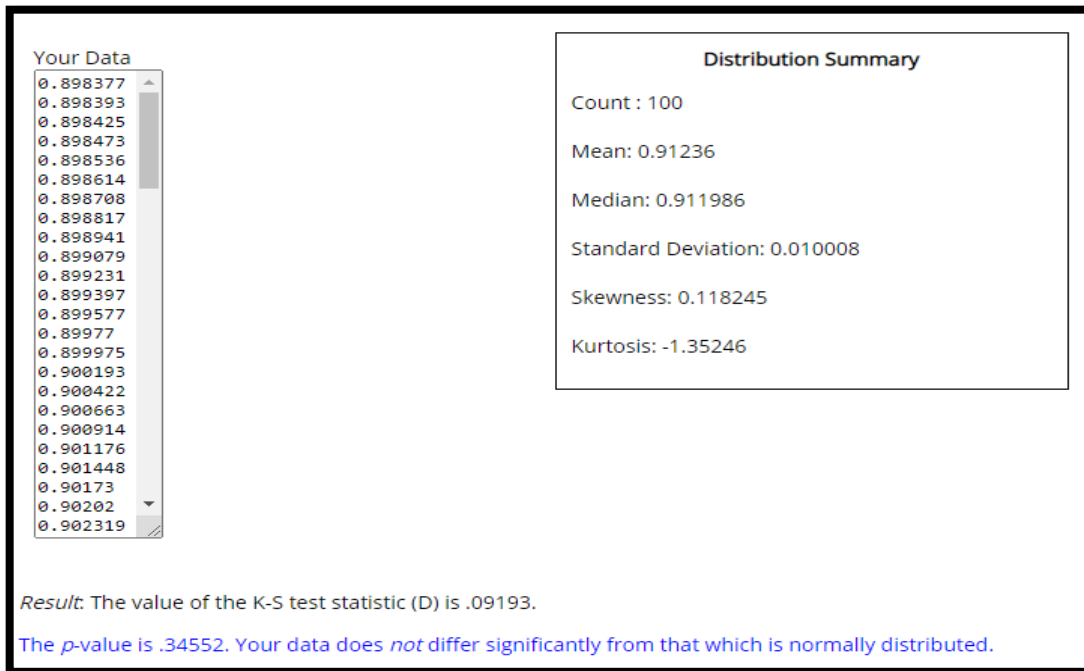


Figure 4.5: Test for normality on learning rates achieved in the fifth run.

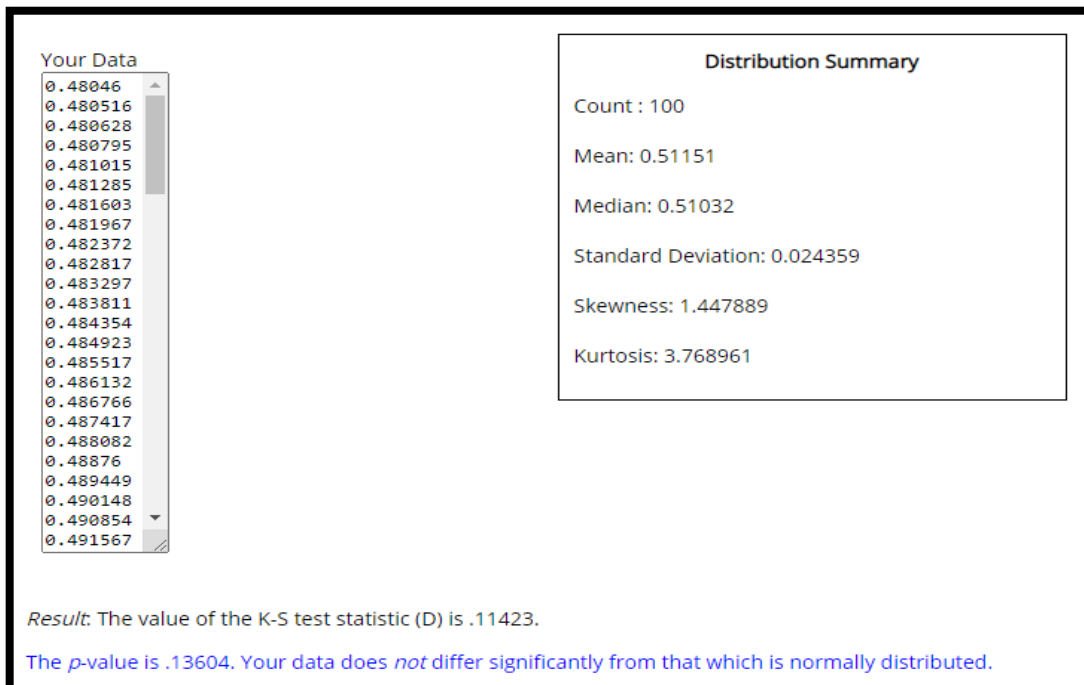


Figure 4.6: Test for normality on learning rates achieved in the sixth run.

observed, along with plausible p – values, all suggesting data sets that do not differ significantly from that which is normally distributed. The central tendencies reported in these runs also connote centrality and commonalities. Related measures of variability connote minimal variations. In all the cases, the observed normality does not show evidence of occurring by chance.

Figure 4.7 shows a heatmap plot used to explain the correlation among data reported from each run. Each square shows the correlation between data labels on each axis. Correlation ranges from -1 to +1. Values close to zero imply that there are no linear patterns between the two data labels under investigation. A strong positive correlation is achieved when the values are closer to 1. In that case, the values of one data field increase when those for the other one also increases. A negative correlation is achieved when the values are closer to -1. In this case, the values of one data field increase while the other one decreases. The diagonal squares are all 1 because there is a perfect positive correlation between values in the same data fields. The larger the number in the squares the higher the correlation between the two data fields.

The plot is symmetrical since the same two data fields are being paired together in those squares. We are encouraged by observing strong positive correlations between all data sets, connoting these results coming from a normally distributed population. Every set of results can be mapped to the results achieved in a different run. Precisely, we can regressively decide on missing factors from one run using results achieved from another run. These are all pointers towards possibilities of arriving at generalized opinions. Table 4.2 summarizes the statistics reported,

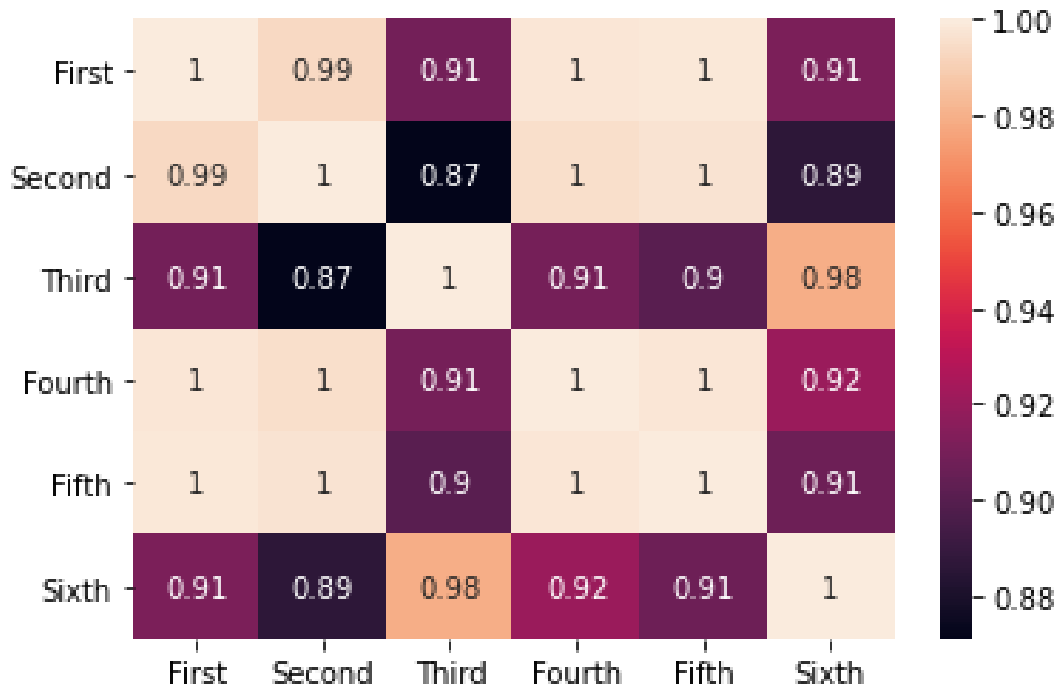


Figure 4.7: Correlation between training results.

| SUMMARY | | | | | |
|---------|----------|----------|----------|-----------|--------------------|
| Groups | Median | Mode | Mean | Kurtosis | Standard Deviation |
| First | 0.616002 | 0.541442 | 0.618885 | -1.228544 | 0.049103 |
| Second | 0.587245 | 0.528595 | 0.585908 | -1.484263 | 0.040548 |
| Third | 0.316038 | 0.277996 | 0.322651 | 2.969665 | 0.036526 |
| Fourth | 0.405354 | 0.237715 | 0.405303 | -1.253331 | 0.108541 |
| Fifth | 0.911986 | 0.898377 | 0.912365 | -1.35246 | 0.010008 |
| Sixth | 0.51032 | 0.48046 | 0.511512 | 3.768961 | 0.024359 |

Table 4.2: Summary of the neural network's training rates.

| ANOVA | | | | | |
|---------------------|----------|-----|----------|---------|----------|
| Source of Variation | SS | df | MS | P-value | F value |
| Between Groups | 21.09145 | 5 | 4.21829 | 0.002 | 2.229193 |
| Within Groups | 1.76854 | 594 | 0.002977 | | |
| Total | 22.85999 | 599 | | | |

Table 4.3: Analysis of variances in the neural network's training rates.

Following the summary of the neural network's training rates, we have adopted the one-way ANOVA to check the relationship between the means of the six sets of results. Table 4.3 summarizes these relationships.

Our null hypothesis stated that the inclusion of a neural network to learn data integrity and authenticity in RSA encrypted data has no significant effect on the enhancement of the algorithm towards CIA compliance. ANOVA p-value is 0.002, which is less than the significance level of 0.05. This provides evidence for us to reject the null hypothesis in favor of an alternate hypothesis. Positive correlations are observed throughout, connoting the alternative hypothesis that plausible improvement of the RSA algorithm from integration with a neural network is noted, which brings integrity and authenticity into the RSA algorithm towards a CIA compliant hybrid.

Further inferential statistics confirmed these observations. We used T-tests to compare the means achieved in different experiment runs. This test checks whether the means show commonalities with known hypothetical population mean. F-tests were also used to compare variations in the training rates achieved in different runs. Inferential statistics are used to allow us to reach

better-informed conclusions. In the six runs reported in this study, a hypothetical population mean of 0.3 is estimated when T-tests are evaluated with a significance level of 0.05 and a one-tailed hypothesis. All the sets of data produced T-values above 5, with probability values less than 0.00001 respectively. This outcome suggests that all the T-values achieved are significant at $p < 0.05$. All the means can be mapped to the population at a 95% level of confidence. The variation we observe in the data sets are mere random insignificant variations which occur by chance. We can conclude that a neural network can learn data integrity and data authenticity in RSA encrypted messages, allowing the model to then exhibit properties of a CIA-compliant model.

4.3.2 Testing of the neural network after training

We shifted our focus to the processes involved in testing the neural network's performances after the training process. In this case, we now used the testing data set as input data to be passed into the subsequent layers of a neural network, implying a feedforward analogy. Logistic regression has been used as an activation function. The backpropagation was used to perform derivations of the hidden layer's activation function into the output layer. Also, the backpropagation was responsible for updating the weights in the hidden layer and the output layer with the derivatives of the main activation function (logistic regression) and weights for the initial input data. The same regularization function used in section 4.3.1 was also adopted, tackling the problem of overfitting.

