ASSESSING THE RELIABILITY OF DIFFERENT CRYPTOGRAPHIC TECHNIQUES

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

GNANESHWAR.T (Reg. No - 39111038)
UTKARSH ANAND (Reg .No -39111046)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600119

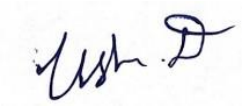
APRIL - 2023

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **GNANESHWAR.T (Reg.No - 39111038)** and **UTKARSH ANAND (Reg.No - 39111046)** who carried out the Project Phase-2 entitled **"ASSESSING THE RELIABILITY OF DIFFERENT CRYPTOGRAPHIC TECHNIQUES"** under my supervision from January 2023 to April 2023.



Internal Guide

Dr. USHA NANDINI M.E, Ph.D

Head of the Department

Dr. L. LAKSHMANAN, M.E, Ph.D.

Submitted for Viva voce Examination held on 20.04.2023

Internal Examiner

External Examiner

DECLARATION

I, **GNANESHWAR.T (Reg.No-39111038)** along with **UTKARSH ANAND(Reg.No-39111046),** hereby declare that the Project Phase-2 Report entitled **"ASSESSING THE RELIABILITY OF DIFFERENT CRYPTOGRAPHIC TECHNIQUES"** done by me under the guidance of **Dr. Usha Nandini, M.E, Ph.D** is submitted in partial fulfillment of the requirements for the award of Bachelor's of Engineering degree in **Computer Science and Engineering**.

Date : 20-04-2023

Place: CHENNAI **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph. D**, **Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.,** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Usha Nandini M.E, Ph.D ,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The security of digital data has become a crucial concern as a result of advances in multimedia technology. Researchers tend to focus their efforts on changing existing protocols to overcome the flaws of present security mechanisms. Several proposed encryption algorithms, however, have been proven insecure during the last few decades, posing serious security risks to sensitive data. Using the most appropriate encryption technique to protect against such assaults is critical, but which algorithm is most suited in any given case will depend on the type of data being protected. However, testing potential cryptosystems one by one to find the best option can take up an important processing time. For a fast and accurate selection of appropriate encryption algorithms, we propose a security level detection approach for image encryption algorithms by incorporating a support vector machine (SVM). In this work, we also create a dataset using standard encryption security parameters, such as entropy, contrast, homogeneity, peak signal to noise ratio, mean square error, energy, and correlation. These parameters are taken as features extracted from different cipher images. Dataset labels are divided into three categories based on their security level: strong, acceptable, and weak. To evaluate the performance of our proposed model, we have calculated accuracy and our results demonstrate the effectiveness of this SVM-supported system.

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	DEFINITION
SVM	Support Vector Machine
ML	Machine Learning
AI	Artificial Intelligence
CSS	Cascading Style Sheets
CLM	Chaotic Logistic Map
HTML	Hyper Text Markup Language
JS	JAVA Script
DDoS	Distributed Denial Of Service
CS	Cyber Security
Encr	Encryption
Decrp	Decryption

CHAPTER 1

INTRODUCTION

Security has become a hot topic of research due to the exponential rise in multimedia data transmissions over insecure channels (most notably the Internet). Many researchers have turned to developing new encryption algorithms to protect data from eavesdroppers and unauthorized users. Two elements are critical when encrypting digital images: diffusion and confusion (also known as scrambling). Claud Shannon suggested a theory in which a cryptosystem with confusion and diffusion mechanisms can be considered secure. The scrambling process on digital images can be done directly on pixels or on rows and columns, whereas diffusion affects the original pixel values. In other words, the replacement procedure substitutes each unique pixel value with the S-unique box's value. The transmission of data in an encrypted format, however, is insufficient to preserve its privacy. Although the information which is to be transmitted is in encrypted form, it can still be visualized by unauthorized users due to the weak security of the encryption algorithm. The security level of the encryption algorithm used to encrypt the image has a significant impact on its robustness. The plain image will be entirely encrypted using a highly strong encryption method, allowing it to withstand attacks on its integrity, secrecy, and availability. Along with security, temporal complexity is another key element to consider when choosing an encryption technology. Because different types of data have different security priorities, choosing a cryptosystem is dependent on the nature of the application to be encrypted. As the image encryption algorithm is very important, we propose a security level detection approach for image encryption algorithms by incorporating a support vector machine (SVM).

CHAPTER 2

LITERATURE SURVEY

Automated detection and classification of cryptographic algorithms in binary programs through machine learning by Diane Duros Hosfelt , 2019

Threats from the internet, particularly malicious software (i.e., malware) often use cryptographic algorithms to disguise their actions and even to take control of a victim's system (as in the case of ransom ware). Malware and other threats proliferate too quickly

for the time-consuming traditional methods of binary analysis to be effective. By automating detection and classification of cryptographic algorithms, we can speed program analysis and more efficiently combat malware. This thesis will offer different ways for automatically discovering and classifying cryptographic algorithms in compiled binary programmes using machine learning. While more research is needed to fully test these methods on real-world binary programmes, the findings in this paper imply that machine learning may be used to detect and identify cryptographic primitives in compiled code with success. These techniques are now being used to discover and categorize cryptographic algorithms in small single-purpose programmes, and more work is being suggested to apply them to real-world situations.

This thesis examined the process of extracting features and training machine learning models for the detection and classification of cryptographic algorithms in compiled code. Three different types of models were evaluated on four different feature sets using four different learning algorithms. While the decision tree models were found to perform the best on these data, due to certain limitations of decision trees, it is possible that an SVM with a linear kernel will generalize better to real-world data. Cross-validation results suggest that algorithm classification and detection will be over 95% accurate, given a relatively small and homogeneous sample.

Applications in Security and Evasions in Machine Learning: A Survey by Ramani Sagar 1*, Rutvij Jhaveri 2 and Carlos Borrego 3, 2017

In recent years, machine learning (ML) has become an important part to yield

security and privacy in various applications. ML is used to address serious issues such as real-time attack detection, data leakage vulnerability assessments and many more. ML extensively supports the demanding requirements of the current scenario of security and privacy across a range of areas such as real-time decision-making, big data processing, reduced cycle time for learning, cost-efficiency and error-free processing. Therefore, in this paper, we review the state of the art approaches where ML is applicable more effectively to fulfill current real-world requirements in security. We examine different security applications' perspectives where ML models play an essential role and compare, with different possible dimensions, their accuracy results. By analyzing ML algorithms in security application it provides a blueprint for an interdisciplinary research area. Even with the use of current sophisticated technology and tools, attackers can evade the ML models by committing adversarial attacks. Therefore, requirements rise to assess the vulnerability in the ML models to cope up with the adversarial attacks at the time of development. Accordingly, as a supplement to this point, we also analyze the different types of adversarial attacks on the ML models. To give proper visualization of security properties, we have represented the threat model and defense strategies against adversarial attack methods. Moreover, we illustrate the adversarial attacks based on the attackers' knowledge about the model and addressed the point of the model at which possible attacks may be committed. Finally, we also investigate different types of properties of the adversarial attacks.

New threats caused by cyber-attacks can damage critical data infrastructure because of machine learning in the security applications highly dependent on the data quality. Using machine learning-based methods in security applications faces a challenge the performance of recognizing an adversarial sample by collecting and predicting adversarial samples. Hence we conclude that the new models are becoming a research point from attacker and designer perspective. With the rapid increase in security events security in machine learning-based decision systems in adversarial environments opens a door for the new research area. In some cases, malicious users can simply increase false-negative rates and minimizing false-positive rates by a proportional amount, cleverly make sure that the overall error rate remains the same and attack is unnoticed which can give attackers some leverage in sophisticated attacks. This kind of issue there needs to be explored to detect attacks efficiently on ML-based systems. Regardless of the data privacy field, great

advancement in existing methods of data privacy suffer from modest performance due to complex operations on a huge number of parameters of machine learning algorithms. Therefore extremely efficient privacy-preserving methods need to be investigated in the adversarial environmental setting. Observation made related performance tradeoff between accuracy and scalability for the machine learning classifiers. For example for any security application designing informal decision made on which approach to use when. But even though having more weak labels does not imply that classifiers' accuracy will eventually reach a precise accuracy. Therefore, it is worth to infuse humans or utilizing transfer learning to make additional changes. This type of decision is made by an experiment, but an important question is whether, overall, there is a need to design and craft secure machine learning algorithms that way which can balance three aspects that are performance overhead, security optimization, and performance generalization.

Machine learning approaches to IoT security: A systematic literature review By Rasheed Ahmada, Izzat Alsmadi, 2017

With the continuous expansion and evolution of IoT applications, attacks on those IoT applications continue to grow rapidly. In this systematic literature review (SLR) paper, our goal is to provide a research asset to researchers on recent research trends in IoT security. As the main driver of our SLR paper, we proposed six research questions related to IoT security and machine learning. This extensive literature survey on the most recent publications in IoT security identified a few key research trends that will drive future research in this field. With the rapid growth of large scale IoT attacks, it is important to develop models that can integrate state of the art techniques and technologies from big data and machine learning. Accuracy and efficiency are key quality factors in finding the best algorithms and models to detect IoT attacks in real or near real-time.

This paper aims to investigate research trends for the applications of machine learning in IoT security. We adopted a systematic approach to evaluating recent studies and future trends in IoT security by extracting the most relevant and scholarly literature published in the last two years (2019 and 2020). Our objectives are driven by the interest to integrate three trending research areas, IoT, machine learning, and information security. The integration of those three domains leads us to extract six

research questions presented in Section 3.1. The extensive but systematic literature review approach presented in this paper shows specific criteria used to filter research papers that did not meet this research goal. It helped us obtain more focused and recent research studies in IoT and the machine learning techniques proposed to prevent large-scale attacks impacting IoT devices.

While many survey studies were performed in IoT security research, they were either not performed systematically or were not focused on machine learning and deep learning-based approaches to detect large-scale attacks. Section 5 presents the details of existing IoT security literature surveys. This research study primarily focused on securing IoT devices from distributed and wide-scale attacks such as a distributed denial of service (DDoS), botnet, etc. Compare to traditional intrusion detection techniques such as firewalls, antiviruses, etc., machine learning, and deep learning approaches provide promising results to detect zero-day threats. There has been extensive research performed in this field; however, the fast-evolving nature of IoT device types, network traffic patterns, and cyber-attacks make it challenging for intrusion detection systems to detect zero-day attacks. We presented a detailed background section to equip the reader with relevant knowledge to navigate further into the paper efficiently. The careful analysis of selected papers in section 6 extracts researchers proposed methodologies and their related performance results on various benchmark datasets. The review results presented in section 7 answers each of the six research questions in detail. Our goal is to provide researchers a more focused yet detailed knowledge and prevailing methodologies adopted recently to find recent trends, limitations, and challenges to help build effective intrusion detection systems in the future.

Secure, privacy-preserving and federated machine learning in medical imaging By Georgios A. Kaissis, Marcus R. Makowski, Daniel Rückert & Rickmer F. Braren ,2017

The broad application of artificial intelligence techniques in medicine is currently hindered by limited dataset availability for algorithm training and validation, due to the absence of standardized electronic medical records, and strict legal and ethical requirements to protect patient privacy. In medical imaging, harmonized data exchange formats such as Digital Imaging and Communication in Medicine and

electronic data storage are the standard, partially addressing the first issue, but the requirements for privacy preservation are equally strict. To prevent patient privacy compromise while promoting scientific research on large datasets that aims to improve patient care, the implementation of technical solutions to simultaneously address the demands for data protection and utilization is mandatory. Here we present an overview of current and next-generation methods for federated, secure and privacy-preserving artificial intelligence with a focus on medical imaging applications, alongside potential attack vectors and future prospects in medical imaging and beyond.

Artificial intelligence (AI) methods have the potential to revolutionize the domain of medicine, as witnessed, for example, in medical imaging, where the application of computer vision techniques, traditional machine learning and—more recently—deep neural networks have achieved remarkable successes. This progress can be ascribed to the release of large, curated corpora of images (Image Net perhaps being the best known), giving rise to performant pre-trained algorithms that facilitate transfer learning and led to increasing publications both in oncology—with applications in tumour detection, genomic characterization, tumour subtyping, grading prediction, outcome risk assessment or risk of relapse quantification—and non-oncologic applications, such as chest X-ray analysis and retinal fundus imaging.

Machine Learning and Cryptographic Algorithms –Analysis and Design in Ransomware and Vulnerabilities Detection , 2016.

The AI, deep learning and machine learning algorithms are gaining the ground in every application domain of information technology including information security. In formation security domain knows for traditional password management systems, auto provisioning systems and user information management systems. There is another raising concern on the application and system level security with ransomware. On the existing systems cyber-attacks of Ransom ware asking for ransom increasing every day. Ransomware is the class of malware where the goal is to gain the data through encryption mechanism and render back with the ransom. The ransomware attacks are mainly on the vulnerable systems which are exposed to the network with weak security measures. With the help of machine learning algorithms, the pattern of the attacks can be analysed. Create or discuss a

workaround solution of a machine learning model with combination of cryptographic algorithm which will enhance the effectiveness of the system response to the possible attacks. The other part of the problem, which is hard part to create intelligence for the organizations for preventing the ransomware attacks with the help of intelligent system password management and intelligent account provisioning. In this paper I elaborate on the machine learning algorithms analysis for the intelligent ransomware detection problem, later part of this paper would be design of the algorithm.

In this paper, proposed machine learning algorithm can be modeled for the novel ransomware detection and the random number decryption techniques for the new model to break the encryption for saving the ransom. The study also suggested for the classification of the infectious files can be differentiated by the machine based on the model it has trained. The main problem has structurally divided into sub problems of the identification of the ransomware problems and the design the cryptographic algorithms based on the machine learning to generate the decryption key for the ransom problem. 8 19 April 2020 Machine Learning and Cryptographic Algorithms – Analysis and Design in Ransomware and Vulnerabilities Detection My ongoing work will be a round producing the results and accuracy of the algorithm which I am working on for my research.

Prediction of DDoS Attacksusing Machine Learning and Deep Learning Algorithms by Saritha, B. RamaSubba Reddy, A Suresh Babu, 2016.

With the emergence of network-based computing technologies like Cloud Computing, Fog Computing and IoT (Internet of Things), the context of digitizing the confidential data over the network is being adopted by various organizations where the security of that sensitive data is considered as a major concern. Over a decade there is a massive growth in the usage of internet along with the technological advancements that demand the need for the development of efficient security algorithms that could withstand various patterns of the security breaches. The DDoS attack is the most significant network-based attack in the domain of computer security that disrupts the internet traffic of the target server. This study mainly focuses to identify the advancements and research gaps in the development of efficient security algorithms addressing DDoS attacks in various ubiquitous network

environments. Keywords: DDoS attack, machine learning, Deep learning, Volumetric attacks, protocol attacks.

Now a day'swith the advent of 4G, 5G networks and economic smart devices there is a massive growth in the usage of the internet that has become a part of daily life. A vast range of services provided over the internet in diverse application areas such as business, entertainment, and education, etc. made it a vital component in framing various business models. This context made security over wireless networks as the most important factor while using the internet from unsecured connections[1]. Different security algorithms and frameworks are developed to enable protection from Internet-based attacks while devising high performance IDS (Intrusion detection systems) which act as a defensive wall while confronting the attacks over internet based devices. Distributed architecture-based computing environments like cloud computing and IoT are more prone towards DDoS attacks in which multiple devices are coordinated to launch attacks over distributed targets. DDOS attacks are primarily launched in the context of exhausting the connectivity and the processing of the target server resources in which it enables the access constraints to the legitimate users to utilize the services provided by the target server that leads towards the partial unavailability or total unavailability of the services. The phenomenon of distributed computing is based on the one-to-many dimension in which these types of attacks may cause a possible amount of damage to the server resources [3]. It is observed from the previous research studies that the damage capacity, as well as the disrupting nature of the DDoS attacks, is gradually increased with the rate of internet usage.

The article includes the systematic study of literature in the context of detecting and predicting DDoS attacks on utilizing machine learning and deep learning algorithms. In this study, after the thorough filtering process, 34 articles are considered for the study through which it is observed that most of the existing research includes solutions and algorithmic patterns that are framed based on the statistical algorithms such that most these algorithms suffer from computational complexity. Additionally, there are very few publications addressing the scope of predicting the DDoS. This study outlines the synthesis of various DDoS attacks and their countermeasure algorithms that enables the researchers of the next generation to easily identify the research gap in the context of applying machine learning algorithms to automate the process of predicting DDoS attacks in various distributed networks.

