A Detailed Project Report

On

GRU-Based Chaotic Pixel Encryption

Submitted to

University of Petroleum and Energy Studies

In Accordance with The Award of The Degree Of

BACHELORS IN TECHNOLOGY(NON-HONS.)

In

COMPUTER SCIENCE AND ENGINEERING

(With Major in cyber security and Forensics)

By

Specialization	**SAP ID**	**Name**
CSF	500087942	MAYANK SONWANE
CSF	500086709	TANUSHPREET KAUR
CSF	500083113	VIJAY MALIK
CSF	500086715	VISHAL OHDAR

6/12/23

Under the guidance of

Dr. Virender Kadyan.

University of Petroleum and Energy Studies

Dehradun-India

UPES

CANDIDATE'S DECLARATION

We hereby certify that the project entitled " GRU-Based Chaotic Pixel Encryption" submitted to the Department of Systemic at the, is **School of Computer Science, University of Petroleum & Energy Studies, Dehradun** an authentic record of our work and satisfies the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with a focus on CYBER SECURITY AND FORENSICS. An authentic record of our work completed from August 2023 to December 2023 under the direction of **Dr. Virender Kadyan,** Associate Professor, School of cyber security, has been submitted to Dr. ~~Alok~~ Ajay Prasad the Department of Systemic.

We have not submitted the subject matter covered in this project for the granting of any other degree from this or any other University. This is to confirm that, to the best of my knowledge, the candidate's above statement is accurate.

Date: 1 December 2023

Dr. Virender Kadyan
Project Guide

Dr. Ajay Prasad
Head – Department of

School of Computer Science
University of Petroleum & Energy
Studies, Dehradun – 248001
(Uttarakhand)

ACKNOWLEDGEMENT

We would like to extend our sincere thanks to our guide, **Dr. Virender Kadyan**, for all the guidance, inspiration, and unwavering support he provided us with during the course of our project work. Without his encouragement and helpful recommendations, this effort would not have been feasible.

We would like to express our heartfelt gratitude to **Dr.** , Head of the Department, for her exceptional assistance with our project, **GRU-based chaotic pixel encryption.**

We are also grateful to **Prof. S. Ravi Shankar**, the **School of Computer Science Dean at UPES**, who gave us the resources we need to successfully finish our project work.

We would like to thank all of our friends for their support and constructive feedback during the course of our project effort. Finally, we can only express our sincere gratitude to our parents for introducing us to the outside world and for all of the assistance they have provided.

Table of Contents

Mentor Meet Report

S.NO	DATE	DISCUSSION
1	24 Aug 2023	Finalized topic for Minor project -1 with mentor.
2	23 Sep 2023	Synopsis presentation
3	23 Nov 2023	Documented SRS and report for mid-sem presentation
4	25 Nov 2023	Mid sem Presentation
5	28 Nov 2023	Analysing mistakes focused by mid-sem evaluation
6	30 Dec 2023	Documenting the Report file
7	1 Dec 2023	Final Documentation

1. Abstract

In today’s era, the digital information is paramount. However, a malicious user can steal and misuse this information. Hence, it becomes vital to have a strong encryption method to save data from various types of malicious attacks. The proposed approach, developed by Atul Kumar and Mohit Dua, introduces an image encryption method that leverages the power of Gated Recurrent Units (GRU) and Chaos-based techniques to protect the transport related data. This approach offers a robust method to the ever-evolving challenges of image security. By combining the strengths of GRU, a recurrent neural network architecture and the unpredictability of chaos theory, this encryption method provides an exception defence against unauthorised attacks and ensures the confidentiality and integrity of sensitive visual data during transmission and storage. This research represents a huge advancement in the field of image encryption.

2. Introduction

In today’s digital age, the secure transmission and storage of sensitive visual information, such as transport images, have become important in ensuring the integrity and confidentiality of critical data [1]. The management of traffic system is very complex so to manage it An Intelligent Transport System (ITS) is used which uses smart devices to capture the traffic data in the form of images. This system uses smart devices to perform real-time traffic management, public data collection, and vehicles speed checking [2]. These smart devices capture and transmit traffic data in the form of images to server via public channel, which can be vulnerable to theft and misuse during communication. To address these challenges, a novel image encryption approach has been developed [3].

An image encryption method that combines the power of Gated Recurrent Units (GRU) and chaos theory principles. GRU is a type of recurrent neural network architecture, exhibits its use in sequential data processing tasks, while chaos theory generates unpredictable and complex cryptographic keys. The fusion of these two elements results in a robust image encryption scheme which provides a solution to the specific demands of securing transport-related visual data.[4]

The encryption algorithm that uses the Gated Recurrent Unit (GRU) and Sine-Cosine chaotic map to encrypt transport images is divided into three phases. In the first phase, two intermediate keys and the seed value required for creating chaotic sequence are generated using unique combinations of 128-bit share key and 128-bit initial vector. In the second phase, permutation is performed using one of the intermediate keys and the chaotic sequence generated by the novel Sine-Cosine chaotic map. The final phase performs the diffusion process using the other intermediate key and GRU approach that uses the chaotic sequence generated by the Sine-Cosine map [5].

2.1. Problem Statement

Use of encryption techniques to protect traffic-related image data collected by Intelligent Transport Systems (ITS) from potential adversaries. As these smart devices capture and transmit traffic data in the form of images, this can be vulnerable to theft and misuse during communication. To address this issue, we are using encryption algorithm that combines the Gated Recurrent Unit (GRU) and the Sine-Cosine chaotic map. This encryption approach aims to safeguard transport images through three key phases: key generation, permutation using chaotic sequences, and diffusion using the GRU approach.

2.2. Objectives

The objective is to develop an efficient encryption scheme for securing transport images that resists against various security attacks.

Sub-Objective 1:

Generating intermediate keys and a seed value for creating chaotic sequences from a combination of 128-bit shared keys and a 128-bit initial vector. To enhance the security of the encryption process, we will be using the Sine-Cosine chaotic map.