Similarly, the cost function calculates the average cost of a neural network, allowing optimization of the model. The train function has parameters including the number of epochs, learning rate, data used, node per layer, regularization, and regularization parameter. These parameters were assigned the same

values as in the training process; 10000, 0.008, train data set, [10, 10], True, and 0.01 respectively. The hot function contained the target values, while on the other hand, the average cost values were contained by the testing epoch function which was then referenced by a parameter in the training method. The results obtained thereof were appended to a single data frame using Python code's pandas library. To avoid biases in the two data sets, the developed model used an input layer of 5 nodes, a hidden layer of 10 nodes, and an output layer of 2 nodes. A simple supervised neural network was used to learn data integrity and authenticity towards a CIA-compliant hybrid.

The test results for the test data set are presented in table 4.4 below. The results are acquired after the model was run six sequential times in an experimental setup. In these testing results, epochs are also sampled to be equal to 10000 in an experimental setup. Each value in the table represents the average value of the first and last 100 average costs of the learned testing data in terms of a neural network's learning rate. Going down, the numbers represent increased target values, that is, predictions per run.

| Testing rate averages | | | | | | |
|-----------------------|----------|----------|----------|----------|----------|----------|
| | First | Second | Third | Fourth | Fifth | Sixth |
| 1 | 0.175374 | 0.802344 | 0.619979 | 0.923925 | 0.742672 | 0.788377 |
| 2 | 0.177367 | 0.804074 | 0.621028 | 0.924456 | 0.742724 | 0.788432 |
| 3 | 0.179543 | 0.80578 | 0.622089 | 0.924977 | 0.742826 | 0.788542 |
| 4 | 0.181927 | 0.807459 | 0.623163 | 0.925487 | 0.742976 | 0.788704 |
| 5 | 0.184547 | 0.809113 | 0.624249 | 0.925986 | 0.743171 | 0.788915 |
| 6 | 0.187434 | 0.810741 | 0.625347 | 0.926475 | 0.743405 | 0.789173 |
| 7 | 0.190625 | 0.812344 | 0.626459 | 0.926954 | 0.743674 | 0.789472 |
| 8 | 0.194161 | 0.813922 | 0.627583 | 0.927424 | 0.743974 | 0.789809 |
| 9 | 0.198089 | 0.815475 | 0.62872 | 0.927884 | 0.744301 | 0.790179 |
| 10 | 0.202463 | 0.817004 | 0.629871 | 0.928335 | 0.744651 | 0.790578 |
| 11 | 0.20734 | 0.818508 | 0.631035 | 0.928778 | 0.745021 | 0.791001 |

| | | | | | | |
|----|----------|----------|----------|----------|----------|----------|
| 12 | 0.212785 | 0.819988 | 0.632212 | 0.929213 | 0.745408 | 0.791446 |
| 13 | 0.218869 | 0.821444 | 0.633403 | 0.929639 | 0.74581 | 0.791907 |
| 14 | 0.225664 | 0.822877 | 0.634608 | 0.930057 | 0.746224 | 0.792383 |
| 15 | 0.233245 | 0.824286 | 0.635828 | 0.930468 | 0.74665 | 0.79287 |
| 16 | 0.241685 | 0.825673 | 0.637061 | 0.930872 | 0.747084 | 0.793366 |
| 17 | 0.251045 | 0.827038 | 0.638309 | 0.931268 | 0.747527 | 0.793869 |
| 18 | 0.261372 | 0.82838 | 0.639572 | 0.931657 | 0.747977 | 0.794377 |
| 19 | 0.272689 | 0.829701 | 0.64085 | 0.932039 | 0.748434 | 0.79489 |
| 20 | 0.284981 | 0.831 | 0.642142 | 0.932415 | 0.748896 | 0.795405 |
| 21 | 0.298191 | 0.832279 | 0.64345 | 0.932785 | 0.749364 | 0.795923 |
| 22 | 0.312209 | 0.833537 | 0.644773 | 0.933148 | 0.749836 | 0.796443 |
| 23 | 0.326876 | 0.834775 | 0.646112 | 0.933505 | 0.750328 | 0.796963 |
| 24 | 0.341986 | 0.835993 | 0.647467 | 0.933856 | 0.750863 | 0.797484 |
| 25 | 0.357303 | 0.837192 | 0.648837 | 0.934202 | 0.751401 | 0.798006 |
| 26 | 0.372578 | 0.838372 | 0.650224 | 0.934542 | 0.751941 | 0.798528 |
| 27 | 0.387571 | 0.839534 | 0.651627 | 0.934876 | 0.752483 | 0.79905 |
| 28 | 0.40207 | 0.840678 | 0.653046 | 0.935205 | 0.753026 | 0.799572 |
| 29 | 0.415905 | 0.841805 | 0.654482 | 0.935529 | 0.75357 | 0.800094 |
| 30 | 0.428955 | 0.842914 | 0.655935 | 0.935848 | 0.754116 | 0.800616 |
| 31 | 0.441146 | 0.844007 | 0.657405 | 0.936162 | 0.754663 | 0.801138 |
| 32 | 0.452445 | 0.845084 | 0.658893 | 0.936472 | 0.755211 | 0.801661 |
| 33 | 0.462857 | 0.846145 | 0.660398 | 0.936776 | 0.75576 | 0.802183 |
| 34 | 0.47241 | 0.847191 | 0.661921 | 0.937076 | 0.75631 | 0.802706 |
| 35 | 0.481149 | 0.848223 | 0.663461 | 0.937372 | 0.756862 | 0.803228 |
| 36 | 0.489131 | 0.84924 | 0.66502 | 0.937663 | 0.757414 | 0.803752 |
| 37 | 0.496416 | 0.850244 | 0.666596 | 0.93795 | 0.757968 | 0.804275 |
| 38 | 0.503067 | 0.851234 | 0.668192 | 0.938233 | 0.758523 | 0.804799 |
| 39 | 0.509142 | 0.852213 | 0.669805 | 0.938512 | 0.759079 | 0.805323 |
| 40 | 0.514698 | 0.853179 | 0.671438 | 0.938787 | 0.759636 | 0.805848 |
| 41 | 0.519789 | 0.854135 | 0.67309 | 0.939058 | 0.760194 | 0.806373 |
| 42 | 0.524462 | 0.855079 | 0.67476 | 0.939326 | 0.760754 | 0.806898 |
| 43 | 0.52876 | 0.856014 | 0.67645 | 0.93959 | 0.761314 | 0.807425 |
| 44 | 0.532723 | 0.85694 | 0.67816 | 0.93985 | 0.761876 | 0.807951 |
| 45 | 0.536384 | 0.857857 | 0.679889 | 0.940106 | 0.762439 | 0.808479 |
| 46 | 0.539776 | 0.858767 | 0.681638 | 0.940359 | 0.763003 | 0.809007 |
| 47 | 0.542925 | 0.859669 | 0.683407 | 0.940609 | 0.763567 | 0.809535 |
| 48 | 0.545855 | 0.860566 | 0.685196 | 0.940856 | 0.764134 | 0.810065 |
| 49 | 0.548589 | 0.861458 | 0.687005 | 0.941099 | 0.764701 | 0.810595 |
| 50 | 0.551145 | 0.862345 | 0.688835 | 0.941339 | 0.765269 | 0.811125 |

| | | | | | | |
|----|----------|----------|----------|----------|----------|----------|
| 51 | 0.553541 | 0.863229 | 0.690684 | 0.941576 | 0.765838 | 0.811656 |
| 52 | 0.555791 | 0.864111 | 0.692554 | 0.94181 | 0.766409 | 0.812188 |
| 53 | 0.557909 | 0.864992 | 0.694445 | 0.942042 | 0.76698 | 0.812721 |
| 54 | 0.559907 | 0.865872 | 0.696355 | 0.94227 | 0.767553 | 0.813254 |
| 55 | 0.561796 | 0.866754 | 0.698286 | 0.942495 | 0.768127 | 0.813788 |
| 56 | 0.563586 | 0.867637 | 0.700238 | 0.942718 | 0.768701 | 0.814323 |
| 57 | 0.565284 | 0.868523 | 0.702209 | 0.942938 | 0.769277 | 0.814858 |
| 58 | 0.566899 | 0.869413 | 0.7042 | 0.943155 | 0.769854 | 0.815394 |
| 59 | 0.568438 | 0.870307 | 0.70621 | 0.94337 | 0.770432 | 0.81593 |
| 60 | 0.569907 | 0.871207 | 0.708238 | 0.943582 | 0.771012 | 0.816468 |
| 61 | 0.571312 | 0.872112 | 0.710285 | 0.943791 | 0.771592 | 0.817005 |
| 62 | 0.572659 | 0.873023 | 0.71235 | 0.943998 | 0.772173 | 0.817544 |
| 63 | 0.573951 | 0.873939 | 0.714431 | 0.944203 | 0.772756 | 0.818083 |
| 64 | 0.575195 | 0.874861 | 0.716527 | 0.944405 | 0.773339 | 0.818623 |
| 65 | 0.576393 | 0.875786 | 0.718637 | 0.944605 | 0.773924 | 0.819163 |
| 66 | 0.577552 | 0.876713 | 0.720759 | 0.944803 | 0.774509 | 0.819704 |
| 67 | 0.578673 | 0.87764 | 0.722892 | 0.944998 | 0.775096 | 0.820245 |
| 68 | 0.579763 | 0.878564 | 0.725032 | 0.945191 | 0.775684 | 0.820787 |
| 69 | 0.580826 | 0.87948 | 0.727179 | 0.945382 | 0.776273 | 0.82133 |
| 70 | 0.581866 | 0.880385 | 0.729327 | 0.945571 | 0.776863 | 0.821873 |
| 71 | 0.582889 | 0.881273 | 0.731474 | 0.945758 | 0.777454 | 0.822416 |
| 72 | 0.583903 | 0.882139 | 0.733617 | 0.945943 | 0.778046 | 0.82296 |
| 73 | 0.584916 | 0.882976 | 0.735751 | 0.946126 | 0.778639 | 0.823505 |
| 74 | 0.585939 | 0.883779 | 0.737872 | 0.946307 | 0.779233 | 0.82405 |
| 75 | 0.58699 | 0.884541 | 0.739977 | 0.946486 | 0.779828 | 0.824596 |
| 76 | 0.588091 | 0.885255 | 0.742061 | 0.946663 | 0.780425 | 0.825142 |
| 77 | 0.589275 | 0.885917 | 0.744121 | 0.946838 | 0.781022 | 0.825688 |
| 78 | 0.590592 | 0.886522 | 0.746158 | 0.947011 | 0.78162 | 0.826235 |
| 79 | 0.592119 | 0.88702 | 0.748172 | 0.947183 | 0.78222 | 0.826782 |
| 80 | 0.593974 | 0.887214 | 0.750169 | 0.947352 | 0.78282 | 0.82733 |
| 81 | 0.596342 | 0.887362 | 0.752163 | 0.94752 | 0.783422 | 0.827879 |
| 82 | 0.599513 | 0.887509 | 0.754175 | 0.947687 | 0.784025 | 0.828427 |
| 83 | 0.603934 | 0.887654 | 0.756242 | 0.947851 | 0.784628 | 0.828976 |
| 84 | 0.610284 | 0.887797 | 0.758417 | 0.948014 | 0.785233 | 0.829525 |
| 85 | 0.61953 | 0.887937 | 0.760779 | 0.948175 | 0.785838 | 0.830075 |
| 86 | 0.632911 | 0.888072 | 0.76344 | 0.948335 | 0.786445 | 0.830625 |
| 87 | 0.651694 | 0.888201 | 0.766546 | 0.948493 | 0.787053 | 0.831175 |
| 88 | 0.676571 | 0.888325 | 0.770283 | 0.94865 | 0.787662 | 0.831726 |
| 89 | 0.706851 | 0.888443 | 0.774868 | 0.948805 | 0.788272 | 0.832277 |

| | | | | | | |
|-----|----------|----------|----------|----------|----------|----------|
| 90 | 0.740164 | 0.888553 | 0.780527 | 0.948958 | 0.788882 | 0.832828 |
| 91 | 0.773295 | 0.888655 | 0.787453 | 0.94911 | 0.789494 | 0.83338 |
| 92 | 0.803573 | 0.888749 | 0.795754 | 0.949261 | 0.790107 | 0.83395 |
| 93 | 0.829642 | 0.888833 | 0.805395 | 0.94941 | 0.790721 | 0.839465 |
| 94 | 0.851311 | 0.888909 | 0.816173 | 0.949558 | 0.791336 | 0.849063 |
| 95 | 0.869031 | 0.888974 | 0.827733 | 0.949704 | 0.791952 | 0.858756 |
| 96 | 0.883468 | 0.889029 | 0.839641 | 0.949849 | 0.792569 | 0.868277 |
| 97 | 0.895276 | 0.889073 | 0.851461 | 0.949993 | 0.793186 | 0.877415 |
| 98 | 0.905007 | 0.889107 | 0.862833 | 0.950135 | 0.793805 | 0.886024 |
| 99 | 0.913104 | 0.889129 | 0.873502 | 0.950276 | 0.794425 | 0.894019 |
| 100 | 0.91991 | 0.88914 | 0.880746 | 0.950416 | 0.795046 | 0.901365 |

Table 4.4: Average Testing Rates.