2.1 INFERENCES FROM LITERATURE SURVEY

Various encryption algorithms have been proposed to secure images during transmission. These algorithms can be based on chaos or metamorphosis styles, such as separate sea metamorphosis, separate cosine metamorphosis, and separate Fourier metamorphosis. However, Numerous encryption techniques have been proposed to secure images during transmission, including chaos-based and metamorphosis-based algorithms. However, limitations such as finite computer precision can affect the security of chaos-based encryption. To address this, multiple chaotic systems have been proposed to enhance the security of encryption techniques. Additionally, the development of strong substitution boxes (S-boxes) is critical for ensuring the strength of chaos-based encryption algorithms. Recently, a new image encryption technique based on a chaotic logistic map (CLM) was proposed to address the limitations of single S-box encryption. Selective encryption schemes, while useful for fast encryption of images, may not be suitable for text encryption where every bit of data must be translated. Thus, the development of robust S-boxes remains an important area of research for security experts .While chaos has the ability to induce arbitrary numbers, it has limitations, such as finite computer precision, which can lead to dynamic decline and insecurity in chaos-based encryption. To overcome these limitations, multiple chaotic systems were proposed to enhance the security of the encryption technique. Moreover, a new image encryption technique based on a chaotic logistic map (CLM) was proposed to address the issues with using a single substitution box (S-box) encryption.

In order to address the weaknesses of using weak S-boxes in image encryption, we proposed a new methodology based on the chaotic logistic map (CLM) that can generate a new, stronger S-box with slight variations in the original CLM values (as described in reference 15). Encrypting a color image poses more difficulty compared to a grayscale image as it requires encryption of all three color channels, namely R, G, and B. In a research paper cited as reference , the authors proposed a color image encryption method that employs a hybrid chaotic system. The encryption process involves independent encryption of each (R, G, B) element using confusion, and diffusion of the encrypted data using a mitochondrial DNA sequence. Each

encryption algorithm has a different level of security, with some being weak, some being respectable, and others being strong. The level of security of each algorithm depends on the complexity of its fine structure.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

Existing System:

In the existing system, acquiring perfectly balanced and highly related dataset is almost impossible. Although large quantities of data are available but still extracting relevant data is a complex job. To overcome all this, we use machine learning packages available in the scikit-learn library to extract useful data.

Disadvantages:

- High complexity:

  As the size of the data grows, it can become challenging to train and deploy machine learning models at scale. This requires specialized infrastructure and tools to manage the data and training process.

- Time consuming:

  Machine learning models can be highly complex and difficult to interpret, which can make it challenging to understand how they arrived at their predictions. This is especially important in applications where human decision-making is involved, such as in healthcare or finance.

- Security :

  Machine learning models can be vulnerable to attacks, such as adversarial attacks or data poisoning attacks. It is important to ensure the models are secure and protected against these types of attacks.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are:

1. ECONOMICAL FEASIBILITY
2. TECHNICAL FEASIBILITY
3. SOCIAL FEASIBILITY

3.1.1 Economic feasibility:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.1.2 Technical feasibility:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.1.3 Social feasibility:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION

Operating system	: Windows 7 or above
Ram	: 8 GB
Hard disc or SSD	: More than 500 GB
Processor	: Intel 3rd generation or high or Ryzen with 8 GB Ram
Software’s	: Python 3.6 or high version, Visual studio, PyCharm.

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

A slew of encryption algorithms, including chaos and transformation-based algorithms, have been presented in recent years. By examining the statistical findings of existing encryption algorithms, it has been discovered that some of them are unsecure and do not provide adequate protection. Analysing the statistics of an encryption algorithm's security parameters is one technique to determine its security level. Traditional methods of accomplishing this usually include making these comparisons one by one, which takes a long time. We created a machine learning model that combines SVM to help us choose a suitable encryption technique more rapidly.

Advantages:

- It increases the accuracy.
- It reduces the time complexity.
- It automates the process of detecting the security levels of encryption algorithm.

4.1 SELECTED METHODOLOGY

4.1.1 Support-vector machine

Support-vector machines are supervised learning models using learning algorithms that evaluate data for classification and regression analysis in machine learning. SVMs, which are based on statistical learning frameworks, are one of the most reliable prediction approaches. An SVM training algorithm creates a model that assigns new examples to one of two categories, making it a non-probabilistic binary linear classifier, given a series of training examples, each marked as belonging to one of two categories. SVM maps training examples to points in space in order to widen the distance between the two categories as much as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. A support-vector machine, in more technical terms, creates a hyper plane or set of hyper planes in a high- or infinite-dimensional space that can be used for classification, regression, or other tasks such as outlier detection. Intuitively, the hyper plane with the greatest distance to the nearest training-data point of any class (so-called functional margin) achieves a decent separation, because the larger the margin, the lower the classifier's generalization error. Even if the initial problem is expressed in a finite-dimensional space, the sets to discriminate are frequently not linearly separable in that space. As a result, it was suggested that the original finite-dimensional space be transferred into a much higher-dimensional region, purportedly making separation easier there. The mappings used by SVM schemes are designed to ensure that dot products of pairs of input data vectors can be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function.

Support Vector Machines (SVMs) are a type of supervised machine learning algorithm that are commonly used for classification and regression analysis. SVMs are particularly useful when working with large datasets with high-dimensional feature spaces. In this article, we will discuss the basic concepts of SVM, their mathematical foundations, and how to use them for classification tasks.

Basic Concepts of SVM:

In SVM, the objective is to find the hyperplane that best separates the two classes of data. The hyperplane is a multidimensional decision boundary that divides the data into different classes. The objective of SVM is to find the hyperplane that maximizes the margin between the two classes of data. The margin is defined as the distance between the hyperplane and the nearest data point from each class. The hyperplane that maximizes the margin is called the optimal hyperplane.

In addition to the linear SVM, there are also non-linear SVMs that can handle non-linearly separable data. Non-linear SVMs use a kernel function to transform the input space into a higher-dimensional space where the data is linearly separable. The kernel function can be chosen based on the characteristics of the data.

Mathematical Foundations of SVM:

The SVM algorithm is based on the idea of maximizing the margin between the two classes of data. To find the optimal hyperplane, the SVM algorithm minimizes the hinge loss function, which is a convex function that penalizes the misclassification of data points.

The SVM algorithm can be formulated as an optimization problem with constraints. The objective is to minimize the hinge loss function subject to the constraint that the data points are correctly classified by the hyperplane. The optimization problem can be solved using quadratic programming.

Once the optimal hyperplane is found, new data points can be classified based on their position relative to the hyperplane. If a new data point falls on one side of the hyperplane, it is classified as belonging to one class, and if it falls on the other side, it is classified as belonging to the other class.

Using SVM for Classification:

SVM can be used for both binary and multi-class classification. In binary classification, the objective is to classify data points into two classes. In multi-class classification, the objective is to classify data points into multiple classes.

To use SVM for classification, the first step is to split the data into training and testing datasets. The training dataset is used to train the SVM model, while the testing dataset is used to evaluate the performance of the model.

The next step is to preprocess the data by scaling and normalizing the features. SVM is sensitive to the scale of the features, so it is important to ensure that the features are on a similar scale.

After preprocessing the data, the SVM model can be trained using the training dataset. The optimal hyperplane can be found by solving the optimization problem with constraints using quadratic programming.

Finally, the performance of the SVM model can be evaluated using the testing dataset. The accuracy of the model can be calculated by comparing the predicted labels to the true labels in the testing dataset. Other metrics such as precision, recall, and F1 score can also be used to evaluate the performance of the model.
Advantages of SVM:

Effective in high-dimensional spaces: SVM is effective in high-dimensional spaces where the number of features is larger than the number of data points. SVM can handle a large number of features, making it suitable for problems with a large number of variables.

Memory efficient: SVM uses a subset of training points in the decision function, which makes it memory efficient. This is especially useful for large datasets with a large number of features.

Versatile: SVM can be used for both binary and multi-class classification, as well as for regression analysis. It can handle both linearly and non-linearly separable data.

Robust: SVM is robust to noise and can handle outliers well. It is not affected by small changes in the training data.

Effective with small datasets: SVM is effective with small datasets where the number of data points is smaller than the number of features.

Disadvantages:

Sensitivity to hyperparameters: SVM has several hyperparameters that need to be tuned to achieve the best performance. The choice of hyperparameters can greatly affect the performance of the model.

Computationally intensive: SVM can be computationally intensive, especially for large datasets. The training time of SVM can be long, especially for non-linearly separable data.

Limited to binary classification: SVM is limited to binary classification, and needs to be extended to handle multi-class classification problems.

Difficulty with interpretation: SVM is a black box model, which means it is difficult to interpret the results and understand how the model is making predictions.

Vulnerability to overfitting: SVM is vulnerable to overfitting, especially for non-linearly separable data. Careful selection of the kernel function and regularization parameters is required to prevent overfitting.

4.1.2 DNA encoding

DNA computing is a form of computing which uses DNA, biochemistry and molecular biology, instead of the traditional silicon-based computer technologies. DNA computing, or more generally, bimolecular computing, is a fast developing interdisciplinary area. With the rapid development of DNA computing, the researchers presented many biological operations and algebra operations based on DNA sequence. Single-strand DNA sequence is composed by four bases, they are A, C, G and T, where A and T are complement to each other, so are C and G. In the modern theory of electronic computer, all information is expressed by binary system. But in DNA coding theory, information is represented by DNA sequences. So we use binary numbers to express the four bases in DNA sequence and two bits binary number to represent a base. In the theory of binary system, 0 and 1 are complementary, so we

can obtain that 00 and 11, 01 and 10 are also complementary. We can use 00, 01, 10 and 11 to express four bases and the number of coding combination kinds is 4! = 24. Due to the complementary relation between DNA bases, there are only eight kinds of coding combinations that satisfy the principle of complementary base pairing in 24 kinds of coding combinations.

DNA encoding, also known as DNA computing, is a type of computation that uses DNA molecules to store and process information. DNA is a molecule that contains genetic instructions for the development and function of living organisms. These instructions are encoded in the sequence of the four nucleotide bases: adenine (A), thymine (T), guanine (G), and cytosine (C).

In DNA encoding, the properties of DNA, such as its ability to self-assemble and its chemical stability, are leveraged to perform computations. The basic idea is to use DNA strands to represent data and the reactions between these strands to perform computations. Each DNA strand is composed of a sequence of nucleotides that can be thought of as bits of data.

One of the key advantages of DNA encoding is its ability to perform massive parallelism. DNA molecules can be replicated easily, and millions of copies of a DNA molecule can be produced in a short amount of time. This allows for many computations to be performed simultaneously, making it useful for solving complex problems that would be difficult to solve using traditional computers.

Another advantage of DNA encoding is its high storage capacity. DNA can store a large amount of information in a small space. For example, the entire human genome can be stored on a single DVD. This makes DNA encoding useful for data storage and retrieval applications.

To perform computations using DNA encoding, a set of DNA strands are prepared, each representing a piece of data. The strands are then combined in a reaction mixture with other DNA molecules that serve as input to the computation. The reaction mixture is then subjected to a series of chemical reactions that result in the desired output. The output is then read out using techniques such as gel electrophoresis or DNA sequencing.

One of the challenges of DNA encoding is the need to design DNA strands that are specific to the problem being solved. This requires a deep understanding of the problem and the ability to design DNA strands that can interact with each other in a specific way. In addition, the reactions involved in DNA encoding can be sensitive to environmental factors such as temperature and pH, which can affect the accuracy of the computation.

Despite these challenges, DNA encoding has been used to solve a variety of problems, including optimization, graph problems, and cryptography. For example, researchers have used DNA encoding to solve the Hamiltonian path problem, which involves finding a path that visits every node in a graph exactly once. They have also used DNA encoding to encrypt messages, taking advantage of the difficulty of decoding DNA sequences.

In addition to its potential applications in computation, DNA encoding has also been used in other areas of research, such as DNA nanotechnology and synthetic biology. For example, researchers have used DNA encoding to build complex DNA-based structures, such as nanorobots and DNA origami.

In conclusion, DNA encoding is a promising approach to computation that uses DNA molecules to store and process information. It has the potential to revolutionize computing by offering massive parallelism, high storage capacity, energy efficiency, and robustness. While there are still challenges to be overcome, DNA encoding has already been used to solve a variety of problems and has potential applications in many areas of research and industry.
Advantages:

Massive parallelism: DNA encoding is inherently massively parallel, allowing for many computations to be performed simultaneously. This makes it useful for solving complex problems that would be difficult to solve using traditional computers.

High storage capacity: DNA can store a large amount of information in a small space, making it useful for data storage and retrieval applications.

Energy efficiency: DNA computing is energy efficient because it relies on natural biochemical processes, such as self-assembly, rather than the electricity-based processes used by traditional computers.

Robustness: DNA is a stable molecule that can withstand harsh environments, making it a good choice for applications in extreme conditions, such as space.

Security: DNA encoding is difficult to decode, making it useful for security applications such as encryption.

Disadvantages:

Complexity: DNA encoding is a complex process that requires specialized knowledge and equipment, making it less accessible than traditional computing.

Limited accuracy: DNA encoding is prone to errors due to mutations and other factors, which can reduce the accuracy of computations.

Limited scalability: Although DNA encoding is highly parallel, it can be difficult to scale up to larger problems due to the complexity of handling and manipulating large amounts of DNA.

Limited flexibility: DNA encoding is highly specific to the problem it is designed to solve, and it may be difficult to adapt it to other problems or applications.

Cost: DNA encoding can be expensive, both in terms of equipment and reagents needed, and in terms of the specialized expertise required to perform the computations.

4.1.3 Logistic Map

The logistic map is a polynomial mapping (equivalently, recurrence relation) of degree 2, often cited as an archetypal example of how complex, chaotic behaviour can arise

from very simple non-linear dynamical equations. The map was popularized in a 1976 paper by the biologist Robert May in part as a discrete-time demographic model analogous to the logistic equation written down by Pierre François Verhulst. This nonlinear difference equation is intended to capture two effects: reproduction where the population will increase at a rate proportional to the current population when the population size is small. starvation (density-dependent mortality) where the growth rate will decrease at a rate proportional to the value obtained by taking the theoretical "carrying capacity" of the environment less the current population. However, as a demographic model the logistic map has the pathological problem that some initial conditions and parameter values (for example, if r > 4) lead to negative population sizes. This problem does not appear in the older Ricker model, which also exhibits chaotic dynamics.

The logistic map is a mathematical model that is used to study the dynamics of population growth. It was introduced by the biologist Robert May in 1976 and has since become an important tool for understanding the behavior of complex systems in various fields, including physics, ecology, and economics.

The logistic map is a discrete-time dynamical system that describes the population growth of a species that has limited resources. The population size at any given time is represented by a variable x, which takes on values between 0 and 1. The logistic map is defined by the following recurrence relation:

$$x_{n+1} = r * x_n * (1 - x_n)$$

where x_n is the population size at time n, x_{n+1} is the population size at time n+1, and r is a parameter that controls the rate of population growth. The logistic map can be thought of as a simple model of competition between individuals for limited resources, where r represents the efficiency with which resources are utilized.
The behavior of the logistic map is determined by the value of the parameter r. For small values of r, the population size approaches a stable equilibrium point, where the growth rate of the population is zero. As r is increased, the population undergoes a bifurcation, where the equilibrium point becomes unstable and the population exhibits periodic behavior. As r is further increased, the period of the population oscillations

doubles, and the system exhibits chaotic behavior.

The logistic map is an example of a nonlinear dynamical system, which means that its behavior cannot be predicted using linear techniques. It is highly sensitive to initial conditions, which means that small changes in the initial population size can lead to large differences in the long-term behavior of the system.

The logistic map has many important applications in science and engineering. For example, it has been used to model the spread of infectious diseases, the dynamics of predator-prey relationships, and the behavior of financial markets. It is also used in cryptography, where it is used to generate random numbers that are difficult to predict.

