A PROJECT REPORT ON

NEURAL NETWORK BASED KEY CONCEALMENT SCHEME

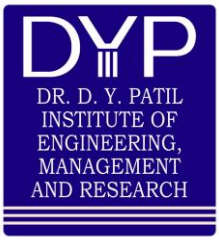
SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

SUBMITTED BY

Prasann Shimpi	**Exam No: B151094331**
Sanket Halake	**Exam No: B151094245**
Shivam Dharmshetti	**Exam No: B151094234**
Shashank Singh	**Exam No: B151094327**



DEPARTMENT OF COMPUTER ENGINEERING

DR. D. Y. PATIL INSTITUTE OF ENGINEERING, MANAGEMENT & RESEARCH

AKURDI, PUNE 411044
(2021-22)

SAVITRIBAI PHULE PUNE UNIVERSITY
2021 -2022

DR. D. Y. PATIL INSTITUTE OF ENGINEERING, MANAGEMENT AND RESEARCH

CERTIFICATE

This is to certify that the project report entitled

" NEURAL NETWORK BASED KEY CONCEALMENT SCHEME"

Submitted by

Prasann Shimpi	**Exam No: B151094331**
Sanket Halake	**Exam No: B151094245**
Shivam Dharmshetti	**Exam No: B151094234**
Shashank Singh	**Exam No: B151094327**

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of **Dr. Amol Dhakne** and it is approved for the fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

Dr. Amol Dhakne
Guide
Department of Computer Engineering

Prof. P.P.Shevatekar
Head,
Department of Computer Engineering

Dr. Anupama V. Patil
Principal,
Dr. D. Y. Patil Institute of Engineering, Management & Research Akurdi, Pune – 41

Place : Pune
Date : 19/05/2022

ACKNOWLEDGEMENT

It gives us great pleasure in presenting the project report on 'Neural Network based message concealment scheme'.

I would like to take this opportunity to thank my internal guide Dr. Amol Dhakne for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.

I am also grateful to Prof. P.P. Shevatekar, Head of Computer Engineering Department, Dr. D. Y. Patil Institute of Engineering, Management & Research for his indispensable support, suggestions.

In the end our special thanks to Dr. Amol Dhakne for providing various resources such as laboratory with all needed software platforms, continuous guidance, for Our Project.

Prasann Shimpi **Exam No: B151094331**
Sanket Halake **Exam No: B151094245**
Shivam Dharmshetti **Exam No: B151094234**
Shashank Singh **Exam No: B151094327**

ABSTRACT

The goal of cryptography is to make it impossible to take a cipher and reproduce the original plain text without the proper key. With good cryptography, your messages are encrypted in such a way that brute force attacks against the algorithm or key are almost impossible. Good cryptography keeps you safe by using incredibly long keys and using encryption algorithms resistant to other attacks. The neural network application is a form of the next development in good cryptography. This project is about the use of neural networks in cryptography, for example, the design of a neural network that is practically used in the field of cryptography.

Neural cryptography is a new field aimed at combining cryptography and neural networks for cryptanalysis and cryptography applications. This project shows that neural networks can perform symmetric cryptography in a hostile environment to improve the known literature on this topic. It also shows that neural networks can detect known unencrypted communications by playing well- known cryptographic games based on ciphertext**.**

In this project, we study the problem of key exchange using Artificial Neural Network. The purpose of this project is to examine the feasibility and performance of Artificial Intellegence in cryptography. We compare the performance and reliability of traditional key exchange against key exchange using ANNs. The Neural Network is trained in an adversarial fashion with distinct constructions for loss. The Accuracy of the performance of the neural network is compared using various out of sample performance measures.

TABLE OF CONTENTS

I LIST OF ABBREVATIONS

ABBREVIATION	ILLUSTRATION
HTTP	Hyper Text Transfer Protocol
MQTT	Message Query Telemetry Transport
ANN	Artificial Neural Networks
CNN	Convolutional Neural Networks

II. LIST OF FIGURES

III LIST OF TABLES

1. INTRODUCTION

1.1 OVERVIEW

Nowadays, we live in a data-driven world, and in most use cases, businesses collect and store sensitive personal and non-personal information, which can be exploited by cyber criminals.

Good cryptography keeps you safe by using incredibly long keys and using encryption algorithms resistant to other attacks. The neural network application is a form of the next development in good cryptography. This project is about the use of neural networks in cryptography, for example, the design of a neural network that is practically used in the field of cryptography. To demonstrate an application of ANNs in the field of cyber security, we create three neural networks: Alice, Bob and Eve. Two of these participate in a message exchange using symmetric encryption and the third acts as an adversary and observe the rate of success.

1.2 MOTIVATON

Cryptography is a method that enables a method of decrypting secrets, and is a function that transforms information into information that is not explicitly understood. The basic idea of encryption is the ability to send information between participants in a way that no one else can read. There are many number-theoretic cryptographic methods available, but they have the disadvantage of requiring a great deal of computational power, complexity, and wasting time. To overcome these shortcomings, artificial neural networks (ANNs) have been used to solve many problems. ANN has many characteristics such as learning, generalization, reduced data requirements, fast computing, easy implementation, software and hardware availability, and is veryattractive to many applications.

Work on artificial neural network has been motivated right from its inception by the recognition that the human brain computes in an entirely different way from the conventional digital computer. With the growth of the Internet, social networks, and online social interactions, getting daily user predictions is feasible job. Thus, our motivation is to

successfully incorporate artificial intelligence in order to make the cyber space safe that will benefit everyone

1.3 PROBLEM STATEMENT AND OBJECTIVE

1.3.1 Problem Statement:

To demonstrate the current progress of implementing Artificial Intelligence in Cryptography and see how it performs against traditional methods. The project is based in the demonstration using three artificial neural networks that will generate and exchange keys for symmetric encryption while the third acts as an adversary. This will not only help test the strength of the encryption and key exchange algorithm but also supersede the need of a human adversary. When executed via the right strategies, cryptography helps you safeguard this sensitive information, preventing it from falling prey to cyber threats and threat actors.

1.3.2 Objectives:

- To ensure data integrity through hashing algorithms and message digests. By providing digital codes and keys to ensure that what is received is genuine and originates from the intended sender, the recipient can ensure that the data received has not been tampered with in transit
- To test key exchange between two neural networks in presence of an adversary.

1.4 PROJECT SCOPE

Symmetric Key encryption using artificial intelligence will be useful to safeguard internet exchanges and to defend against man-in-middle attacks.

2 LITERATURE TABLE

Sr. No	Paper Title	Authors & Published on	Methodology
1	Deep Neural Networks based key concealment schemes	Taehyuk Kim, Tae Young Youn 04 Nov 2020(IEEE Explore)	In this paper, we propose a new DNNs-based key concealment scheme for concealing cryptographic keys within DNNs.
2	A Neural Network based Approach for Cryptographic Function Detection	Li Jia, Anmin Zhou, Peng Jia, Luping Liu, Yan Wang, Liang Liu 2020 IEEE	This paper proposed a novel approach for cryptographic function detection in malware.
3	Neural Cryptography based on Complex Valued Neural Networks	Tao Dong,Tingwen Huang 2019 IEEE	This paper took a complex value based parity machine (CVTPM) approach to neural cryptography.
4	Neural Cryptography Based on Topology Evolving Neural Networks	Yuetong Zhu, Danilo Vasconcellos Vargas, Kouichi Sakurai 2018 IEEE	This paper suggested a neural network architecture to learn neural cryptography.
5	Neural Cryptography: From symmetric encryption to Adverarial Steganography	Dylan Modesitt, Tim Henry, Jon Coden, and Rachel Lathe 2018 SemanticsScholar	This paper discussed various research in the field of neural cryptography from symmetric encryption to steganography.

6	Use of Neural Networks in Cryptography: A Review	Pranita P. Hadke, Swati G. Kale 2016 IEEE	This paper gave a review of the current usage and possibilities of the utilisation of neural networks in cryptography.
7	Neural Network based Cryptography	Apdullah Yayik,, Yapik Kutlu 2014 ResearchGate	This paper discussed the iteration of ANNs in cryptography and their advantages over other methods (symmetric encryption).
8	Cryptography based on Neural Networks	Eva Volna, Michael Janosek, Martin Kotyrba 2014 ResearchGate	This paper brought into light the method of practical implementation of neural cryptography in real world applications.
9	Cryptography using Artificial Neural Networks	Vikas Gujral, Satish Pradhan 2009 Cryptography using Artificial Neural Networks	This paper represented a comparative study between different neural network architectures for a adder and discussed their possible applications in cryptography.
10	Neural Cryptography	Wolfgang Kiesel, Ido Kanter. 2002 IJRASET	This paper discussed the first iteration of ANNs in cryptography and their advantages over other methods (symmetric encryption).

Table 1.1 Literature Table

3 SOFTWARE REQQUIREMENT SPECIFICATION

3.1. ASSUMPTIONS AND DEPENDENCIES

- Users must have the knowledge of web based applications.
- Users must have knowledge of English.
- Users must have all required software to run the application.
- End users will be available to test during the time they agree to.
- Training rooms will be available at the training center as needed.
- Project servers arrive configured as expected.
- Project will follow a waterfall methodology throughout execution.
- The proposed system will help to strengthen the traditional key exchange.
- The given system will create a secret key and transfuse this secret key using several convolution layers

3.2. FUNCTIONAL REQUIREMENTS

It should be able to encrypt and decrypt the data. It shouldn't be cracked easily.

Performance of the functions and every module must be well.

The overall performance of the algorithm will enable the users to rely with respect to security.

The system is designed in modules where errors can be detected and fixed easily.

This makes it easier to install and update new functionality if required.

3.2.1. Encryption Operation - Strings

Functional Requirement Id	FR1

Requirement Title	Encryption of Strings
Requirement Description	Should take input strings from a user Should use the ANN model trained for encryption and creates a bit-based cipher text Should make sure that only the intended receiver is able to decrypt it, even if the intruder or eavesdropper has a ANN with similar architecture.
Business Rationale	Should allows safe and secure transmission of sensitive text messages.
Exception Scenarios	If the receiver's trained decryption model is available to the intruder, then the encryption model is not able to protect the communication channel.
Dependencies	Python - Tensorflow, Numpy, Flask Browser

3.2.2. Decryption Operation - Strings

Functional Requirement Id	FR2
Requirement Title	Decryption of Cipher Strings
Requirement Description	Should take an encrypted bit-based array as input.This array of bits is the output of the encryption model that works on the plain text message. Should decrypt the input into plaintext if it is the intended receiver using the trained model.
Business Rationale	Allows conversion of encrypted messages to its original form whilst ensuring that only the intended receiver is able to do so.

Dependencies	Python - Tensorflow, Numpy, Flask Browser

3.2.3. Encryption - Images

Functional Requirement Id	FR3
Requirement Title	Encryption of Images
Requirement Description	Should takes an png image as input.This Should Encrypt the input into a cipher image and transmit it to the receiver.
Business Rationale	Allows safe and secure transmission of sensitive images.
Dependencies	Python - Tensorflow, Numpy, Flask Browser

3.2.4. Decryption - Images

Functional Requirement Id	FR4
Requirement Title	Decryption of Cipher Images
Requirement Description	Should take an encrypted bit-based array as input.This array of bits is the output of the encryption model that works on the plain text image. Should decrypt the input into plaintext if it is the intended receiver using the trained model.

Business Rationale	Allows conversion of encrypted images to its original form whilst ensuring that only the intended receiver is able to do so.
Dependencies	Python - Tensorflow, Numpy, Flask Browser

3.3. EXTERNAL INTERFACE REQUIREMENTS

3.3.1. User Interfaces

The requirements section of hardware includes minimum of 180 GB hard disk and 4 GB RAM with 2 GHz or higher speed. The primary requirements include a memory of 4GB for the Android Application development and MySQL

3.3.2. HARDWARE REQUIREMENTS

As this is an online application for product management we are not enabling or installing anyhardware components for user interface.

It's not an embedded system

- Processor - Pentium IV 2.4 GHZ
- Speed - 1.5 Ghz and Above
- RAM - 4 GB (min)
- Hard Disk - 220 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse

3.3.3. Software Interfaces

This is the software configuration in which the project was shaped. The programming language used, tools used are described here.

- Operating System :Windows
- Front End :html,css,boostrap,javascript
- Tool :pycharm,Keras
- Database :MySql

3.3.4. Communication Interfaces

- Standard internet connection is required.
- TCP/UDP connection will be required.

3.4. NON-FUNCTIONAL REQUIREMENTS

3.4.1. Perfomance Requirements

- High Speed:
  System should process requested task in parallel for various action to give quick response. Then system must wait for process completion.
- Accuracy
  System should correctly execute process, display the result accurately. System output should be in user required format.

3.4.2. Safety Requirements

The data safety must be ensured by arranging for a secure and reliable transmission media. The source and destination information must be entered correctly to avoid any misuse or malfunctioning

3.4.3. Security Requirements

Secure access of confidential data (user's details).

- Information security means protecting information and information systems from

unauthorized access, use, disclosure, disruption, modification or destruction.

- The terms information security, computer security and information assurance are frequentlyincorrectly used interchangeably. These fields are interrelated often and share the common

  goals of protecting the confidentiality, integrity and availability of information; however,there are some subtle differences between them.

3.4.4. Software Quality Assurance

- Availability [related to Reliability]
- Modifiability [includes portability, reusability, scalability]
- Performance
- Security
- Testability
- Usability[includes self-adaptability and user adaptability]

3.5. SYSTEM REQUIREMENTS

3.5.1 Database Requirements

MySQL : MySQL is an open-source relational database management system (RDBMS). Its name is a combination of “My”, the name of co-founder Michael Widenius’s daughter, and “SQL”, the abbreviation for Structured Query Language.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQLproject to create MariaDB.

MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube

3.5.2. Software Requirements

Operating System: Windows7 and above

Language: Python

IDE: Pycharm, VS Code

3.5.3 Hardware Requirements

System	:	Intel I3 Processor and above.
Hard Disk	:	5 GB.
Monitor	:	15 VGA Color.
Ram	:	4 GB.