We also determine Kolmogorov Smirnov test values of the data reported in table 4.4 using the same online tool which also reports some central tendencies and measures of dispersion. Figure 4.8 summarizes tests on the results achieved in the first run using test data. A Kolmogorov - Smirnov test value of 0.13157 and the p-value of 0.05712 are achieved, similarly connoting data which does not significantly differ from that which is normally distributed. Measures of central tendencies, in this case, equally showed a similar picture, with a mean of 0.5046907, a mode of 0.575374, and a median of 0.552343. Striking is how close these scores are to one another, further connoting normality in the data distribution. Dispersion measures are also consistent. A standard deviation of 0.191575 and kurtosis of -0.301438 are observed.

Figures 4.9 to 4.13 report Kolmogorov – Smirnov tests, central tendencies, and measures of dispersions observed in the second to the sixth run of the experiment. Consistent Kolmogorov – Smirnov values and p-values pointing to results that do not significantly differ from data that is normally distributed are persistent. Central tendencies plausibly remain close to each other (e.g., a

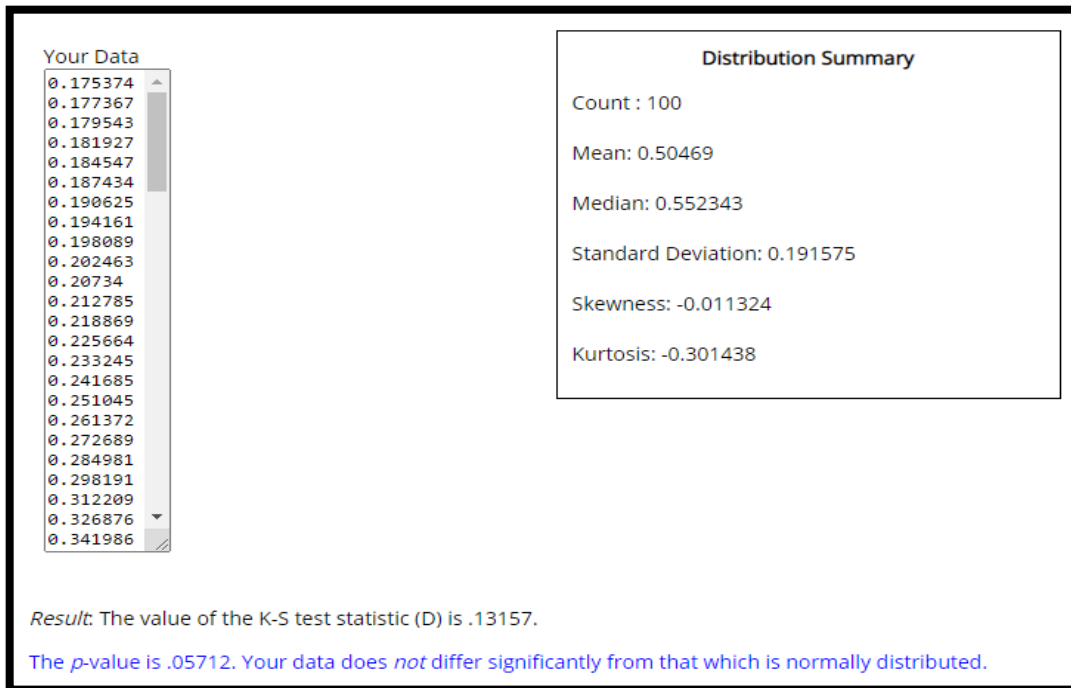


Figure 4.8: Normality test for the first column.

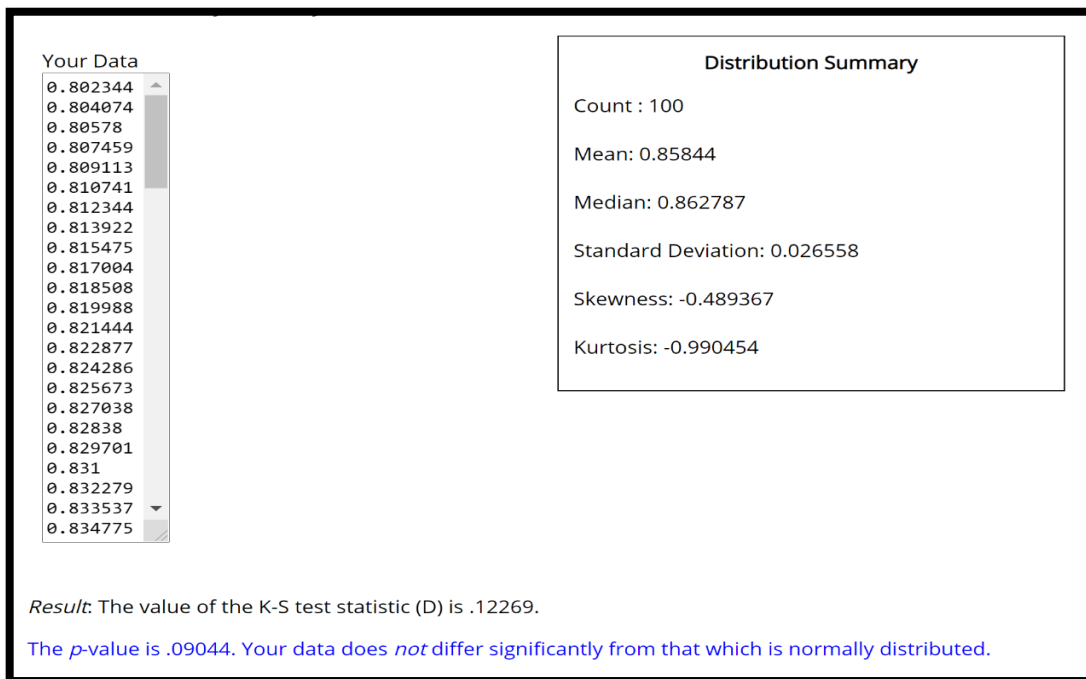


Figure 4.9: Normality test for the second column.

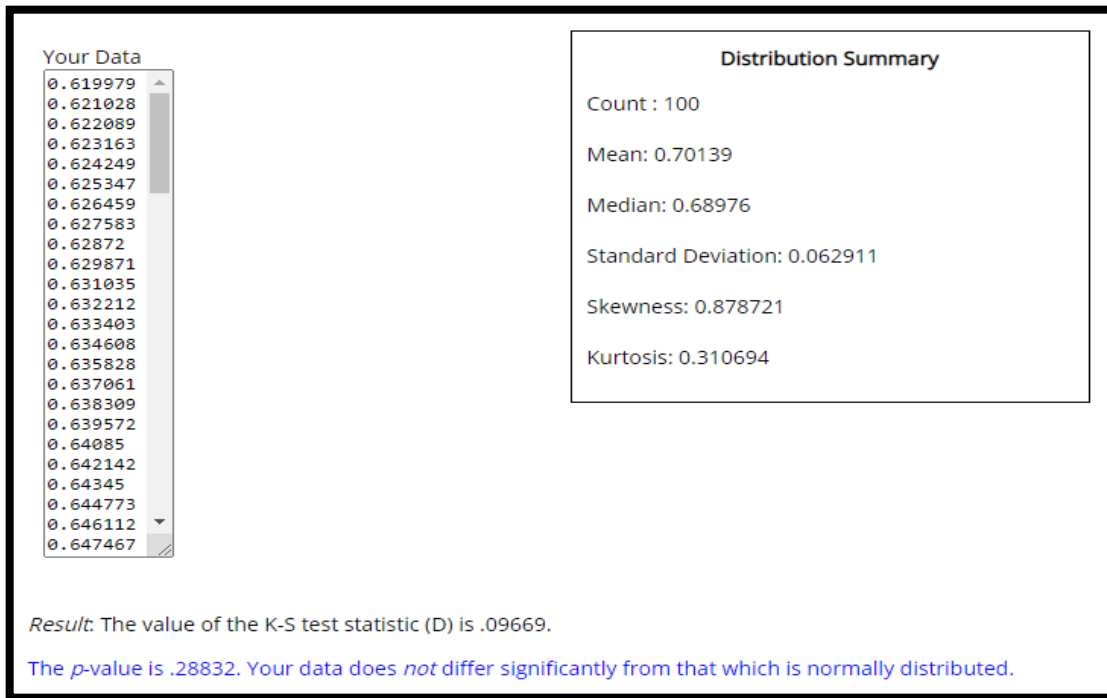


Figure 4.10: Normality test for the third column.

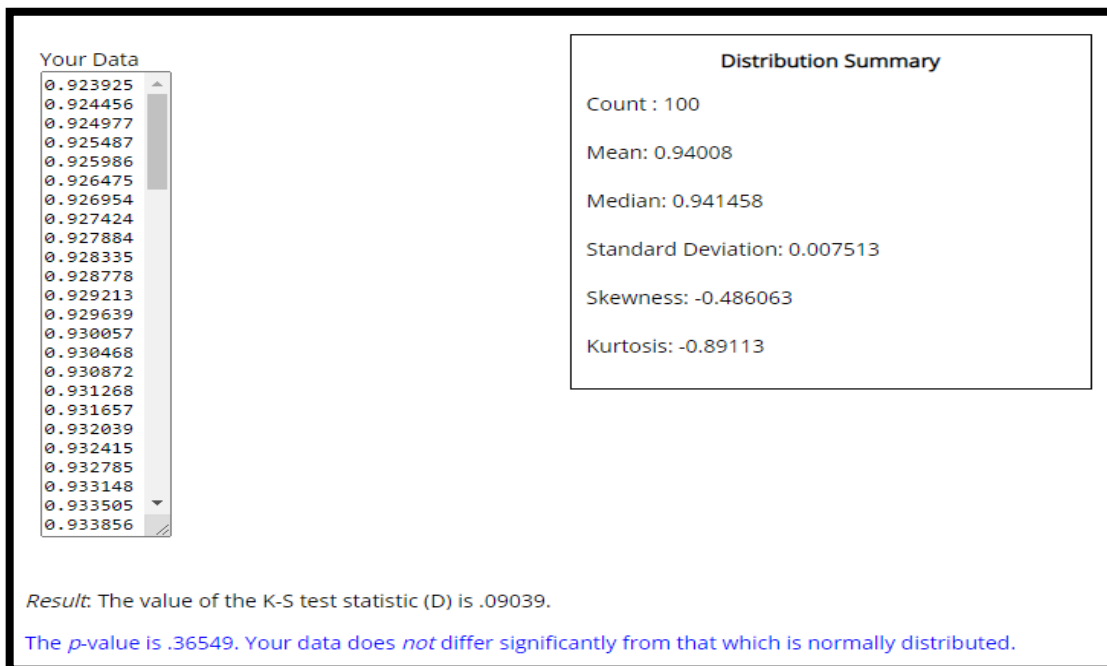


Figure 4.11: Normality test for the fourth column.