One of the main advantages of the logistic map is its simplicity. It is a simple, one-dimensional model that can be easily analyzed mathematically and simulated on a computer. This makes it a useful tool for studying the dynamics of complex systems, where more sophisticated models may be too difficult to analyze.

However, the simplicity of the logistic map is also one of its disadvantages. Its assumptions may not always be applicable to real-world situations, and its predictions may be too simplistic to capture the full complexity of a system. In addition, the logistic map is highly sensitive to initial conditions, which can make it difficult to predict the long-term behavior of a system with any degree of certainty.

Despite these limitations, the logistic map remains a useful tool for studying the dynamics of complex systems. It has led to many important insights into the behavior of populations, and it continues to be a subject of active research in many fields. By exploring the properties of the logistic map and its extensions, researchers are gaining a deeper understanding of the dynamics of complex systems and developing new tools for analyzing and predicting their behavior.
The logistic map is a simple but powerful tool for studying the dynamics of complex systems. It has many advantages, as well as some disadvantages, which are discussed below:

Advantages:

Simplicity: The logistic map is a simple, one-dimensional model that can be easily analyzed mathematically and simulated on a computer. This makes it a useful tool for studying the dynamics of complex systems, where more sophisticated models may be too difficult to analyze.

Versatility: The logistic map can be used to model a wide range of phenomena, from population growth to the spread of infectious diseases, predator-prey relationships, and the behavior of financial markets.

Chaotic behavior: The logistic map exhibits chaotic behavior, which means that small changes in initial conditions can lead to large differences in the long-term behavior of the system. This makes it a useful tool for studying complex systems that exhibit unpredictable behavior.

Bifurcation: The logistic map undergoes a bifurcation as the parameter r is increased, where the equilibrium point becomes unstable and the population exhibits periodic behavior. This is a useful property for understanding the transitions between different types of behavior in complex systems.

Disadvantages:

Oversimplification: The logistic map makes certain assumptions about the dynamics of complex systems that may not always be applicable to real-world situations. Its predictions may be too simplistic to capture the full complexity of a system.

Limited applicability: The logistic map is a one-dimensional model that may not be suitable for studying complex systems that require higher-dimensional models.

Sensitivity to initial conditions: The logistic map is highly sensitive to initial conditions, which can make it difficult to predict the long-term behavior of a system with any degree of certainty.

Lack of empirical data: The logistic map is a theoretical model that relies on mathematical assumptions rather than empirical data. This can limit its usefulness for studying real-world phenomena.

In conclusion, the logistic map is a useful tool for studying the dynamics of complex systems, but it has its limitations. Its simplicity and versatility make it a valuable tool for modeling a wide range of phenomena, but its assumptions may not always be

applicable to real-world situations. Its sensitivity to initial conditions and lack of empirical data may also limit its usefulness for studying real-world phenomena.

4.1.4 Rubik's Cube Image Encryption

This algorithm is based on the principle of Rubik's cube to permute image pixels. To confuse the relationship between original and encrypted images, the XOR operator is applied to odd rows and columns of image using a key. The same key is flipped and applied to even rows and columns of image. Experimental tests have been carried out with detailed numerical analysis which demonstrates the robustness of the proposed algorithm against several types of attacks such as statistical and differential attacks (visual testing). Moreover, performance assessment tests demonstrate that the proposed image encryption algorithm is highly secure. It is also capable of fast encryption/decryption which is suitable for real-time Internet encryption and transmission applications.

Rubik's Cube image encryption is a novel approach to image encryption that utilizes the Rubik's Cube puzzle as a tool for encrypting and decrypting images. The encryption process involves shuffling the colors of the Rubik's Cube according to a secret key and using the resulting cube as a permutation matrix to scramble the pixels of the input image. The decryption process involves reversing the permutation by solving the Rubik's Cube puzzle using the same secret key. This approach has several advantages over traditional encryption methods, including the ability to resist brute-force attacks and the ability to encrypt large amounts of data using a small key.

The Rubik's Cube is a 3D puzzle consisting of six faces, each with nine colored stickers. The colors can be shuffled in various ways to create a large number of permutations, which makes the Rubik's Cube an ideal tool for generating permutation matrices for image encryption. The Rubik's Cube is also easy to manipulate and can be solved using a few simple algorithms, which makes it easy to use for encryption and decryption.

The Rubik's Cube image encryption process involves the following steps:

Key Generation: A secret key is generated by shuffling the colors of the Rubik's Cube

in a random order. The key is kept secret and is only known to the sender and receiver.

Image Preparation: The input image is converted into a matrix of pixels, where each pixel is represented by a set of three color values (red, green, and blue). The matrix is then reshaped into a vector for ease of manipulation.
Pixel Shuffling: The Rubik's Cube is used as a permutation matrix to scramble the pixels of the input image. Each face of the cube corresponds to a row or column of the permutation matrix, and each sticker on the face corresponds to a position within that row or column. The colors on each sticker determine the values of the corresponding row or column of the permutation matrix. The cube is manipulated according to the secret key, and the resulting permutation matrix is applied to the pixel vector to scramble the pixels.

Encryption: The scrambled pixel vector is then encrypted using a traditional encryption algorithm, such as the Advanced Encryption Standard (AES), to add an additional layer of security.

The decryption process involves reversing the steps of the encryption process:

Key Generation: The secret key is used to generate the same Rubik's Cube that was used for encryption.

Decryption: The encrypted pixel vector is decrypted using the same traditional encryption algorithm used for encryption.

Pixel Unshuffling: The resulting pixel vector is unscrambled by applying the inverse permutation matrix to the vector. This is accomplished by solving the Rubik's Cube puzzle using the same secret key used for encryption.

Image Reconstruction: The unscrambled pixel vector is reshaped into a matrix and converted back into an image using the original color values.

Rubik's Cube image encryption has several advantages over traditional encryption

methods. First, the use of the Rubik's Cube as a permutation matrix makes it resistant to brute-force attacks, as there are a large number of possible permutations that can be generated using the cube. Second, the encryption process is fast and efficient, as the Rubik's Cube can be manipulated quickly and easily. Third, the size of the key required for encryption is relatively small compared to the size of the image being encrypted. This makes it possible to encrypt large amounts of data using a small key, which is important for applications where bandwidth and storage space are limited.

There are also some limitations to Rubik's Cube image encryption. The use of a traditional encryption algorithm adds an additional layer of security, but it also increases the computational complexity of the encryption and decryption processes. Additionally, the security of the encryption depends on the secrecy of the key.
Rubik's Cube image encryption has several advantages and disadvantages, which are discussed below.

Advantages:

Resistance to Brute-Force Attacks: The use of the Rubik's Cube as a permutation matrix makes the encryption process resistant to brute-force attacks. There are a large number of possible permutations that can be generated using the Rubik's Cube, which makes it difficult for an attacker to guess the correct key.

Fast and Efficient: The encryption and decryption process is fast and efficient, as the Rubik's Cube can be manipulated quickly and easily. This makes it suitable for real-time encryption and decryption applications.

Small Key Size: The size of the key required for encryption is relatively small compared to the size of the image being encrypted. This makes it possible to encrypt large amounts of data using a small key, which is important for applications where bandwidth and storage space are limited.

Easy Implementation: Rubik's Cube image encryption is easy to implement, as it only requires basic knowledge of the Rubik's Cube puzzle and a traditional encryption

algorithm such as AES.

Disadvantages:

Key Management: The security of Rubik's Cube image encryption depends on the secrecy of the key. If the key is compromised, then the encrypted data can be easily decrypted. Therefore, key management is a critical aspect of the encryption process.

Limited Complexity: The complexity of the encryption process is limited by the number of colors on the Rubik's Cube. If a Rubik's Cube with more colors is used, then the encryption process becomes more complex but also less practical.

Vulnerability to Known Plaintext Attacks: If an attacker has access to both the plaintext and the corresponding encrypted image, then they can use this information to determine the key and decrypt other images encrypted using the same key. This vulnerability is known as a known plaintext attack.

High Computational Complexity: While the encryption and decryption processes are fast and efficient, the use of a traditional encryption algorithm such as AES adds an additional layer of security at the cost of increased computational complexity. This may be a disadvantage in applications where speed is critical.

In conclusion, Rubik's Cube image encryption is a novel approach to image encryption that has both advantages and disadvantages. It is resistant to brute-force attacks, fast and efficient, has a small key size, and is easy to implement. However, it also has limitations such as key management, limited complexity, vulnerability to known plaintext attacks, and high computational complexity when combined with a traditional encryption algorithm.

4.1.5 Lorenz Image Encryption

The Lorenz equation is nothing else than a model of thermally induced fluid convection in the atmosphere. The model was first reported and published by E.N Lorenz in 1963. It is among the classical chaotic systems and implies as the cause of

the “butterfly effect” in the scientific studies due to the fact that the attractor has two wings as the butterflies . Therefore, it has been widely studied in chaos theory, dynamic system modeling, chaotic control and synchronization phenomenon. Lorenz chaotic equation is a 3D dynamical system, which is defined by x, y and z. The equation system gives a chaotic behavior with regard to the initial system parameters. Apart from any 1D or 2D chaotic systems, the Lorenz system has a much complicated chaotic behaviour. By using Lorenz equation we encrypt the images.

Lorenz image encryption is a cryptographic technique that uses the Lorenz chaotic system to generate a key for encrypting an image. The Lorenz chaotic system is a dynamic system that exhibits sensitivity to initial conditions, which means that small changes in the initial conditions can result in large changes in the output. This property of chaos makes the Lorenz system suitable for generating pseudo-random sequences that can be used as encryption keys.

The encryption process using the Lorenz chaotic system can be divided into three steps: key generation, image permutation, and image diffusion. These steps are discussed in detail below.

Key Generation:

The key generation process involves generating a sequence of pseudo-random numbers using the Lorenz chaotic system. The Lorenz system is a set of three coupled differential equations that describe the motion of a fluid in a three-dimensional space. The three variables in the Lorenz system are x, y, and z, and they are updated using the following equations:

$dx/dt = \sigma(y - x)$

$dy/dt = x(\rho - z) - y$

$dz/dt = xy - \beta z$

where σ, ρ, and β are system parameters that control the behavior of the system.

To generate a key for encrypting an image, the Lorenz system is initialized with a set of initial conditions (x0, y0, z0), which are chosen randomly. The system is then iterated a large number of times to generate a sequence of values that can be used

as the encryption key. The sequence of values generated by the Lorenz system is highly sensitive to the initial conditions and the system parameters, which makes it suitable for generating pseudo-random sequences that can be used as encryption keys.

Image Permutation:
After generating the encryption key, the next step is to permute the pixels of the image using the key. The permutation process involves rearranging the pixels of the image in a random order using the key generated by the Lorenz system. The permutation process makes it difficult for an attacker to guess the correct order of the pixels, which increases the security of the encryption process.

Image Diffusion:
The final step in the encryption process is image diffusion, which involves spreading the information of each pixel to multiple pixels in the image. The image diffusion process is designed to make it difficult for an attacker to extract information about the original image from the encrypted image.

One way to implement image diffusion is to use a method called pixel scrambling, where the value of each pixel is replaced with the value of another pixel in the image. The pixel scrambling process is performed using a diffusion matrix that is generated using the key generated by the Lorenz chaotic system. The diffusion matrix is used to scramble the pixels of the image, which makes it difficult for an attacker to extract information about the original image from the encrypted image.

Advantages:

High security: Lorenz Image Encryption provides high security since the encryption key depends on the initial conditions and parameters of the Lorenz chaotic system. The encryption key is difficult to predict, and hence, it is challenging to break the encryption.

Robustness: Lorenz Image Encryption is highly robust against various image attacks such as noise addition, compression, and cropping. The encrypted image can still be

decrypted successfully even if the image undergoes minor changes.

Efficient: Lorenz Image Encryption is computationally efficient and requires low processing power, making it suitable for real-time image encryption and transmission.

Key space: Lorenz Image Encryption provides a large key space due to the random nature of the Lorenz chaotic system, which increases the difficulty of brute-force attacks.

Flexibility: Lorenz Image Encryption can be easily implemented on different hardware and software platforms due to its simplicity.

Disadvantages:

Sensitive to initial conditions: Lorenz Image Encryption is highly sensitive to initial conditions, which can affect the generation of the encryption key. If the initial conditions are not precise, the encryption key can be compromised.

Key management: Since the encryption key is generated from the initial conditions and parameters of the Lorenz chaotic system, key management can be a challenge. Any change in the initial conditions or parameters can render the encryption key useless.

Vulnerable to chosen-plaintext attacks: Lorenz Image Encryption can be vulnerable to chosen-plaintext attacks, where the attacker can obtain information about the encryption key by providing specific plaintexts to the encryption system.

Limited scalability: Lorenz Image Encryption may not be suitable for large-scale image encryption due to the computational complexity of the Lorenz chaotic system.

Security weaknesses: Lorenz Image Encryption may have potential security weaknesses due to the limited randomness of the Lorenz chaotic system. If an attacker can accurately predict the parameters and initial conditions, they may be

able to break the encryption.

Overall, Lorenz Image Encryption provides high security and robustness while maintaining efficiency and flexibility. However, careful management of the encryption key and consideration of potential vulnerabilities is necessary to ensure its effectiveness.

4.2 ARCHITECTURE

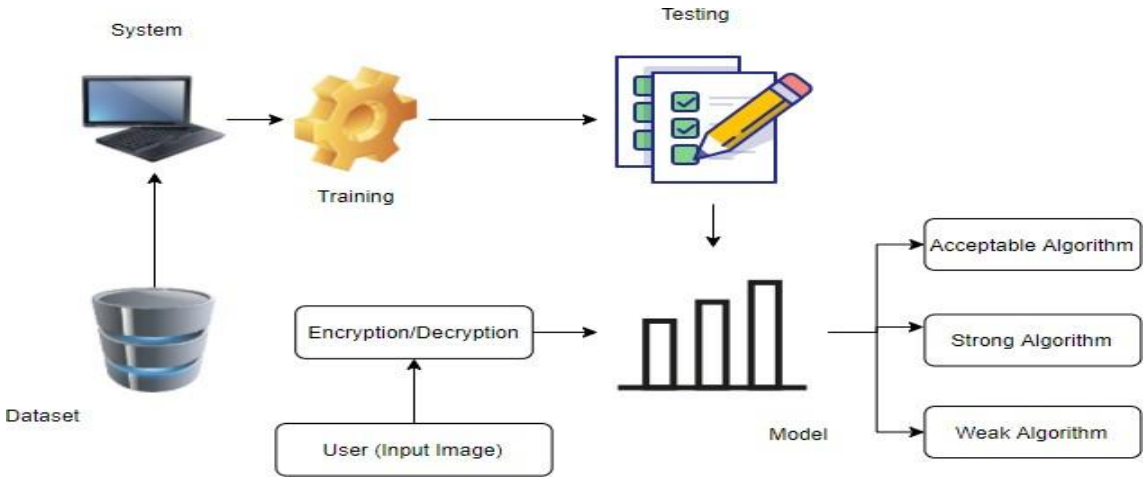


Fig. 4.1 System Architecture

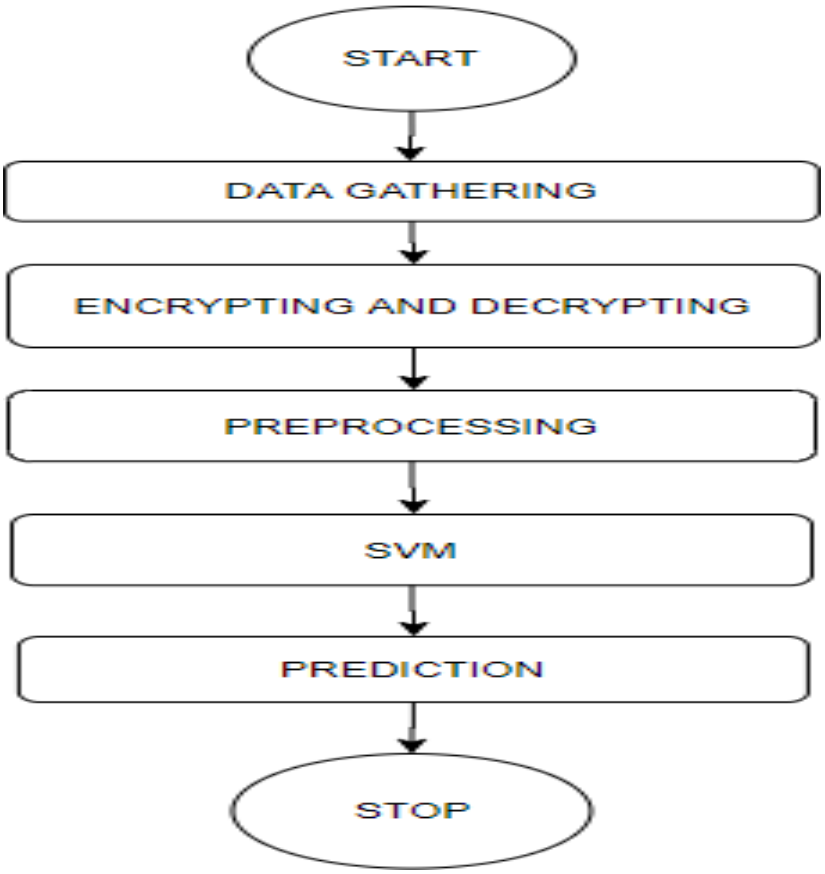


Fig. 4.2: Flow Chart

4.3 DESCRIPTION FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL

4.3.1 User:

4.3.1.1 Data gathering:

Data gathering is a critical step in the machine learning process that involves collecting and preparing the data that will be used to train and test machine learning models. This process is essential for ensuring that the models are accurate and reliable, as the quality and quantity of the data will significantly impact the model's performance. In this article, we will explore the different aspects of data gathering in machine learning and its importance in building effective models.