Sub-Objective 2:

Integrating the Gated Recurrent Unit (GRU), a type of recurrent neural network, into the encryption process. The GRU is used in the diffusion phase, which involves modifying pixel values in the encrypted images.

2.3. Methodology

Step 1: Collection of datasets

Step 2: Generating a 128-bit shared key and a 128-bit initial vector

Step 3: Generation of permuted image using sine-cosine map

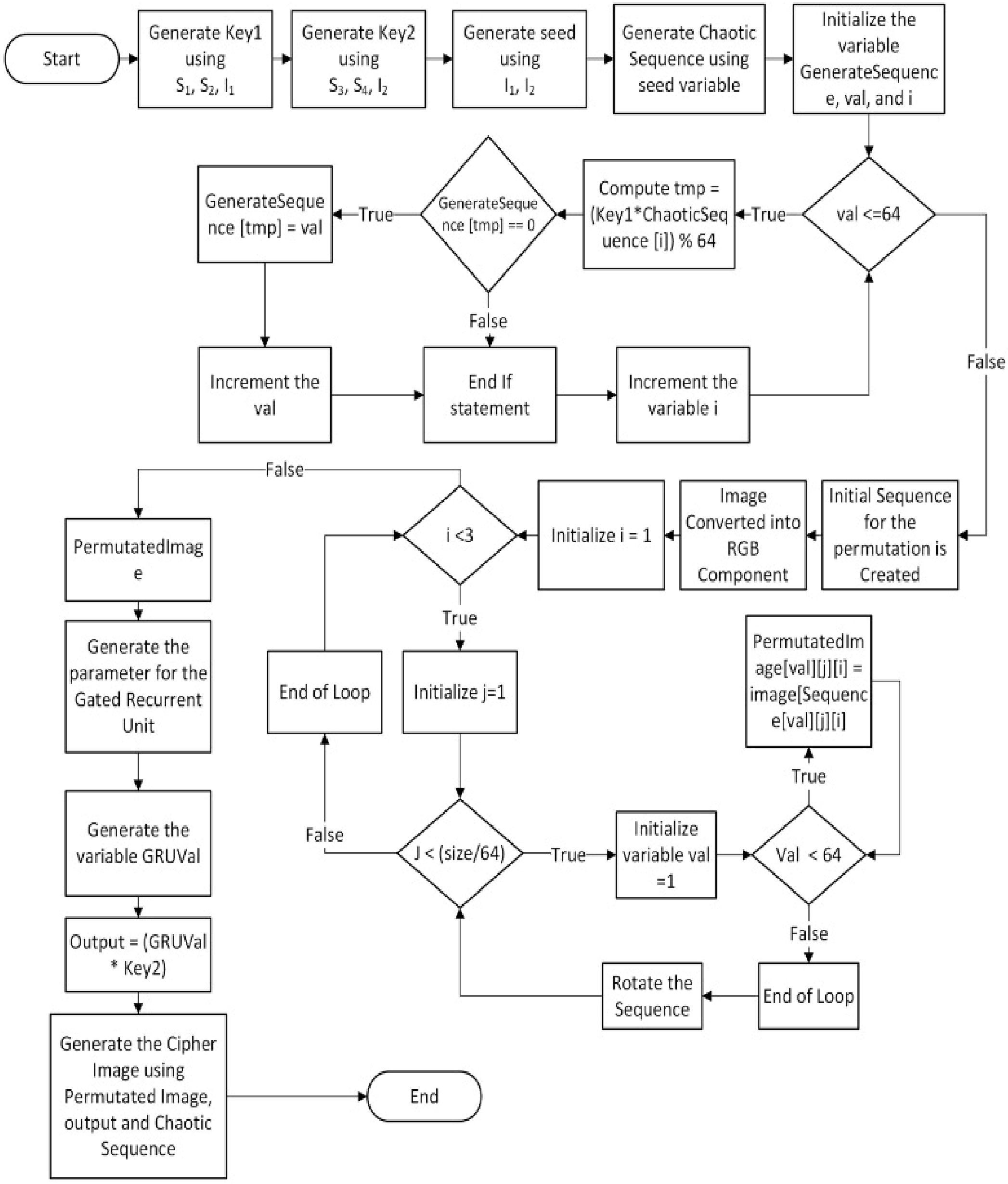
Step 4: Encryption using Gated Recurrent Unit (GRU)

Step 5: Performance Evaluation using a Bifurcation diagram, Lyapunov exponent and Shannon entropy