3.6. ANALYSIS MODELS: SDLC MODEL TO BE APPLIED

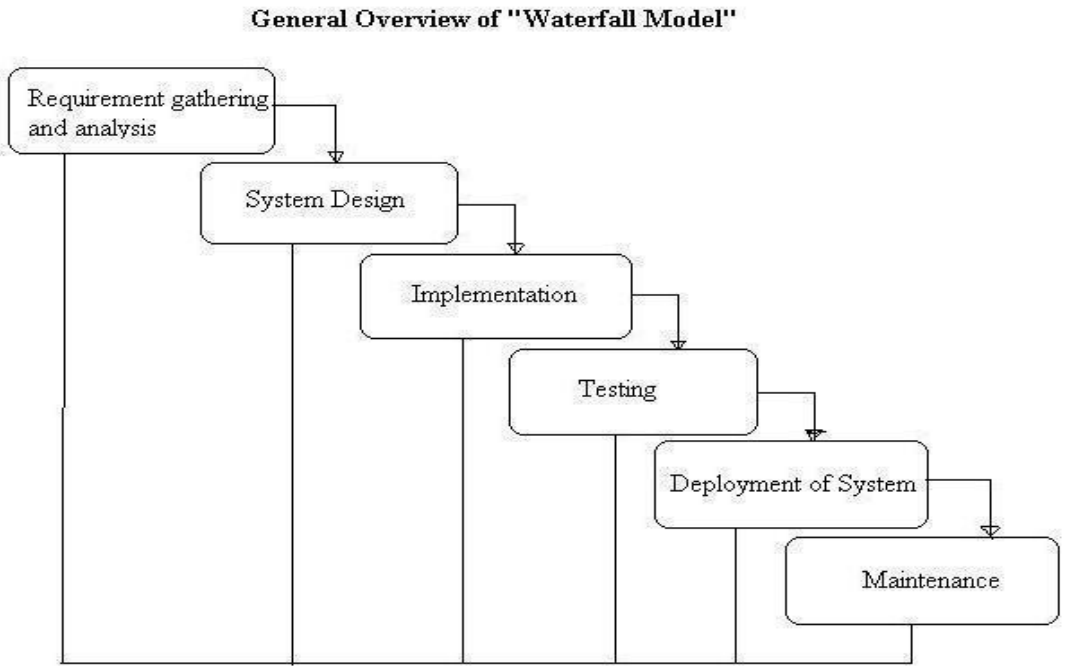


Fig 3.2: Waterfall Model

4. SYSTEM DESIGN

4.1. System Architecture

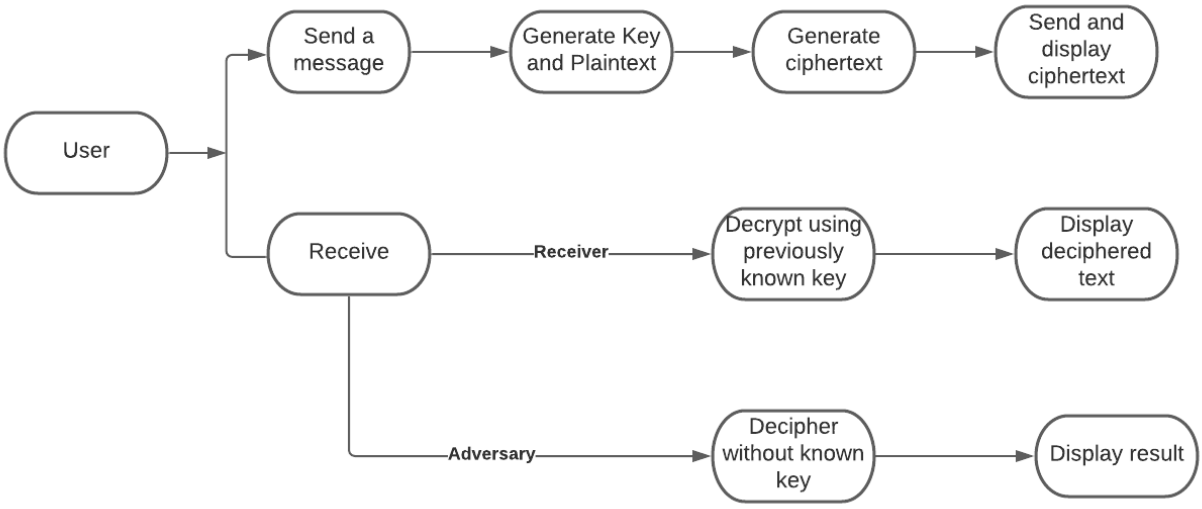


Figure 4.1. System Architecture

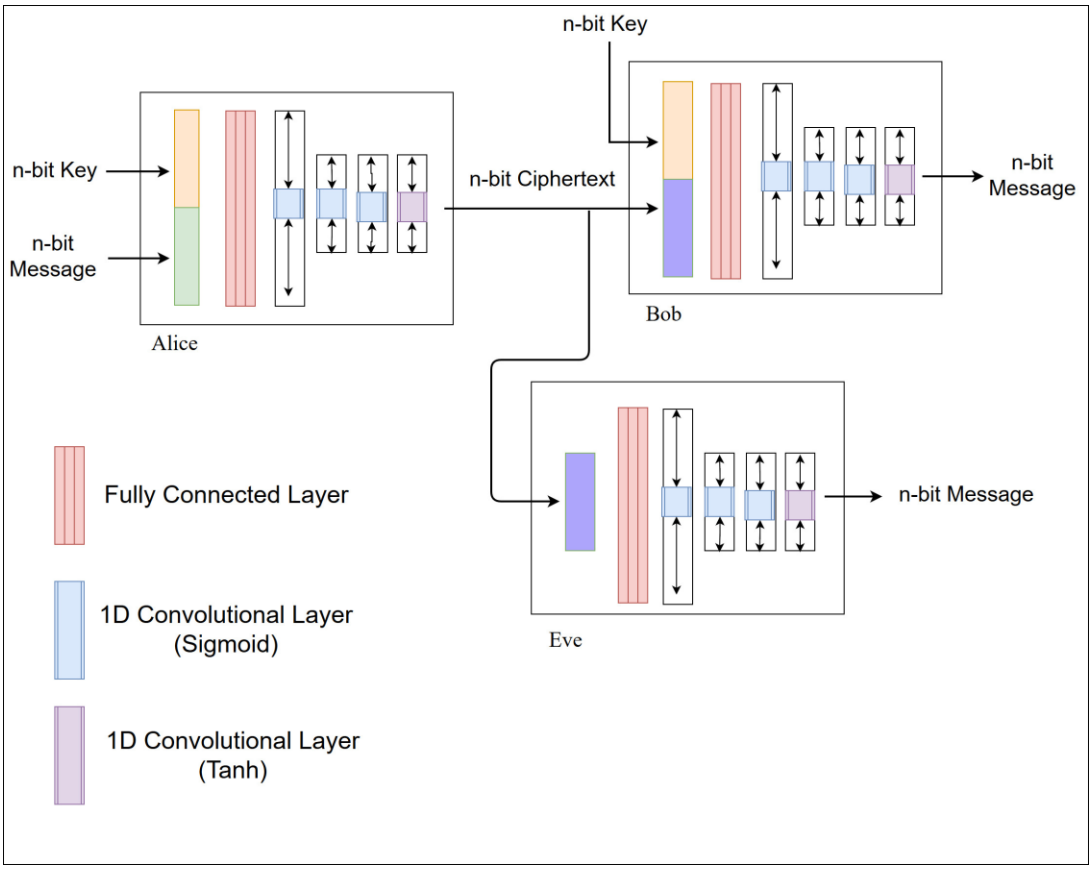


Figure No. 4.2. Neural Network Architecture

4.2. MATHEMATICAL MODEL

An artificial neural network consists of a pool of simple processing units which communicate by sending signals to each other over a large number of weighted connections. A set of major aspects of ANN are:

A state of activation yk for every unit, which is equivalent to the output of the unit; Connections between the units. Generally each connection is defined by a weight wjk

which determines the effect which the signal of unit j has on unit k;

A propagation rule, which determines the effective input sk of a unit from its external inputs;

An activation function Fk, which determines the new level of activation based on the effective input sk(t) and the current activation yk(t) (i.e., the update);)[2][3][4]

An external input (aka bias, offset) θk for each unit;

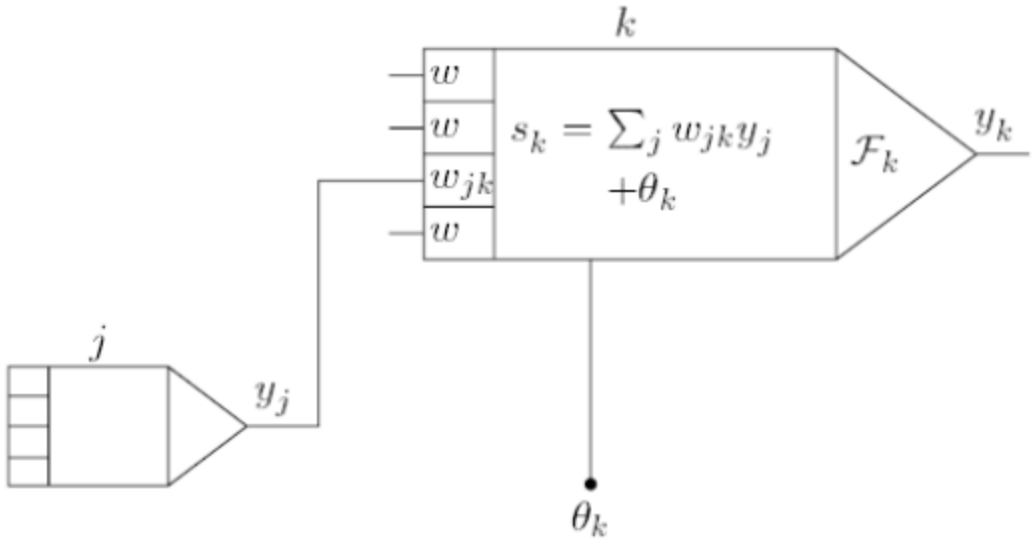


Figure No. 4.3. Block Diagram Mathematical Model of an Artificial Neural Network

Connections between units

$$s_k(t) = \sum_j w_{jk}(t)\, y_j(t) + \theta_k(t).$$

$$s_k(t) = \sum_j w_{jk}(t) \prod_m y_{j_m}(t) + \theta_k(t).$$

Activation and the output rules

$$y_k = \mathcal{F}(s_k) = \frac{1}{1 + e^{-s_k}}$$

$$y_k(t+1) = \mathcal{F}_k(y_k(t), s_k(t)).$$

$$y_k(t+1) = \mathcal{F}_k(s_k(t)) = \mathcal{F}_k\left(\sum_j w_{jk}(t)\, y_j(t) + \theta_k(t)\right),$$

Failures:
1. Data packet loss during the transmission within the network
2. Hardware failure.
3. Software failure.

Success:
1. Decrypt the message without compromising integrity of data.
2. Encryption is done very fast so is the decryption

Space Complexity:
The space complexity depends on Presentation and visualization of discovered patterns. More the storage of data more is the space complexity.

Time Complexity:
Check No. of patterns available in the datasets= n
If (n>1) then retrieving of information can be time consuming. So the time complexity of this algorithm is $O^{(n^n)(n^n)}$.