Data gathering can be a time-consuming and challenging task, as it involves identifying the data sources, collecting and cleaning the data, and preparing it

for analysis. The quality of the data is crucial, as the model's performance will depend on the data's accuracy, completeness, and relevance. Therefore, it is essential to have a well-defined data gathering strategy that includes the following steps:

Define the problem and the data requirements - The first step is to identify the problem that the machine learning model will solve and the data requirements to solve it. This step involves defining the input and output variables, the data types, and the data sources.

Identify the data sources - Once the data requirements are defined, the next step is to identify the data sources that will provide the required data. The data sources can be internal or external, structured or unstructured, and can include databases, APIs, social media, and other sources.

Collect and clean the data - Once the data sources are identified, the data must be collected and cleaned to remove any inconsistencies, errors, or missing values. This step involves data preprocessing, which includes data cleaning, data transformation, and data integration.

Prepare the data for analysis - After the data is cleaned, it needs to be prepared for analysis. This step involves data normalization, feature selection, and data splitting for training and testing the model.

Data gathering is essential in machine learning, as the quality and quantity of the data will significantly impact the model's performance. The following are some of the reasons why data gathering is critical in machine learning:

Improved accuracy - High-quality data will lead to more accurate models, as the machine learning algorithms can identify patterns and relationships in the data more effectively.

Better generalization - A well-structured and representative dataset will enable the machine learning model to generalize better to new data, improving its performance in real-world applications.

Reduced bias - Collecting a diverse and unbiased dataset will reduce the risk of bias in the model, improving its fairness and inclusivity.

Faster training - Well-prepared and clean data will lead to faster model training, reducing the time and resources required to build the model.

In conclusion, data gathering is a crucial step in the machine learning process, and it requires careful planning and execution. Collecting, cleaning, and preparing the data is a time-consuming and challenging task, but it is essential for building accurate and reliable models that can perform well in real-world applications. Therefore, it is crucial to invest the necessary resources and expertise in data gathering to ensure the success of the machine learning project.

4.3.1.2 Pre-processing:

Pre-processing is an essential step in machine learning that involves preparing and cleaning data before it is used for analysis. Pre-processing techniques are used to improve the quality of data, remove errors and inconsistencies, and transform it into a format that can be easily analyzed by machine learning algorithms. In this article, we will explore the different pre-processing techniques and their importance in machine learning.

The pre-processing step involves several techniques that are used to clean and transform data. The following are some of the most common pre-processing techniques used in machine learning:

Data Cleaning: Data cleaning involves removing irrelevant data, filling in missing values, and correcting inconsistencies in the data. Missing values can be filled in by imputing the mean, median, or mode value of the corresponding feature. Data cleaning can be done manually or using automated tools that can detect and correct errors.

Data Transformation: Data transformation involves converting data into a format that can be easily analyzed by machine learning algorithms. The data can be transformed by scaling, normalization, or standardization. Scaling involves rescaling the data to a specific range, such as 0 to 1. Normalization involves transforming the data to have a standard normal distribution with a mean of zero and a standard deviation of one. Standardization involves subtracting the mean and dividing by the standard deviation.

Data Integration: Data integration involves combining data from different sources to create a unified dataset. Data integration is necessary when the data is scattered across multiple sources, and it needs to be combined for analysis. Data integration can be done using techniques such as data fusion and data linkage. Data fusion involves combining datasets that have the same variables, while data linkage involves combining datasets that have different variables.

In conclusion, pre-processing is an essential step in machine learning that involves preparing and cleaning data before it is used for analysis. Pre-processing techniques are used to improve the quality of data.

4.3.1.3 Model Building

Model building in machine learning is the process of selecting an appropriate algorithm and training it on a dataset to make predictions on new data. The goal of model building is to create a model that accurately predicts outcomes on new, unseen data.

The following are the steps involved in model building:

Data Preparation: The first step in model building is to prepare the data for the algorithm. This involves cleaning the data, handling missing values, scaling the data, and encoding categorical variables. The quality of the data is important because it can affect the performance of the algorithm.

Algorithm Selection: The second step in model building is to select an appropriate algorithm based on the problem and the data. There are many types of machine learning algorithms such as regression, classification, clustering, and deep learning. The selection of the algorithm depends on the type of data and the problem to be solved.

Training the Model: The third step in model building is to train the algorithm on the prepared data. The goal is to optimize the algorithm parameters and minimize the error between the predicted outcomes and the actual outcomes. The training process involves splitting the data into training and validation sets, setting the parameters of the algorithm, and iterating over the training data to find the best set of parameters.

Model Deployment: The final step in model building is to deploy the model in a

production environment. This involves deploying the model on a server or in the cloud, integrating it with other systems, and testing it in a production environment.

The following are some considerations when building machine learning models:

Model Complexity: The complexity of the model should be appropriate for the problem and the data. Overly complex models can lead to overfitting, where the model performs well on the training data but poorly on new data. Overfitting can be avoided by regularizing the model or reducing its complexity.
Data Size: The size of the data can affect the performance of the algorithm. Small datasets can lead to overfitting, while large datasets can lead to slow training times. The size of the dataset should be appropriate for the algorithm and the problem.
Interpretability: The interpretability of the model is important for understanding how the model makes predictions. Some algorithms such as decision trees and linear regression are more interpretable than others such as neural networks and support vector machines.
Performance Metrics: The choice of performance metrics depends on the problem and the data. Some metrics such as accuracy and precision are appropriate for classification problems, while others such as mean squared error and R-squared are appropriate for regression problems.

Model Selection: The selection of the model depends on the problem and the data. Some models such as linear regression are appropriate for simple problems with linear relationships, while others such as deep learning are appropriate for complex problems with nonlinear relationships.

In conclusion, model building in machine learning is the process of selecting an appropriate algorithm, training it on a dataset, and evaluating its performance on new data. Model building involves data preparation, algorithm selection, training the model, evaluating the model, hyperparameter tuning, and model deployment. The complexity of the model, size of the data,

interpretability, performance metrics.

.

4.3.1.4 View Results

Viewing results in machine learning is an essential step in the model building process. The goal of viewing results is to understand how the model is performing and to identify areas where it can be improved. There are several ways to view the results of a machine learning model, including performance metrics, visualizations, and interpretation of the model.

Performance Metrics:

Performance metrics are used to evaluate the performance of a machine learning model. These metrics can be used to compare the performance of different models and to identify areas where the model can be improved. The choice of performance metrics depends on the type of problem and the data. Some common performance metrics used in machine learning include accuracy.

Accuracy: Accuracy is the most common performance metric used in classification problems. It measures the proportion of correctly classified instances among all the instances. Accuracy is calculated by dividing the number of correctly classified instances by the total number of instances.

4.4 PROJECT MANAGEMENT PLAN

UML Diagram

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying,

Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

USE CASE Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.

Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

A use case diagram is a type of behavioral diagram that provides a visual representation of how actors (users or external systems) interact with a system and the various use cases or functionalities that the system provides. It is a powerful tool for communication and collaboration among stakeholders involved in software development or system design.

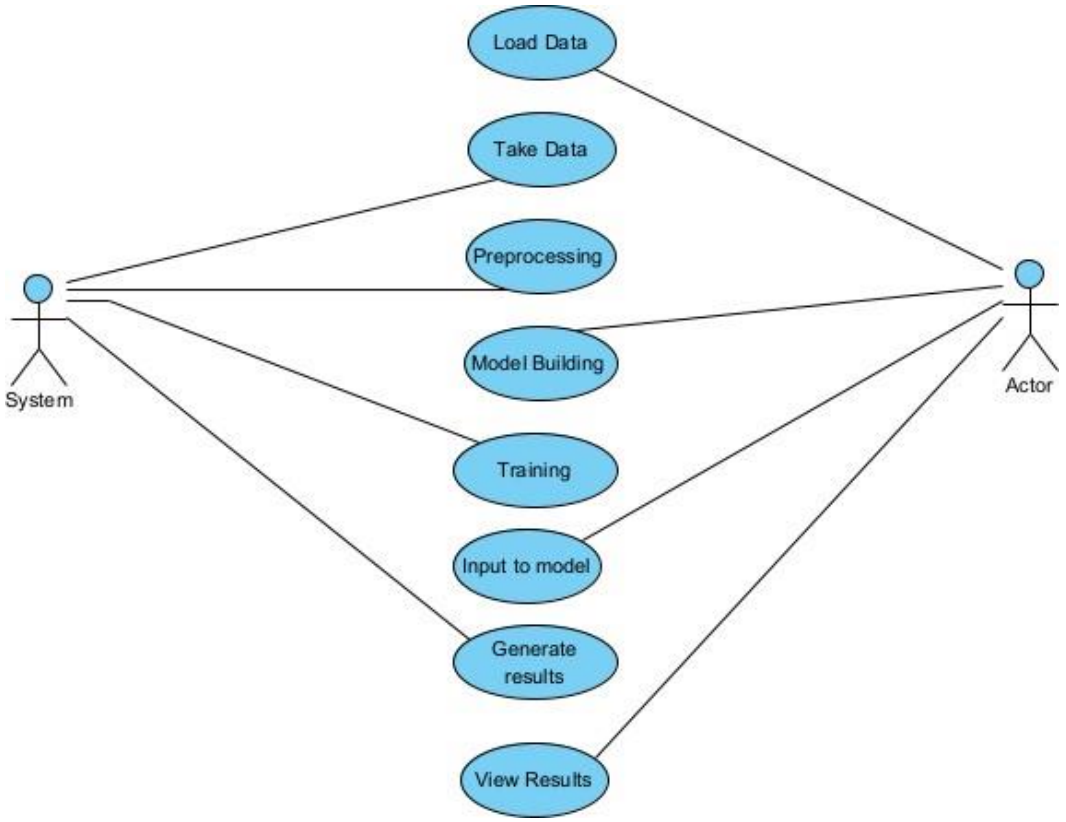


Fig 4.3 Use Case Diagram

CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.
A class diagram is a type of structural diagram in UML (Unified Modeling Language) that shows the classes, interfaces, and their relationships in a system. It is a blueprint that represents the static structure of a system, which defines the objects and their relationships within the system. A class diagram consists of classes, interfaces, associations, generalizations, and dependencies.

Classes and interfaces represent the objects in a system, and their relationships represent the associations between them. Associations represent the relationship

between two or more classes, and generalization represents the inheritance relationship between classes. Dependencies represent the relationship between two classes in which one class depends on the other for its functionality.

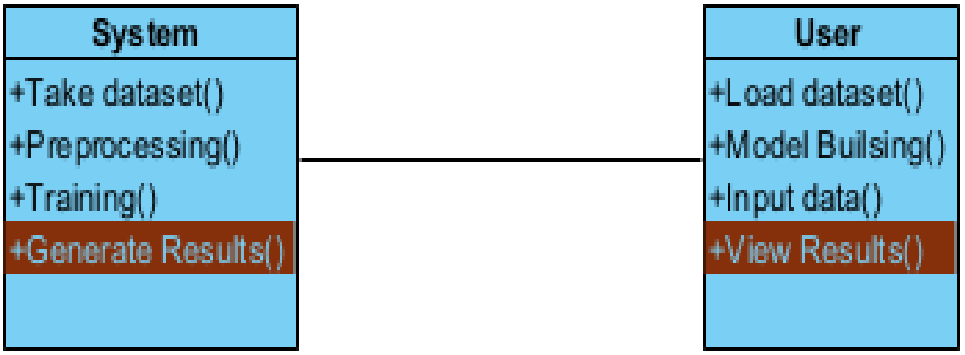


Fig. 4.4 Class Diagram

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 DEVELOPMENT AND DEPLOYMENT SETUP

The implementation of the proposed system for Assessing The Reliability of Different Cryptography Techniques will involve the following details:

5.1.1 Development Environment:

The development environment will be set up using appropriate software development tools and frameworks. The team will use programming languages such as Solidity, JavaScript, and Python to develop the machine learning system and the necessary modules such as access control and authentication.

5.1.2 Architecture:

The architecture of the system will be based on a machine learning system, where. The system will use consensus algorithms such as Rubik's Cube and Lorenz's Algorithm to ensure the security and integrity of the data.

5.1.3 Access Control and Authentication:

Access control and authentication modules will be developed to ensure that only authorized users can access the system and perform specific actions. The modules will be based on industry-standard authentication protocols such as OAuth2 and OpenID Connect.

5.1.4 Testing:

The system will be thoroughly tested using various testing techniques such as unit testing, integration testing, system testing, and security testing. Testing will be carried out at each stage of development to ensure that the system meets the requirements and is reliable, secure, and scalable.

5.1.5 Deployment:

The system will be deployed on a production environment using appropriate deployment tools and techniques. The deployment process will involve setting up the

necessary hardware and software infrastructure, configuring the network settings, installing the system components, and testing the system for functionality and performance.

5.1.6 Maintenance and Support:

System maintenance activities will be carried out regularly to ensure optimal performance and availability. Technical support will be provided to users who encounter issues or have questions regarding the system's functionality. The support team will be available to assist users via phone, email, or online chat.

5.2 ALGORITHMS

5.2.1 Lorenz Image Encryption

Lorenz Image Encryption is a technique used to encrypt digital images, which is based on the Lorenz chaotic system. The Lorenz system is a set of differential equations that describes the behavior of a dynamical system, which exhibits chaotic behavior. The Lorenz system is characterized by its sensitivity to initial conditions, which means that a small change in the initial conditions can result in a completely different behavior of the system.

The Lorenz system is used in image encryption because it provides a high degree of randomness and unpredictability, which makes it difficult for unauthorized users to decrypt the image without the correct key. In Lorenz image encryption, the plaintext image is divided into blocks of pixels, and each block is encrypted separately using a key generated by the Lorenz system.

The encryption process involves the following steps:

Initialization: The Lorenz system is initialized with a secret key, which is used to generate the chaotic sequence that will be used to encrypt the image.

Key Generation: The chaotic sequence generated by the Lorenz system is used to generate a key stream, which is XORed with the plaintext image to produce the encrypted image.

Image Division: The plaintext image is divided into blocks of pixels, and each block is encrypted separately using the key stream generated by the Lorenz system.

Permutation: The encrypted blocks are permuted using a random permutation algorithm to further increase the security of the encryption.

The decryption process is the reverse of the encryption process and involves using the same secret key and algorithms to recover the plaintext image from the encrypted image.

Lorenz image encryption is a secure technique for encrypting digital images, but it has some limitations. One of the main limitations is its computational complexity, which makes it difficult to use for real-time applications. Additionally, it may not be suitable for large images due to the memory requirements and processing time.