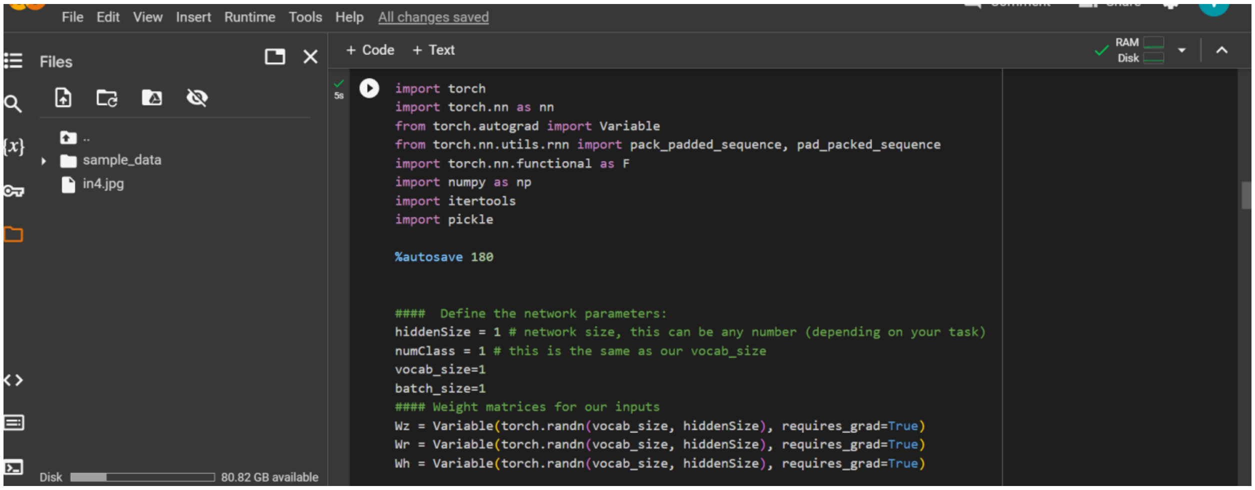
2.4. Algorithm

Step 1. Load the input image ('in.jpg') using OpenCV.
Step 2. Display the original image.
Step 3. Get the height and width of the image.
Step 4. Generate a chaotic sequence using the `chaoticSequ_Sine_Cosine_Map` function. This sequence will be used for pixel permutation.
Step 5. Permute the image using the generated chaotic sequence and display the shuffled image.
Step 6. Define a hidden unit count and calculate the GRU matrix using the chaotic sequence.
Step 7. Multiply the GRU matrix by a key (key2) to generate an output matrix.
Step 8. Encrypt the shuffled image using the output matrix and key2, displaying the encrypted image.
Step 9. Decrypt the encrypted image using the same key and output matrix, displaying the decrypted image.
Step 10. Un shuffle the decrypted image using the same chaotic sequence and display the unshuffled image.
Step 11. Save the unshuffled image as 'original_row.jpg'.

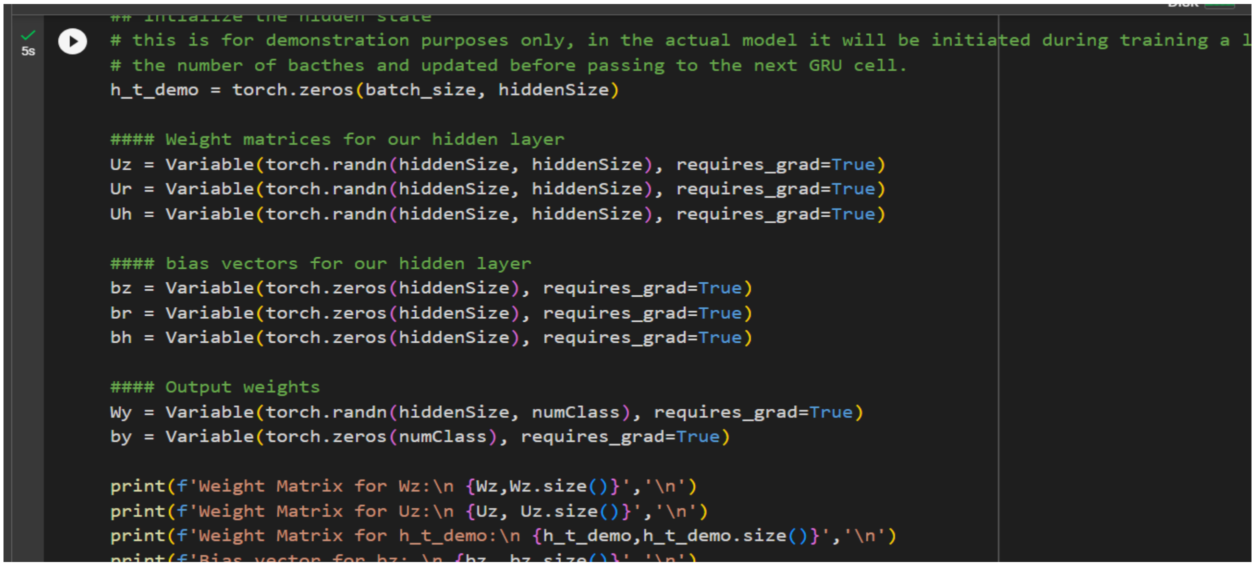
3. CODE


Start
Generate Key1 using S_1, S_2, I_1
Generate Key2 using S_3, S_4, I_2
Generate seed using I_1, I_2
Generate Chaotic Sequence using seed variable
Initialize the variable GenerateSequence, val, and i
val <=64
True
Compute tmp = (Key1*ChaoticSequence [i]) % 64
GenerateSequence [tmp] == 0
True
GenerateSequence [tmp] = val
False
Increment the val
End If statement
Increment the variable i
False
Initial Sequence for the permutation is Created
Image Converted into RGB Component
Initialize i = 1
i <3
False
True
PermutatedImage
Generate the parameter for the Gated Recurrent Unit
Generate the variable GRUVal
Output = (GRUVal * Key2)
Generate the Cipher Image using Permutated Image, output and Chaotic Sequence
End
Initialize j=1
End of Loop
J < (size/64)
False
True
Initialize variable val =1
Val < 64
True
PermutatedImage[val][j][i] = image[Sequence[val][j][i]
False
End of Loop
Rotate the Sequence