4.3. DATA FLOW DIAGRAM

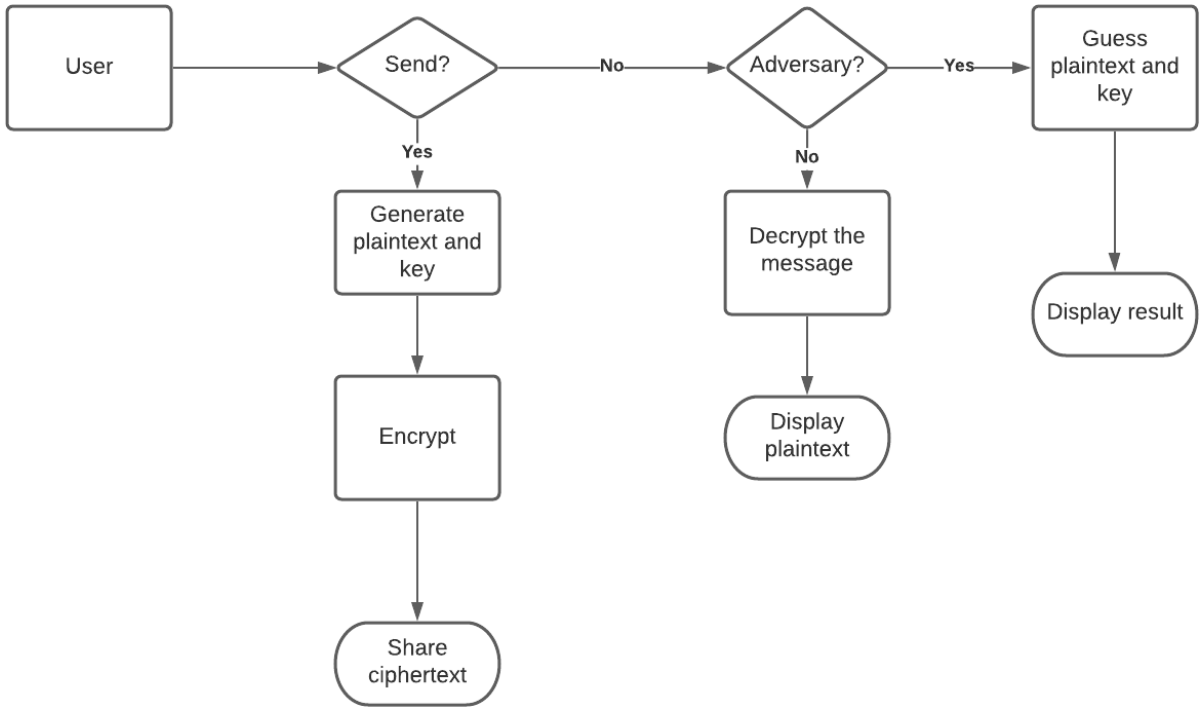


Figure No. 4.4. Data flow diagram

4.4. ENTITY RELATIONSHIP DIAGRAMS

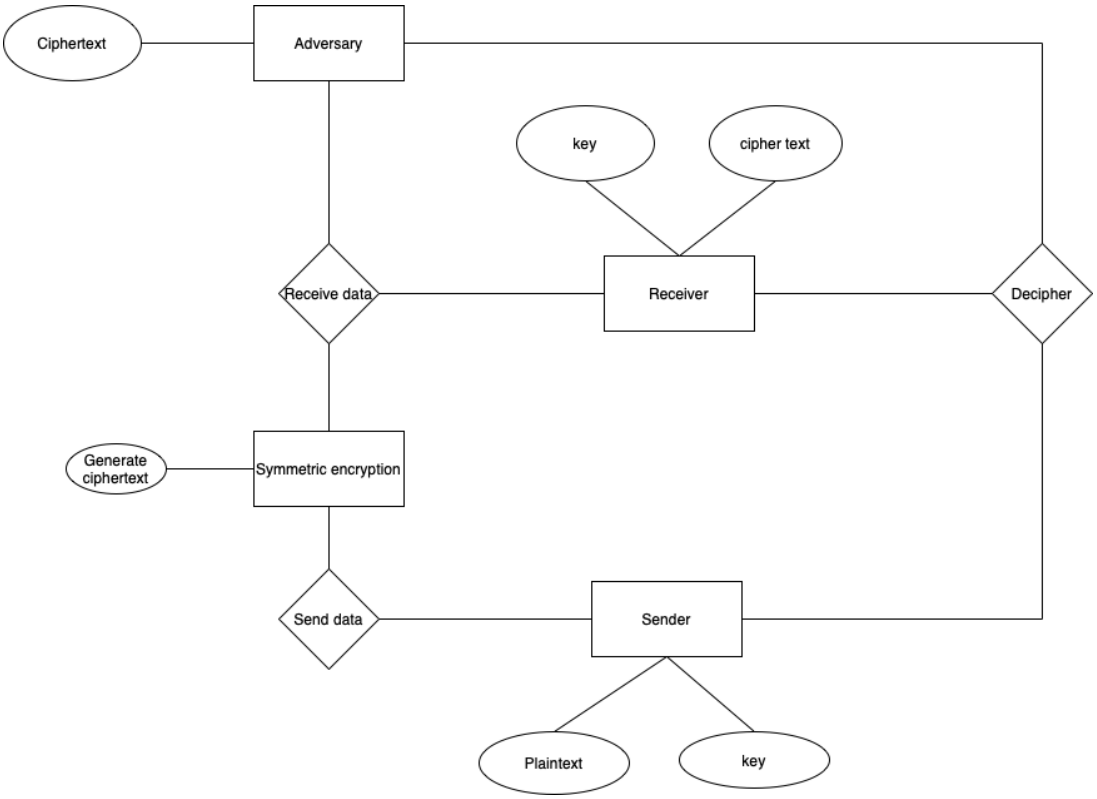


Figure No. 4.5. ER Diagram

4.5. UML DIAGRAMS

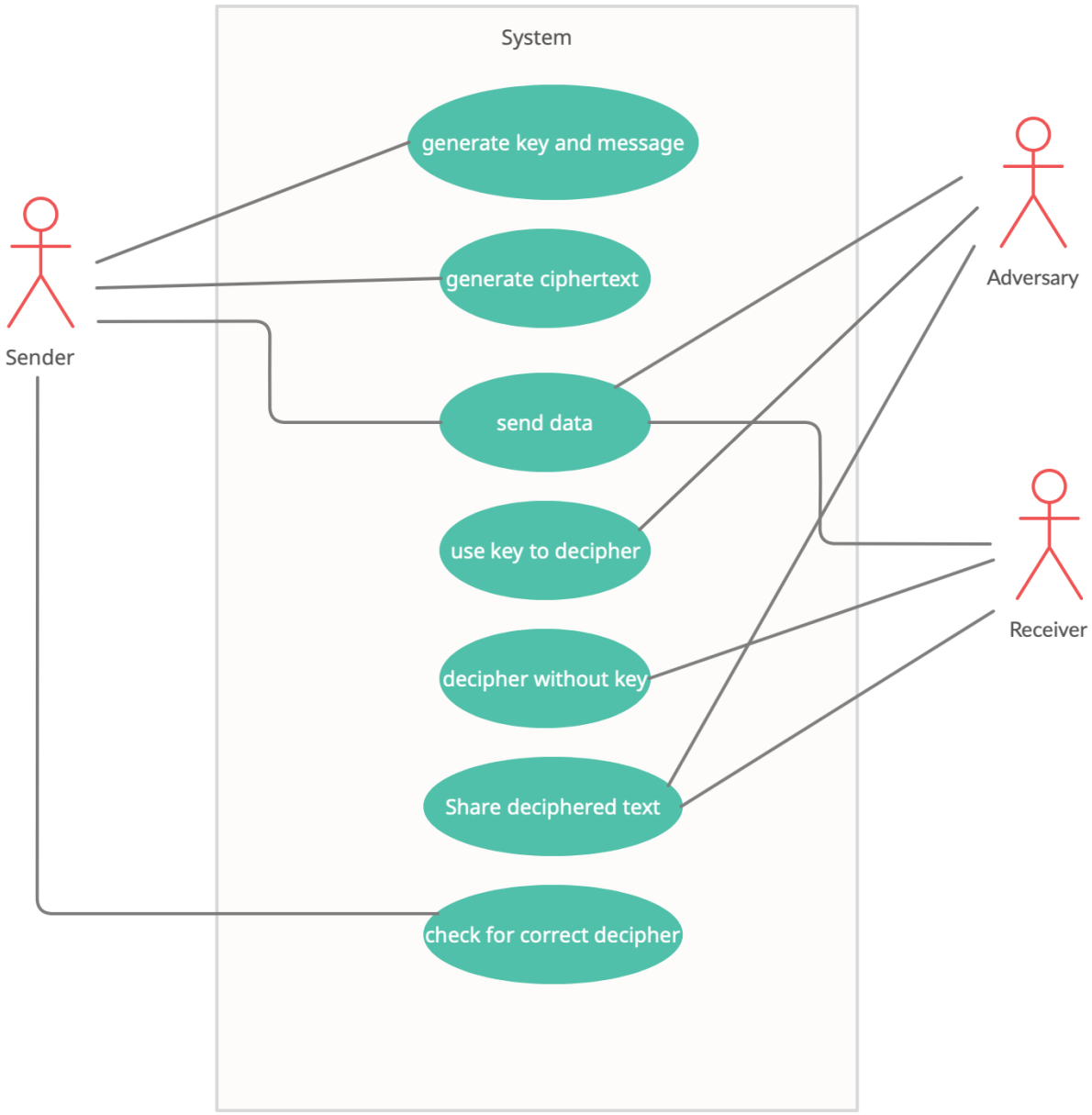


Figure No.4.6. Use Case Diagram

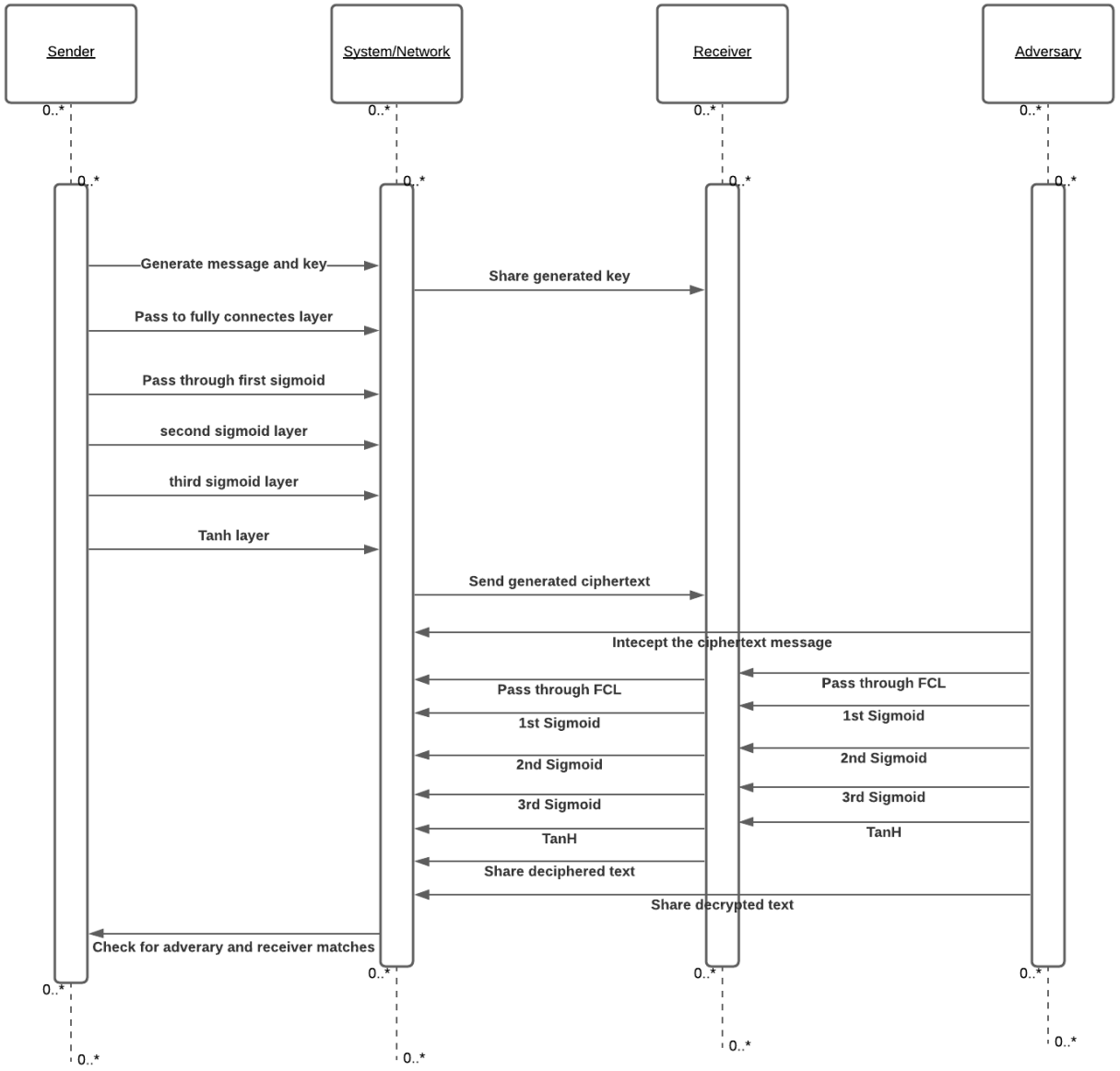


Figure No.4.7. Sequence Diagram

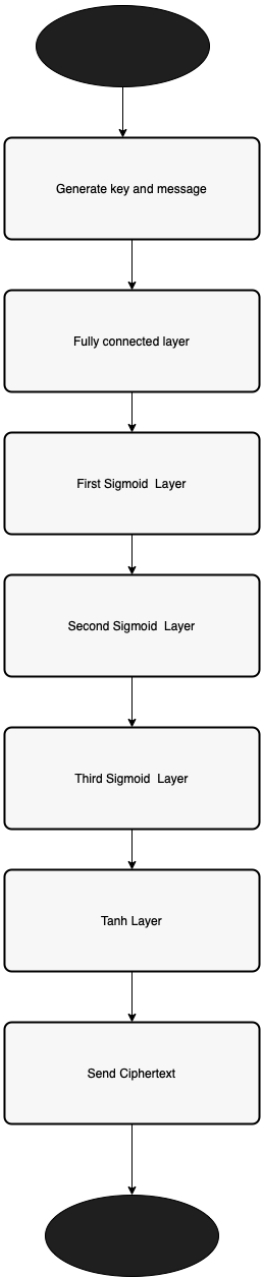


Figure No. 4.8. Activity Diagram

5. PROJECT PLAN

5.1. PROJECT ESTIMATE

5.1.1. Reconciled Estimates

Time Estimates -

The time estimate of this project is approximately 11 months. Software project estimation is a form of problem solving. The complex software is hard to estimate hence it is divided into smaller pieces. The estimation of the project will be correct only when the estimation of size of the project is correct. In the context of project planning size refers to the qualifiable outcome of the project. Here,the direct approach is chosen and hence , the size is estimated in the Line of Codes.

The feasibility study comprising an initial investigation into personnel will be required.

Feasibility study will help you make informed and transparent decisions at crucial points during the developmental process. All projects are feasible given unlimited times and resources. Unfortunately, the development of computer based systems is more likely to be plagued by scarcity of resources .It is both necessary and prudent to evaluate the feasibility of a projectat the earliest possible time.

Feasibility analysis -

Whatever we think need not be feasible .It is wise to think about the feasibility of any problem

we undertake. Feasibility is the study of impact, which happens in the organization by the development of a system. The impact can be either positive or negative. When the positives nominate the negatives, then the system is considered feasible.

Feasibility study -

Feasibility study is made to see if the project on completion will serve the purpose of the

organization for the amount of work, effort and the time that is spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system

proposal is according to its work ability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus when a new application is proposed it normally goes through a feasibility study before it is approved for development. The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project.

Technical feasibility -

The system must be evaluated from the technical point of view first. The assessment of this

feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

5.1.2. Project Resources

Well configured Laptop, Python Environment, 2 GHZ CPU speed, 4 GB RAM, Internet connection .

5.2. RISK MANAGEMENT

5.2.1. Risk Identification

1. Have top software and customer managers formally committed to support the project? Ans-Not applicable.
2. Are end-users enthusiastically committed to the project and the system/product to be built?
Ans-Not known at this time.
3. Are requirements fully understood by the software engineering team and its customers?
Ans-Yes
4. Have customers been involved fully in the definition of requirements? Ans-Not applicable
5. Do end-users have realistic expectations? Ans-Not applicable

6. Does the software engineering team have the right mix of
skills? Ans-yes
7. Are project requirements
stable? Ans-Not applicable
8. Is the number of people on the project team adequate to do the
job? Ans-Not applicable
9. Do all customer/user constituencies agree on the importance of the project
and on the requirements for the system/product to be built?
Ans-Not applicable

5.2.2. Risk Analysis

The risks for the Project can be analyzed within the constraints of time and quality

Risk Table:

Id	Risk Description	Probability	Schedule	Quality	Overall
1	Correctness	Low	Low	High	High
2	Availability	High	Low	High	High

Table 5.1. Risk Table

Risk Probability definitions:

Probability	Value	Description
High	Probability of occurrence is	$> 75\%$
Medium	Probability of occurrence is	$26 - 75\%$
Low	Probability of occurrence is	$< 25\%$

Table 5.2. Risk Probability definitions

Risk Impact definitions:

Impact	Value	Description
Very high	$> 10\%$	Schedule impact or Unacceptable quality
High	$5 - 10\%$	Schedule impact or Some parts of the project have low quality
Medium	$< 5\%$	Schedule impact or Barely noticeable degradation in quality Low Impact on schedule or Quality can be incorporated

Table 5.3 Risk Impact definitions

5.2.3. Overview of Risk Mitigation, Monitoring, Management.

If a software team adopts a proactive approach to risk, avoidance is always the best strategy.
This is achieved by developing a plan for risk mitigation. To mitigate this risk, you would develop a strategy for reducing turnover. Among the possible steps to be taken are:

. Meet with current staff to determine causes for turnover (e.g., poor working conditions, low pay, competitive job market). Mitigate those causes that are under your control before the project starts. Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave. Organize project teams so that information about each development activity is widely dispersed. Define work product standards and establish mechanisms to be sure that all models and documents are developed in a timely manner. Conduct peer reviews of all work (so that more than one person is "up to speed"). Assign a backup staff member for every critical technologist.

2. Risk Monitoring -

As the project proceeds, risk-monitoring activities commence. The project manager monitors factors that may provide an indication of whether the risk is becoming more or less likely. In the case of high staff turnover, the general attitude of team members based on project pressures, the degree to which the team has gelled, interpersonal relationships among team members, potential problems with compensation and benefits, and the availability of jobs within the company and outside it are all monitored.

3. Risk Management -

Risk management and contingency planning assumes that mitigation efforts have failed and that the risk has become a reality. Continuing the example, the project is well under way and a number of people announce that they will be leaving. If the mitigation strategy has been followed, backup is available, information is documented, and knowledge has been dispersed across the team. In addition, you can temporarily refocus resources (and readjust the project schedule) to those functions that are fully staffed, enabling newcomers who must be added to the team to "get up to speed." Those individuals who are leaving are asked to stop all work and spend their last weeks in "knowledge transfer mode." This might include video-based knowledge capture, the development of "commentary documents or Wikis," and/or meeting with other team members who will remain on the project