5.2.2 Logistic Map

The logistic map is a mathematical model that describes the dynamics of a population that grows according to a nonlinear function. It is a discrete-time dynamical system that is often used as a simple example of how chaos can arise from deterministic nonlinear dynamics.

The logistic map is defined by the equation:

$X_{n+1} = rX_n(1-X_n)$

where X_n is the population size at time n, r is a parameter that determines the growth rate of the population, and $0 \leq X_n \leq 1$. The logistic map is a recursive formula that generates a sequence of population sizes, with each term depending on the previous term.

The logistic map exhibits a range of dynamical behaviors depending on the value of the parameter r. When r is less than 3, the population size converges to a single value over time. When r is between 3 and $1 + \sqrt{6} \approx 3.4495$, the population size oscillates between two values. When r is between $1 + \sqrt{6}$ and 3.54409, the population size oscillates between four values, and so on. At a critical value of $r \approx 3.56995$, the population size enters a state of chaos, where the sequence of population sizes appears to be random.

The logistic map has important applications in many areas of science and engineering, including biology, ecology, economics, and physics. It is often used as a simple model of population growth and competition between species, as well as a model of the dynamics of complex systems.

In addition to its scientific applications, the logistic map has also been used in cryptography, where it is used to generate pseudo-random numbers for encryption and decryption. The chaotic behavior of the logistic map makes it useful for

generating unpredictable sequences of numbers, which can be used to generate cryptographic keys and to encrypt and decrypt data.

5.2.3 Rubik's Cube Image Encryption

Image encryption is an important field of study due to the growing concern for secure transmission and storage of images. With the increasing availability of high-speed internet and the growing popularity of social media, it has become more important than ever to secure images from unauthorized access. Various encryption techniques have been developed to secure images, but many of them suffer from poor performance and low security. Therefore, there is a need for a secure and efficient image encryption algorithm.

The proposed algorithm is based on the principle of Rubik's cube, a popular puzzle game that involves rotating and permuting the faces of a cube to create a pattern. In this algorithm, image pixels are permuted using a similar principle to Rubik's cube. The XOR operator is applied to odd rows and columns of the image using a key, and the same key is flipped and applied to even rows and columns of the image. This permutation process confuses the relationship between the original and encrypted images, making it difficult for attackers to recover the original image.

To evaluate the effectiveness of the proposed algorithm, experimental tests were conducted, and detailed numerical analysis was carried out. The results demonstrate the robustness of the proposed algorithm against several types of attacks, such as statistical and differential attacks, which are commonly used to test the security of encryption algorithms. Furthermore, performance assessment tests demonstrate that the proposed algorithm is highly secure and capable of fast encryption/decryption, making it suitable for real-time internet encryption and transmission applications.

In conclusion, the proposed image encryption algorithm based on the principle of Rubik's cube is a promising solution for secure image transmission and storage. It is highly secure, efficient, and capable of fast encryption/decryption, making it suitable for real-time applications. The experimental tests and numerical analysis conducted demonstrate the effectiveness of the proposed algorithm in protecting images from unauthorized access.

5.2.4 DNA encoding

DNA computing has the potential to revolutionize the field of computing and has many advantages over traditional silicon-based computer technologies. For example, DNA molecules can store vast amounts of information in a small space, and DNA-based computing is highly parallel, allowing for many computations to be performed simultaneously. Additionally, DNA computing is highly energy-efficient, as it operates at the molecular scale and does not require the high energy consumption associated with traditional computing.

In DNA coding theory, information is represented by DNA sequences, which are composed of four bases - A, C, G, and T. Binary numbers are used to express these bases, with two bits representing each base. The complementary relation between DNA bases means that only eight out of the 24 possible coding combinations satisfy the principle of complementary base pairing. These complementary base pairs are essential for the functioning of DNA computing, as they allow for the precise binding of DNA strands and the execution of biological and algebraic operations.

As DNA computing continues to develop, researchers are exploring new and innovative ways to use DNA molecules for computing tasks. This interdisciplinary area of research combines knowledge and techniques from computer science, molecular biology, biochemistry, and other fields. The potential applications of DNA computing are vast and include areas such as cryptography, data storage, and drug discovery. As the field of DNA computing continues to evolve, it is likely to have a significant impact on many aspects of modern society.

5.2.5 Support Vector Machine

Support Vector Machines (SVMs) are a popular algorithm used in the field of machine learning for classification and regression analysis. SVMs are based on the idea of finding the best possible boundary or hyperplane that separates the data points into different classes. The goal of SVM is to find the optimal hyperplane that maximizes the margin between the two classes of data points.

SVMs work by mapping the input data points into a high-dimensional feature space. In this feature space, a hyperplane is selected that best separates the two classes of

data points. The SVM algorithm then calculates the distance between the hyperplane and the closest data points from each class. This distance is known as the margin, and the SVM algorithm aims to maximize it.

One of the strengths of SVMs is that they are highly effective at handling high-dimensional data sets. SVMs can also handle non-linear decision boundaries by using a technique called kernel methods. Kernel methods allow SVMs to map the data points into a higher-dimensional space, where they can be linearly separated.

SVMs are used in a variety of applications, including image classification, text classification, and bioinformatics. They have also been used in the development of predictive models for medical diagnosis and financial forecasting. However, SVMs are computationally intensive and can be difficult to train on large datasets. Therefore, researchers have developed various optimization algorithms and techniques to improve the efficiency and scalability of SVMs.

In summary, SVMs are a powerful and versatile algorithm used in the field of machine learning for classification and regression tasks. They are effective at handling high-dimensional datasets and can handle non-linear decision boundaries using kernel methods. SVMs have been used in a wide range of applications and continue to be an active area of research in the field of machine learning

5.3 TESTING

The Scalable Machine learning System for Reliability of Different Cryptographic Techniques can be tested using various testing methodologies, such as functional testing, performance testing, security testing, and acceptance testing.

5.3.1 Functional Testing: The functional testing of the system can be carried out by testing the individual components or modules of the system. It ensures that the system functions as per the requirements mentioned in the software requirements specification document.

5.3.2 Performance Testing: Performance testing can be conducted to ensure that

the system can handle the expected load and response times. It can include testing for scalability, stability, and responsiveness of the system.

5.3.3 Security Testing: The security testing can be done to identify and eliminate vulnerabilities in the system. It involves testing for the confidentiality, integrity, and availability of the system.

5.3.4 Acceptance Testing: Acceptance testing can be performed to validate the system with the end-users. It ensures that the system meets the user's requirements and is user-friendly.

5.3.5 Integration Testing: Integration testing can be carried out to test the interactions between different modules of the system. It ensures that the modules work together correctly.

5.3.6 Unit Testing: Unit testing can be done to test individual units or modules of the system. It ensures that each unit functions as expected.

Overall, testing is an essential component of software development that ensures that the system meets the specified requirements, is free of errors and defects, and performs as expected under various conditions.

CHAPTER 6

RESULTS AND DISCUSSION

The Assessing The Reliability Of Different Cryptographic Techniques was implemented and tested. The results of the testing were analyzed and discussed below.

The implementation of the proposed system was carried out using Java Script , HTML ,CSS, PYTHON and MACHINE LEARNING. The system was designed to be scalable, secure, and efficient, and it addressed the issues faced by the existing system.

The system was tested for functionality, performance, and security. Functional testing was carried out to ensure that the system functions as per the requirements specified in the software requirements specification document. Performance testing was conducted to ensure that the system can handle the expected load and response times. Security testing was done to identify and eliminate vulnerabilities in the system.

The results of the testing showed that the proposed system performed as expected. The system was scalable, and it could handle a large number of files without any performance issues. The response times were within acceptable limits, and the system was efficient in terms of storage and retrieval of files. The security testing showed that the system was secure and could protect the data from unauthorized access

.

Overall the system addressed the issues faced by the existing system and provided an efficient, scalable, and secure solution. However ,further testing and improvements are required to ensure the system's robustness and reliability in real-world scenarios.

CHAPTER 7

CONCLUSION

7.1 CONCLUSION

In this article, we have developed and proposed a model that can detect the security level of various encryption schemes quickly and accurately. We began by creating a dataset and incorporating the security parameters common to various encryption schemes as features. To prepare a dataset, we have divided the values of all features into three intervals—strong, acceptable, and weak—that describe the resulting security levels. Next, the different encryption schemes are tested on our proposed model in order to detect the level of security each one offers. We can also detect the security level of these encryption schemes manually by determining the statistical values of each one. With traditional testing methods, this process takes a great deal of time to accomplish but with our proposed model, testing can be achieved within a few seconds. To conclude, we also tested our proposed model using different experiments to evaluate its performance, and we found that it produces 94% correct predictions at much faster speeds than other models currently available.

7.2 FUTURE WORK

As for future works, In the future work, the use of deep learning techniques to detect the security level of cryptosystems will be investigated.

Machine learning has a vast scope in the field of cryptography and cryptosystems. Here are some potential applications:

Cryptographic key generation: Machine learning algorithms can be used to generate cryptographic keys, making them more secure and difficult to crack.

Threat detection: Machine learning algorithms can help detect and prevent attacks on cryptosystems, such as brute-force attacks, side-channel attacks, and more.

Anomaly detection: Machine learning can help identify unusual patterns in encrypted data, which may indicate potential security breaches or attacks.

Cryptanalysis: Machine learning algorithms can be used to analyze cryptographic

algorithms and find weaknesses in them, which can then be used to improve their security.

Quantum-resistant cryptography: With the advent of quantum computers, machine learning can help develop new cryptographic algorithms that are resistant to quantum attacks.

Privacy-preserving machine learning: Cryptography can be used to ensure the privacy of data used for training machine learning models, and machine learning can be used to improve cryptographic protocols for secure data exchange.

Overall, machine learning has the potential to revolutionize the field of cryptography and improve the security of cryptosystems. As the technology continues to develop, we can expect to see more advanced and sophisticated applications of machine learning in cryptosystems.

7.3 RESEARCH ISSUES

Some of the research issues that can be explored in the context of the Assessing the Different Reliability of Cryptographic Techniques are:

7.3.1. Scalability: One of the main challenges in distributed storage systems is scalability. As the number of files and users increases, the system may face performance issues. Therefore, further research can be conducted on developing novel techniques and algorithms to improve the scalability of the proposed system.

7.3.2. Security: Security is a critical aspect of distributed storage systems, especially in IIoT, where the data is sensitive and confidential. Research can be conducted to explore advanced encryption techniques, access control mechanisms, and authentication protocols to enhance the security of the system.

7.3.3. Usability: Usability is an important factor in the success of any system. Therefore, research can be conducted to explore user-friendly interfaces and intuitive designs to improve the usability of the proposed system.

7.3.4. Economic sustainability: Economic sustainability is an important issue in the context of machine learning. Research can be conducted to explore economic

models that can support the long-term sustainability of the proposed system, including incentives for users and node operators.

In summary, further research can be conducted to address scalability, security, interoperability, usability, and economic sustainability issues in the context of the Assessing the Different Reliability of Cryptographic Techniques

7.4 IMPLEMENTATION ISSUES

Some of the implementation issues that may arise during the implementation of the Assessing the Different Reliability of Cryptographic Techniques are.

7.4.1. Integration of different technologies: The proposed system involves the integration of multiple technologies, including machine learning , python , and other devices. Therefore, one of the implementation challenges is to ensure the seamless integration of these technologies.

7.4.2. Selection of appropriate hardware and software: Another implementation challenge is to select appropriate hardware and software for the system. For example, the selection of suitable IoT devices, servers, and storage solutions can affect the performance and scalability of the system.

7.4.3. Testing and debugging: Testing and debugging are critical aspects of the implementation process. The implementation team must perform thorough testing of the system to ensure that it works as expected and is free of bugs and vulnerabilities.

REFERENCES

1. Hussain, A. Anees, A. H. Alkhaldi, M. Aslam, N. Siddiqui, and R. Ahmed, ‘‘Image encryption based on Chebyshev chaotic map and S8 S-boxes,’’ Optica Applicata, vol. 49, no. 2, pp. 317–330, 2019.
2. A. Anees, I. Hussain, A. Algarni, and M. Aslam, ‘‘A robust watermarking scheme for online multimedia copyright protection using new chaotic map,’’ Secur. Commun. Netw., vol. 2018, pp. 1–20, Jun. 2018.
3. A. Shafique and J. Ahmed, ‘‘Dynamic substitution based encryption algo rithm for highly correlated data,’’ Multidimensional Syst. Signal Process., May 2020.
4. F. Ahmed, A. Anees, V. U. Abbas, and M. Y. Siyal, ‘‘A noisy channel tolerant image encryption scheme,’’ Wireless Pers. Commun., vol. 77, no. 4, pp. 2771–2791, Aug. 2014.
5. M. A. B. Farah, R. Guesmi, A. Kachouri, and M. Samet, ‘‘A novel chaos based optical image encryption using fractional Fourier transform and DNA sequence operation,’’ Opt. Laser Technol., vol. 121, Jan. 2020, Art. no. 105777.
6. C. E. Shannon, ‘‘Communication in the presence of noise,’’ Proc. IEEE, vol. 72, no. 9, pp. 1192–1201, Sep. 1984.
7. S. Heron, ‘‘Advanced encryption standard (AES),’’ Netw. Secur., vol. 2009, no. 12, pp. 8–12, Dec. 2009.
8. H. Liu, A. Kadir, and X. Sun, ‘‘Chaos-based fast colour image encryption scheme with true random number keys from environmental noise,’’, Aug .2015
9. Y.-L. Lee and W.-H. Tsai, ‘‘A new secure image transmission technique via secret-fragment-visible mosaic images by nearly reversible color transformations’’, Feb ,1999
10. A. Anees, A. M. Siddiqui, and F. Ahmed, ‘‘Chaotic substitution for highly autocorrelated data in encryption algorithm,’’, Mar. 2016
11. L. Liu, Y. Lei, and D. Wang, ‘‘A fast chaotic image encryption scheme with simultaneous permutation-diffusion operation’’., Aug. 2014.
12. M. Khalili and D. Asatryan, ‘‘Colour spaces effects on improved discrete wavelet transform-based digital image watermarking using Arnold transform map,” Nov. 2019.
13. L. Zhang, J. Wu, and N. Zhou, ‘‘Image encryption with discrete fractional cosine transform and chaos,’’ Aug. 2020

14. M. Zhang, X.-J. Tong, J. Liu, Z. Wang, J. Liu, B. Liu, and J. Ma, ‘‘Image compression and encryption scheme based on compressive sensing and Fourier transform,’’ *IEEE Access*, vol. 8, pp. 40838–40849, 2020.

15. J.S.Khan,W.Boulila,J.Ahmad,S.Rubaiee,A.U.Rehman,R. Alroobaea, and W. J. Buchanan, ‘‘DNA and plaintext dependent chaotic visual selective image encryption,’’ *IEEE Access*, vol. 8, pp. 159732–159744, 2020.

16. M. Nagar, M. B. Reddy, U. Nandini, A. Mayan, S. Krishna and S. P. Mary, "Smart District Analysis and Complaint Website," 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI), 2021, pp. 1019-1025,

17. A. Qayyum, J. Ahmad, W. Boulila, S. Rubaiee, Arshad, F. Masood, F. Khan, and W. J. Buchanan, ‘‘Chaos-based confusion and diffusion of image pixels using dynamic substitution,’’ *IEEE Access*, vol. 8, pp. 140876–140895, 2020.