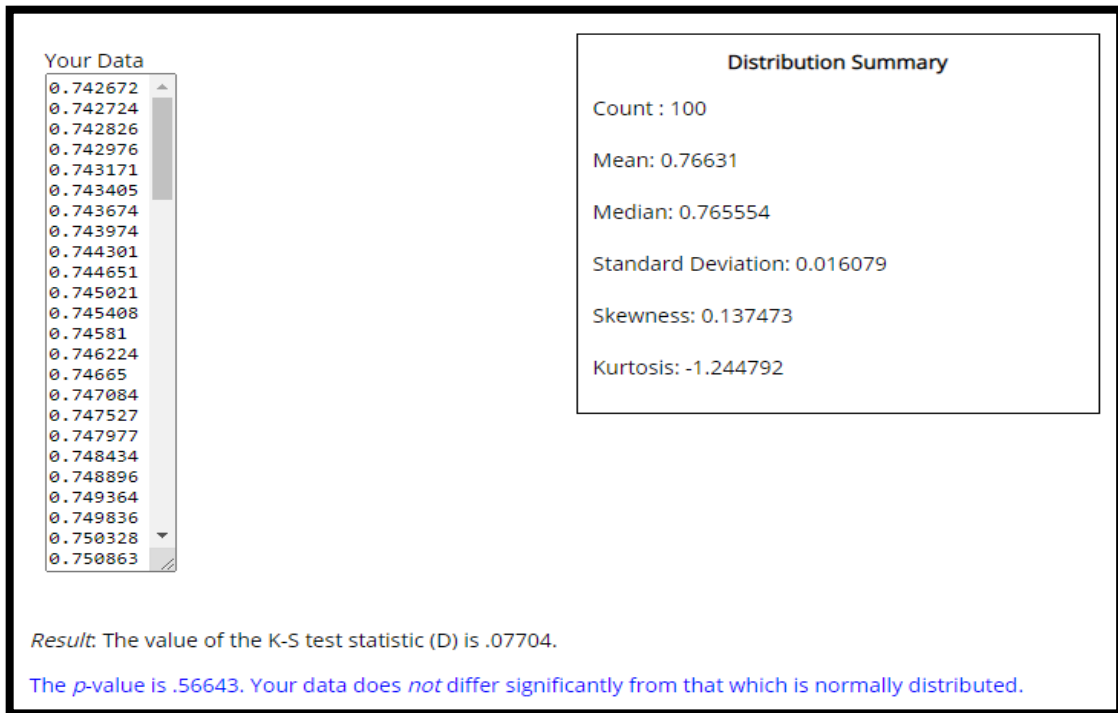


Figure 4.12: Normality test for the fifth column.

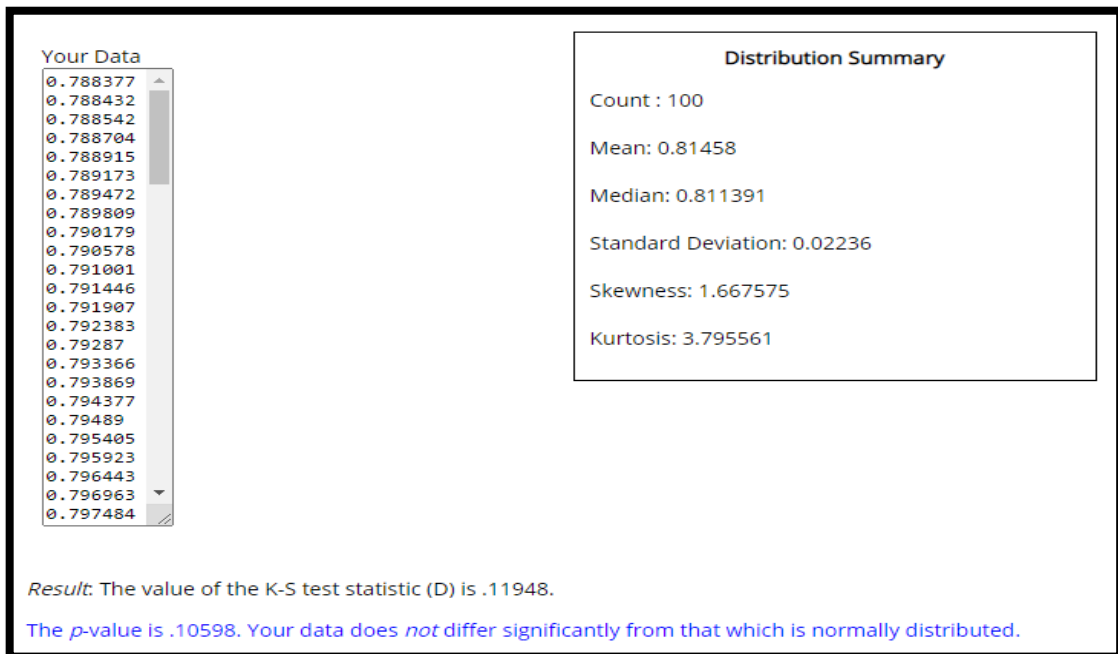


Figure 4.13: Normality test for the sixth column.

mean of 0.8584446, mode of 0.802344, and median of 0.862787 achieved in the second set of results), purporting commonality in the different data sets. Variability tests also persistently achieved low standard deviations and acceptable kurtosis measures.

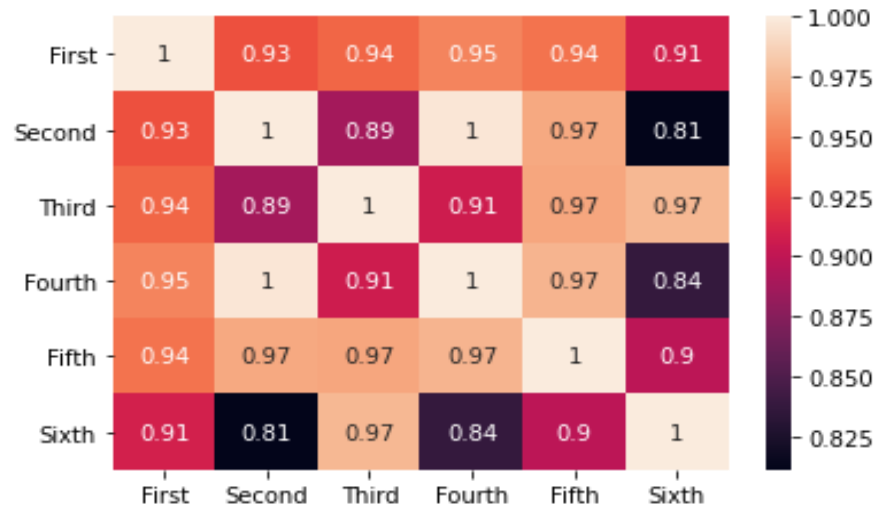


Figure 4.14: Correlation between testing results.

Figure 4.14 shows a heatmap plot used to explain the correlation among these data labels. The plot is also symmetrical since the same data fields are being paired. The high positive correlation coefficients observed throughout are indicative of well-related outcomes in each run. They indicate common trends. They point to possibilities of drawing inferences towards generalizations. However, we proceed to extract inferential views before arriving at conclusions.

T-tests are administered on pairs of the sets of results reported from every run in order to compare the means. F-tests follow to compare variation. A hypothetical population mean value of 0.3 is proposed before the T-tests are administered with the significance level of 0.05 and a one-tailed hypothesis. All

outcomes from T-tests are above 10.684625, with probability values less than 0.00001. Precisely, we observe T-values that significantly suggest means arising from the same hypothetical population. This indicates that the means we observe are true means not occurring by chance. It is possible to infer a mean to a new sample from observing these means. An analysis of variance (ANOVA) confirmed the same observations. Table 4.5 summarizes the measures reported in this category. Following this summary, table 4.6 shows a one-way ANOVA on the same data.

Because the p-value reported here is 0.00129, which is less than the significance level of 0.05, we do not have sufficient evidence with which to accept the null hypothesis. Rather, we conclude that the means and variations we observe are truly not occurring by chance. We accept the alternative hypothesis that plausible improvements to the RSA algorithm are observed when a neural network is incorporated to learn data integrity and authenticity towards a CIA-compliant RSA hybrid.

| SUMMARY | | | | | |
|---------|----------|----------|----------|-----------|--------------------|
| Groups | Median | Mode | Mean | Kurtosis | Standard Deviation |
| First | 0.552343 | 0.175374 | 0.504691 | -0.301438 | 0.191575 |
| Second | 0.862787 | 0.802344 | 0.858445 | -0.990454 | 0.026558 |
| Third | 0.68976 | 0.619979 | 0.701394 | 0.310694 | 0.062911 |
| Fourth | 0.941458 | 0.923925 | 0.940078 | -0.89113 | 0.007513 |
| Fifth | 0.765554 | 0.742672 | 0.766308 | -1.244792 | 0.016079 |
| Sixth | 0.811391 | 0.788377 | 0.814576 | 3.795561 | 0.02236 |

Table 4.5: Summary of the neural network's testing rates.

| ANOVA | | | | | |
|---------------------|----------|-----|----------|---------|----------|
| Source of Variation | SS | df | MS | P-value | F-value |
| Between Groups | 11.36469 | 5 | 2.272939 | 0.00129 | 2.229193 |
| Within Groups | 4.175727 | 594 | 0.00703 | | |
| Total | 15.54042 | 599 | | | |

Table 4.6: Analysis of variances in the neural network's testing rates.

4.4 Discussion of the results

Hypothesis testing is about seeking evidence to accept or reject a claim. In this work, we hypothesized that the inclusion of a neural network to learn data integrity and data authenticity in the RSA model would upgrade the resultant hybrid RSA model towards CIA compliance.

A null hypothesis was formulated which assumed insignificant effects of a neural network to the performance of the hybrid RSA model. An RSA model was developed which assessed a neural network's learning rate on training and testing data. The central tendencies reported, the measures of dispersion achieved, correlation analyses, and tests for normality on all the results achieved connoted mappable outcomes from the reported samples to hypothetical populations. Inferential analyses based on T and F tests yielded consistent results with the descriptive statistics achieved. We, therefore, failed to get grounds with which to accept the null hypothesis. Rather, sufficient deductive evidence based on the derived descriptive and inferential statistics provided grounds on which to believe that inclusion of a neural network to learn data integrity and data authenticity in the RSA model upgrades the resultant

hybrid RSA model towards CIA compliance. This is a notable improvement to the RSA model with the potentials to inspire further research and revive the application of the RSA model in commerce and business. The null hypothesis which states that *a neural network does not affect the enhancement of the RSA algorithm towards a CIA compliant RSA hybrid* was, therefore, rejected. An alternative hypothesis which states that *the observation of plausible improvement of the RSA algorithm from the incorporation of a neural network brings data integrity and data authenticity into the RSA algorithm towards a CIA compliant hybrid* was then accepted.