5.3. PROJECT SCHEDULE

5.3.1. Project Task Set

Major Tasks in the Project stages are: Task

Task 1: correctness

Task 2: availability

Task 3: integrity

Task Network -

5.3.2. Task Network

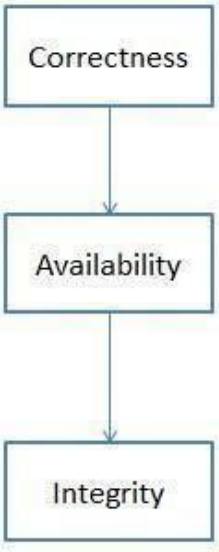


Figure No. 5.1. Task Network

5.3.3. Timeline Chart

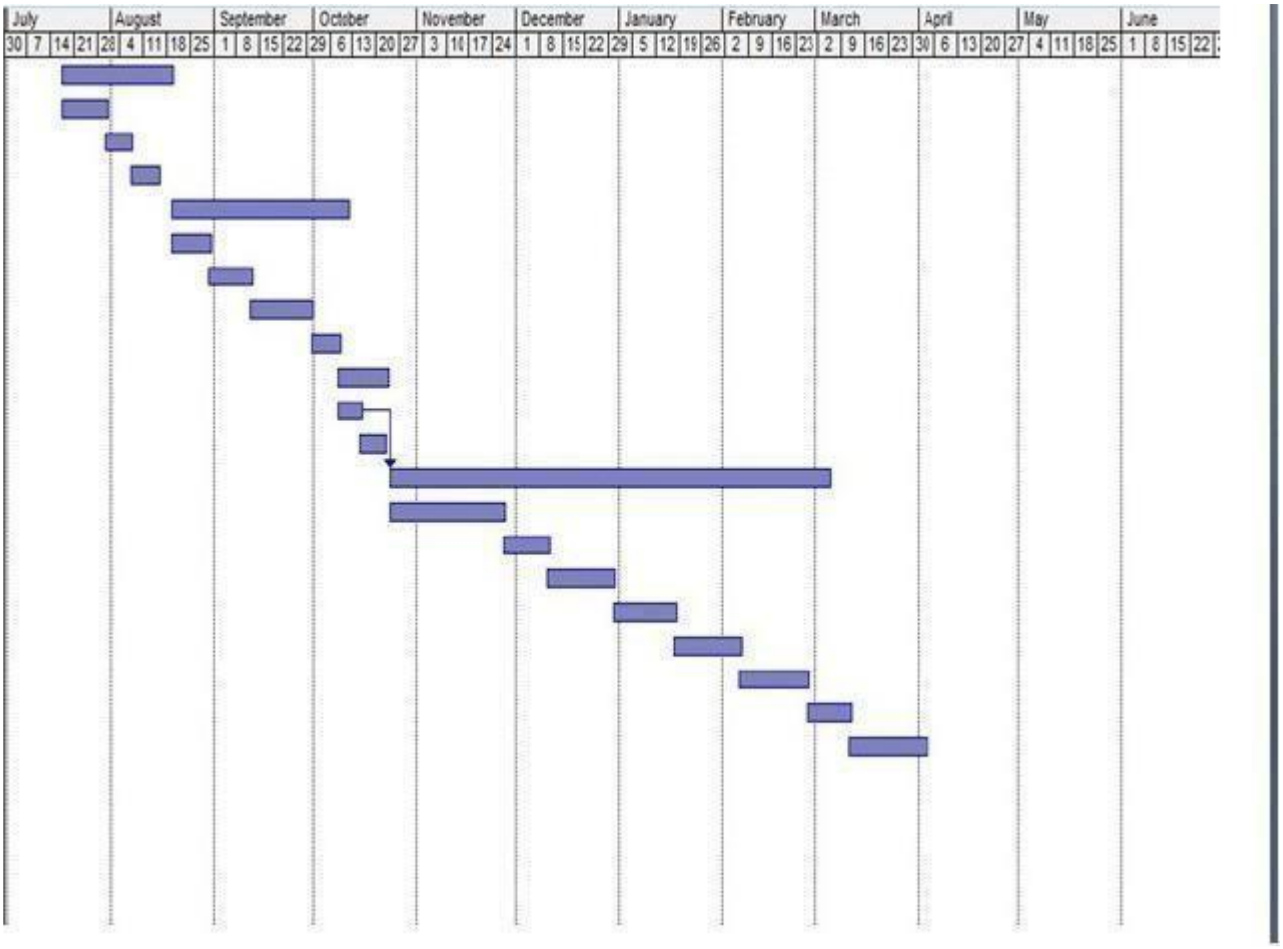


Figure No. 5.2. Timeline Chart

5.4. TEAM ORGANIZATION

5.4.1. Team Structure

The team structure for the project is identified. There are a total 4 members in our team and roles are defined. All members are contributing in all the phases of the project.

5.4.2. Management, Reporting and Communication

. Well planning mechanisms are used for progress reporting and inter/intra team communication are identified as per requirements of the project.

6. PROJECT IMPLEMENTATION

6.1. OVERVIEW OF PROJECT MODULES

The primary purpose of this project is to implement the concepts and adaptive nature of Artificial Neural Networks for advanced encryption and decryption without manually creating an algorithm to do so. For the purpose of demonstration, we create and train three neural network models, and implement their functionality in the application. Various machine learning and data processing modules are required. The python modules used for this project are sys, ctypes and random.

Along with modules from the python language, we use several user defined modules. They are:

- helper.py
- helper_tests.py
- The training notebook

6.2. TOOLS AND TECHNOLOGIES USED

The technologies used for creating this demonstration are:

- Python Machine Learning Libraries
    - Tensorflow
    - keras
    - Numpy
    - Pandas
    - Matplotlib
- Python Web Development Framework
    - Flask
- Basic Python Libraries
    - random
    - string
    - base64
    - werkzeug.utils

- gmpy2
- math
- joblib
- datetime
- Pickle
- padding

- Python Cryptography library
  - cryptography
  - Fernet
- Python Image Processing Libraries
  - pixelshuffle
  - imageshuffle
  - BlockScramble

6.3. ALGORITHM DETAILS

The various algorithms used for encryption are custom and ANN developed. The neural networks are trained and tested to perform successful encryption and decryption in the presence of an adversary. As a result, there is no specific algorithm used for the purpose of encryption and decryption. The model consists of three different categories of hidden layers:

- Fully Connected Layers (FCC)
- Sigmoid Layer
- Tanh Layer

On the other hand, for image encryption we use the combination of the above models and chaos maps. We use logistic encryption for image encipherment and our ANN for key encryption

6.3.1 Hidden Layers of ANN

The basic structure of our neural network is as follows:

```
Model: "alice"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_3 (InputLayer)           [(None, 16)]         0           []

 input_4 (InputLayer)           [(None, 16)]         0           []

 concatenate_1 (Concatenate)    (None, 32)           0           ['input_3[0][0]',
                                                                  'input_4[0][0]']

 dense_1 (Dense)                (None, 32)           1056        ['concatenate_1[0][0]']

 FCC (Activation)               (None, 32)           0           ['dense_1[0][0]']

 Reshape (Reshape)              (None, 32, 1)        0           ['FCC[0][0]']

 conv1d_4 (Conv1D)              (None, 32, 2)        10          ['Reshape[0][0]']

 sigmoid_layer1 (Activation)    (None, 32, 2)        0           ['conv1d_4[0][0]']

 conv1d_5 (Conv1D)              (None, 16, 4)        20          ['sigmoid_layer1[0][0]']

 sigmoid_layer2 (Activation)    (None, 16, 4)        0           ['conv1d_5[0][0]']

 conv1d_6 (Conv1D)              (None, 16, 4)        20          ['sigmoid_layer2[0][0]']

 sigmoid_layer3 (Activation)    (None, 16, 4)        0           ['conv1d_6[0][0]']

 conv1d_7 (Conv1D)              (None, 16, 1)        5           ['sigmoid_layer3[0][0]']

 tanh_non_linear (Activation)   (None, 16, 1)        0           ['conv1d_7[0][0]']

 flatten_1 (Flatten)            (None, 16)           0           ['tanh_non_linear[0][0]']

==================================================================================================
```

Figure No. 6.1. Layers of an ANN

The different layers are as follows:

- Fully Connected Layer(FCC)

FCNNs are artificial neural networks in which all of the nodes, or neurons, in one layer are linked to the neurons in the next layer. While this method is commonly used to analyse some sorts of data, it has several drawbacks when it comes to image recognition and categorization. Overfitting is a problem with such networks because they need a lot of computing power. When such networks are also 'deep,' that is, when there are numerous layers of nodes or neurons, humans may find them challenging to understand. [5].

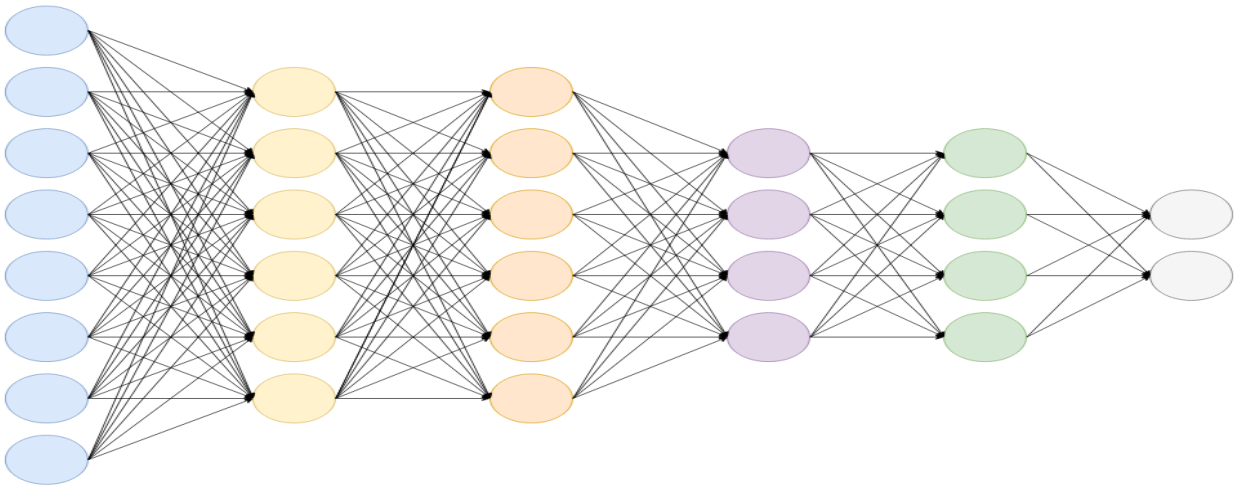


Figure No. 6.2. Fully Connected Neural Layer

- Sigmoid Layer

  The sigmoid function, also known as the o(x) or sigmoid function, is a form of logistic function (x). It is supplied by:

  $$\sigma(x) = 1/(1+exp(-x))$$

  The sigmoid function is used as an activation function in neural networks.
  A weighted sum of inputs is sent through an activation function, and the outcome is used as an input to the next layer.
  The output of this unit will always be between 0 and 1 when the activation function of a neuron is a sigmoid function. Because the sigmoid is a nonlinear function, the output of this unit would similarly be a nonlinear function of the weighted sum of inputs. A sigmoid unit is a type of neuron that activates by using a sigmoid function. [5].

- TanH Layer

  The hyperbolic tangent activation function is also known as the Tanh (also "tanh" and "TanH") function. It looks similar to the sigmoid activation function and even has the same S-shape. The function takes any real number as input and outputs a number between -1 and

1. The output gets closer to 1.0 as the input gets larger (more positive), while the output gets closer to -1.0 as the input becomes smaller (more negative). [6].

The Tanh activation function is calculated as follows:

$$(e^x - e^{-x}) / (e^x + e^{-x})$$

6.3.2. Logistic Functions for Encryption and Decryption

- Encryption

In the encryption algorithm, there are three fundamental processes. In the first stage, chaotic sequences are generated. The pixel values were confused in the second stage, and the pixel position was jumbled in the third stage to form the required encrypted image. Consider the image f, which has the dimensions M N. f(i,j) represents the pixel of f, where I and j are in the range 1=i=M and 1=j=N. The grey value of the picture f at pixel position I j) is now indicated by f. (i,j). The initial state of the logistic map is calculated using a 256-bit (32-character) secret key expressed in ASCII as K = K1K2 K3......K32 (Ki signifies the 8-bit key character in the i-th key position). [14]

The following is a step-by-step technique for logistic encryption using Chaos Maps:

Step 1: Make an array of P =
P,P,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L, Then convert the pixel values to an unsigned integer in the range of 0 to 255 using the mod operation. After each pixel encryption, the initial values of the chaos map are computed using the previous encryption value and the key value.

Step 2: Using the logistic map indicated in Eq., create n number of chaotic sequences xi =x1,x2,x3,L,xn in the range 0 to 1 with the initial condition x0 and the parameter r = 3.999. (1). Turn xi into an unsigned integer in the range of 0 to 255 using the mod operator.

Step 3: Create the sequence C = P x to confuse iii the pixel value. The symbol represents the bitwise XOR operation.

Step 4: Convert Ci = C1,C2,C3,L,Cn to an array of size MN to retrieve the image f'. Add one to the unsigned integer series xi = x1, x2, x3,L, xn to produce X, then convert it to an array of size M N.

Step 5: Finally, to obtain the required encrypted image f, follow the pixel shuffling processes outlined below. j and k have values ranging from 1 to 255. The sign denotes the swapping of the values of two pixel places in f ′.

$f'(X(j,j),k) \Leftrightarrow f'(X(j+1,j+1),k)$

$f'(k,X(j,j)) \Leftrightarrow f'(k,X(j+1,j+1))$. (4)

The final encrypted picture has now been assigned to the letter f.

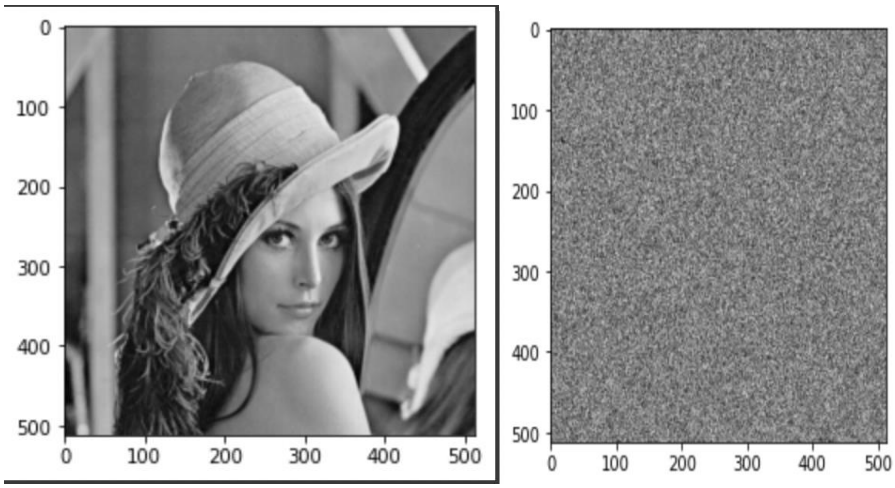


Figure No.6.3: Image Encryption input and result

- Decryption

The process of decrypting an image is the inverse of encryption.

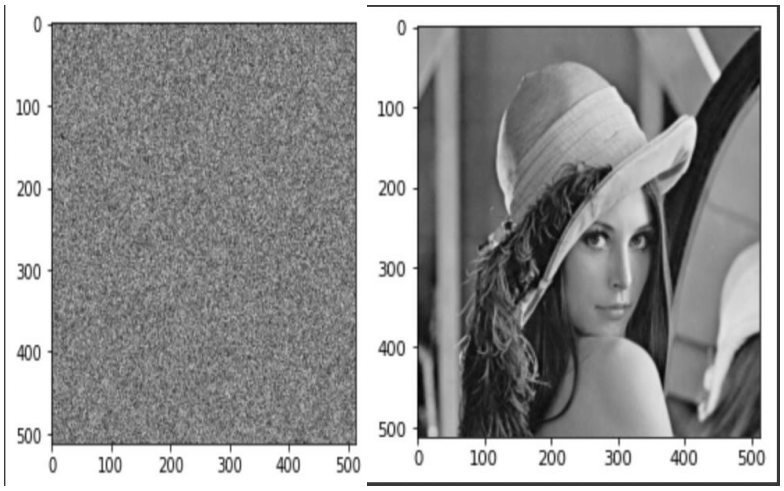


Figure No. 6.4. Image Decryption input and output

7. SOFTWARE TESTING

7.1. TYPE OF TESTING

7.1.1. Unit Testing

It is a software testing method by which individual units of source code, are tested to determine whether they are fit for use. It provides a sort of living documentation of the system. It is performed by using the White Box testing method. It is performed by software developer. It increase the confidence of changing or maintaining code.

It ensure that all statements in the unit have been executed at least once. It tests data structures (like stacks, queues) that represent relationships among individual data elements.

- Function
- Module Interface: Ensure that information flows properly into and out of the module.
- Local data structures: Ensure that data stored temporarily maintain its integrity during all steps in an algorithm execution.
- Driver and Stub:

Driver is a module that takes input from test case, passes this input to the unit to be tested and prints the output produced. Stub is a module that works as a unit referenced by the unit being tested. It uses the interface of the subordinate unit, does minimum data manipulation, and returns control back to the unit being tested.

7.1.2. Integration Testing

It tests integration or interface between components, interactions to different parts of the system. It is to verify the functional, performance, and reliability between the modules that are integrated.

Defined as a systematic technique for constructing the software arch texture At the same time integration is occurring, conduct tests to uncover errors associated with interfaces.

Objective is to take unit tested modules and build a program structure based on the prescribed design.