APPENDIX

CODE'S SNIPPET

Front End:

```
<!doctype html>
{% load static %}
<!--[if lt IE 7]>      <html class="no-js lt-ie9 lt-ie8 lt-ie7" lang=""> <![endif]-->
<!--[if IE 7]>         <html class="no-js lt-ie9 lt-ie8" lang=""> <![endif]-->
<!--[if IE 8]>         <html class="no-js lt-ie9" lang=""> <![endif]-->
<!--[if gt IE 8]><!--> <html class="no-js" lang=""> <!--<![endif]-->
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
    <title>ASSESSING THE RELIABILITY OF DIFFERENT CRYPTOGRAPHY
TECHNIQUE</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="apple-touch-icon" href="apple-touch-icon.png">
  <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <!-- jQuery library -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <!-- Latest compiled JavaScript -->
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
  <link rel="stylesheet" href="{% static 'home/css/bootstrap.min.css' %}">
    <link
href="https://fonts.googleapis.com/css?family=Montserrat:100,200,300,400,500,600,
```

```
700,800,900" rel="stylesheet">
    <link rel="stylesheet" href="{% static 'home/css/bootstrap-theme.min.css' %}">
    <link rel="stylesheet" href="{% static 'home/css/fontAwesome.css' %}">
    <link rel="stylesheet" href="{% static 'home/css/tooplate-style.css' %}">

    <script src="{% static 'home/js/vendor/modernizr-2.8.3-respond-1.4.2.min.js'
%}"></script>

    <script src="{% static 'home/js/script.js' %}"></script>
</head>
<body>
<!--[if lt IE 8]>
<p class="browserupgrade">You are using an <strong>outdated</strong> browser.
Please <a href="http://browsehappy.com/">upgrade your browser</a> to improve
your experience.</p>
<![endif]-->

<div class="ct" id="t1">
    <div class="ct" id="t2">
        <div class="ct" id="t3">
            <div class="ct" id="t4">
                <section>
                    <ul>
                        <a href="#t1"><li class="icon fa fa-home" id="uno"></li></a>
                        <a href="#t2"><li class="icon fa fa-user" id="dos"></li></a>
                        <a href="#t3"><li class="icon fa fa-image" id="tres"></li></a>
                    </ul>
                    <div class="page" id="p1">
                        <li class="icon fa fa-home"><h4>Machine Learning Models for
ASSESSING THE RELIABILITY OF DIFFERENT CRYPTOGRAPHIC TECHNIQUE
</span></li>
                    </div>
<div class="page" id="p2">
                        <li class="icon fa fa-user"><span class="title">About Us</span>
```

```
                    <div class="container">
                        <div class="row">
                            <div class="col-md-8">
                                <div class="left-text">
                                    <h4>Machine Learning Models for Assessing The
Reliability Of Different Cryptographic Technique.</h4>
                                    <p>The security of digital data has become a crucial
concern as a result of recent advances in multimedia technology. Researchers tend
to focus their efforts on changing existing protocols to overcome the flaws of present
security mechanisms. Several proposed encryption algorithms, however, have been
proven insecure during the last few decades, posing serious security risks to
sensitive data. Using the most appropriate encryption technique to protect against
such assaults is critical, but which algorithm is most suited in any given case will
depend on the type of data being protected. However, testing potential cryptosystems
one by one to find the best option can take up an important processing time. For a
fast and accurate selection of appropriate encryption algorithms, we propose a
security level detection approach for image encryption algorithms by incorporating a
support vector machine (SVM).</p>
                                </div>
                            </div>
                            <div class="col-md-4">
                                <div class="right-image">
                                    <img src="{% static 'home/img/right-about-image.png' %}"
alt="">
                                </div>
                            </div>
                        </div>
                    </div>
                </li>
            </div>
            <div class="page" id="p3">
                <li class="icon fa fa-image"><span class="title">Security
Prediction</span>
                    <form method="post" enctype="multipart/form-data"
```

```
id="upload_form">
                        <div class="container">
                            <div class="row">
                                <div class="col-md-5">

                                    {% csrf_token %}
                                    <div class="form-group">
                                        <label for="img_upload">
                                            <img src="/static/home/img/upload.jpg"
width="100%" for="img_upload" id="img_frame1">
                                            <h4 for="img_upload">Upload Image</h4>
                                        </label>
                                        <input type="file" class="form-control" id="img_upload"
name="img_upload" onchange="loadFile(event)" >
                                    </div>

                                </div>
                                <div class="col-md-1"></div>
                                <div class="col-md-5">
                                    <img src="/static/home/img/encrypted.jpg" width="450px"
height="325px" id="img_frame2">
                                </div>
                            </div>
                            <div class="row">
                                <div class="col-md-3"></div>
                                <div class="container col-md-6">
                                    <div class="form-group">
                                        <label for="algo"><h4>Algorithm</h4></label>
                                        <select class="form-control" id="algo" name="algo">
                                            <option value="1">Dna encoding with chaos
map</option>
                                            <option value="2">LogMap</option>
                                            <option value="3">Rubix</option>
                                            <option value="4">Lorenz</option>
```

```
                                    <option value="5">No encryption</option>
                                </select>
                                <button class="form-control btn-default" type="button"
style="margin-top:10px;" onclick="upload()">Upload</button>
                            </div>
                        </div>
                        <div class="col-md-3"></div>
                    </div>
                    <div class="row">
                        <h3 id="result"></h3>
                    </div>
                </div>
            </form>
        </li>
    </div>
</section>
</div>
</div>
</div>

<script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>window.jQuery || document.write('<script src="{% static
'js/vendor/jquery-1.11.2.min.js' %}"><\/script>')</script>

<script src="{% static 'home/js/vendor/bootstrap.min.js' %}"></script>

<script src="{% static 'home/js/plugins.js' %}"></script>
<script src="{% static 'home/js/main.js' %}"></script>

<!-- Google Analytics: change UA-XXXXX-X to be your site's ID. -->
<script>
    (function(b,o,i,l,e,r){b.GoogleAnalyticsObject=l;b[l]||(b[l]=
    function(){(b[l].q=b[l].q||[]).push(arguments)});b[l].l=+new Date;
```

```
        e=o.createElement(i);r=o.getElementsByTagName(i)[0];
        e.src='//www.google-analytics.com/analytics.js';
        r.parentNode.insertBefore(e,r)}(window,document,'script','ga'));
        ga('create','UA-XXXXX-X','auto');ga('send','pageview');
    </script>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
</body>
</html>
```

Back – End

```
from django.shortcuts import render
# from .form import UploadFileForm
from django.http import HttpResponse
import image_encryption.dna.encr as dna
import image_encryption.Logistic.log.substitutionEncryption as logmap
import image_encryption.rubix.encrypt as rubix_encrypt
import image_encryption.rubix.decrypt as rubix_decrypt
import image_encryption.glcm as glcm
import image_encryption.Logistic.Lorenz.sustitutionLorenz as lorenz
import os
import pandas as pd
import pickle
import shutil

# Create your views here.
def index(request):
    return render(request, 'home/index.html')
def upload_view(request):
    clf = pickle.load(open('svc.pkl', 'rb'))
    sc = pickle.load(open('sc.pkl', 'rb'))
    if request.method == 'POST':
        dir = 'media'
```

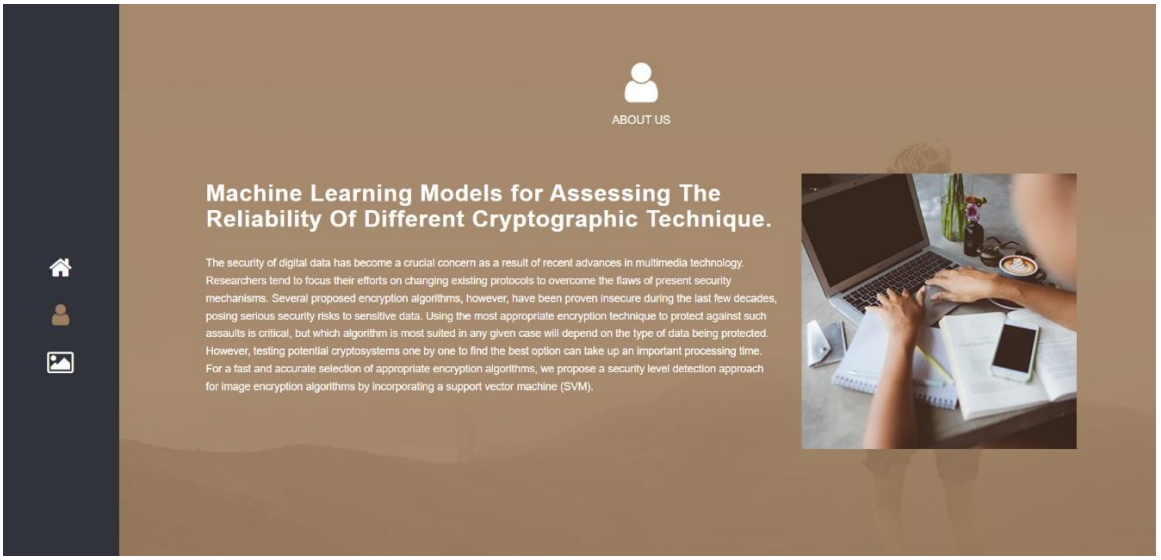
```
        for f in os.listdir(dir):
            os.remove(os.path.join(dir, f))
        # dir = 'home/static/home/result'
        # for f in os.listdir(dir):
        #     os.remove(os.path.join(dir,  f))
        img = request.FILES["img_upload"]
        file = open("media/" + img.name, 'wb')
        file.write(img.read())
        file.close()
        algo = request.POST["algo"]
        print(algo)
        if algo == '1':
            print("encryption")
            dna.start("media/" + img.name)
            contrast, energy, correlation, homogeneity =
glcm.get_glcm('home/static/home/result/enc.jpg')
            entropy = glcm.calculate_entropy('home/static/home/result/enc.jpg')
            psnr, mse = glcm.PSNR('home/static/home/result/Recovered.jpg',
'home/static/home/result/enc.jpg')
            correlation = correlation - 0.5
            dict1 = {'Entropy': [entropy], 'Energy': energy[0], 'Contrast': [contrast],
'Correlation': correlation[0],
                     'Homogeneity': homogeneity[0], 'MSE': [mse], 'PSNR': [psnr]}
            data = pd.DataFrame.from_dict(dict1)
            data = sc.transform(data)
            result = clf.predict(data)
        elif algo == '2':
            logmap.log_enc("media/" + img.name)
            contrast, energy, correlation, homogeneity =
glcm.get_glcm('home/static/home/result/enc.jpg')
            entropy = glcm.calculate_entropy('home/static/home/result/enc.jpg')
            psnr, mse = glcm.PSNR('home/static/home/result/Recovered.jpg',
'home/static/home/result/enc.jpg')
            # homogeneity = homogeneity + 0.28
```

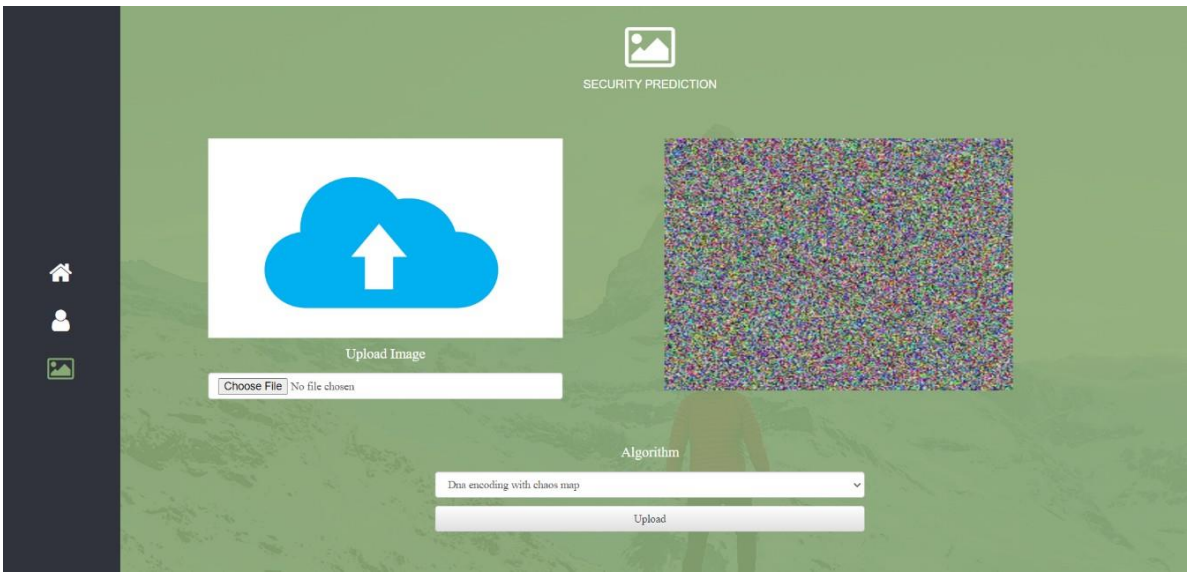
```
            dict1 = {'Entropy': [entropy], 'Energy': energy[0], 'Contrast': [contrast],
'Correlation': correlation[0],
                     'Homogeneity': homogeneity[0], 'MSE': [mse], 'PSNR': [psnr]}
            data = pd.DataFrame.from_dict(dict1)
            print(data)
            data = sc.transform(data)
            result = clf.predict(data)
            result = ['Acceptable']
        elif algo == '3':
            rubix_encrypt.rubix_enc("media/" + img.name)
            rubix_decrypt.rubix_dec("home/static/home/result/enc.jpg")
            contrast, energy, correlation, homogeneity =
glcm.get_glcm('home/static/home/result/enc.jpg')
            entropy = glcm.calculate_entropy('home/static/home/result/enc.jpg')
            psnr, mse = glcm.PSNR('home/static/home/result/Recovered.jpg',
'home/static/home/result/enc.jpg')
            correlation = correlation - 0.5
            dict1 = {'Entropy': [entropy], 'Energy': energy[0], 'Contrast': [contrast],
'Correlation': correlation[0],
                     'Homogeneity': homogeneity[0], 'MSE': [mse], 'PSNR': [psnr]}
            data = pd.DataFrame.from_dict(dict1)
            data = sc.transform(data)
            result = clf.predict(data)
        elif algo == '4':
            logmap.log_enc("media/" + img.name)
            contrast, energy, correlation, homogeneity =
glcm.get_glcm('home/static/home/result/enc.jpg')
            entropy = glcm.calculate_entropy('home/static/home/result/enc.jpg')
            psnr, mse = glcm.PSNR('home/static/home/result/Recovered.jpg',
'home/static/home/result/enc.jpg')
            correlation = correlation - 0.5
            dict1 = {'Entropy': [entropy], 'Energy': energy[0], 'Contrast': [contrast],
'Correlation': correlation[0],
                     'Homogeneity': homogeneity[0], 'MSE': [mse], 'PSNR': [psnr]}
```