4.5 Summary

This chapter mainly collected data on the performance of the RSA predictive cryptosystem built on the notion that a neural network can learn data integrity and data authenticity from RSA's encrypted data towards a CIA-compliant hybrid RSA model. The design of the learning process using a training data set and a testing data set was explained. Neural network learning rates were reported from six replicated experiment runs. Analyses of those learning rates based on descriptive and inferential statistics showed common consistent trends towards normally distributed outcomes. These outcomes provided support with which the null hypothesis was rejected in favour of an alternative hypothesis which supports the view that inclusion of a neural network to learn data integrity and data authenticity in RSA encrypted data upgrades the hybrid RSA model towards CIA compliance. We arrived at and made this conclusion at a 95% level of confidence. The next and final chapter concludes this study.

Chapter 5 : Conclusion

This chapter concludes the project by summarizing the chapters, answering the research questions posed in chapter one, reviewing the contributions of the study, and pointing out avenues for future work. This is where we reflect on the findings of the study and the results yielded.

5.1 Summary of the chapters covered

The project was divided into five chapters. Below is a summary of what each chapter contributed to the project.

- The first chapter introduced the study by stating the statement of the problem as an investigation of ways of integrating the RSA algorithm with a neural network system that learns data integrity and data authenticity in RSA encrypted data towards a CIA compliant hybrid RSA model. The aim of the study was pinpointed as to investigate improvement to the RSA algorithm by incorporating a neural network that learns data integrity and data authenticity towards a CIA-compliant hybrid RSA model. Chapter 1 stated three objectives; (a) to implement the RSA algorithm, (b) to design and embed a neural network into the RSA algorithm to learn data integrity and data authenticity patterns, and (c) to evaluate the performances of the CIA compliant hybrid RSA model. Questions were asked in line with the three objectives. The first question sought to understand how the RSA algorithm is implemented. The second sought an understanding of how a neural network could be incorporated into the RSA model. The third question assessed the extent to which the hybrid RSA model satisfied CIA compliancy. The hypothesis of the study was stated as: H_0 -

incorporation of a neural network into the RSA algorithm will yield an upgraded hybrid RSA model which satisfies three of the five objectives of cryptographic algorithms - data confidentiality, data integrity, and data authenticity. Two motivating factors were pointed out as (a) the desire to assess the application of a neural network in the improvement of the RSA algorithm towards CIA compliance, and (b) curiosity to assess the extent to which bringing a cryptographic model and a machine learning system together works. Envisaged contributions were given as (a) creation of literature and new content in the fields, as well as (b) responding to most organizations' needs. The location of the study was indicated from a conceptual perspective before the limitations of the study closed the chapter.

- Chapter two presented related works on which our work is grounded. Precisely, we reviewed works concerned with the improvement of the RSA algorithm in general. We looked at literature related to the inclusion of neural networks in cryptography. The chapter closed by elucidating the gap we fill in the body of knowledge, also justifying the worthiness of the work here undertaken.
- Chapter three presented the methodology we followed, as well as the theoretical framework upon which most of our reasoning and argumentations are grounded. Precisely, a design science research methodology was assumed which emphasized the spiral design of the artifacts required for improving the RSA model and to prove the concept at hand.
- In chapter four, we generated results, interpreted the same, and discussed the bigger picture emanating. Experiments were conducted towards testing and training the designed predictive cryptosystem.

Plausible performances were noted, confirming the functionality of the proposed hybrid RSA model. The results yielded, on their own, are evidence, a milestone, and a deliverable of the work undertaken and presented in this study.

- In this chapter, we firstly summarize the chapters covered in the study and what each chapter offered to the study. We then explicitly present the answers to the research questions asked in the first chapter. Our philosophical reflection of the work, the findings, the results, and the conclusions arising is also presented in this chapter. The key contributions this study makes to the field and body of knowledge are also revisited in this chapter. Then, the potential direction for future work is discussed at the end of this chapter.

5.2 Answers to the research questions

Three questions were posed in chapter one. The first question sought the design and implementation of the RSA encryption/decryption algorithm. This aspect was successfully achieved (see related code in the appendices). Successful encryption and the reverse process thereto are sufficient results that serve as evidence of the achievement of the first objective, and an answer to the first question of the study.

The second question investigated the design and implementation of a neural network. A convolutional neural network arose which learns data integrity and data authenticity in RSA encrypted messages. The validity of the evidence of achievement of this objective is implicitly inferred in the results presented in chapter four.

The key deliverable of this work was the development of the RSA predictive cryptosystem which incorporates a neural network. This integrated hybrid was

successfully implemented, tried, and tested in chapter four. Data analysis on the results from the performances of the hybrid RSA model indicates generalizable commonalities in the different data sets of the results reported. In our view, the hybrid RSA model responded to the idea of including all three cryptographic objectives. The hybrid RSA model satisfied data confidentiality, data integrity, and data authenticity. Any central tendencies and variations observed when the hybrid RSA model was tested are insignificant, occurring by chance. This was statistically confirmed at a 95% level of confidence. The hypothesis that the inclusion of a neural network in the RSA model upgrades this cipher from handling only data confidentiality to also handling data integrity and data authenticity holds over the null hypothesis.

5.3 Reflections

The combination of the RSA algorithm with a neural network embraced the notion of hybridization. Hybridized cryptosystems apply mathematical rules which diffuse traces that enable brute force attacks to the resultant hybrid cipher. Data encrypted using the RSA model and signed using a neural network satisfy data confidentiality, data integrity, and data authentication. Data integrity and data authentication are digital signature issues that are indispensable in today's online transactions. Our intervention and contribution in this study are, thus, worthwhile.

5.4 Contributions

Successful design and implementation of the proposed RSA predictive cryptosystem came with valuable contributions both from an academic and from a practical point of view. Precisely:

- we added content to the body of knowledge. We provided additional literature upon which future researchers can base arguments on.

- Various matters outside the scope of this research work have been observed. These are recommended as future work which upcoming researchers can explore. We thus contribute ideas for new research.
- Practical and commercial perspectives are envisioned to arise from this work. That alone renders the work worthwhile. Precisely, the RSA model may be re-considered on the market. Current RSA users may also upgrade their systems to enjoy CIA-compliant siblings of the RSA model. Also, the improvements we reported bring us closer to fulfilling the expectation of most organizations and government entities in which data security is paramount.
- Above all, we have learned a few more aspects of computational designs and algorithms.

5.5 Future works

Improvement of the RSA algorithm where a neural network is used to learn features of the RSA algorithm is possible. However, the same improvement could, potentially, be achieved and optimized by incorporating machine learning algorithms such as the support vector machines or the principal component analysis method. We propose further studies which investigate the quality of CIA compliance and the relative performance of other potential hybrid RSA models built using different machine learning algorithms.

References

Agarwal, N. and Agarwal, P., 2013. Use of Artificial Neural Network in the field of Security. *In the MIT International Journal of Computer Science & Information Technology*, 3(1), pp.42-44.

Ahmed, J.M. and Ali, Z.M., 2011. The enhancement of computation technique by combining RSA and El-Gamal Cryptosystems. *In the Proceedings of the 2011 International Conference on Electrical Engineering and Informatics* (pp. 1-5).

Aiguokhian, E., 2013. Supply Chain Security Using RSA Algorithm. *Savonia University of Applied Sciences*.

Aiswarya, P.M., Raj, A., John, D., Martin, L. and Sreenu, G., 2016. Binary RSA encryption algorithm. *In the 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)* (pp. 178-181).

Alallayah, K.M., Alhamami, A.H., AbdElwahed, W. and Amin, M., 2012. Apply neural networks for simplified data encryption standard (SDES) cipher system cryptanalysis. *In the International Arab Journal of Information Technology*., 9(2), pp. 163-169.

Al-Hamami, A.H. and Aldariseh, I.A., 2012. Enhanced method for RSA cryptosystem algorithm. *In the 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT)* (pp. 402-408).

Alhassan, J.K., Ismaila, I., Waziri, V.O. and Abdulkadir, A., 2016. A Secure Method to Hide Confidential Data Using Cryptography and Steganography. A

technical report submitted at the Federal University of Technology, Minna, Nigeria.

Amalarethinam, I.G. and Leena, H.M., 2017. Enhanced RSA algorithm with varying key sizes for data security in cloud. *In the 2017 World Congress on Computing and Communication Technologies (WCCCT) (pp. 172-175).*

Aman, W. and Snekkenes, E., 2013. An empirical research on InfoSec Risk Management in IoT-based eHealth. *In the Proceedings of the Third International Conference on Mobile Services, Resources, and Users (MOBILITY 2013), Lisbon, Portugal (pp. 17-22).*

Andrea, I., Chrysostomou, C. and Hadjichristofi, G., 2015. Internet of Things: Security vulnerabilities and challenges. *In the 2015 IEEE Symposium on Computers and Communication (ISCC) (pp. 180-187). IEEE.*

Ayub, M.A., Onik, Z.A. and Smith, S., 2019. Parallelized RSA Algorithm: An Analysis with Performance Evaluation using OpenMP Library in High Performance Computing Environment. *In the 2019 22nd International Conference on Computer and Information Technology (ICCIT) (pp. 1-6). IEEE.*

Backhouse, R. and Ferreira, J.F., 2011. On Euclide's algorithm and elementary number theory. *In the Science of Computer Programming, 76(3), pp.160-180.*

Bangju, W. and Huanguo, Z., 2006. A new fast modular arithmetic method in public key cryptography. *In the Wuhan University Journal of natural sciences, 11(6), pp.1645-1648.*

Bhanot, R. and Hans, R., 2015. A review and comparative analysis of various encryption algorithms. *In the International Journal of Security and Its Applications, 9(4), pp.289-306.*

Bhattacharjya, A., Zhong, X. and Li, X., 2019. A Lightweight and Efficient Secure Hybrid RSA (SHRSA) Messaging Scheme with Four-Layered Authentication Stack. *In the IEEE Access*, 7, pp.30487-30506.

Blaisdell, J. and Vuong, T., MOCANA CORP, 2010. Firewall propagation. *In the U.S. Patent 7,853,998*.

Blömer, J., Otto, M. and Seifert, J.P., 2003. A new CRT-RSA algorithm secures against bellcore attacks. *In the Proceedings of the 10th ACM conference on Computer and communications security* (pp. 311-320).

Bodkhe, R. and Jethani, V., 2015. Hybrid encryption algorithm based improved RSA and Diffie-Hellman. *In the International Journal of Engineering, Science and Mathematics*, 4(1), pp.1-15.

Botella, J., Legnard, B., Peureux, F. and Vernotte, A., 2014. Risk-based vulnerability testing using security test patterns. *In the International Symposium on Leveraging Applications of Formal Methods, Verification and Validation* (pp. 337-352). Springer, Berlin, Heidelberg.

Cavallar, S., Dodson, B., Lenstra, A.K., Lioen, W., Montgomery, P.L., Murphy, B., Te Riele, H., Aardal, K., Gilchrist, J., Guillerm, G. and Leyland, P., 2000. Factorization of a 512-bit RSA modulus. *In the International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 1-18). Springer, Berlin, Heidelberg.

Cha, I., Schmidt, A., Leicher, A. and Shah, Y.C., 2014. Method and apparatus for trusted federated identity management and data access authorization. *In the InterDigital Patent Holdings Inc, U.S. Patent 8,881,257*.

Chakraborty, M., Jana, B., Mandal, T. and Kule, M., 2018. A Performance Analysis of RSA Scheme Using Artificial Neural Network. *In the 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp1-5).*

Crawford, K. and Schultz, J., 2014. Big data and due process: Toward a framework to redress predictive privacy harms. *In the BCL Rev.*, 55, p.93.

Dey, S., 2012. SD-AREE-I Cipher: Amalgamation of Bit Manipulation, Modified VERNAM CIPHER & Modified Caesar Cipher (SD-AREE). *In the International Journal of Modern Education and Computer Science*, 4(6), p.43.