It ensures that all modules work together properly and transfer accurate data across their interfaces.

It is performed with an intention to uncover errors that lie in the interfaces among the integrated components.

It tests those components that are new or have been modified or affected due to changes.

7.2. TEST CASES AND RESULTS

Evaluation of Algorithm:

Evaluated algorithm for 10000 examples. Evaluation results we got are 99.99% correctness in case of training with inteded recepient who has the key and 0.24% in case of adversary which doesn't have key.

Evaluate

```
n_examples = 10000
m_batch = np.random.randint(0, 2, m_bits * n_examples).reshape(n_examples, m_bits)
k_batch = np.random.randint(0, 2, m_bits * n_examples).reshape(n_examples, m_bits)

m_enc = alice.predict([m_batch, k_batch])
m_dec = (bob.predict([m_enc, k_batch]) > 0.5).astype(int)
m_att = (eve.predict(m_enc) > 0.5).astype(int)

bdiff = np.abs(m_batch - m_dec)
bsum = np.sum(bdiff, axis=-1)
ediff = np.abs(m_batch - m_att)
esum = np.sum(ediff, axis=-1)

print("Bob % correct: ", 100.0*np.sum(bsum == 0) / n_examples, '%')
print("Eve % correct: ", 100.0*np.sum(esum == 0) / n_examples, '%')
```

```
Bob % correct:  99.99 %
Eve % correct:  0.24 %
```

Unit Testing:

Check individual functions are working properly.

```
class UnitTestHelper(unittest.TestCase):

    def test_encstr(self):
        raw_message = 'Hello World!'
        bintext = ' '.join('{0:016b}'.format(ord(x), 'b') for x in raw_message)
        cipher  = bintext.replace(" ", "")
        output = encstr(raw_message)
        correct_bin = '0000000001001000000000000110010100000000011011000000000001101
        self.assertEqual(output[0],correct_bin)
        self.assertEqual(output[1],len(raw_message))

    def test_decstr(self):
        raw_message = 'Hello World!'
        bintext = ' '.join('{0:016b}'.format(ord(x), 'b') for x in raw_message)
        cipher  = bintext.replace(" ", "")
        plaintext = decstr(cipher,BLOCKSIZE)
        self.assertEqual(raw_message,plaintext)

    def test_strToArr(self):
        raw_message = 'Hello World!'
        bintext = ' '.join('{0:016b}'.format(ord(x), 'b') for x in raw_message)
        cipher  = bintext.replace(" ", "")
        result = strToArr(cipher,BLOCKSIZE)

        self.assertEqual(result,result)

    def test_arrToStr(self):
        raw_list = [[1,0,1,0]]
        bintext = arrToStr(raw_list)
        self.assertEqual(bintext,'1010')
      Prasann Shimpi, 4 months ago • functionality done …
```

Integration Testing:

Here we performed an integration testing

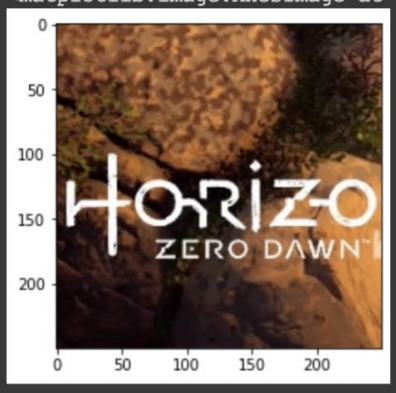
Testing 3 models ie. Sender, receiver, adversary which are integrated to form complete algorithm.

```python
class TestStringMethods(unittest.TestCase):

    def test_encryption(self):
        raw_message = 'Hello World!'
        messages = processRawMessage(raw_message)
        message = messages[0]
        key = messages[1]
        cipher = (alice.predict([message,key]) > 0.5).astype(int)
        ciphertext = processBinaryMessage(cipher)
        # print(ciphertext)
        self.assertNotEqual(raw_message,ciphertext)

    def test_decryption(self):
        raw_message = 'Hello World!'
        messages = processRawMessage(raw_message)
        message = messages[0]
        key = messages[1]
        cipher = alice.predict([message,key])      Prasann Shimpi, 2 days ago •
        decipher = (bob.predict([cipher, key]) > 0.5).astype(int)
        plaintext = processBinaryMessage(decipher)
        self.assertEqual(raw_message, plaintext)

    def test_adversarial_performance(self):
        raw_message = 'Hello World!'
        messages = processRawMessage(raw_message)
        message = messages[0]
        key = messages[1]
        cipher = alice.predict([message,key])
        decipher = (bob.predict([cipher, key]) > 0.5).astype(int)
        plaintext = processBinaryMessage(decipher)
        adversary = (eve.predict(cipher) > 0.5).astype(int)
        adv = processBinaryMessage(adversary)

        self.assertEqual(raw_message, plaintext)
        self.assertNotEqual(raw_message, adv)
```

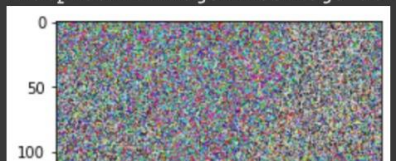
```
(tensor_env) prasann@Prasanns-MacBook-Pro tests % python helper_test.py
Using TensorFlow backend.
....
----------------------------------------------------------------------
Ran 4 tests in 0.001s

OK
(tensor_env) prasann@Prasanns-MacBook-Pro tests % python test.py
Using TensorFlow backend.
...
----------------------------------------------------------------------
Ran 3 tests in 0.426s

OK
(tensor_env) prasann@Prasanns-MacBook-Pro tests %
```

Test for Images:

```
[ ] LogisticDecryption(image + "_LogisticEnc.png", "supersecretke")
    im = Image.open(image + "_LogisticDec.png", 'r')
    imshow(np.asarray(im))

    <matplotlib.image.AxesImage at 0x7f75f232c7d0>
```



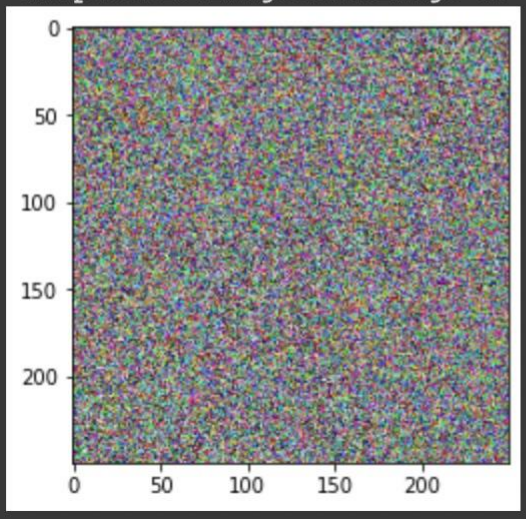

```
    LogisticDecryption(image + "_LogisticEnc.png", "supersecretkelns;lahfj")
    im = Image.open(image + "_LogisticDec.png", 'r')
    imshow(np.asarray(im))

    <matplotlib.image.AxesImage at 0x7f75f260ca90>
```



Decrypt with the key "supersecretkd"

```
[ ]  LogisticDecryption(image + "_LogisticEnc.png", "supersecretkd")
     im = Image.open(image + "_LogisticDec.png", 'r')
     imshow(np.asarray(im))
```

```
<matplotlib.image.AxesImage at 0x7f75f30f9f90>
```



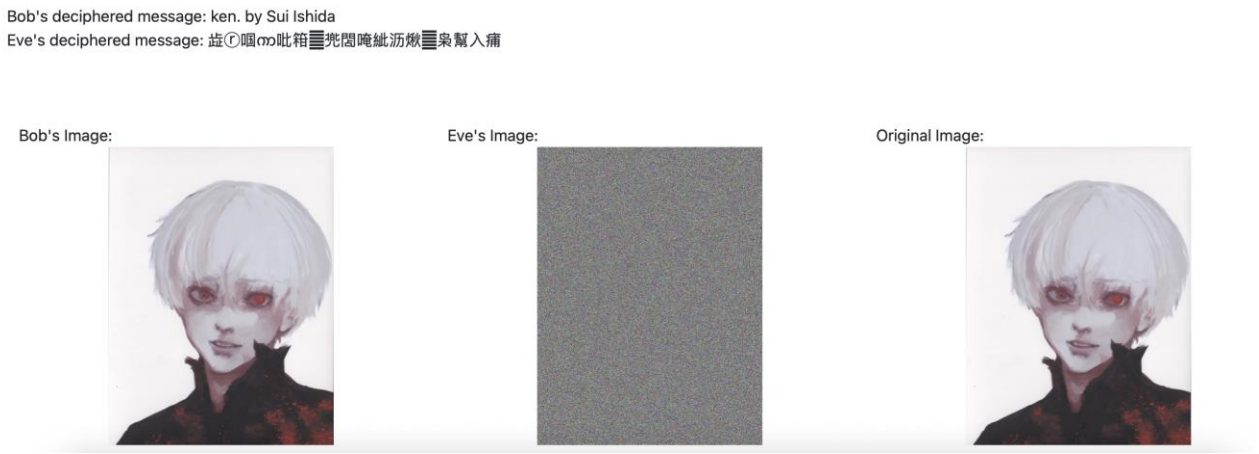
8. RESULTS

8.1. OUTCOMES

- With the use of this encryption scheme encryption and decryeption of messages and documents while maintaining confidentiality and integrity can be achieved
- Overcome major obstacle in symmetric encryption that is key exchange problem.
- Algorithm is also tested in adversial environment to check against Man-In-The-Middle attacks. No adversary can decrypt the message without key or necessary training

8.2. SCREENSHOTS

Text and Image Decryption

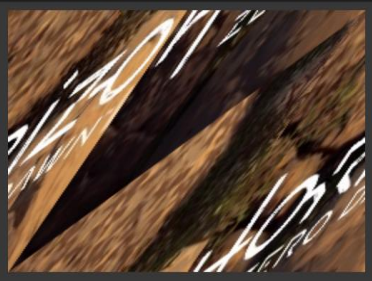


Comparison with Fernet

```
Cryptography.fernet AES Encrypt/Decrypt sessions
-----------------------------------------------
cipher text =  b'gAAAAABh_9q6i6WcvuY8czV0cPWgHaBzmXzDMcQieh5nIW4yk6stYcl7QPxoVc8zqCHuBA6F_a0G-Dm0f
plain_text =
b'\x00\x00\x00\x00\x00\x00\x00\x00\x01\x01\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
cipher text =  [-0.5383039   0.00614445 -0.16977766 -0.32550016  0.21388824 -0.16379495
 -0.09691516 -0.21935992  0.06962691 -0.07455905  0.12722728  0.12503095
 -0.03900693 -0.0020982   0.05760781  0.16742107]
plain_text =   [ 8.8632072e-04  1.3343979e-03  1.6982538e-03  4.2468306e-05
  1.5579746e-03  4.5234542e-03  4.7041122e-03  7.7709532e-04
  3.3884514e-03  9.9946743e-01  9.9957657e-01 -9.7697927e-04
  9.9958336e-01  8.2513895e-03  5.0963014e-03  6.7423154e-03]
*********Performance of NCA versus Cryptography.Fernet********************
Time taken for Alice to Encrypt the Plain Text (seconds):  0:00:00.111218
Time taken for Bob to Decrypt the Cipher Text (seconds):  0:00:00.109432
Time taken for Fernet to Encrypt the Plain Text (seconds):  0:00:00.000213
Time taken for Fernet to Decrypt the Cipher Text (seconds):  0:00:00.000114
```

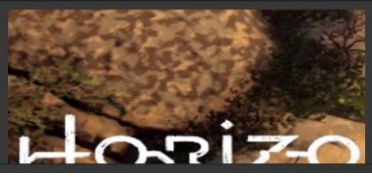
Key Prediction Problem of Arnold Cat Maps

```
[ ]  ArnoldCatDecryptionIm = ArnoldCatDecryption(image + "_ArnoldcatEnc.png", 19)
     cv2_imshow(ArnoldCatDecryptionIm)
```



Decrypt with original key 20

```
[ ]  ArnoldCatDecryptionIm = ArnoldCatDecryption(image + "_ArnoldcatEnc.png", 20)
     cv2_imshow(ArnoldCatDecryptionIm)
```

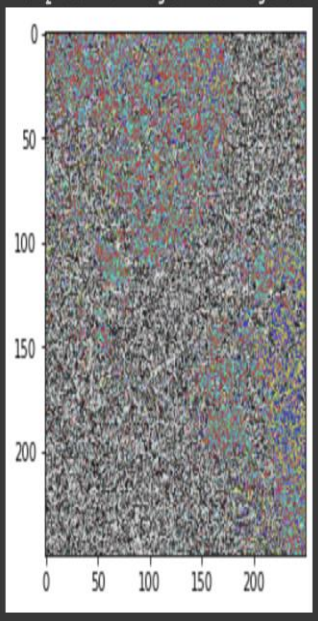


Using Henon Maps

Decrypt with the key (0.1, 0.101)

```
HenonDecryption(image + "_HenonEnc.png", (0.1, 0.101))
im = Image.open(image + "_HenonDec.png", 'r')
imshow(np.asarray(im))
```

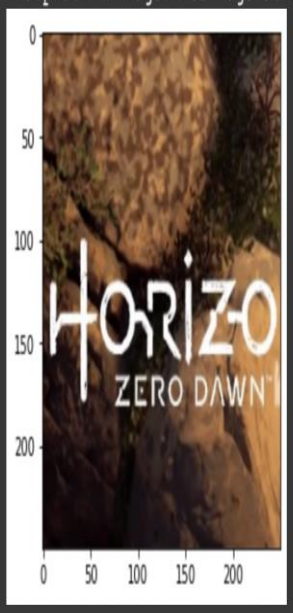
```
<matplotlib.image.AxesImage at 0x7f75f31ad850>
```



Decryption with Original key

```
HenonDecryption(image + "_HenonEnc.png", (0.1, 0.1))
im = Image.open(image + "_HenonDec.png", 'r')
imshow(np.asarray(im))
```

```
<matplotlib.image.AxesImage at 0x7f75f31be990>
```

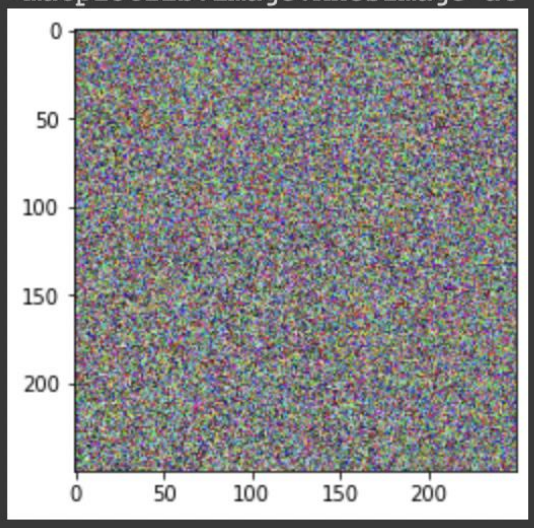


Using Logistic Encryption with key mapping

Decrypt with the key "supersecretkd"

```
LogisticDecryption(image + "_LogisticEnc.png", "supersecretkd")
im = Image.open(image + "_LogisticDec.png", 'r')
imshow(np.asarray(im))
```

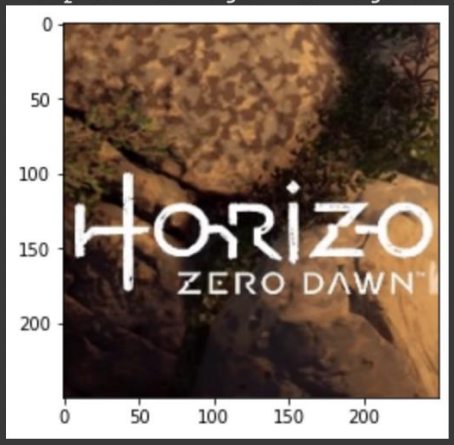
```
<matplotlib.image.AxesImage at 0x7f75f30f9f90>
```



Decryption with original Key

```
LogisticDecryption(image + "_LogisticEnc.png", "supersecretke")
im = Image.open(image + "_LogisticDec.png", 'r')
imshow(np.asarray(im))
```

```
<matplotlib.image.AxesImage at 0x7f75f232c7d0>
```



9. CONCLUSION

9.1. CONCLUSION

Artificial neural networks are a simple yet powerful technique that can emulate highly complex computing machines. In this project, we used this technique to build a simple combinatorial logic and sequential machine using the backpropagation algorithm. Comparative studies have been conducted between two different architectures of the neural network, and their strengths / weaknesses are mentioned. ANN can be used to implement complex combinational and sequential circuits.