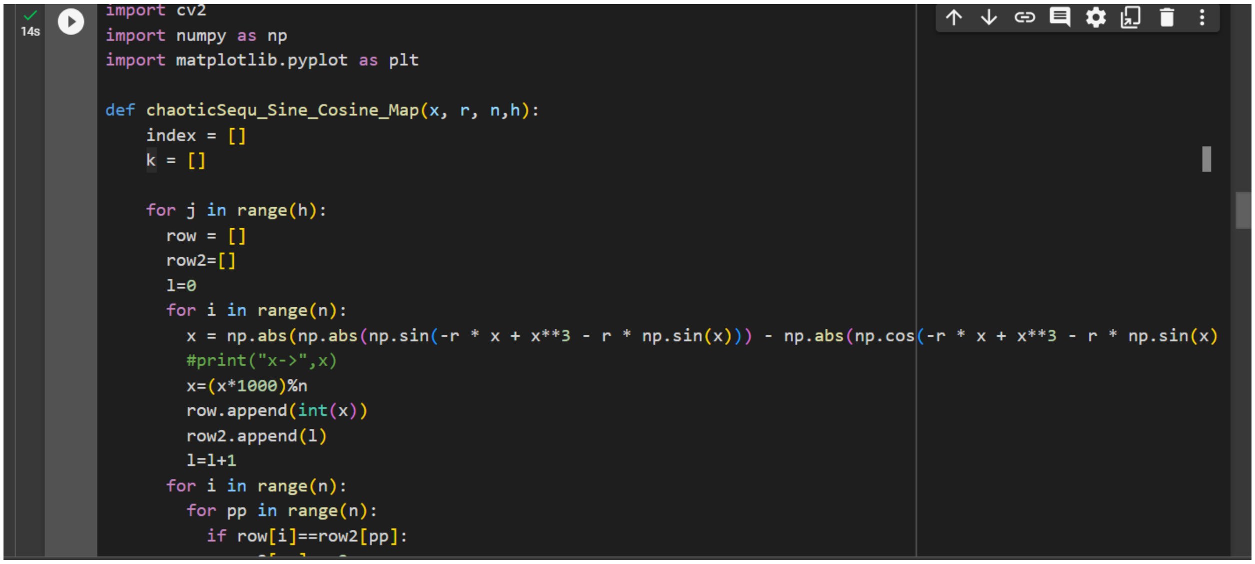
3.CODE



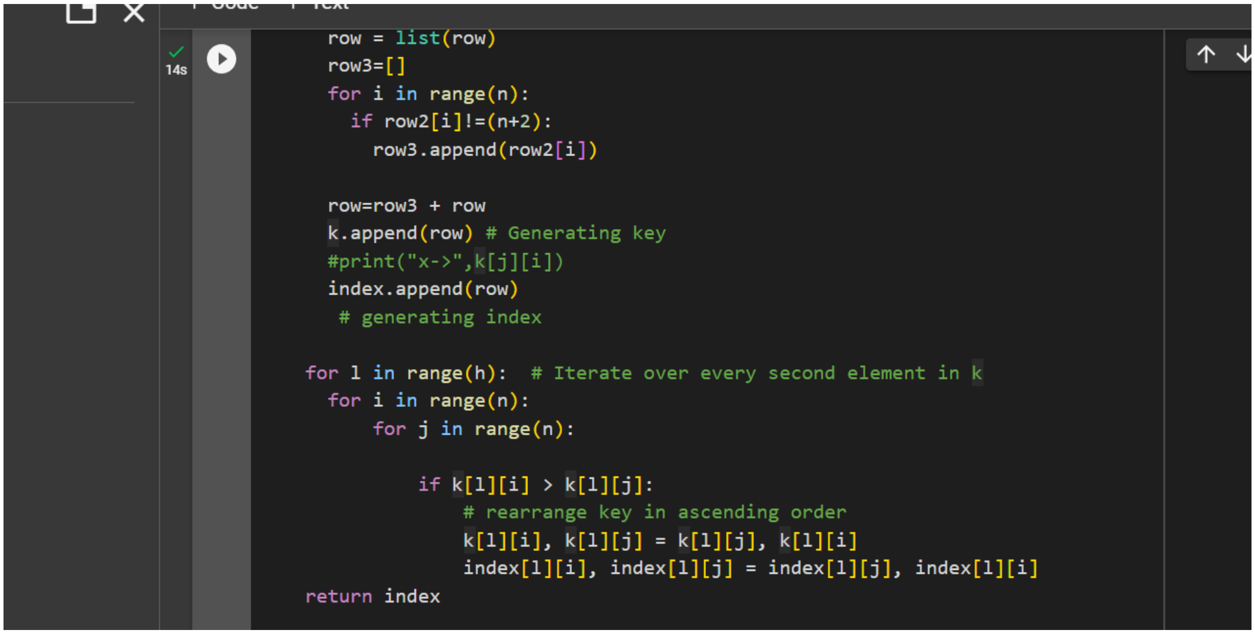
```
import torch
import torch.nn as nn
from torch.autograd import Variable
from torch.nn.utils.rnn import pack_padded_sequence, pad_packed_sequence
import torch.nn.functional as F
import numpy as np
import itertools
import pickle

%autosave 180


####  Define the network parameters:
hiddenSize = 1 # network size, this can be any number (depending on your task)
numClass = 1 # this is the same as our vocab_size
vocab_size=1
batch_size=1
#### Weight matrices for our inputs
Wz = Variable(torch.randn(vocab_size, hiddenSize), requires_grad=True)
Wr = Variable(torch.randn(vocab_size, hiddenSize), requires_grad=True)
Wh = Variable(torch.randn(vocab_size, hiddenSize), requires_grad=True)
```