Dworkin, M.J., 2010. Recommendation for block cipher modes of operation: The XTS-AES mode for confidentiality on storage devices. *In a Special Publication (NIST SP)-800-38E).*

Frunza, M. and Scripcariu, L., 2007. Improved RSA encryption algorithm for increased security of wireless networks. *In the 2007 International symposium on signals, circuits and systems (Vol. 2, pp. 1-4).*

Galbraith, S.D., 2012. Mathematics of public key cryptography. *In the Cambridge University Press.*

Garg, P. and Sharma, V., 2014. An efficient and secure data storage in Mobile Cloud Computing through RSA and Hash function. *In the 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT) (pp. 334-339).*

Gola, K.K., Gupta, B. and Iqbal, Z., 2014. Modified RSA digital signature scheme for data confidentiality. *In the International Journal of Computer Applications*, 106(13).

Goshwe, N.Y., 2013. Data encryption and decryption using RSA algorithm in a network environment. *In the International Journal of Computer Science and Network Security (IJCSNS)*, 13(7), p.9.

Guliyev, N.J. and Ismailov, V.E., 2016. A single hidden layer feedforward network with only one neuron in the hidden layer can approximate any univariate function. *In the neural computation*, 28(7), pp.1289-1304.

Hauer, B., 2015. Data and information leakage prevention within the scope of information security. *In the IEEE Access*, 3, pp.2554-2565.

Hecht-Nielsen, R., 1992. Theory of the backpropagation neural network. *In the Neural networks for perception* (pp. 65-93). Academic Press.

Hevner, A. and Chatterjee, S., 2010. Design science research in information systems. *In the Design research in information systems* (pp. 9-22). Springer, Boston, MA.

Hevner, A.R., 2007. A three-cycle view of design science research. *In the Scandinavian journal of information systems*, 19(2), p.4.

Jahan, I., Asif, M. and Rozario, L.J., 2015. Improved RSA cryptosystem based on the study of number theory and public key cryptosystems. *In the American Journal of Engineering Research (AJER)*, 4(1), pp.143-149.

Kallam, S., 2015. Diffie-Hellman: Key exchange and public key cryptosystems. *A Master of Science degree in Math and Computer Science, India State University, USA*, pp.5-6.

Karssenbergh, D., de Jong, K. and Van Der Kwast, J., 2007. Modelling landscape dynamics with Python. *In the International Journal of Geographical Information Science*, 21(5), pp.483-495.

Khairina, N. and Harahap, M.K., 2019. RSA Cryptographic Algorithm using Cubic Congruential Generator. *In the Journal of Physics: Conference Series (Vol. 1424, No. 1, p. 012010)*. IOP Publishing.

Khalfan, A.M., 2004. Information security considerations in IS/IT outsourcing projects: a descriptive case study of two sectors. *In the International Journal of Information Management, 24(1)*, pp.29-42.

Khan, Z., Pervez, Z. and Abbasi, A.G., 2017. Towards a secure service provisioning framework in a smart city environment. *In Future Generation Computer Systems, 77*, pp.112-135.

Knudsen, L.R., 1993. Practically secure Feistel ciphers. *In the International Workshop on Fast Software Encryption (pp. 211-221)*. Springer, Berlin, Heidelberg.

Krishnamoorthy, M. and Perumal, V., 2017. Secure and efficient hand-over authentication in WLAN using elliptic curve RSA. *In Computers & Electrical Engineering, 64*, pp.552-566.

Kumaravel, S. and Marimuthu, R., 2007. VLSI implementation of high-performance RSA algorithm using vedic mathematics. *In the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007) (Vol. 4, pp. 126-128)*.

Kuswaha, S., Waghmare, S. and Choudhary, P., 2015. Data Transmission using AES-RSA Based Hybrid Security Algorithms. *In the International Journal on Recent and Innovation Trends in Computing and Communication, 3(4)*, pp.1964-1969.

Lee, S. and Choeh, J.Y., 2014. Predicting the helpfulness of online reviews using multilayer perceptron neural networks. *In the journal of Expert Systems with Applications*, 41(6), pp.3041-3046.

Li, M., Lou, W. and Ren, K., 2010. Data security and privacy in wireless body area networks. *In the IEEE Wireless communications*, 17(1), pp.51-58.

Lin, H.Y., Hsu, C.L. and Huang, S.K., 2011. Improved convertible authenticated encryption scheme with provable security. *In the Information Processing Letters*, 111(13), pp.661-666.

Marium, S., Nazir, Q., Ahmed, A., Ahthasham, S. and Mirza, A.M., 2012. Implementation of EAP with RSA for enhancing the security of cloud computing. *In the International Journal of Basic and Applied Science*, 1(3), pp.177-183.

Marshall, G. and Jonker, L., 2011. An introduction to inferential statistics: A review and practical guide. *In Radiography*, 17(1), pp.e1-e6.

Meletiou, G., Tasoulis, D.K. and Vrahatis, M.N., 2002. A first study of the neural network approach to the RSA cryptosystem. In *IASTED 2002 Conference on Artificial Intelligence* (pp. 483-488).

Menezes, A.J., Katz, J., Van Oorschot, P.C. and Vanstone, S.A., 1996. Handbook of applied cryptography. *In the CRC press*.

Miller, W.J. and Trbovich, N.G., RACAL MILGO Inc, 1982. RSA Public-key data encryption system having large random prime number generating microprocessor or the like. *In the U.S. Patent 4,351,982*.

Mishra, S., Singh, S. and Ali, S.T., 2018. RCSDS: RSA based Cross Domain Secure Deduplication on Cloud Storage. *In the 2018 9th International*

Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-7).

Nguyen, P.Q., 2009. Public-key cryptanalysis. *In the Recent Trends in Cryptography, Contemp. Math, 477, pp.67-120.*

Nozaki, H., Motoyama, M., Shimbo, A. and Kawamura, S., 2001. Implementation of RSA algorithm based on RNS Montgomery multiplication. *In the International Workshop on Cryptographic Hardware and Embedded Systems (pp. 364-376). Springer, Berlin, Heidelberg.*

Nunamaker, J.F., Dennis, A.R., Valacich, J.S., Vogel, D. and George, J.F., 1991. Electronic meeting systems. *In Communications of the ACM, 34(7), pp.40-61.*

Odeh, A., Elleithy, K., Alshowkan, M. and Abdelfattah, E., 2013. Quantum key distribution by using public key algorithm (RSA). *In the Third International Conference on Innovative Computing Technology (INTECH 2013) (pp. 83-86).*

Omolara, O.E., Oludare, A.I. and Abdulahi, S.E., 2014. Developing a modified Hybrid Caesar cipher and Vigenere cipher for secure Data Communication. *In the Computer Engineering and Intelligent Systems, 5(5).*

Ora, P. and Pal, P.R., 2015. Data security and integrity in cloud computing based on RSA partial homomorphic and MD5 cryptography. *In the 2015 International Conference on Computer, Communication and Control (IC4) (pp. 1-6).*

Paar, C. and Pelzl, J., 2009. Understanding cryptography: a textbook for students and practitioners. *In the Springer Science & Business Media.*

Peltier, T.R., 2016. Information Security Policies, Procedures, and Standards: guidelines for effective information security management. *In the CRC Press*.

Pfleeger, C.P. and Pfleeger, S.L., 2012. Analyzing computer security: a threat/vulnerability/countermeasure approach. *In the Prentice Hall Professional*.

Plesa, M.I. and Mihai, T., 2018. A new quantum encryption scheme. *In the Advanced Journal of Graduate Research*, 4(1), pp.59-67.

Rahim, R. and Ikhwan, A., 2016. Cryptography technique with modular multiplication block cipher and playfair cipher. *In the International Journal of Science Research. Sci. Technol*, 2(6), pp.71-78.

Rahim, R., Winata, H., Zulkarnain, I. and Jaya, H., 2017. Prime number: an experiment Rabin-Miller and fast exponentiation. *In the Journal of Physics. Conf. Ser (Vol. 930, No. 1, p. 012032)*.

Rahman, M.N.A., Abidin, A.F.A., Yusof, M.K. and Usop, N.S.M., 2013. Cryptography: A new approach of classical Hill cipher. *In the International Journal of Security and Its Applications*, 7(2), pp.179-190.

Ren, W. and Miao, Z., 2010. A hybrid encryption algorithm based on DES and RSA in bluetooth communication. *In the 2010 Second International Conference on Modeling, Simulation and Visualization Methods (pp. 221-225)*.

Rubin, F., 1996. One-time pad cryptography. *In the Cryptologia*, 20(4), pp.359-364.

Sadikin, M.A. and Wardhani, R.W., 2016. Implementation of RSA 2048-bit and AES 256-bit with digital signature for secure electronic health record

application. *In the 2016 International Seminar on Intelligent Technology and Its Applications (ISITIA) (pp. 387-392).*

Samonas, S. and Coss, D., 2014. The CIA strikes back: Redefining confidentiality, integrity and availability in security. *In the Journal of Information System Security, 10(3).*

Saranya Jothi, C., Usha, V. and Alex David, S., 2017. Dynamic Data Integrity and Checkpoint Recovery Using Public Auditing in Cloud Storage. *In the International Journal of Civil Engineering and Technology, 8(9).*

Saxena, R., Jain, M., Singh, D. and Kushwah, A., 2017. An enhanced parallel version of RSA public key crypto based algorithm using OpenMP. *In the Proceedings of the 10th International Conference on Security of Information and Networks (pp. 37-42).*

Shantz, S.C., 2001. From Euclid's GCD to Montgomery multiplication to the great divide. *In the Sun Microsystems, Inc.*

Simon, H.A., 1996. The sciences of the artificial. *In the MIT press.*

Singh, G., 2013. A study of encryption algorithms (RSA, DES, 3DES and AES) for information security. *In the International Journal of Computer Applications, 67(19).*

Singh, S., 2000. The code book: the science of secrecy from ancient Egypt to quantum cryptography. *In the Anchor.*

Smith, J.P., 1995. Authentication of digital medical images with digital signature technology. *In the Radiology, 194(3), pp.771-774.*

Stallings, W., 2006. *Cryptography and network security. 4th Edition. Pearson Education India.*

Stengel, R., 2017. *Introduction to Neural Networks. Robotics and Intelligent Systems, Princeton University, MAE 345.*

Syam Kumar, P. and Subramanian, R., 2012. RSA-based dynamic public audit service for integrity verification of data storage in cloud computing using Sobol sequence. *In the International Journal of Cloud Computing, 1(2-3), pp.167-200.*

Thayananthan, V. and Albeshri, A., 2015. Big data security issues based on quantum cryptography and privacy with authentication for mobile data center. *In the Procedia Computer Science, 50, pp.149-156.*

Van Tilborg, H.C. and Jajodia, S. eds., 2014. *Encyclopedia of cryptography and security. In the Springer Science & Business Media.*

Venable, J., 2006. A framework for design science research activities. In *Emerging Trends and Challenges in Information Technology Management: In the Proceedings of the 2006 Information Resource Management Association Conference (pp. 184-187). Idea Group Publishing.*

Venkatesh, M., Sumalatha, M.R. and SelvaKumar, C., 2012. Improving public auditability, data possession in data storage security for cloud computing. *In the 2012 International Conference on Recent Trends in Information Technology (pp. 463-467).*

Venkatraman, S. and Overmars, A., 2019. New Method of Prime Factorisation-Based Attacks on RSA Authentication in IoT. *In Cryptography, 3(3), p.20.*

Wu, T.S. and Lin, H.Y., 2009. Secure convertible authenticated encryption scheme based on RSA. *In the Informatica*, 33(4).