In data communication systems, data security is a major concern. Two methodologies are used to study the usage of ANN in the realm of cryptography. Data encryption is designed using a sequential machine-based mechanism. Also examined is a chaotic neural network for digital signal cryptography. Better results can be obtained by improving the code or employing more effective training techniques. As a result, Artificial Neural Networks can be used to encrypt and decrypt data in innovative ways.

One such application is to use our system for transmission and exchange of extremely confidential information where confidentiality takes precedence over power and time consumption, this system proves to be a very efficient mechanism for exchange of information with very high confidentiality, integrity and non-repudiation requirements. A more lenient approach is to use this system in tandem with a smart router or a program running on both participants of the conversation that is able to detect any anomalous activity, say the presence of an adversary. The use of our custom encryption approach is made only in the said situation.

9.2. FUTURE WORK

As this demonstration includes programmer-defined ANNs, more complex iterations of these models with a lot more hidden layers are imminent. An example of this is the Jordan network having an architecture similar to one we have created.This project tries to use artificial neural networks for cryptography, doing so ensures integrity of data during transmission, and also strengthens data sharing through the network.

Future scope for the project to do obfuscation as the next step of the encryption and hide the key within the encrypted data, that is neural steganography. That will be an optimization of neural cryptography.

As huge advances are observed in the field of big data and machine learning, it is obvious that there will be practical implementations of the proposed model with a lot of tweaks and improvements. Furthermore, this paper only discusses symmetric encryption using machine learning and ANNs. Even stronger and reliable are asymmetric encryption algorithms like RSA, etc. which are yet to have an implementation through machine learning. As predicted with symmetric encryption, it is no longer a point to ponder upon that ANNs will supersede the traditional cryptographic functions, provided there is availability of the huge pool of resources and computing power required to do the same. As of writing this paper, there are a few systems capable of performing neural cryptographic operations but steady advantages are being made at a never-seen-before pace in computing power and high performance computing. If it is successfully implemented and made feasible for everyday applications, a drastic decrease in man-in-the-middle attacks will be imminent.

9.3. APPLICATIONS

The example IoT environment contains a server and IoT devices. We assume that the IoT devices used in our scenario were produced with the server's public key in the manufacturing process. The detailed use-case scenario is as follows:

1) The IoT devices are installed at specific places.

2) The noisy data collected from the IoT devices are encrypted with the server's public key and transmitted to the server.

3) The server decrypts the encrypted noisy data from each of the IoT devices and trains a key with the two noisy data. The training method follows the one detailed in Section. After the training is complete, the architecture and parameters of the DNN are transmitted to each IoT device

Artificial Neural Networks can be used in different environments.

Some applications of ANNs in cryptography can include:

- Key Hiding
- File Encryption
- Image Encryption

- Message Encryption
- Key Exchange

One such application is to use our system for transmission and exchange of extremely confidential information where confidentiality takes precedence over power and time consumption, this system proves to be a very efficient mechanism for exchange of information with very high confidentiality, integrity and non-repudiation requirements.

APPENDIX A

- **CORRECTNESS**

We use the false negative ratio (FNR) and the false positive rate (FPR) for measuring the correctness. In our experiments, the FNR represents the probability of not reproducing a concealed key for proper data. Additionally, the FPR shows the probability of reproducing a concealed key in the case of improper data. In case 1 presented in Table 1, the FNR is 0%,, thereby implying that concealed key is reproduced reliable when proper images and IMEI numbers are used. Furthermore, the FNR for each key size64, 128, and 256shows a same value, thereby implying that our instantiation can stably reproduce the concealed key regardless of the key size. In cases 2,3 and 4, the FPR is 0%. It shows that our instantiation does not reproduce the concealed key with improper

	input data			key size		
	image	IMEI		64	128	256
case1	Proper images	Proper IMEI	FNR	0.0	0.0	0.0
case2	Proper images	Improper IMEI		0.0	0.0	0.0
case3	Improper images	Proper IMEI	FPR	0.0	0.0	0.0
case4	Improper images	Improper IMEI		0.0	0.0	0.0

FNR(%): False Negative Ratio
FPR(%): False Positive Ratio

- **ROBUSTNESS**

- WEIGHTS AGGREGATE EVALUATION

In the case of steady negative training, the Pearson correlation coefficient is approximately 0.09 between the concealed key and guessed key. It implies that the guessed key is analogous to the randomly generated bit strings, and therefore, there is no exposure of information regarding the concealed key In the case of unsteady negative training, however, the Pearson correlation

coefficient is 0:84. This means that the guessed key contains a significant amount of information regarding the concealed key.

- **STATISTICAL EVALUATION**

- In steady negative training, the Pearson correlation coefficient is 0.1 between the concealed key and guessed key. This implies that the guessed key is analogous to the randomly generated bit strings, such there is no potential risk of key exposure. In unsteady negative training, however, the Pearson correlation coefficient is 0:58. This implies that the guessed key includes a significant amount of information of the concealed key.

APPENDIX B

International Journal for Research in Applied Science and Engineering Technology (IJRASET) is an international peer-reviewed, open-access and multidisciplinary online journal published for the enhancement of research in various disciplines of Applied Science & Engineering Technologies.

IJRASET Journal is a multidisciplinary peer-reviewed, UGC recognized, refereed journal with high impact factor and low publication fee. We also implemented a truly open access publication model that provides open global access to its published material so that anyone can download and read articles free of charge around the clock.

IJRASET relies 100% on high-quality science materials, a rigorous peer review analysis of all our research and evaluation papers, ensuring that technology-based research is conducted across a wide variety of engineering fields.


ISSN No. : 2321-9653
iJRASET
International Journal for Research in Applied Science & Engineering Technology
IJRASET is indexed with Crossref for DOI-DOI : 10.22214
Website : www.ijraset.com, E-mail : ijraset@gmail.com
ISRA Journal Impact Factor: 7.429
45.98
INDEX COPERNICUS
THOMSON REUTERS
doi
cross ref
TOGETHER WE REACH THE GOAL
SJIF 7.429
Certificate
It is here by certified that the paper ID : IJRASET42522, entitled
Neural Network based Message Concealment Scheme
by
Prasann Shimpi
after review is found suitable and has been published in
Volume 10, Issue V, May 2022
in
International Journal for Research in Applied Science & Engineering Technology
Good luck for your future endeavors
Editor in Chief, iJRASET


ISSN No. : 2321-9653
iJRASET
International Journal for Research in Applied Science & Engineering Technology
IJRASET is indexed with Crossref for DOI-DOI : 10.22214
Website : www.ijraset.com, E-mail : ijraset@gmail.com
ISRA Journal Impact Factor: 7.429
45.98
INDEX COPERNICUS
THOMSON REUTERS
doi crossref
TOGETHER WE REACH THE GOAL
SJIF 7.429
Certificate
It is here by certified that the paper ID : IJRASET42522, entitled
Neural Network based Message Concealment Scheme
by
Sanket Halake
after review is found suitable and has been published in
Volume 10, Issue V, May 2022
in
International Journal for Research in Applied Science & Engineering Technology
Good luck for your future endeavors
Editor in Chief, IJRASET


ISSN No. : 2321-9653
iJRASET
International Journal for Research in Applied Science & Engineering Technology
IJRASET is indexed with Crossref for DOI-DOI : 10.22214
Website : www.ijraset.com, E-mail : ijraset@gmail.com
ISRA Journal Impact Factor: 7.429
45.98
INDEX COPERNICUS
THOMSON REUTERS
doi
cross ref
TOGETHER WE REACH THE GOAL
SJIF 7.429
Certificate
It is here by certified that the paper ID : IJRASET42522, entitled
Neural Network based Message Concealment Scheme
by
Shashank Singh
after review is found suitable and has been published in
Volume 10, Issue V, May 2022
in
International Journal for Research in Applied Science & Engineering Technology
Good luck for your future endeavors
By
Editor in Chief, iJRASET


ISSN No. : 2321-9653
iJRASET
International Journal for Research in Applied Science & Engineering Technology
IJRASET is indexed with Crossref for DOI-DOI : 10.22214
Website : www.ijraset.com, E-mail : ijraset@gmail.com
ISRA Journal Impact Factor: 7.429
45.98
INDEX COPERNICUS
THOMSON REUTERS
doi cross ref
TOGETHER WE REACH THE GOAL
SJIF 7.429
Certificate
It is here by certified that the paper ID : IJRASET42522, entitled
Neural Network based Message Concealment Scheme
by
Shivam Dharmshetti
after review is found suitable and has been published in
Volume 10, Issue V, May 2022
in
International Journal for Research in Applied Science & Engineering Technology
Good luck for your future endeavors
Editor in Chief, iJRASET

APPENDIX C: PLAGIARISM REPORT

Plagiarism Detector.net

Date: March, 25 2022

PLAGIARISM SCAN REPORT

Excluded Url : None

Content Checked For Plagiarism

REFERENCES

[1] Chandra, Sourabh & Bhattacharyya, Siddhartha & Paira, Smita & Alam, Sk. (2014). A Study and Analysis on Symmetric Cryptography. 10.1109/ICSEMR.2014.7043664. K. Elissa.

[2] Sooksatra, Korn & Rivas, Pablo. (2020). A Review of Machine Learning and Cryptography Applications. 591-597. 10.1109/CSCI51800.2020.00105.

[3] Kalsi, Shruti & Kaur, Harleen & Chang, Victor. (2017). DNA Cryptography and Deep Learning using Genetic Algorithm with NW algorithm for Key Generation. Journal of Medical Systems. 42. 17. 10.1007/s10916-017-0851-z.

[4] P. P. Hadke and S. G. Kale, "Use of Neural Networks in cryptography: A review," 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), 2016, pp. 1-4, doi: 10.1109/STARTUP.2016.7583925.

[5] https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

[6] Mandal, Mrinal & Dutta Banik, Gourab & Chattopadhyay, Debasish & Nandi, Debashis. (2012). An Image Encryption Process based on Chaotic Logistic Map. IETE Tech. Rev.. 29. 395-404. 10.4103/0256-4602.103173.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue V May 2022- Available at www.ijraset.com

Neural Network Based Message Concealment Scheme

Dr. Amol Dhakne[1], Prasann Shimpi[2], Sanket Halake[4], Shivam Dharmshetti[3], Shashank Singh[5]
[1]*Associate Professor, Dept. of Computer Engineering, DYPIEMR*
[2, 3, 4]*Student, Dept. of Computer Engineering, DYPIEMR*
[5]*Associate Professor, Dept. of Computer Engineering, DYPIEMR*

Abstract: ***Neural Cryptography is a new thread that integrates cryptography and neural networks for cryptanalysis and encryption applications. We show that Neural Networks can execute symmetric encryption in an adversarial context in this paper, and we build on the existing literature on the subject. Cryptography's purpose is to make it difficult to decipher a cypher and recreate the plain text without the associated key. Your messages are encrypted with excellent cryptography in such a way that brute force attacks against the algorithm or key are nearly impossible. Good cryptography is secure because it employs extremely long keys and encryption techniques that are resistant to various types of attack. The neural net application is the next step in the evolution of good cryptography. This paper discusses the use of neural networks in cryptography, including how to create neural networks that can be utilized in cryptography.***
Keywords: Cryptography key, encryption system, encryption algorithm, artificial neural network,chaos maps, logistic encryption.

I. INTRODUCTION TO CRYPTOGRAPHY

The creation of new security methods that safeguard anyone from trespasser reading is the focus of cryptography. Data privacy systems are referred to as "cypher systems." The cypher key is a set of rules that are used to encrypt all data. The process of transforming open text, such as a message, into encrypted text using rules is known as encryption. Cryptanalysis of the news is the inverse technique, in which the recipient of the encryption converts it back to the original text. The cypher key must comprise a number of crucial qualities. The best illustration is the singularity of encryption and cryptanalysis. Letters, digits, and punctuation marks in the international alphabet are all the same. All the while with mystery of data the propensity for perusing the code news without realizing the code key was advanced. Code keys were observed intently. The primary objective of cryptology is to figure the code news and to recreate the utilized keys with the assistance of good examination of code news. It utilizes numerical insights, variable based math, numerical phonetics, and so on, just as realized slip-ups made by figures as well. The legitimateness of the open text and the applied code key are reflected in each code framework. Further developing the code key assists with diminishing this legitimacy. The wellbeing of the code framework lies in its insusceptibility against interpretation.

The objective of cryptanalysis is to make it conceivable to take a code message and imitate the first plain message without the comparing key. Two significant strategies utilized in encryption are symmetric and hilter kilter encryption. In symmetric encryption, two gatherings share a solitary encryption-unscrambling key (Khaled, Noaman, Jalab 2005). The sender scrambles the first message (P), which is alluded to as plain message, utilizing a key (K) to create an obviously irregular hogwash, alluded to as code message (C), i.e.:[4]

$$C = \text{Encrypt}(K,P)$$

When the code text is created, it can be sent. Upon receipt, the code text can be changed back to the first plain text by utilizing a decoding calculation and the very key that was utilized for encryption, which can be communicated as follows:

$$P = \text{Decrypt}(K,C)$$

In asymmetric encryption, two keys are utilized, one key for encryption and one more key for unscrambling. The length of a cryptographic key is quite often estimated in bits. The more pieces that a specific cryptographic calculation permits in the key, the more keys are conceivable and the safer the calculation becomes.[4]

A. Introduction to Symmetric Encryption

Symmetric encryption is a type of encryption in which the sender and receiver use the same key to encrypt and decrypt plaintext and ciphertext, respectively.[4] Block or stream cyphers have traditionally been employed in symmetric encryption algorithms. It has been shown, however, that with end-to-end adversarial training, a system of neural networks may learn how to conduct types of 'encryption' and 'decryption' without the usage of a specific cryptographic algorithm.[4]

©IJRASET: All Rights are Reserved

B. Introduction to Artificial Neural Networks

A neural organization is an organization or circuit of neurons, or in this day and age, a counterfeit neural organization consisting of fake neurons or hubs.[1] Along these lines, a neural organization can be either natural (composed of natural neurons) or counterfeit (composed of fake neurons) and used to tackle computerized reasoning (AI) challenges. Counterfeit neural organizations model the associations of natural neurons as loads between hubs. An excitatory connection has a positive weight, while inhibitory associations have a negative weight. A weight is applied to all contributions before they are added. A direct mix is the name for this action.Finally, the yield's sufficiency is constrained by actuation work.[1][2] For example, an adequate yield range is ordinarily somewhere in the range of 0 and 1, in spite of the fact that it may likewise be somewhere in the range of 1 and 1.