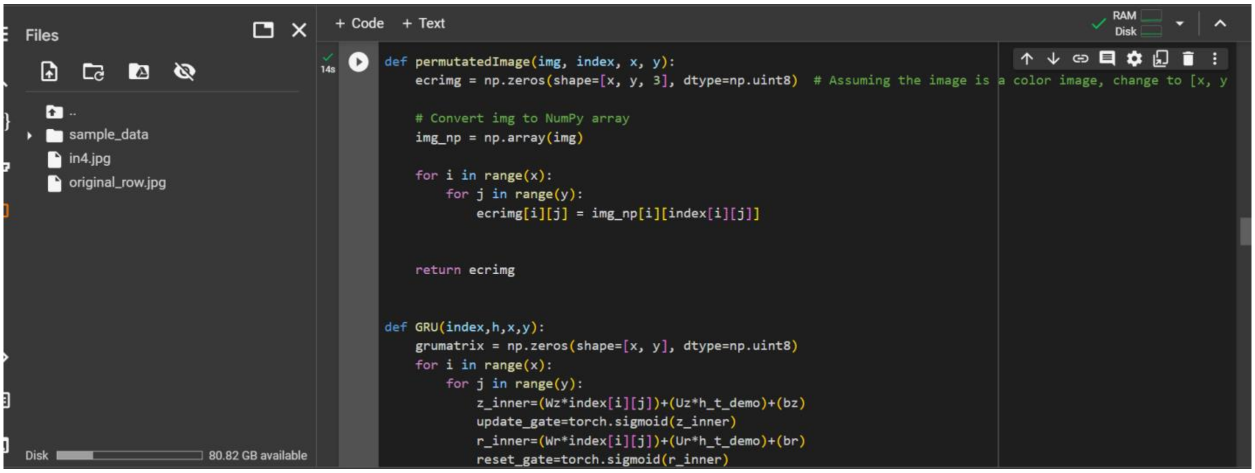
```
## initialize the hidden state
# this is for demonstration purposes only, in the actual model it will be initiated during training a l
# the number of bacthes and updated before passing to the next GRU cell.
h_t_demo = torch.zeros(batch_size, hiddenSize)

#### Weight matrices for our hidden layer
Uz = Variable(torch.randn(hiddenSize, hiddenSize), requires_grad=True)
Ur = Variable(torch.randn(hiddenSize, hiddenSize), requires_grad=True)
Uh = Variable(torch.randn(hiddenSize, hiddenSize), requires_grad=True)

#### bias vectors for our hidden layer
bz = Variable(torch.zeros(hiddenSize), requires_grad=True)
br = Variable(torch.zeros(hiddenSize), requires_grad=True)
bh = Variable(torch.zeros(hiddenSize), requires_grad=True)

#### Output weights
Wy = Variable(torch.randn(hiddenSize, numClass), requires_grad=True)
by = Variable(torch.zeros(numClass), requires_grad=True)

print(f'Weight Matrix for Wz:\n {Wz,Wz.size()}','\n')
print(f'Weight Matrix for Uz:\n {Uz, Uz.size()}','\n')
print(f'Weight Matrix for h_t_demo:\n {h_t_demo,h_t_demo.size()}','\n')
print(f'Bias vector for bz:\n {bz, bz.size()}','\n')
```



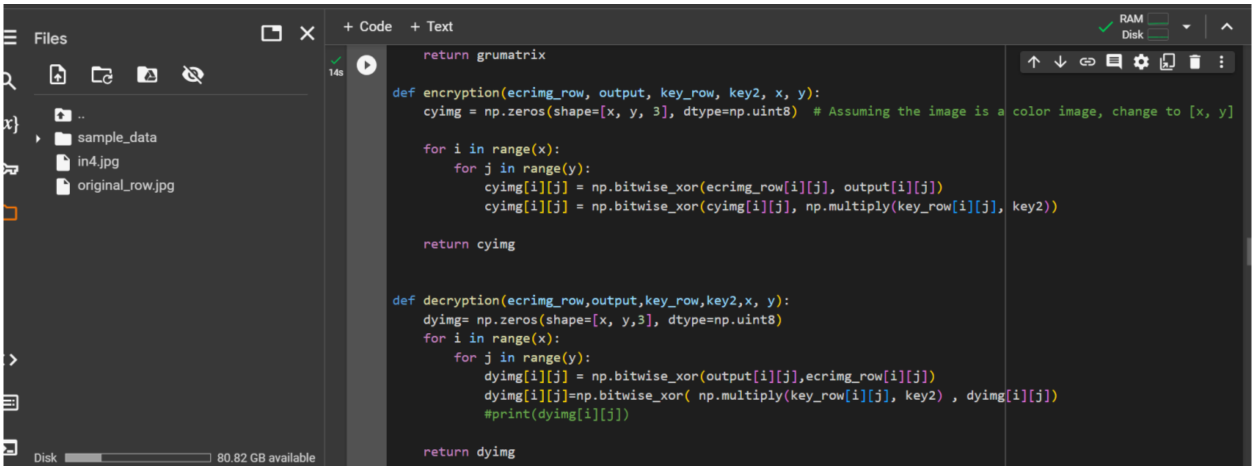
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

def chaoticSequ_Sine_Cosine_Map(x, r, n,h):
    index = []
    k = []

    for j in range(h):
      row = []
      row2=[]
      l=0
      for i in range(n):
        x = np.abs(np.abs(np.sin(-r * x + x**3 - r * np.sin(x))) - np.abs(np.cos(-r * x + x**3 - r * np.sin(x)
        #print("x->",x)
        x=(x*1000)%n
        row.append(int(x))
        row2.append(l)
        l=l+1
      for i in range(n):
        for pp in range(n):
          if row[i]==row2[pp]:
```



```
     row = list(row)
     row3=[]
     for i in range(n):
      if row2[i]!=(n+2):
       row3.append(row2[i])

     row=row3 + row
     k.append(row) # Generating key
     #print("x->",k[j][i])
     index.append(row)
      # generating index

    for l in range(h):  # Iterate over every second element in k
     for i in range(n):
       for j in range(n):

         if k[l][i] > k[l][j]:
           # rearrange key in ascending order
           k[l][i], k[l][j] = k[l][j], k[l][i]
           index[l][i], index[l][j] = index[l][j], index[l][i]
    return index
```



```
def permutatedImage(img, index, x, y):
    ecrimg = np.zeros(shape=[x, y, 3], dtype=np.uint8)  # Assuming the image is a color image, change to [x, y

    # Convert img to NumPy array
    img_np = np.array(img)

    for i in range(x):
        for j in range(y):
            ecrimg[i][j] = img_np[i][index[i][j]]


    return ecrimg


def GRU(index,h,x,y):
    grumatrix = np.zeros(shape=[x, y], dtype=np.uint8)
    for i in range(x):
        for j in range(y):
            z_inner=(Wz*index[i][j])+(Uz*h_t_demo)+(bz)
            update_gate=torch.sigmoid(z_inner)
            r_inner=(Wr*index[i][j])+(Ur*h_t_demo)+(br)
            reset_gate=torch.sigmoid(r_inner)
```