Xu, R., Joshi, J.B. and Li, C., 2019, July. Cryptonn: Training neural networks over encrypted data. *In the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)* (pp. 1199-1209).

Yamuna, V. and Anusha Priya, A., 2015. Efficient and Secure Data Storage in Cloud Computing RSA and DSE Function. *In the International Journal of Innovative Research in Computer and Communication Engineering*, 3(7), pp.6758-6763.

Yavuz, E., Yazici, R., Kasapbaşı, M.C. and Yamaç, E., 2016. A chaos-based image encryption algorithm with simple logical functions. *In Computers & Electrical Engineering*, 54, pp.471-483.

Yu, Y., Xue, L., Au, M.H., Susilo, W., Ni, J., Zhang, Y., Vasilakos, A.V. and Shen, J., 2016. Cloud data integrity checking with an identity-based auditing mechanism from RSA. *In the Future Generation Computer Systems*, 62, pp.85-91.

Zhou, X. and Tang, X., 2011. Research and implementation of RSA algorithm for encryption and decryption. *In the Proceedings of 2011 6th international forum on strategic technology* (Vol. 2, pp. 1118-1121).

Appendix A- Screenshots of Important code

```
▶ import random
import sys
import math
import pandas as pd
import numpy as np
from random import randrange
import csv
import itertools
from sklearn.model_selection import train_test_split
from matplotlib import pyplot as plt
import scipy.stats as stats
#import keras
```

```
▶ def rabinMiller(n, k=10):
    if n == 2:
        return True
    if not n & 1:
        return False

    def check(a, s, d, n):
        x = pow(a, d, n)
        if x == 1:
```

```
        return True
    for i in range(1, s - 1):
        if x == n - 1:
            return True
        x = pow(x, 2, n)
    return x == n - 1

s = 0
d = n - 1

while d % 2 == 0:
    d >>= 1
    s += 1

for i in range(1, k):
    a = randrange(2, n - 1)
    if not check(a, s, d, n):
        return False

return True
```

```
def isPrime(n):  
    lowPrimes = [3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,  
                ,101,103,107,109,113,127,131,137,139,149,151,157,163,167,173,179,  
                ,181,191,193,197,199,211,223,227,229,233,239,241,251,257,263,269,  
                ,271,277,281,283,293,307,311,313,317,331,337,347,349,353,359,367,  
                ,373,379,383,389,397,401,409,419,421,431,433,439,443,449,457,461,  
                ,463,467,479,487,491,499,503,509,521,523,541,547,557,563,569,571,  
                ,577,587,593,599,601,607,613,617,619,631,641,643,647,653,659,661,  
                ,673,677,683,691,701,709,719,727,733,739,743,751,757,761,769,773,  
                ,787,797,809,811,821,823,827,829,839,853,857,859,863,877,881,883,  
                ,887,907,911,919,929,937,941,947,953,967,971,977,983,991,997,1009,  
                ,1013,1019,1021,1031,1033,1039,1043,1051,1061,1063,1069,1087,1091,  
                ,1093,1097,1103,1109,1117,1123,1129,1151,1153,1163,1171,1181,1187,  
                ,1193,1201,1213,1217,1223]  
  
    if (n >= 3):  
        if (n&1 != 0):  
            for p in lowPrimes:  
                if (n == p):  
                    return True  
                if (n % p == 0):  
                    return False  
            return rabinMiller(n)  
    return False
```

```
▶ def generateLargePrime(k):
    #k is the desired bit length
    r = 1000*(math.log(k,2)+1) #number of attempts max
    r_ = r
    while r>0:
        #randrange is mersenne twister and is completely deterministic
        #unusable for serious crypto purposes
        n = random.randrange(2**(k-1),2**(k))
        r -= 1
        if isPrime(n) == True:
            return n

    str_failure = "Failure after" + str(r_) + "tries."
    return str_failure
```

```
▶ def gcd(a, b):
    '''
    Euclid's algorithm for determining the greatest common divisor
    Use iteration to make it faster for larger integers
    '''
    while b != 0:
        a, b = b, a % b
    return a
```

```
▶ def multiplicative_inverse(a, b):  
    """Returns a tuple (r, i, j) such that  $r = \gcd(a, b) = ia + jb$   
    """  
    x = 0  
    y = 1  
    lx = 1  
    ly = 0  
    oa = a # Remember original a/b to remove  
    ob = b # negative values from return results  
    while b != 0:  
        q = a // b  
        (a, b) = (b, a % b)  
        (x, lx) = ((lx - (q * x)), x)  
        (y, ly) = ((ly - (q * y)), y)  
    if lx < 0:  
        lx += ob # If neg wrap modulo original b  
    if ly < 0:  
        ly += oa # If neg wrap modulo original a  
    return lx
```

```
▶ def rwh_primes2(n):  
    """ Input  $n \geq 6$ , Returns a list of primes,  $2 \leq p < n$  """  
    correction = (n%6>1)  
    n = {0:n,1:n-1,2:n+4,3:n+3,4:n+2,5:n+1}[n%6]  
    sieve = [True] * (n/3)
```

```
sieve[0] = False
for i in xrange(int(n**0.5)/3+1):
    if sieve[i]:
        k=3*i+1|1
        sieve[ ((k*k)/3) ::2*k]=[False]*((n/6-(k*k)/6-1)/k+1)
        sieve[(k*k+4*k-2*k*(i&1))/3::2*k]=[False]*((n/6-(k*k+4*k-2*k*(i&1))/6-1)/k+1)
return [2,3] + [3*i+1|1 for i in xrange(1,n/3-correction) if sieve[i]]
```

```
def multiply(x, y):
    _CUTOFF = 1536
    if x.bit_length() <= _CUTOFF or y.bit_length() <= _CUTOFF: # Base case
        return x * y
    else:
        n = max(x.bit_length(), y.bit_length())
        half = (n + 32) // 64 * 32
        mask = (1 << half) - 1
        xlow = x & mask
        ylow = y & mask
        xhigh = x >> half
        yhigh = y >> half
        a = multiply(xhigh, yhigh)
        b = multiply(xlow + xhigh, ylow + yhigh)
        c = multiply(xlow, ylow)
        d = b - a - c
        return (((a << half) + d) << half) + c
```



```
▶ def generate_keypair(keySize=10):  
    p = generateLargePrime(keySize)  
    q = generateLargePrime(keySize)  
    if p == q:  
        p = generateLargePrime(keySize)  
        q = generateLargePrime(keySize)  
    n = multiply(p, q)  
    phi = multiply((p-1),(q-1))  
    e = random.randrange(1, phi)  
    g = gcd(e, phi)  
    while g != 1:  
        e = random.randrange(1, phi)  
        g = gcd(e, phi)  
    d = multiplicative_inverse(e, phi)  
    return ((n, e), (p, q, phi, d))
```

```
▶ def encrypt(pk, plaintext):  
    key, n = pk  
    cipher = [(ord(char) ** key) % n for char in plaintext]  
    return cipher
```

```
▶ def getheaders(headers):  
    coln = {}  
    for h in headers:  
        coln[h] = []  
    return coln
```

```
▶ with open('../data/Health_Records.csv') as f:

    spamreader = csv.reader(f, delimiter=",")
    headers = getheaders(next(spamreader, None))

    for row in itertools.islice(spamreader, 1, 101, 1):
        res = []
        #output = []
        for h, word in zip(headers, row):
            res.extend(ord(num) for num in word)
            ascii_values = str(res)
            public, private = generate_keypair()
            ascii_values = str(sys.argv[1])
            encrypted_msg = encrypt(public, ascii_values) #C = P^e(mod n)

            encrypted_value = ''.join(map(lambda x: str(x), encrypted_msg))
            headers[h].append(encrypted_value)

df = pd.DataFrame(data = headers)
df
```

```
▶ df_train, df_test = train_test_split(df, test_size=0.5)
df_train.to_csv('../data/Train_Encrypted_data.csv')
df_test.to_csv('../data/Test_Encrypted_data.csv')
```

```
▶ mean_ls = []
mode_ls = []
median_ls = []
for _ in range(6):
    #-----|-----
    def act_fun(x):
        '''
            the activation function (currently logistic function)
        input :
            x: a n x 1 numpy ndarray
        return: a n x 1 numpy ndarray
        '''
        bias = -.5
        return 1/(1 + np.exp(-x + bias))

    #-----|-----
    def der_act_fun(x):
        '''the derivative of the activation function
        input :
            a_vec: a nx1 numpy ndarray
        return:
            a nx1 numpy ndarray
        '''
        bias = -.5
        return (np.exp(-x + bias)/np.square(1 + np.exp(-x + bias)))
```

```
def one_hot(labels, n_classes):
    """given a vector of n numerical labels, it creates an array of dimension
    where each row contains a 'one hot' vector.
    ie. for 10 classes labeled by the digits 1-10:
    1 -> 1 0 0 0 0 0 0 0
    2 -> 0 1 0 0 0 0 0 0
    ...
    10 -> 0 0 0 0 0 0 0 1
    input:
    labels : type ndarray of size (1 x n_examples)
    n_classes : type int
    output:
    one_hot_labels : type ndarray of size (n_examples x n_classes)
    """
    num_labels=len(labels)

    one_hot_labels = np.zeros([num_labels,n_classes])

    for i in range(num_labels):
        j = labels[i]
        one_hot_labels[i,j] = 1

    return one_hot_labels
```

```
class layer(object):
    '''layer objects contains several important attributes:
    → self.weights - a ndarray of dimension (n_in x n_nodes)
    → self.wsum - the weighted sum of inputs into the layer
    → self.deltas - the deltas for this layer
    → self.activations - the activations of the layer
    '''
    def __init__(self, n_in, n_nodes, act_fun='sigmoid'):
        '''random initialization of weights for this layer
        → inputs:
        → n_in: type: int
        → n_nodes: type: ints
        '''
        if (act_fun=='sigmoid'):
            scale_fac = 4*np.sqrt(6)/(np.sqrt(n_in+n_nodes))
            self.weights = scale_fac*(2*np.random.rand(n_nodes,n_in) - 1)
            self.size = n_nodes
            self.wsum = np.zeros(n_nodes)
            self.deltas = np.zeros(n_nodes)
            self.activ = np.zeros(n_nodes)

    def activate(self,inp):
        self.wsum = self.weights.dot(inp)
        self.activ = act_fun(self.wsum)
        return self.activ
```

```

class neural_net(object):
    → '''A neural_net object is simply a list of layers'''

    → #-----
    → def __init__(self, n_in, nodes_per_layer):
    →     self.n_layers = len(nodes_per_layer)
    →     self.ni = n_in
    →     self.no = nodes_per_layer[self.n_layers-1]
    →     self.layer_sizes=nodes_per_layer
    →     self.layers = []
    →
    →     #create first layer (layer 0)
    →     self.layers += [layer(n_in, nodes_per_layer[0])]
    →
    →     #create hidden layers and output layer
    →     for i in range(1, self.n_layers):
    →         self.layers += [layer(nodes_per_layer[i-1],nodes_per_layer[i])]

```

```

→ def activate(self, vec):
→     '''activate each layer of the neural network
→     input :
→     vec: the input vector. type: ndarray of size (1 x n_inputs)
→     output:
→     vec: activation of the last layer. type: ndarray of size (1 x n_out)
→     '''
→     for i in range(self.n_layers):
→         vec = self.layers[i].activate(vec)
→     return vec