II. LITERATURE SURVEY

Sr. No	Paper Title	Authors & Published on	Methodology
1	Deep Neural Networks based key concealment schemes	Taehyuk Kim, Tae Young Youn 04Nov2020	In this paper, we propose a new DNNs-based key con- cealment scheme for concealing cryptographic keys within DNNs.
2	A Neural Network based Approach for Cryptographic Function Detection	Li Jia, Anmin Zhou, Peng Jia, Luping Liu, Yan Wang, Liang Liu 2020	This paper proposed a novel approach for cryptographic function detection in malware.
3	Neural Cryptography based on Complex Valued Neural Networks	Tao Dong,Tingwen Huang 2019	This paper took a complex value based parity machine (CVTPM) approach to neural cryptography.
4	Neural Cryptography Based on Topology Evolving Neural Networks	Yuetong Zhu, Danilo Vasconcellos Vargas, Kouichi Sakurai 2018	This paper suggested a neural network architecture to learn neural cryptography.
5	Neural Cryptography: From symmetric encryption to Adversarial Steganography	Dylan Modesitt, Tim Henry, Jon Coden, and Rachel Lathe 2018	This paper discussed various research in the field of neural cryptography from symmetric encryption to steganography.
6	Use of Neural Networks in Cryptography: A Review	Pranita P. Hadke, Swati G. Kale 2016	This paper gave a review of the current usage and possibilities of the utilization of neural networks in cryptography.
7	Neural Network based Cryptography	Apdullah Yayik,, Yapik Kutlu 2014	This paper discussed the iteration of ANNs in cryptography and their advantages over other methods (symmetric encryption).
8	Cryptography based on Neural Networks	Eva Volna, Michael Janosek, Martin Kotyrba 2014	This paper brought into light the method of practical implementation of neural cryptography in real world applications.
9	Cryptography using Artificial Neural Networks	Vikas Gujral, Satish Pradhan 2009	This paper represented a comparative study between different neural network architectures for an adder and discussed their possible applications in cryptography.
10	Neural Cryptography	Wolfgang Kiesel,Ido Kanter. 2002	This paper discussed the first iteration of ANNs in cryptography and their advantages over other methods (symmetric encryption).

1) A Neural Network based Approach for Cryptographic Function Detection:-

They suggested an unique neuralnet-based technique for identifying cryptographic features in malware in this research, and we constructed a prototype system. Their design is made up of two key parts: Instruction-2-vec, which extracts the semantic information from assembly instructions and converts it into 100-dimensional vectors, and an enhanced neural network. K-Max-CNN- Calculate function embeddings and complete the process of categorizing cryptographic functions with care.

2) Deep Neural Networks based key concealment schemes

To keep the Internet-of-things (IoT) environment secure, employing a cryptographic function to various IoT devices has become vital. An important factor to consider is how to store a cryptographic key (or passwords) securely. A popular method is to store the key in the storage protected by some hardware-based security functions. This paper presents a novel concept to conceal cryptographic keys into deep neural networks (DNNs), named DNNs-based key concealment scheme. In this scheme, a key can be concealed into a proper deep neural network model which is trained with secret input data. We demonstrate the practical applicability of our concept by presenting an instance and a use-case scenario of the DNNs-based key concealment scheme and show its correctness. To prove its robustness, two fundamental security evaluation methods are proposed for investigating the security of the instantiation. To the best of our knowledge, this is the first attempt of its kind.

3) Neural Cryptography based on Complex Valued Neural Networks

A public key exchange technique based on the notion of neural network synchronization is known as neural cryptography. The two neural networks adjust their own weight by sharing output from each other using a neural network's learning method. The weights of the two neural networks are the same after the synchronization is complete. The secret key may be created using the neural network's weights. All existing research, on the other hand, is based on the real-valued neural network paradigm. Rarely are papers on neural cryptography based on a complex-valued neural network model published. A neural cryptography based on the complex-valued tree parity machine network is presented in this technical note.

4) Neural Cryptography Based on Topology Evolving Neural Networks

Mathematical theory is used to build modern encryption methods. Recent research demonstrates a new approach in cryptography based on neural networks. A cryptographic scheme is developed automatically rather than knowing a specific technique. While one type of neural network is employed to implement the method, it is uncertain if the notion of neural cryptography can be accomplished using other neural network design. This attribute is used in this study to develop a neural cryptography scheme based on the Spectrum-diverse unified neuroevolution architecture, a novel topology changing neural network architecture.

5) Neural Cryptography: From symmetric encryption to Adversarial Steganography

Neural Cryptography is an emergent field that aims to combine cryptography with Neural Networks for applications in cryptanalysis and encryption. They (1) demonstrate that Neural Networks may perform symmetric encryption in an adversarial scenario and improve on the existing research on the subject in this study.

They also (2) demonstrate that by putting Neural Networks through known cryptographic games based on Ciphertext Indistinguishability, they may discover known cryptographically unsafe communication. Finally, they (3) discuss more work in Neural Steganography, including building neural end-to-end steganographic (imagein-image, text-in-image, video-in-video) algorithms in the face of adversarial networks seeking to censor.

6) Use of Neural Networks in Cryptography: A Review

Secret information is made illegible for unauthorized users using cryptography. Many cryptographic methods exist, but they are more complicated procedures that demand more processing capacity. This study examines how neural networks aid cryptography and how neural networks and cryptography may be used together to improve security.

7) Neural Network based Cryptography

The use of neural networks for cryptography is demonstrated in this study. There are two steps to the system. In the first stage, neural network-based pseudo-random numbers (NPRNGs) are generated, and the outputs are checked for randomness using randomness tests developed by the National Institute of Standards and Technology (NIST).

8) Cryptography based on Neural Networks

The neural net application is the next step in the evolution of excellent cryptography. This paper discusses the use of neural networks in cryptography, including how to create neural networks that can be utilized in cryptography. An experimental demonstration is also included in this publication.

9) Cryptography using Artificial Neural Networks

Different neural network topologies for an Adder are compared, and their benefits and drawbacks are highlighted. A finite state sequential machine was successfully developed using a Jordan (Recurrent network) trained using the back-propagation technique. The resulting sequential machine was utilized for encryption, with the initial key serving as the decryption key. A chaotic neural network with weights determined by a chaotic sequence was also used to accomplish cryptography.

10) Neural Cryptography

The first iteration of ANNs in cryptography was addressed in this study, as well as its advantages over other approaches (symmetric encryption)

III. PROPOSED MODEL

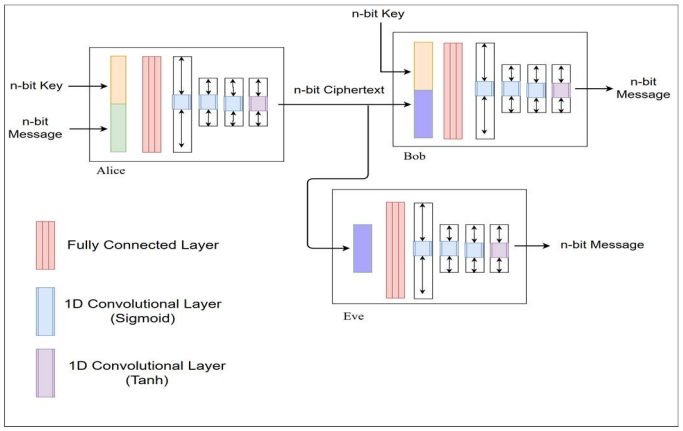


Fig.1 Neural Architecture

The proposed project aims to create three neural networks, Alice, Bob and Eve. Alice and Bob participate in a conversation protected by symmetric encryption. Simultaneously, Eve eavesdrops the conversation. As a result both the receiver and Eve have access to the cipher-text. However, the receiver does have the associated n-bit key. On the other hand, Eve doesn't have the associated key. While the receiver deciphers the cipher text, Eve does try to decrypt without the associated key.[3][5]

A. Dataset

The dataset consists of two equally sized randomly generated strings: one as the plaintext and the second that functions as its associated keys. A uniform random distribution is achieved for this purpose. The generated keys and plaintext have the same length of N bits. The lengths may be 16, 32 or 64 bits. The length may be picked randomly out of the three values.

B. Artificial Neural Networks

Artificial neural networks consist of a pool of simple processing units that communicate with each other by sending signals to each other over a number of weighted links. The set of key aspects of ANN's is:

The activation state *yk* of each unit, corresponding to the output of units. connections between units. In general, each connection is defined by the weight *wjk* . This determines the effect of the signal on unit j on unit k. Propagation rule that determines the effective input *sk* of the unit from the external input. Activation function *Fk* that determines the new activation level based on the valid input sk (t) and the current activation yk (t) (ie update).[6][7][8]

External input of each unit (also called bias or offset) θk.

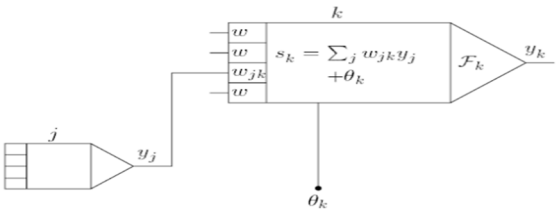


Fig.2 Neural Network

The figure above depicts the basic structure of an artificial neural network and its different components. For the purpose of this project, we construct three neural networks acting as sender, receiver and adversary respectively[6][7][8]. This proposed model, for the purpose of demonstration, includes four layers that make the structure of the convolutional neural networks[4].

C. Convolutional Neural Networks

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning system that can take an input image, assign relevance (learnable weights and biases) to various aspects/objects in the image, and distinguish between them. When compared to other classification methods, the amount of pre-processing required by a ConvNet is significantly less. While basic approaches require hand-engineering of filters, ConvNets can learn these filters/characteristics with enough training.

CNN encryption's robustness against crypto assaults was not examined, because it is beyond the scope of this research. The final results show how well a particular neural network may be used for symmetric cryptography[11]. For the purpose of demonstration in this context, there are three neural networks. They are as follows:

1) *Alice:* Assuming that the sending or initiating network is Alice, the generation of the dataset is done in this neural network. A uniform random generator is implemented for the said generation of the dataset. This stage produces two strings of equal length: the plaintext or message and its associated key. These strings of n-bit each are then passed over to a fully connected layer of this network. The next step for this network is to convert the message and key into a 2n-bit vector, which is then passed to the fully connected layer. It takes in 2n bits and produces a n bit output. This is then passed through a series of sigmoid convolutional layers. At last, the output of the sigmoid layers is then passed to a non-linear tanh layer which scales it to n-bits within the range of -1 to 1 as required.[10]

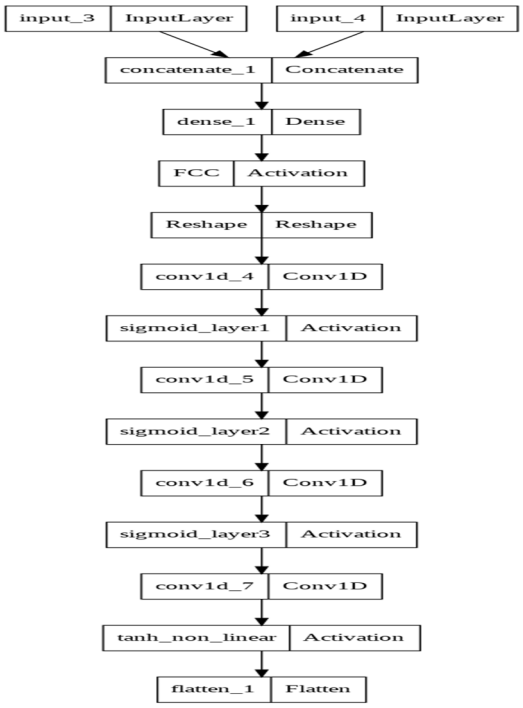


Fig 3: Layers of Alice and Bob

The purpose of each layer is studied later on in this paper.

2) *Bob:* Consider Bob as the neural network as the receiver of the message. The architecture of Bob is an exact replica of Alice's. It consists of a fully connected layer, followed by three sigmoid convolutional layers and a non-linear tanh layer. The input to Bob, however, is the n-bit cipher text generated by the sender, in this case Alice, and the key used for the symmetric encryption. All layers perform the same operation as Alice and the output produced is the deciphered text that is the same as the one that was the input to Alice. This process concludes the last stage of symmetric encryption.[10]
3) *Eve:* In the scenario in consideration, Eve is the adversary, performing a man-in-middle attack.Traditionally, a man-in-the-middle would be able to view messages in transit by eavesdropping through the network. By using machine learning, the neural networks are able to learn crypto-operations and thus eliminates the need for exchanging keys with every message. Here, the architecture of the adversary networks is the same as the others. The inputs to this ANN, however, is only the ciphertext (an identity string is also passed along with the cipher text). This ANN attempts to crack the ciphertext without the use of the secret key known only to Alice and Bob. The primary purpose of this is to test the strength and reliability of using ANNs for symmetric encryption while saving human effort.[10]

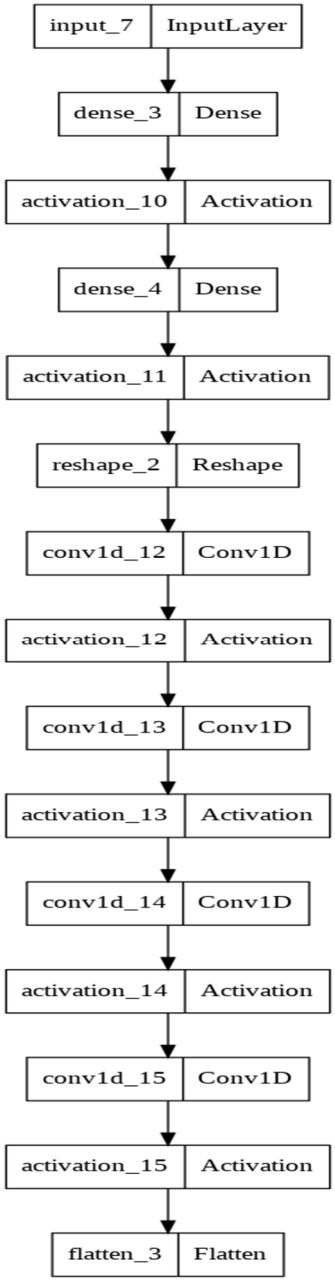


Fig 4: Layers of Eve

D. Proposed Neural Network Architecture

The proposed architecture of each ANN consists of four layers. They are as follows:

1) *Fully Connected Layer:* FCNNs are a sort of artificial neural network in which all of the nodes, or neurons, in one layer are connected to the neurons in the following layer. While this type of technique is often used to process certain types of data, it has certain limitations when it comes to image identification and classification. Such networks require a lot of processing power and are prone to overfitting. When such networks are also 'deep,' that is, when there are multiple layers of nodes or neurons, they can be very difficult to comprehend for humans.

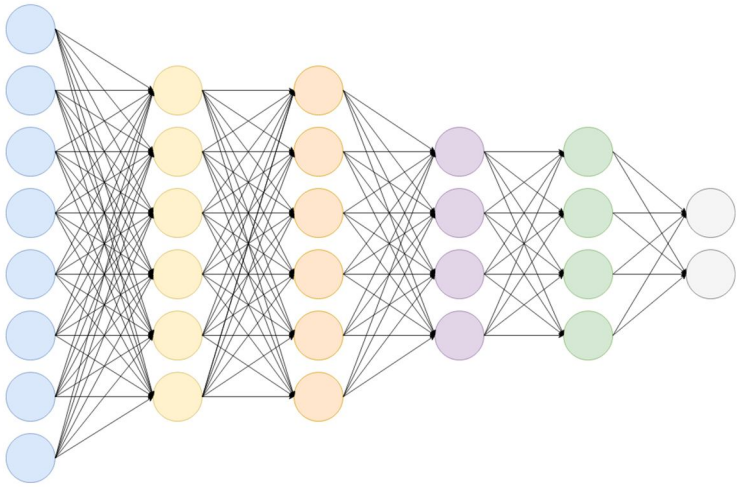


Fig 5: Fully Connected Layer

2) *Sigmoid Layer:* The sigmoid function is a type of logistic function that is commonly symbolised by the o(x) or sig (x). It is provided by:

$$\sigma(x) = 1/(1+\exp(-x))$$

In neural networks, the sigmoid function is utilized as an activation function. An activation function is used to pass a weighted sum of inputs through, and the output is used as an input to the following layer. When a neuron's activation function is a sigmoid function, the output of this unit will always be between 0 and 1. The output of this unit would also be a nonlinear function of the weighted sum of inputs, as the sigmoid is a nonlinear function. A sigmoid unit is a type of neuron that uses a sigmoid function as an activation function[12].