```
    return grumatrix

def encryption(ecrimg_row, output, key_row, key2, x, y):
    cyimg = np.zeros(shape=[x, y, 3], dtype=np.uint8)  # Assuming the image is a color image, change to [x, y]

    for i in range(x):
        for j in range(y):
            cyimg[i][j] = np.bitwise_xor(ecrimg_row[i][j], output[i][j])
            cyimg[i][j] = np.bitwise_xor(cyimg[i][j], np.multiply(key_row[i][j], key2))

    return cyimg


def decryption(ecrimg_row,output,key_row,key2,x, y):
    dyimg= np.zeros(shape=[x, y,3], dtype=np.uint8)
    for i in range(x):
        for j in range(y):
            dyimg[i][j] = np.bitwise_xor(output[i][j],ecrimg_row[i][j])
            dyimg[i][j]=np.bitwise_xor( np.multiply(key_row[i][j], key2) , dyimg[i][j])
            #print(dyimg[i][j])

    return dyimg
```

4. GRAPHS

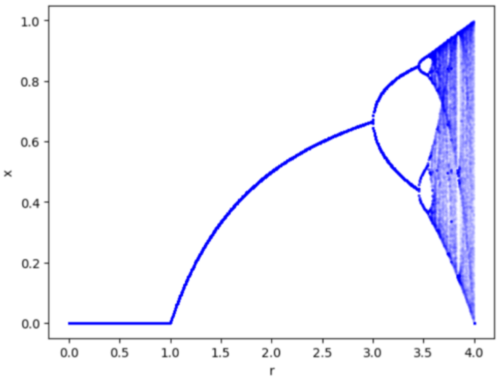
4.1. Logistic Map

The logistic map is a mathematical equation that describes a simple model of population growth. It is also used as a basic example of chaotic dynamics, where small differences in initial conditions can lead to vastly different outcomes.

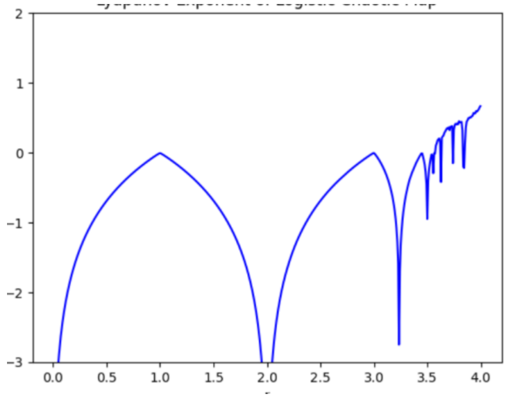
The logistic map equation is expressed as:

$$Xn+1 = rXn(1 - Xn)$$

where Xn is the population at time step n, Xn+1 is the population at the next time step, and r is the growth rate parameter. The term (1 - Xn) represents the carrying capacity of the population, i.e., the maximum population size that can be sustained in the given environment.



[Bifurcation Diagram]



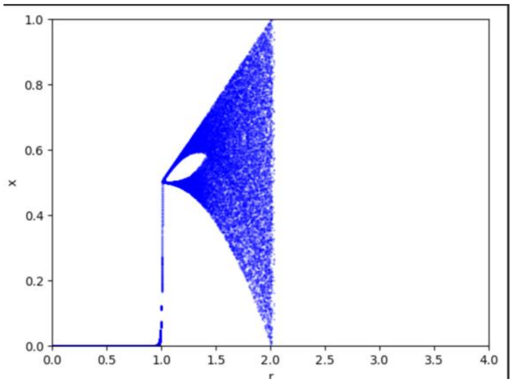
[Lyapunov Exponent]

4.2 Sine Map

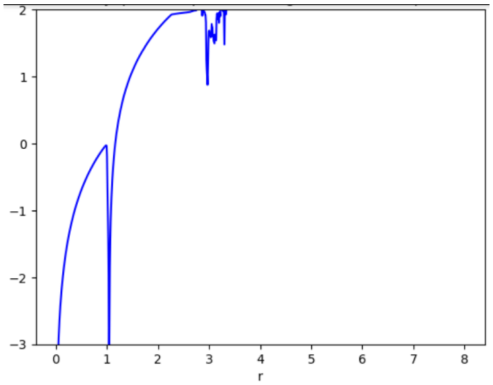
The sine map is a one-dimensional discrete-time dynamical system that is often used in chaos theory. It is defined by the following iterative equation:

$$x\_\{n+1\} = a * \sin(pi * x\_n)$$

where x_n is the value of the map at time n, a is a parameter that determines the behavior of the map, and pi is the mathematical constant pi.



[Bifurcation Diagram]



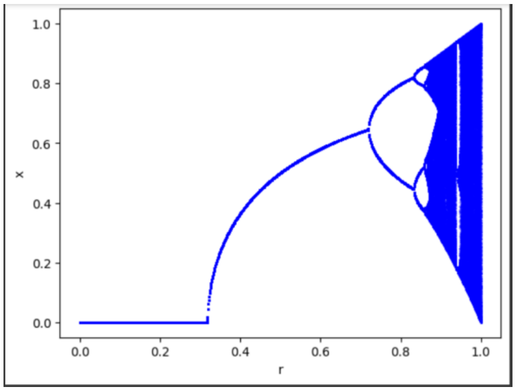
[Lyapunov Exponent]

4.3 Tent Map

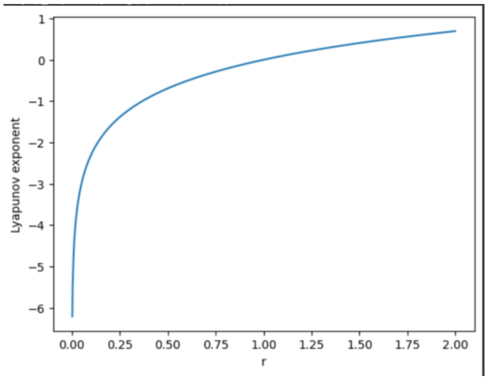
The tent map is a one-dimensional chaotic map that exhibits sensitive dependence on initial conditions, which is a key feature of chaos. It is a piecewise linear map that maps the interval [0, 1] onto itself. The tent map is defined by the following equation:

$$x[n+1] = r (1 - |2x[n] - 1|)$$

where $x[n]$ is the value of the map at time step n, and r is a parameter that controls the behavior of the map.



[Bifurcation Diagram]



[Lyapunov Exponent]

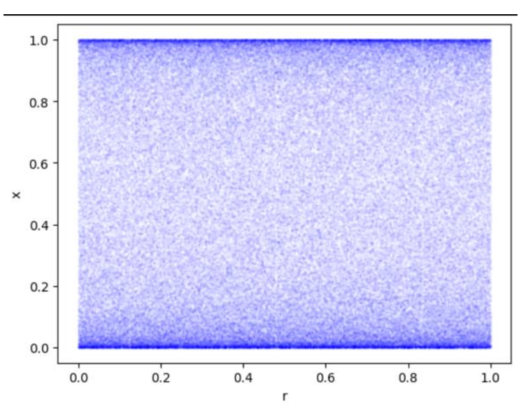
4.4 Logistic Sine Map

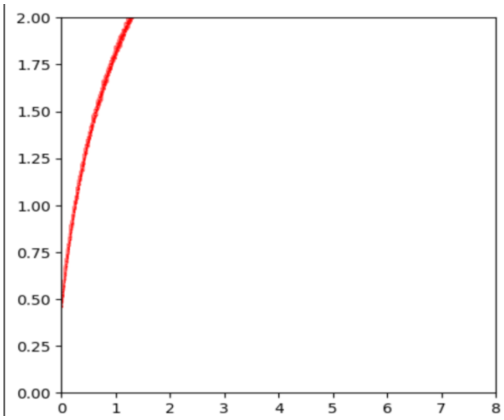
The Logistic-Sine-Cosine (LSC) map is a chaotic map that combines the logistic map with sine and cosine functions. The logistic map is a non-linear, iterative equation that produces chaotic behavior when certain parameters are chosen. The sine and cosine functions are added to the logistic map equation to introduce more complexity and increase the randomness of the output.

The equation for the LSC map is given by:

Xn+1 = (np.cos(math.pi*(4*r*x*(1-x)+(1-r)*np.sin(math.pi*x)-0.5)))

where Xn is the current value of the map, Xn+1 is the next value, and λ is a parameter that controls the amount of chaos. The sine and cosine functions are applied to Xn in a cyclic manner, with cos applied first, followed by sin.