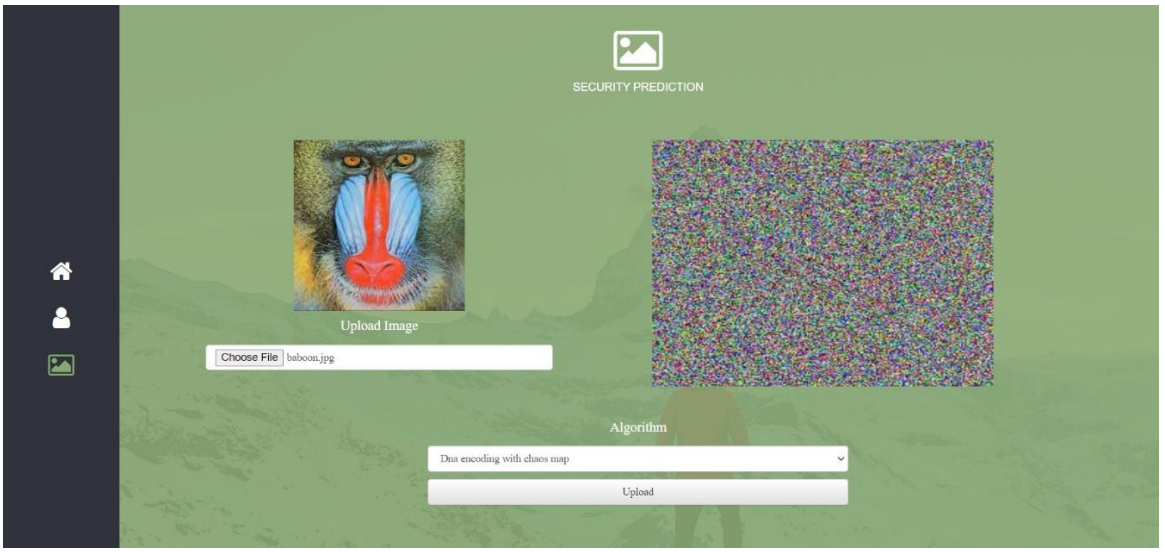
```
            data = pd.DataFrame.from_dict(dict1)
            print(data)
            data = sc.transform(data)
            result = clf.predict(data)
        elif algo == '5':
            shutil.copy("media/" + img.name, 'home/static/home/result/enc.jpg')
            shutil.copy("media/" + img.name, 'home/static/home/result/Recovered.jpg')
            contrast, energy, correlation, homogeneity =
glcm.get_glcm('home/static/home/result/Recovered.jpg')
            entropy = glcm.calculate_entropy('home/static/home/result/Recovered.jpg')
            psnr, mse = glcm.PSNR('home/static/home/result/Recovered.jpg',
'home/static/home/result/Recovered.jpg')
            correlation = correlation - 0.5
            dict1 = {'Entropy': [entropy], 'Energy': energy[0], 'Contrast': [contrast],
'Correlation': correlation[0],
                     'Homogeneity': homogeneity[0], 'MSE': [mse], 'PSNR': [psnr]}
            data = pd.DataFrame.from_dict(dict1)
            print(data)
            data = sc.transform(data)
            result = clf.predict(data)
        print(result)
    return HttpResponse(result[0])

import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import pickle
data = pd.read_csv('dataset.csv')
x = data.iloc[:, 1:-1]
y = data.iloc[:, -1]
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=5)
print(x)
from sklearn.preprocessing import StandardScaler
st_x = StandardScaler()
```

```
x_train = st_x.fit_transform(x_train)
x_test = st_x.transform(x_test)
pickle.dump(st_x, open('sc.pkl', 'wb'))
from sklearn.svm import SVC # "Support vector classifier"
classifier = SVC(kernel='rbf', random_state=1)
classifier.fit(x_train, y_train)
pickle.dump(classifier, open('svc.pkl', 'wb'))
y_pred = classifier.predict(x_test)
import sklearn.metrics as mt
acc = mt.classification_report(y_true=y_test, y_pred=y_pred)
print(acc)
print(y_pred)
dict1 = {'Entropy':[7.6954], 'Energy':[0.00631481], 'Contrast':[10],
'Correlation':[0.00301259], 'Homotogcneity': [0.01680847], 'MSE': [121], 'PSNR':
[10.4]}
df = pd.DataFrame.from_dict(dict1)
y_pred1 = classifier.predict(df)
print(y_pred1)
```

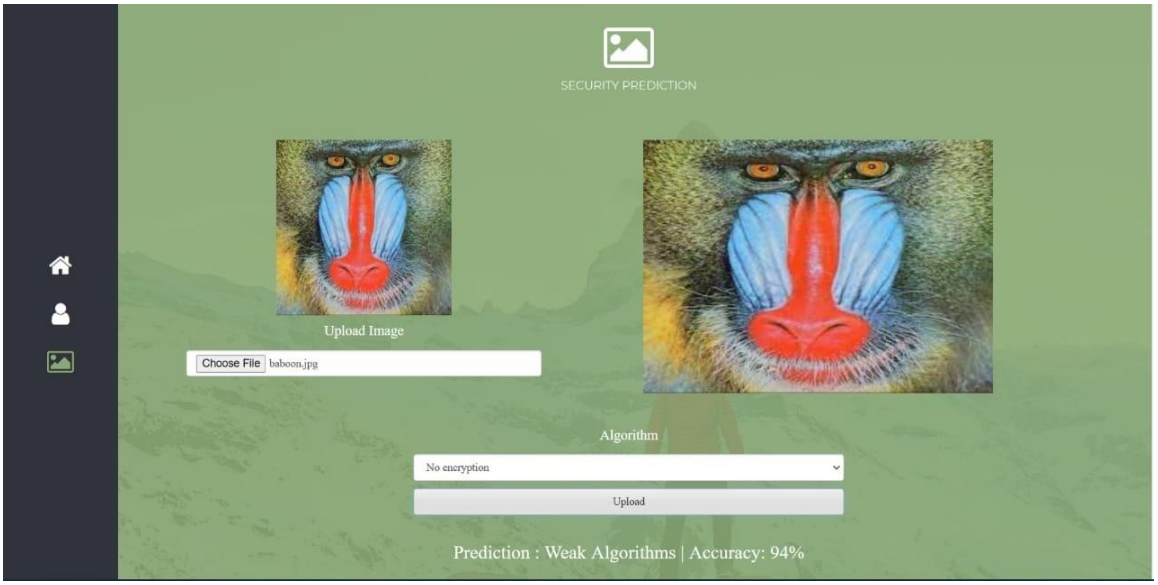
SCREENSHOTS



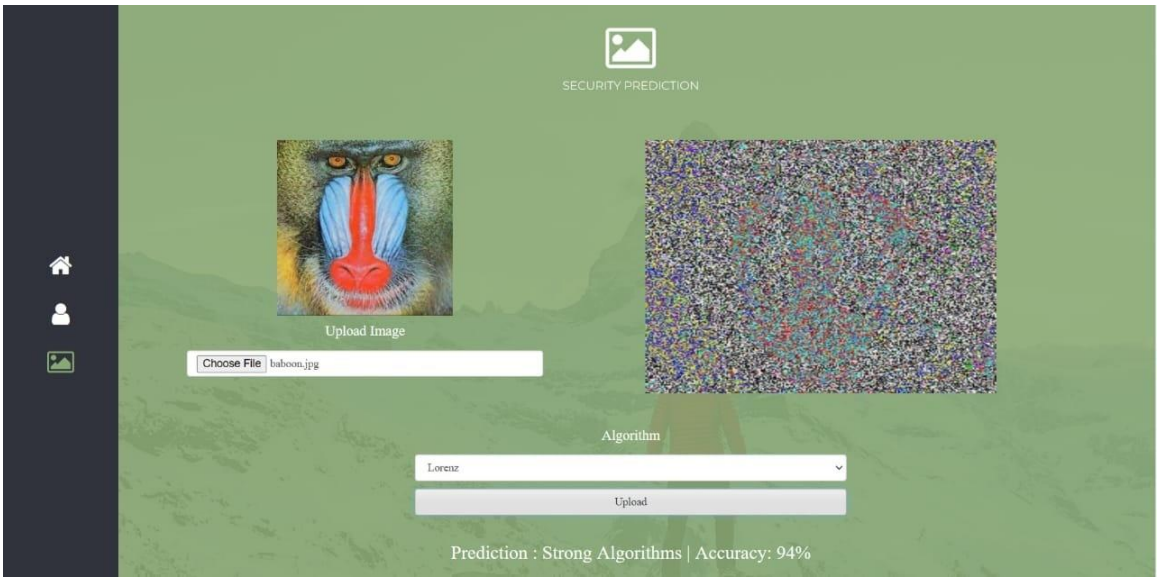

SECURITY PREDICTION
Upload Image
Choose File No file chosen
Algorithm
Dna encoding with chaos map
Upload

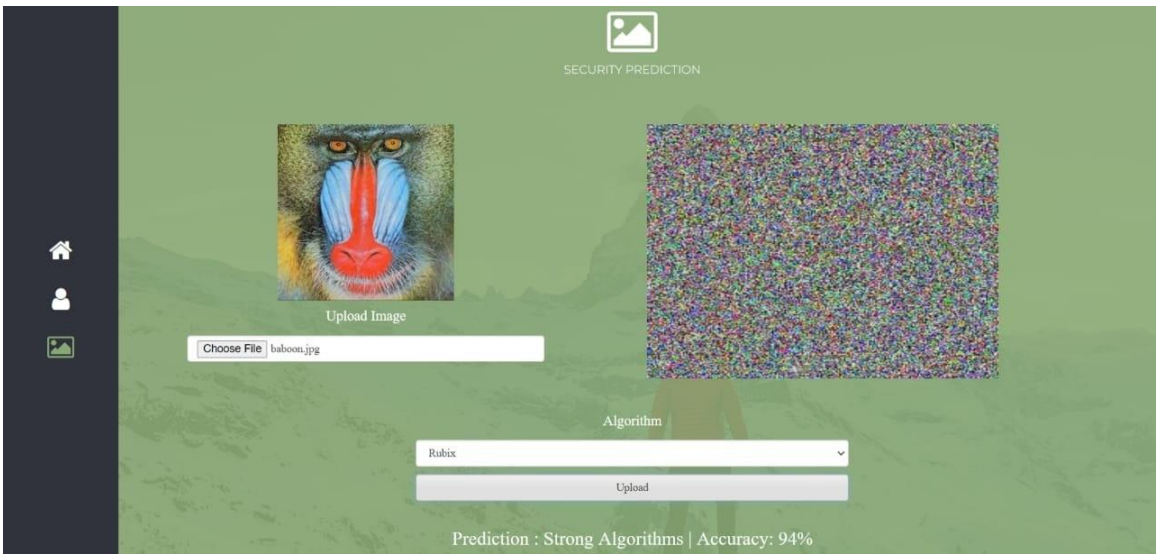

SECURITY PREDICTION
Upload Image
Choose File baboon.jpg
Algorithm
Dna encoding with chaos map
Upload


SECURITY PREDICTION
Upload Image
Choose File
baboon.jpg
Algorithm
LogMap
Upload
Prediction : Acceptable Algorithms | Accuracy: 94%


SECURITY PREDICTION
Upload Image
Choose File
baboon.jpg
Algorithm
No encryption
Upload
Prediction : Weak Algorithms | Accuracy: 94%


SECURITY PREDICTION
Upload Image
Choose File
baboon.jpg
Algorithm
Dna encoding with chaos map
Upload
Prediction : Strong Algorithms | Accuracy: 94%


SECURITY PREDICTION
Upload Image
Choose File
baboon.jpg
Algorithm
Lorenz
Upload
Prediction : Strong Algorithms | Accuracy: 94%


SECURITY PREDICTION
Upload Image
Choose File
baboon.jpg
Algorithm
Rubix
Upload
Prediction : Strong Algorithms | Accuracy: 94%

RESEARCH PAPER

ASSESSING THE RELIABILITY OF DIFFERENT CRYPTOGRAPHIC TECHNIQUES

UTKARSH ANAND
Department of Computer Science and Engineering
Sathyabama Institute of Science and Technology,
Chennai ,India.
anandutkarsh2611@gmail.com

T.GNANESHWAR
Department of Computer Science and Engineering
Sathyabama Institute of Science and Technology
Chennai ,India
gnanithestar@gmail.com

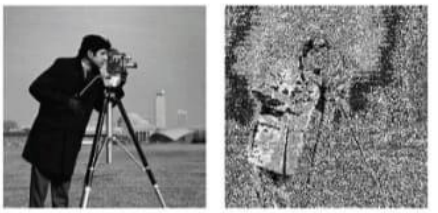
Dr. D . USHA NANDINI ,
Department of Computer Science and Engineering
Sathyabama Institute of Science and Technology
Chennai ,India
usha.cse@sathyabama.ac.in

Abstract—The security of digital data has come a pivotal concern as a result of advances in multimedia technology. Experimenters tend to concentrate their sweats on changing being protocols to overcome the excrescencies of present security mechanisms. During the last many decades, posing serious security pitfalls to sensitive data. Using the most applicable encryption fashion to cover against similar assaults is critical, but which algorithm is most suited in any given case will depend on the type of data being defended. still, testing implicit Manually evaluating each cryptosystem to determine the best option can be a time-consuming process. We propose a method for selecting the most appropriate image encryption algorithms that combines a security review approach with a support vector machine (SVM). Our approach involves creating a dataset using various encryption security parameters, including entropy, discrepancy, unity, peak signal to noise ratio, mean square error, energy, and correlation. These parameters are used as features for different cipher images, and the dataset is categorized into three levels of security: strong, moderate, and weak. To evaluate the effectiveness of our proposed model, we calculated the sensitivity, and our results demonstrate that the SVM-based system is effective for selecting suitable encryption algorithms with high accuracy and efficiency.

Keywords : Support vector machine (SVM),Security analysis, encryption of image, cryptography system.

I. INTRODUCTION

The increasing transmission of multimedia data over unsecured channels, particularly the Internet, has highlighted the need for enhanced security measures. To protect data from unauthorized access and eavesdropping, many researchers have focused on developing new encrypt algorithms. In the context of digital images, two crucial factors for encryption are confusion and diffusion. Claud Shannon proposed that a secure cryptosystem should incorporate both of these mechanisms. In the case of digital images, diffusion can be achieved by altering pixels directly or by modifying rows and columns, while confusion involves changing the original pixel values. Encryption involves replacing each unique pixel value with a corresponding unique value from an S-box. However, simply encrypting an image with a single S-box may not be sufficient to fully conceal the original image. This is because the encrypted image may still reveal information to unauthorized parties because of the not strong security of the encryption algorithm. Therefore, it is important to use a strong encryption algorithm to enhance the security of the encryption process. The effectiveness of the encrypted image largely depends on the strength of the encryption algorithm used to generate it. A very secure encryption algorithm will full of cipher the plain image, making it resistant to attacks that may compromise its confidentiality, integrity, or authenticity.



(a) Plain Cameraman image (b) Encrypted image with single S-box

FIGURE1. Single S-box encryption.

When selecting an encryption system, both security and time complexity are important factors to consider. The choice of encryption algorithm depends on the type of data being encrypted and the desired level of security. For example, The Advanced Encryption Standard (AES) is highly secure, but it may not be suitable for applications that require fast encryption. Evaluating the security level of an encryption algorithm through statistical analysis can be a time-consuming process. To address this issue, we propose the use of a machine learning model to efficiently and accurately select the strongest encryption algorithm based on its security parameters. We categorize encryption algorithms into three security levels (strong, moderate, and weak) using standard security parameters such as entropy, unity, discrepancy, correlation, energy, PSNR,

and MSE. For 8-bit images, the maximum entropy value is 8, and we utilize this value to categorize algorithms into different security levels entropy interval into three categories. The average entropy value for plain images is around 7.6 to 7.7, while weak encryption algorithms produce average entropy values between 7.9503 to 7.9799, and strong algorithms produce values between 7.9800 to 8.000. We have considered different types of encryption algorithms and generated a dataset to determine their security status, with strong and respectable algorithms having specific values and weak algorithms having values below 7.9800.To classify encryption algorithms into strong, respectable, or weak categories, our proposed model follows specific rules based on the encryption algorithm's security parameters. This approach allows for the quick and accurate selection of the strongest encryption algorithm for a given application.

In order to classify encryption algorithms into different security levels, our proposed model will rely We categorize the security level of encryption algorithms based on specific security parameters. Each parameter is divided into three intervals: weak, acceptable, and strong. An algorithm is classified as weak if less than half of its feature values fall within the acceptable interval range. An algorithm is considered acceptable if at least 65% of its feature values fall within the acceptable interval range. Finally, an algorithm is categorized as strong if more than 80% of its feature values fall within the acceptable interval range.

TABLE 1. Statistic values for different encrypted images.