```

```

def backpropagation(self, activation, target, learning_rate):
    '''performs backpropagation on the network
    Input:
    activation: the activations. type: ndarray of size (1 x n_inputs)
    target: the target. type: ndarray of size (1 x n_inputs)
    learning_rate: type: float
    Output:
    '''
    n_layers = self.n_layers
    layers = self.layers

    #calculate deltas for last layer
    layers[n_layers-1].deltas = der_act_fun(layers[n_layers-1].wsum)*(target - activation)

    #calculate deltas for hidden layers
    for l in range(n_layers-2, 0, -1):
        layers[l].deltas = der_act_fun(layers[l].wsum)*(np.transpose(layers[l+1].weights).d
    #update weights for last and hidden layers with the deltas
    for l in range(n_layers-1, 1, -1):
        for i in range(layers[l].size):
            for j in range(layers[i-1].size):
                layers[l].weights[i,j] = layers[l].weights[i,j] + learning_rate*layers[l].d

```

```

*—> for i in range(layers[0].size):
*—> *—> for j in range(len(target)):
*—> *—> *—> layers[0].weights[i,j] = layers[0].weights[i,j] + learning_rate*layers[0].delta

#-----
*—> def training_epoch(self, data, targets, learning_rate):
*—> *—> '''Input:
*—> *—> *—> data : batch of training inputs, stored in rows. type: ndarray of size (n_exam
*—> *—> *—> targets : batch of training target outputs, stored in rows. type: ndarray of si
*—> *—> *—> learning_rate : type: float
*—> *—> *—> Output:
*—> *—> *—> avg_cost: the avg squared error during this epoch. type: float
*—> *—> *—> '''
*—> *—> n_examples = np.size(data,1)
*—> *—>
*—> *—> avg_cost = 0
*—> *—>
*—> *—> #online learning
*—> *—> for i in range(n_examples):
*—> *—> *—> activation = self.activate(data[i,:])
*—> *—> *—> avg_cost += sum(np.square(activation - targets[i,:]))
*—> *—> *—> self.backpropagation(activation, targets[i, :], learning_rate)
*—> *—> *—>
*—> *—> *—> avg_cost = avg_cost/n_examples
*—> *—> *—> return avg_cost

```



```
def regularization(self):
    '''return a regularization term for the neural network'''
    reg = 0
    for layer in self.layers:
        reg += sum(layer.weights**2)
    return reg

#-----
def cost(self, data):
    '''given a set of input data, calculates the cost, or error outputed by the nn
    this method is provided for use with advanced optimization routines
    '''

    num_inps = np.size(data,1)

    cost=0

    for i in range(num_inp):
        outvec = self(data[:,i])
        cost = cost + sum( ans[:,i]*np.log(outvec) + (1-ans[:,i])*np.log(1-outvec) )
        cost = cost + .1*nn.regularization()
    return cost
```

----- *Setting up and training* -----

```
def train(nn, n_epochs = 10000,
          learning_rate=.008,
          datafile='../data/Test_Encrypted_data.csv',
          nodes_per_layer=[10,10],
          regularization=True,
          reg_param=.01):
    raw_data = np.array([[0,0,0], [1,0,1], [1,1,0], [0,1,1]])
    targets = one_hot(labels=raw_data[:,0], n_classes=2)
    data = raw_data[:,1:]
    results = []
    for n in range(n_epochs):
        avg_cost = nn.training_epoch(data, targets, learning_rate)
        results.append(avg_cost)
    return results
if __name__ == '__main__':
    #build neural net
    nn = neural_net(n_in=2, nodes_per_layer=[5,10,2])
    results = train(nn)
```

```
df_test_results = pd.DataFrame({'test_values': results})
df_test_results.to_csv('../data/Testing_results.csv')
df_train_results = pd.DataFrame({'trained_values': results})
df_train_results.to_csv('../data/Training_results.csv')
```

Appendix B- Editing Certificate



Dr. J. Sibanda (Senior Lecturer: English)
School of Education
Private Bag X 5003, Kimberley, 5100
North Campus, Chapel Street, Kimberley

Website: www.spu.ac.za

18 March 2021

Certificate of language editing

To whom it may concern

I hereby confirm that I have proofread and edited the following MSc dissertation in Applied Mathematics using the Windows Tracking System and Grammarly to reflect my comments and suggested corrections for the author (s) to action.

Title: A Neural Network Enhanced RSA Model Towards a CIA Compliant Hybrid

Author(s) Promise Tshepiso Magoma supervised by Dr. Colin Chibaya (Sol Plaatje University) and Dr. Dephney Mathebula (University of Venda)

Although the greatest care was taken in the editing of this document, the final responsibility for the product rests with the author (s) who may accept or reject the proposed changes at their discretion.



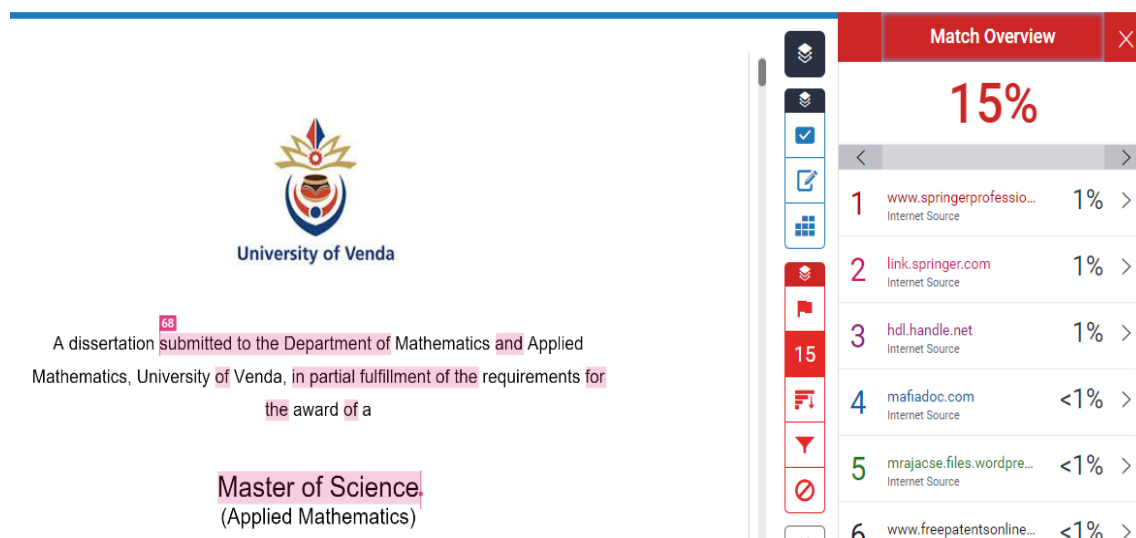
Signature

18 March 2021

Date

Appendix C – Similarity Report

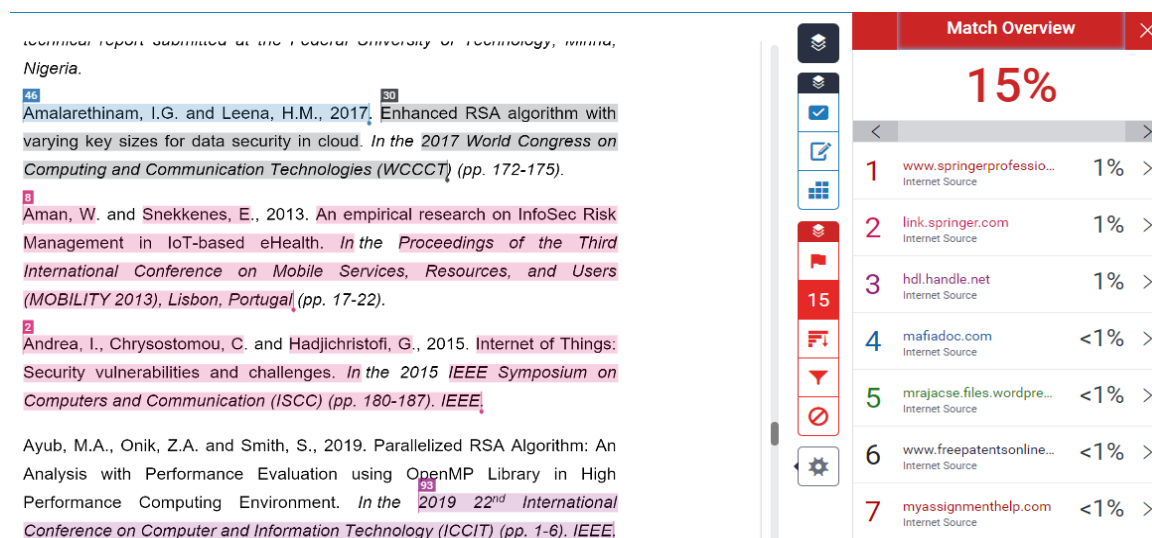
When searched from all repositories (internet, student papers, periodicals, journals, and publications) including title, preamble sections, and reference list.



The screenshot shows a document snippet on the left and a 'Match Overview' sidebar on the right. The document text is: 'A dissertation submitted to the Department of Mathematics and Applied Mathematics, University of Venda, in partial fulfillment of the requirements for the award of a Master of Science (Applied Mathematics)'. The sidebar shows a 15% match rate and a list of 6 matches, each with a 1% match rate.

| Match Number | Source | Match Rate |
|--------------|---|------------|
| 1 | www.springerprofessio... Internet Source | 1% |
| 2 | link.springer.com Internet Source | 1% |
| 3 | hdl.handle.net Internet Source | 1% |
| 4 | mafiadoc.com Internet Source | <1% |
| 5 | mrjacse.files.wordpre... Internet Source | <1% |
| 6 | www.freepatentonline... Internet Source | <1% |

Hits are mainly in the declaration, chapter headings, section headings, and in the reference list. See screenshot of one of the reference list pages.




The screenshot shows a reference list snippet on the left and a 'Match Overview' sidebar on the right. The reference list text includes: 'Amalarethinam, I.G. and Leena, H.M., 2017, Enhanced RSA algorithm with varying key sizes for data security in cloud. In the 2017 World Congress on Computing and Communication Technologies (WCCCT) (pp. 172-175).', 'Aman, W. and Snekkenes, E., 2013. An empirical research on InfoSec Risk Management in IoT-based eHealth. In the Proceedings of the Third International Conference on Mobile Services, Resources, and Users (MOBILITY 2013), Lisbon, Portugal (pp. 17-22).', and 'Andrea, I., Chrysostomou, C. and Hadjichristofi, G., 2015. Internet of Things: Security vulnerabilities and challenges. In the 2015 IEEE Symposium on Computers and Communication (ISCC) (pp. 180-187). IEEE.' The sidebar shows a 15% match rate and a list of 7 matches, each with a 1% match rate.

| Match Number | Source | Match Rate |
|--------------|---|------------|
| 1 | www.springerprofessio... Internet Source | 1% |
| 2 | link.springer.com Internet Source | 1% |
| 3 | hdl.handle.net Internet Source | 1% |
| 4 | mafiadoc.com Internet Source | <1% |
| 5 | mrjacse.files.wordpre... Internet Source | <1% |
| 6 | www.freepatentonline... Internet Source | <1% |
| 7 | myassignmenthelp.com Internet Source | <1% |

Below is the similarity report after excluding the reference list, keeping everything else the same.

A Neural Network Enhanced RSA Model Towards a CIA Compliant Hybrid



University of Venda

²¹
 A dissertation submitted to the Department of Mathematics and Applied
 Mathematics, University of Venda, in partial fulfillment of the requirements for
 the award of a

Master of Science
 (Applied Mathematics)

Match Overview

6%

| | | |
|---|--|-------|
| 1 | hdl.handle.net Internet Source | 1% > |
| 2 | mrajacse.files.wordpre... Internet Source | <1% > |
| 3 | www.freepatentsonline... Internet Source | <1% > |
| 4 | archive.org Internet Source | <1% > |
| 5 | id.scribd.com Internet Source | <1% > |
| 6 | mafiadoc.com Internet Source | <1% > |
| 7 | www.is.informatik.uni-... Internet Source | <1% > |

===== IN GOD WE TRUST =====