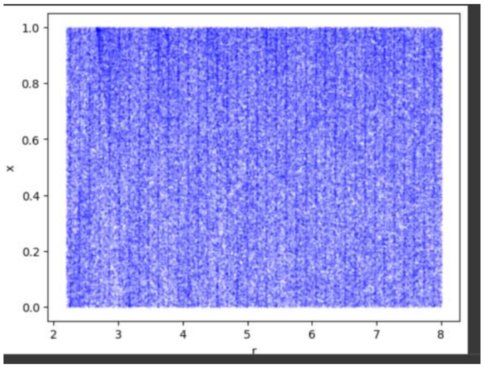
[Bifurcation Diagram] [Lyapunov Exponent]

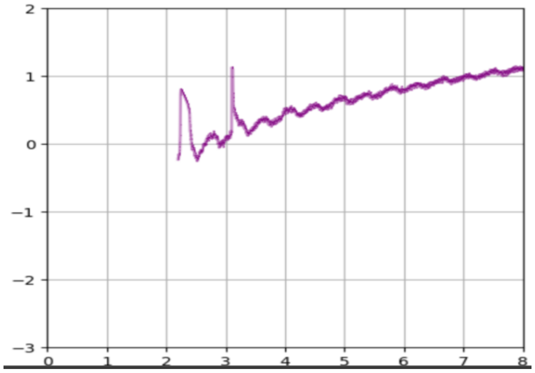
4.5 Sine Cosine Map

The Sine-Cosine map is a type of chaotic map that is widely used in cryptography and other applications that require randomness. It is a one-dimensional map that is defined by the following equation:

x(n+1) = abs(abs(np.sin(-r*x + x**3-r*np.sin(x))) - abs(np.cos(-r*x + x**3-r*np.sin(x))))

where x(n) is the value of the map at the nth iteration, a and b are two constant parameters, and sin and cos are the sine and cosine functions, respectively.





[Bifurcation Diagram] [Lyapunov Exponent]

5. SWOT Analysis

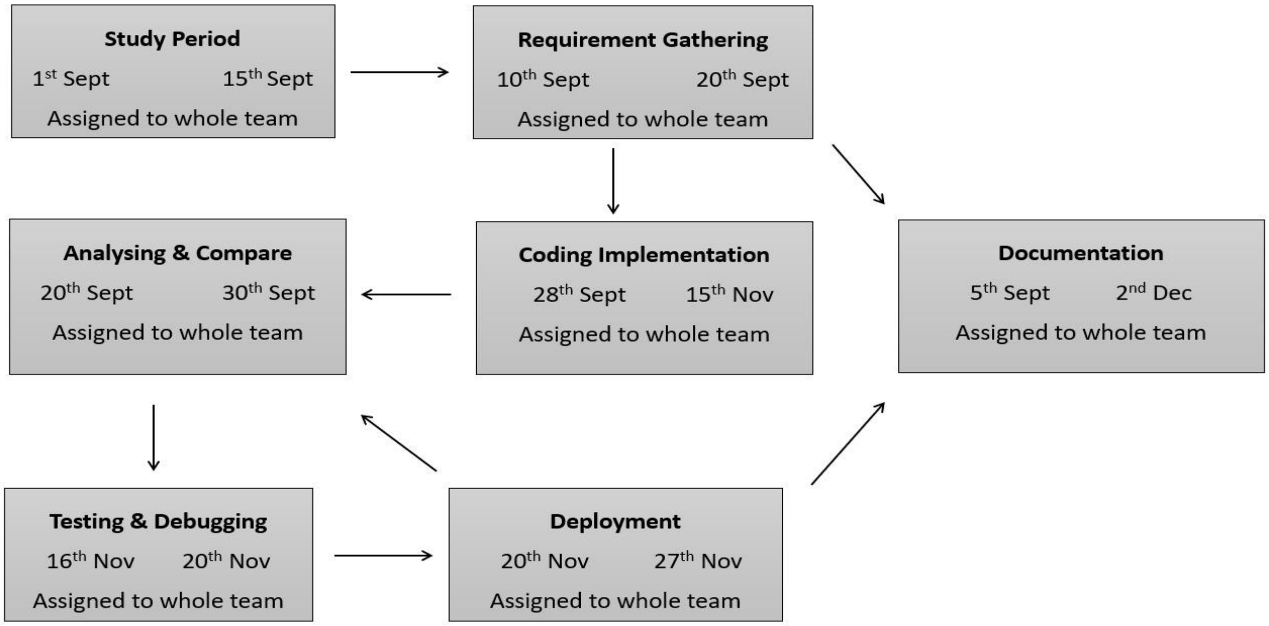
STRENGTHS	WEAKNESS
Innovative approach to encryption: The project presents a new encryption method that combines chaotic GRU maps and sine cosine, which could be a unique and effective approach to secure transport images. Tailored for transport images: The project focuses on the specific needs of transport image encryption, making it potentially more efficient and relevant to the intended application. Performance Evaluation: The inclusion of comprehensive performance analysis demonstrates a commitment to rigorous testing and validation of the proposed encryption method.	Complexity: The use of advanced techniques such as GRU and chaos-based maps can introduce complexity into the encryption process, which could be a potential weakness in terms of implementation and understanding. Resource intensive: These advanced encryption methods can be resource intensive in terms of computing power and memory, which may limit their practicality for certain systems.
OPPORTUNITIES	**THREATS**
Security Enhancement: With the growing need for secure image transmission in the field of transportation and surveillance, the project offers an opportunity to significantly increase security and privacy. Potential Applications: The encryption method developed in this project may have wider applications beyond transport images and opens the door to other areas where secure image encryption is required.	Competing Approaches: The field of image encryption is highly competitive and there may be other encryption methods that are equally or more effective, making it challenging to gain widespread adoption. Security Threats: Any encryption method is vulnerable to attack, and the project may face threats from adversaries attempting to break the encryption and compromise the security of transmission images. Technological Advances: Rapid advances in encryption technologies may render the

	proposed method obsolete or less effective over time.

6. PERT Chart

An effective project management tool for planning and scheduling tasks inside of a project is a PERT (Program Evaluation and Review Technique) chart. It is a graphical representation of the project's timetable that aids in identifying potential delays and the project's essential path.

A PERT chart is made up of nodes and arrows, where each node corresponds to a single task and each arrow shows the connections between those tasks. To depict the order of tasks and their interdependencies, arrows are used to connect the nodes. A duration estimate for each node indicates how long it should take to finish that task.