3) *Tanh Layer:* The Tanh (also "tanh" and "TanH") function is another name for the hyperbolic tangent activation function. It resembles the sigmoid activation function in appearance and even has the same S-shape. The function accepts any real value as input and returns a value between -1 and 1. The larger the input (more positive), the closer the output to 1.0, and the smaller the input (more negative), the closer the output to -1.0.

This is how the Tanh activation function is calculated:

$$(e^x - e^{-x}) / (e^x + e^{-x})$$

Where e is the base of the natural logarithm and is a mathematical constant[13].

E. Encrypting and Decrypting Images

The model proposed above works only for text messages. In order to work with files and media content like images, this proposes a combined implementation of chaos maps and the model proposed above. Chaotic theory is a branch of mathematics that studies the dynamic behavior of natural and manmade systems that are affected by initial conditions such as weather, climate, and road traffic. It can be investigated using a chaotic mathematical model, as well as recurrence plots and Poincare maps. Emerging technologies such as neurology, cardiology, control and circuit theory, weather prediction, and others use chaos theory. Chaos is defined as "when the present determines the future, yet the approximate present cannot approximate the future." In chaos, even little changes in the original conditions can result in a sequence that is completely unrelated. For our purpose we have used logistic chaotic maps.[14]

1) Logistic Maps

Mathematically, the logistic map is represented using the equation

$f(x) = rx(1 - x)$

$xn+1 = f(xn)$ …(1)[14].

Here, xn depicts the chaotic sequence ranging between 0 and 1 as illustrated in Figure 5.

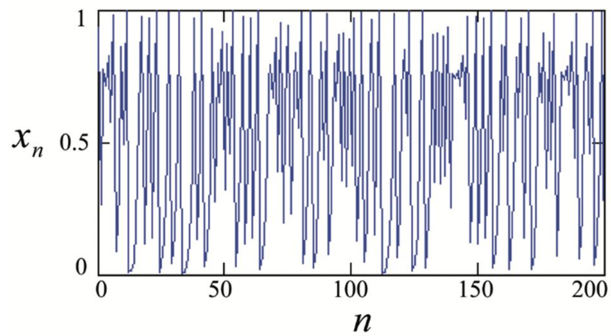


Fig. 5: Variation of chaotic logistic maps with iteration values[14].

Initially, the map is in the xn=0 = $x0$ ∈[0,1] state. The variable r takes a value between 0 and 4.When r is between 0 and 1, xn = 0 regardless of the beginning conditions x0. When r is between 1 and 3, the value of xn stabilizes at (r 1) r, regardless of the initial conditions x0. When r is between 3 and 1+ 6 (about 3.45), the value of xn oscillates between two values for as long as r is between 3 and 1+ 6 (roughly 3.45)[14]. The value of xn oscillates between four values forever when r is between 3.45 and 3.54 (roughly). The value of xn oscillates between 8 values when r is somewhat larger than r, then 16, 32, and so on. The commencement of chaos occurs at roughly 3.57 r, at the end of the period doubling cascade[14]. Small changes in the starting state produce significantly different effects over time in this region, which is a key feature of chaos. Beyond 3.57, the behavior is chaotic, but there are isolated values of r that appear to be non-chaotic; these are commonly referred to as islands of stability.[14].The logistic map is a one-dimensional discrete-time map with an astonishing level of complexity despite its formal simplicity. It was historically one of the most important and archetypal systems in the early days of deterministic chaos study[15].

2) Image Encryption

There are three major processes in the encryption algorithm. The chaotic sequences are generated in the first stage. To create the requisite encrypted image, the second stage confused the pixel values, and the third step shuffled the pixel position. Let f be a picture with the dimensions M N. The pixel of f is represented by f(i,j), where i and j are in the range of 1<=i<=M and 1<=j<=N. The grey value at the pixel position (i, j) of the picture f is now denoted by f(i,j). The logistic map's initial condition is derived from a 256-bit (32-character) secret key written in ASCII as K = K1K2 K3......K32 (Ki signifies the 8-bit key character in the i-th key position).[14]

The step by step procedure for logistic encryption using Chaos Maps is as follows[14]:

Step 1: Convert the picture of size M N pixels into an array of P = P,P,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L,P,L Then, using the mod operation, transform the pixel values to an unsigned integer in the range of 0 to 255.The initial values of the chaos map is recalculated after every pixel encryption based on the previous encryption value as well as the key value.

Step 2: With the initial condition x0 and the parameter r = 3.999, generate n number of chaotic sequences xi ={x1,x2,x3,L,xn} in the range 0 to 1 using the logistic map mentioned in Eq. (1). Then, using the mod operation, turn xi into an unsigned integer in the range of 0 to 255.

Step 3: To confuse iii the pixel value, generate the sequence C = P x. The bitwise XOR operation is indicated by the symbol.

Step 4: To get the image f', convert Ci = {C1,C2,C3,L,Cn} to an array of size MN. Then, to get X, add one to the unsigned integer sequence xi = {x1, x2, x3,L, xn} and convert it to an array of size M N.

Step 5: Finally, perform the pixel shuffling procedures below to obtain the needed encrypted image f. The values of j and k range from 1 to 255. The sign indicates that the values of two pixel positions of f ′ are swapped.

f'(X(j,j),k)⇔ f'(X(j+1,j+1),k)

f'(k,X(j,j))⇔ f'(k,X(j+1,j+1)). (4)

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

The final encrypted image is now f.

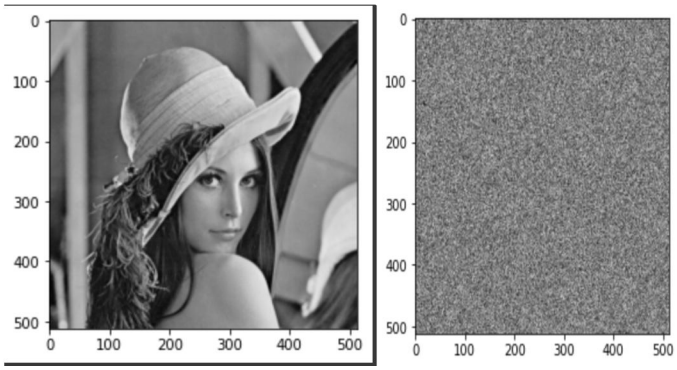


Fig.6 : Image Encryption using Logistic Chaotic Maps

3) Image Decryption

The process of decrypting an image is the inverse of encryption.

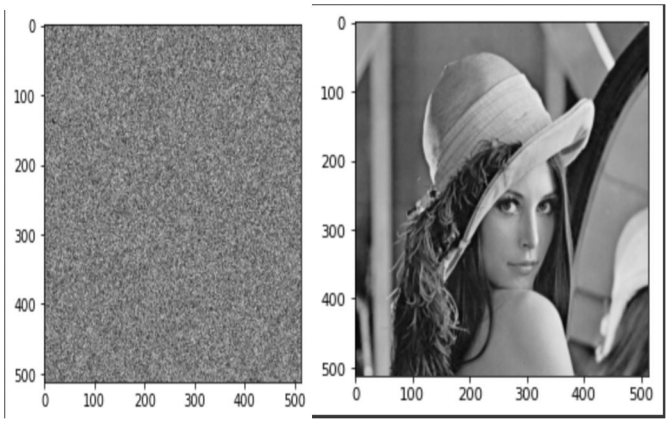


Fig 7: Image Decryption using Logistic Decryption

4) Proposed Methodology for Image-based Cryptography:

For the purpose of implementing image encryption in our demonstration, we use logistic encryption with key mixing. The generation of the chaos sequence is according to the steps mentioned above but each time a new sequence is generated, it is done with respect to the previous sequence and the key.

The key required for logistic encryption is a string in which only a fixed number of characters are implemented for the purpose of cryptography. This key is then encrypted using the ANN for encryption, in this case Alice, and thus the encrypted key along with the cipher image is transmitted.

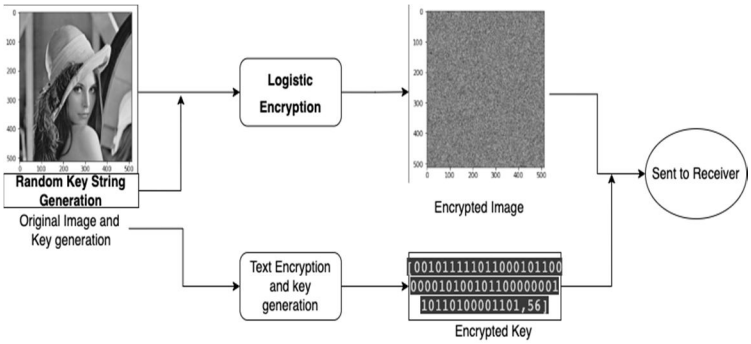


Fig 8: Proposed Image Encryption Methodology

The process of decryption is the reverse of the above process. The encrypted key is first decrypted using the decryption ANN model, in this case Bob. On obtaining the key for image decryption, the “plain” image is then obtained from the encrypted image.

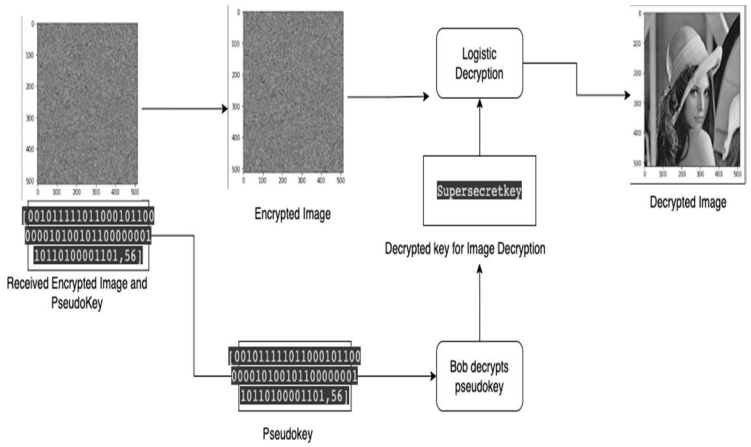


Fig. 9: Decryption methodology for said encryption scheme.

F. Advantages

If successfully implemented, this method will have the following advantages:

1) A large number of attackers have been trained, and each new time step is multiplied to cover the possible internal representation of the current output.
2) The dynamics of a successful attacker precedes, so the attacker stays while the unsuccessful attacker leaves. A Probabilistic attack in which an attacker attempts to track the probability of each weighting element by calculating the local field distribution for each input and using publicly known outputs.
3) As mentioned in point 2 on the learning concept that weight adjustment is done by accident, this randomization is unknown to the attacker, so these ideas lead to a very distant adjustment range.
4) The need for sharing the secret key every time a message is sent and received (though this method is implemented in some deprecated systems) is eliminated.
5) Once the sender and receiver networks are in a stable synchronization, the need for sharing the key is eliminated as the network itself acts as a key.

G. Limitations

1) As it involves machine learning, a lot of time is spent in training the models, which is saved in the traditional Diffe-Hillman algorithms.
2) The limitations of this type of system are minor, but potentially important. This is effectively a private key system where the key is the weight and architecture of the network. Breaking the weights and architecture makes encryption easier. However, encryption and decryption require both weight and architecture. Knowing one or the other is not enough to break it.
3) The advantage of this system is that it seems very difficult to break through without knowledge of the methodology behind it.
4) Due to its implementation being through machine learning, it is tolerant towards noise. Most messages cannot be modified by even a single bit with standard encryption schemes. A neural network-based system allows the encoded message to be varied and still accurate.

H. Future Possibilities

As huge advances are observed in the field of big data and machine learning, it is obvious that there will be practical implementations of the proposed model with a lot of tweaks and improvements. Furthermore, this paper only discusses symmetric encryption using machine learning and ANNs. Even stronger and reliable are asymmetric encryption algorithms like RSA, etc. which are yet to have an implementation through machine learning. As predicted with symmetric encryption, it is no longer a point to ponder upon that ANNs will supersede the traditional cryptographic functions, provided there is availability of the huge pool of resources and computing power required to do the same. As of writing this paper, there are a few systems capable of performing neural cryptographic operations but steady advantages are being made at a never-seen-before pace in computing power and high performance computing. If it is successfully implemented and made feasible for everyday applications, a drastic decrease in man-in-the-middle attacks will be imminent.

REFERENCES

[1] Grossi, Enzo & Buscema, Massimo. (2008). Introduction to artificial neural networks. European journal of gastroenterology & hepatology. 19. 1046-54. 10.1097/MEG.0b013e3282f198a0.
[2] Zupan, Jure. (1994). Introduction to Artificial Neural Network (ANN) Methods: What They Are and How to Use Them. Acta Chimica Slovenica. 41.
[3] T. Dong and T. Huang, "Neural Cryptography Based on Complex-Valued Neural Network," in IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 11, pp. 4999-5004, Nov. 2020, doi: 10.1109/TNNLS.2019.2955165.
[4] Chandra, Sourabh & Bhattacharyya, Siddhartha & Paira, Smita & Alam, Sk. (2014). A Study and Analysis on Symmetric Cryptography. 10.1109/ICSEMR.2014.7043664. K. Elissa.
[5] O'Shea, Keiron & Nash, Ryan. (2015). An Introduction to Convolutional Neural Networks. ArXiv e-prints.
[6] Volna, Eva & Kotyrba, Martin & Kocian, Vaclav & Janosek, Michal. (2012). Cryptography Based On Neural Network. 10.7148/2012-0386-0391.
[7] Sooksatra, Korn & Rivas, Pablo. (2020). A Review of Machine Learning and Cryptography Applications. 591-597. 10.1109/CSCI51800.2020.00105.
[8] Kalsi, Shruti & Kaur, Harleen & Chang, Victor. (2017). DNA Cryptography and Deep Learning using Genetic Algorithm with NW algorithm for Key Generation. Journal of Medical Systems. 42. 17. 10.1007/s10916-017-0851-z.
[9] P. P. Hadke and S. G. Kale, "Use of Neural Networks in cryptography: A review," 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), 2016, pp. 1-4, doi: 10.1109/STARTUP.2016.7583925.
[10] T. Kim, T. Y. Youn and D. Choi, "Deep Neural Networks Based Key Concealment Scheme," in IEEE Access, vol. 8, pp. 204214-204225, 2020, doi: 10.1109/ACCESS.2020.3036650.
[11] R. Forgáč and M. Očkay, "Contribution to Symmetric Cryptography by Convolutional Neural Networks," 2019 Communication and Information Technologies (KIT), 2019, pp. 1-6, doi: 10.23919/KIT.2019.8883490.
[12] https://machinelearningmastery.com/a-gentle-introduction-to-sigmoid-function/
[13] https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/
[14] Mandal, Mrinal & Dutta Banik, Gourab & Chattopadhyay, Debasish & Nandi, Debashis. (2012). An Image Encryption Process based on Chaotic Logistic Map. IETE Tech. Rev.. 29. 395-404. 10.4103/0256-4602.103173.
[15] https://www.complexity-explorables.org/flongs/logistic/