7. Applications:

1. Secure Image Transmission: Encrypting transport images ensures secure transmission over networks, preventing unauthorized access or tampering during transportation.

2. Privacy Protection in Smart Transportation: Applying encryption to images in smart transportation systems enhances privacy protection for passengers and sensitive information related to transportation services.

3. Traffic Surveillance: Securing images captured by traffic surveillance cameras ensures the confidentiality and integrity of the data, preventing malicious activities such as tampering or unauthorized access.

4. Vehicle-to-Everything (V2X) Communication: Encrypting images exchanged in V2X communication ensures that visual data shared between vehicles, infrastructure, and pedestrians is secure and cannot be compromised.

5. Logistics and Supply Chain Security: Protecting images related to the transportation of goods ensures the confidentiality of sensitive information, preventing theft, sabotage, or unauthorized access.

6. Emergency Response Systems: In situations like accidents or emergencies, encrypted images can be transmitted securely to emergency services, ensuring that critical information remains confidential and accurate.

7. Remote Sensing and Satellite Imaging: Applying encryption to images captured by satellites or remote sensing devices is crucial to protect sensitive data related to geographical and environmental monitoring.

8. Autonomous Vehicles Security: Ensuring the security of images processed by autonomous vehicles is vital for preventing cyber attacks and ensuring the safety of passengers and pedestrians.

9. Military Applications: Transport of sensitive military information via images can benefit from encryption, safeguarding critical data during transmission or storage.

10. Medical Image Transport: In the context of medical transportation, encrypting images ensures patient privacy and the confidentiality of medical data during transit.

11. Drone Surveillance: Images captured by drones for surveillance purposes can be encrypted to prevent unauthorized access and protect sensitive information.

12. Insurance and Claims Processing: Encrypting images related to transportation incidents aids in maintaining the confidentiality and integrity of evidence during insurance claims processing.

8. Reference:

[1] Atul Kumar, Mohit Dua (2022) A GRU and chaos-based novel image encryption approach for transport images. https://doi.org/10.1007/s11042-022-13902-z

[2] Chai X, Chen Y, Broyde L (2017) A novel chaos-based image encryption algorithm using DNA sequence operations. Opt Lasers Eng 88:197–213. https://doi.org/10.1016/j.optlaseng.2016.08.009

[3] Chen J, Zhu Z, Fu C, Yu H (2013) An improved permutation-diffusion type image cipher with a chaotic orbit perturbing mechanism. Opt Express 21(23):27873. https://doi.org/10.1364/oe.21.027873

[4] Dua M, Suthar A, Garg A, Garg V (2020) An ILM-cosine transform-based improved approach to image encryption. Complex Intell Syst 7:1–17. https://doi.org/10.1007/s40747-020-00201-z

[5] Thoms GRW, Muresan R, Al-Dweik A (2019) Chaotic encryption algorithm with key controlled neural networks for intelligent transportation systems. IEEE Access 7:158697–158709