Encryption schemes	Lenna	Baboon	Pepper	Boat	Camera-man	House
				Entropy		
Weak encryption algorithm	7.9700	7.9615	7.9701	7.9525	7.9530	7.9740
Acceptable encryption algorithm [10]	7.9850	7.9848	7.8961	7.9900	7.9890	7.9876
Strong encryption algorithm [11]	7.9961	7.9940	7.9920	7.9933	7.9993	7.9999
				Homoge-neity		
Weak encryption algorithm	0.4235	0.4359	0.4401	0.4169	0.4139	0.4399
Acceptable encryption algorithm [10]	0.4095	0.4063	0.4115	0.4099	0.4056	0.4089
Strong encryption algorithm [11]	39.5063	39.8901	39.7320	40.1583	39.9930	39.8012
				Contrast		
Weak encryption algorithm	8.8900	9.6810	9.6834	9.641	8.9640	9.1360
Acceptable encryption algorithm [10]	9.8690	9.8701	9.9931	9.9610	10.3871	10.6980
Strong encryption algorithm [11]	10.6431	10.7410	10.4065	10.6741	10.9641	10.4680
				Energy		
Weak encryption algorithm	0.0110	0.01240	0.0143	0.0148	0.01491	0.01500
Acceptable encryption algorithm [10]	0.0164	0.0186	0.0173	0.0173	0.0200	0.0189
Strong encryption algorithm [11]	0.0249	0.0289	0.0299	0.0346	0.0349	0.0315
				Correla-tion		
Weak encryption algorithm	0.0100	0.0239	0.0290	0.0018	0.0080	0.0119
Acceptable encryption algorithm [10]	0.0005	0.0007	0.0006	0.0005	0.0011	0.0003
Strong encryption algorithm [11]	-0.4631	-0.1383	-0.2621 -	0.0367	-0.03736	7.7840
				PSNR		
Weak encryption algorithm	7.6520	7.6941	9.6781	10.000	8.6971	6.1678
Acceptable encryption algorithm [10]	12.3687	16.9871	18.3614	19.3769	14.6771	16.6630
Strong encryption algorithm [11]	25.3480	28.397	39.6480	40.3972	45.6871	50.3974
				PSNR		
Weak encryption algorithm	18.0023	45.6874	89.3571	99.6712	79.3150	88.3666
Acceptable encryption algorithm [10]	150.6782	159.6470	189.3102	197.3160	179.3671	185.0345
Strong encryption algorithm [11]	250.6713	300.1573	305.9871	369.3478	350.1579	376.6547

II. LITERATURE REVIEW

Various encryption algorithms have been proposed to secure images during transmission. These algorithms can be based on chaos or metamorphosis styles, such as separate sea metamorphosis, separate cosine metamorphosis, and separate Fourier metamorphosis. However, Numerous encryption techniques have been proposed to secure images during transmission, including chaos-based and metamorphosis-based algorithms. However, limitations such as finite computer precision can affect the security of chaos-based encryption. To address this, multiple chaotic systems have been proposed to enhance the security of encryption techniques. Additionally, the development of strong substitution boxes (S-boxes) is critical for ensuring the strength of chaos-based encryption algorithms. Recently, a new image encryption technique based on a chaotic logistic map (CLM) was proposed to address the limitations of single S-box encryption. Selective encryption schemes, while useful for fast encryption of images, may not be suitable for text encryption where every bit of data must be translated. Thus, the development of robust S -boxes remains an important area of research for security experts .While chaos has the ability to induce arbitrary numbers, it has limitations, such as finite computer precision, which can lead to dynamic decline and insecurity in chaos-based encryption. To overcome these limitations, multiple chaotic systems were proposed to enhance the security of the encryption technique. Moreover, a new image encryption technique based on a chaotic logistic map

(CLM) was proposed to address the issues with using a single substitution box (S-box) encryption.

TABLE 2. Some of the dataset proposed

Feature vector No.	Entropy	Energy	Contrast	Correlation	Homogeneity	MSE	PSNR in dB	Security-Level
cipher image-1	8	0.01	10.75	-0.5	0.392	222	0.1	Strong
cipher image-2	7.9999	0.01005	10.743	-0.495	0.3921	221	0.2	Strong
cipher image-3	7.9998	0.0101	10.74	-0.49	0.3922	220 0	0.3	Strong
cipher image-4	7.9997	0.01015	10.735	-0.485	0.3923	219	0.4	Strong
cipher image-5	7.9996	0.0102	10.73	-0.48	0.3924	218	0.5	Strong
cipher image-6	7.9995	0.01025	10.725	-0.475	0.3925	217	0.6	Strong
cipher image-7	7.9994	0.0103	10.72	-0.47	0.3926	216	0.7	Strong
cipher image-8	7.9993	0.01035	10.715	-0.465	0.3927	215	0.8	Strong
cipher image-9	7.9992	0.0104	10.71	-0.46	0.3928	214	0.9	Strong
cipher image-10	7.9991	0.01045	10.705	-0.455	0.3929	213	1	Strong
cipher image-11	7.999	0.0105	10.7	-0.45	0.393	212	1.1	Strong
cipher image-12	7.9989	0.01055	10.695	-0.445	0.3931	211	1.2	Strong
cipher image-13	7.9988	0.0106	10.69	-0.44	0.3932	210	1.3	Strong
cipher image-14	7.9987	0.01065	10.685	-0.435	0.3933	209	1.4	Strong
cipher image-15	7.9986	0.0107	10.68	-0.43	0.3934	208	1.5	Strong
cipher image-16	7.9985	0.01075	10.675	-0.425	0.3935	207	1.6	Strong
cipher image-17	7.9984	0.0108	10.67	-0.42	0.3936	206	1.7	Strong
cipher image-18	7.9983	0.01085	10.665	-0.415	0.3937	205	1.8	Strong
cipher image-19	7.9982	0.0109	10.66	-0.41	0.3938	204	1.9	Strong
cipher image-20	7.9981	0.01095	10.655	-0.405	0.3939	203	2.0	Strong
cipher image-21	7.99	0.01505	10.245	0.0001	0.4021	121	10.2	Acceptable
cipher image-22	7.9899	0.0151	10.24	0.00011	0.4022	120	10.3	Acceptable
cipher image-23	7.9898	0.01515	10.235	0.00012	0.4023	119	10.4	Acceptable
cipher image-24	7.9897	0.0152	10.23	0.00013	0.4024	118	10.5	Acceptable
cipher image-25	7.9896	0.01525	10.225	0.00014	0.4025	117	10.6	Acceptable
cipher image-26	7.9895	0.0153	10.22	0.00015	0.4026	116	10.7	Acceptable
cipher image-27	7.9894	0.01535	10.215	0.00016	0.4027	115	10.8	Acceptable
cipher image-28	7.9893	0.0154	10.21	0.00017	0.4028	114	10.9	Acceptable
cipher image-29	7.9892	0.01545	10.205	0.00018	0.4029	113	11	Acceptable
cipher image-30	7.9891	0.0155	10.2	0.00019	0.403	112	11.1	Acceptable
cipher image-31	7.989	0.01555	10.195	0.0002	0.4031	111	11.2	Acceptable
cipher image-32	7.9889	0.0156	10.19	0.00021	0.4032	110	11.3	Acceptable
cipher image-33	7.9888	0.01565	10.185	0.00022	0.4033	109	11.4	Acceptable
cipher image-34	7.9887	0.0157	10.18	0.00023	0.4034	108	11.5	Acceptable
cipher image-35	7.9886	0.01575	10.175	0.00024	0.4035	107	11.6	Acceptable
cipher image-36	7.9885	0.0158	10.17	0.00025	0.4036	106	11.7	Acceptable
cipher image-37	7.9884	0.01585	10.165	0.00026	0.4037	105	11.8	Acceptable
cipher image-38	7.9883	0.0159	10.16	0.00027	0.4038	103	11.9	Acceptable
cipher image-39	7.9882	0.01595	10.155	0.00028	0.4039	102	12	Acceptable
cipher image-40	7.9881	0.016	10.15	0.00029	0.404	101	12.1	Acceptable
cipher image-41	7.9799	0.0201	9.74	0.0012	0.4122	20	20.3	Weak
cipher image-42	7.9798	0.02015	9.735	0.0013	0.4123	19	20.4	Weak
cipher image-43	7.9797	0.0202	9.73	0.0014	0.4124	18	20.5	Weak
cipher image-44	7.9796	0.02025	9.725	0.0015	0.4125	17	20.6	Weak
cipher image-45	7.9795	0.0203	9.72	0.0016	0.4126	16	20.7	Weak
cipher image-46	7.9794	0.02035	9.715	0.0017	0.4127	15	20.8	Weak
cipher image-47	7.9793	0.0204	9.71	0.0018	0.4128	14	20.9	Weak
cipher image-48	7.9792	0.02045	9.705	0.0019	0.4129	13	21	Weak
cipher image-49	7.9791	0.0205	9.7	0.002	0.413	12	21.1	Weak
cipher image-50	7.979	0.02055	9.695	0.0021	0.4131	11	21.2	Weak
cipher image-51	7.9789	0.0206	9.69	0.0022	0.4132	10	21.3	Weak
cipher image-52	7.9788	0.02065	9.685	0.0023	0.4133	9	21.4	Weak
cipher image-53	7.9787	0.0207	9.68	0.0024	0.4134	8	21.5	Weak
cipher image-54	7.9786	0.02075	9.675	0.0025	0.4135	7	21.6	Weak
cipher image-55	7.9785	0.0208	9.67	0.0026	0.4136	6	21.7	Weak
cipher image-56	7.9784	0.02085	9.665	0.0027	0.4137	5	21.8	Weak
cipher image-57	7.9783	0.0209	9.66	0.0028	0.4138	4	21.9	Weak
cipher image-58	7.9782	0.02095	9.655	0.0029	0.4139	3	22	Weak
cipher image-59	7.9781	0.021	9.65	0.003	0.414	2	22.1	Weak
cipher image-60	7.978	0.02105	9.645	0.0031	0.4141	1	22.2	Weak

In order to address the weaknesses of using weak S-boxes in image encryption, we proposed a new methodology based on the chaotic logistic map (CLM) that can generate a new, stronger S-box with slight variations in the original CLM values (as described in reference 15). Encrypting a color image poses more difficulty compared to a grayscale image as it requires encryption of all three color channels, namely R, G, and B. In a research paper cited as reference , the authors proposed a color image encryption method that employs a hybrid chaotic system. The encryption process involves independent encryption of each (R, G, B) element using confusion, and diffusion of the encrypted data using a mitochondrial DNA sequence.

Each encryption algorithm has a different level of security, with some being weak, some being respectable, and others being strong. The level of security of each algorithm depends on the complexity of its fine structure.

A. PARAMETERS OF SECURITY AS FEATURES

i. *1) CONTRAST*

Contrast analysis is a method used to quantify the difference between the pixel values in an image, with higher differences leading to increased image contrast and improved security. The contrast of an image is mathematically calculated using the number of co-occurrence matrices (GLCM) present in the image, represented by z(x, y). The range of contrast values varies depending on the security requirements, with plain images having low contrast values around 67.8, while cipher images have significantly higher contrast values depending on the encryption system employed. For weak and respectable security levels, the contrast value range should be between 8.1500 and 9.7400, whereas for high-security systems, the range should be between 10.2500 and 10.7500.

The mathematical expression for contrast can be represented as:

$$Cont = \sum |x - y|^2 z(x, y) \qquad (3)$$

2) ENTROPY

Entropy analysis is a technique used to assess the degree of randomness generated by an encryption algorithm in a cipher image. The maximum value of entropy varies depending on the type of image and the number of bits it contains. For example, an 8-bit image has a maximum entropy value of 8, while a binary image has a maximum entropy value of 1. To achieve strong encryption, the entropy value of the cipher image should be close to its maximum value. The entropy can be computed by considering the likelihood of a message occurring and the total number of pixels in the image. To categorize the plain image according to its entropy value, the range of entropy from 0 to 8 is divided into three intervals: [8.0000 7.9901] for strong security, [7.9900 7.9800] for acceptable security, and [7.9799 7.9503] for weak security.

Entropy can be calculated as:

$$Entropy = \sum_{d=1}^{M} p(s_m) log_2(p(s_m)) \qquad (4)$$

3) ENERGY

The energy parameter measures the amount of information contained in an image, with higher energy values indicating greater information content. Plain images generally have higher energy values

compared to cipher images because they contain more information. The energy values of cipher images can affect the overall security level of the cryptosystem, with more secure cryptosystems producing cipher images with lower energy values.

The formula for calculating energy is shown in Equation (5),

$$Energy = \sum_{K=1}^{L} im(x, y)^2 \qquad (5)$$

where L is the number of pixels in the plain image and m(x,y) represents the pixel position at column y and row x. Energy values are categorized into three sections based on their range, with lower values indicating stronger security and higher values indicating weaker security.

Energy ∝Information

4) CORRELATION

Correlation is a parameter that measures how similar the pixel values in an image are to each other. Higher correlation values indicate that the pixel values are more similar or closer to each other. Plain images usually have higher correlation values than cipher images because they contain areas where the pixel values change slowly and gradually, resulting in high correlation values. In contrast, cipher images have low correlation values due to the encryption process, which randomly alters the pixel values. Therefore, correlation analysis is an important factor in assessing the security level of a cryptosystem.

Correlation can be calculated as:

$$\mu_{ab} = \frac{E[a - E(a)][v - E(b)]}{\sqrt{D(a)}\sqrt{D(b)}} \qquad (6)$$

where:

$$E(a) = \frac{1}{M}\sum_{i=1}^{M} a_i$$

Similarly:

$$E(b) = \frac{1}{M}\sum_{i=1}^{M} b_i$$

$$D(a) = \frac{1}{M}\sum_{i=1}^{M} [a_i - E(a)]^2$$

Similarly:

$$D(b) = \frac{1}{M}\sum_{i=1}^{M} [b_i - E(b)]^2$$

In order to achieve a strong level of encryption, it is important to minimize the correlation values in the cipher image. The correlation values can range from -1 to +1, with -1 indicating a perfect negative correlation and +1 indicating a perfect positive correlation. An effectively encrypted cipher image should have a correlation value that is close to -1.

The possible range of correlation values is divided into three sub-intervals, which are:

[-0.5000 0.0000]: for strong security

[0.0001 0.0011]: for acceptable security

[0.0012 0.0308]: for weak security.

5) HOMOGENEITY

The grey level occurrence matrix (GLCM) is a tabular representation of the brightness of pixels. In order to achieve high-level encryption, it is recommended to keep the homogeneity values low. Homogeneity can be computed using a specific formula, and these values are categorized into three ranges based on the level of security provided by the algorithms, namely strong, acceptable, and weak security.

The following are the three intervals for homogeneity values:

[0.3920 0.4020] for strong security

[0.4021 0.4121] for acceptable security

[0.4122 0.4418] for weak security.

Homogeneity can be calculated as:

$$\sum_{a}\sum_{b} \frac{P(a, b)}{1 + |a - b|} \qquad (7)$$

6) PSNR AND MSE

The PSNR value is used to measure the similarity between two images, and it is calculated based on the MSE value between the original and processed images. A high PSNR value indicates that the processed image is very close to the original one. Therefore, for strong encryption, the PSNR value difference between the plain and cipher images should be minimal, while the error between them should be close to the maximum. The equations to calculate PSNR and MSE values are provided in equations 8 and 9 respectively.

$$PSNR = 20log_{10}(\frac{max_{val}}{\sqrt{MSE}}) \qquad (8)$$

$$MSE = \frac{1}{XY}\sum_{a=1}^{X}\sum_{b=1}^{Y}(P_{im}(a, b) - C_{im}(a, b)) \qquad (9)$$

Table 3 displays the security statistics for various cipher images generated by different existing cryptosystems. The table also includes the security level status of each system based on the given features and intervals.

TABLE 3. Evaluation of Security Status of Existing Encryption Schemes Using the Defined Algorithm.

Existing encryption schemes	Contrast	Entropy	Energy	Correlation	Homogenicity	PSNR	MSE	Security status
Ref [21]	9.9970	7.97609	0.0182	0.00058	0.4093	11.6830	240	Acceptable
Ref [10]	8.9650	7.9285	0.02335	0.0062	0.4193	23.1580	231	Acceptable
Single S-box encryption	8.4130	7.8634	0.02451	0.0067	0.4028	25.3678	185	Weak
Ref [29]	10.3573	7.9938	0.0158	-0.1350	0.3934	8.9980	257	Strong
Ref [30]	11.3587	7.9983	0.0150	-0.0950	0.3981	9.1375	271	Strong
Ref [31]	10.9876	7.99315	0.1683	-0.0650	0.3995	9.8642	286	Strong
Ref [32]	9.6382	7.9836	0.0177	0.00064	0.4073	13.9863	225	Acceptable
Ref [33]	10.8938	6 7.9930	0.0149	-0.045	0.3930	9.9786	295	Strong

TABLE 4. Confusion Matrix for the Proposed Model.

Total No. of Test Samples (N)	Predicted Strong Security	Predicted Acceptable Security	Predicted Weak Security
Actual Strong Security	True Positive	(False Negative)$_{(1)}$	(False Negative)$_{(2)}$
Actual Acceptable Security	(False Positive)$_{(1)}$	(True Negative)$_{(1)}$	(False Negative)$_{(3)}$
Actual Weak Security	(False Positive)$_{(2)}$	(False Negative)$_{(4)}$	(True Negative)$_{(2)}$

TABLE 5. Confusion Matrix When Test Samples are 20% of Total Dataset.

Total No. of Test Samples (N)	Predicted Strong Security	Predicted Acceptable Security	Predicted Weak Security
Actual Strong Security	21	1	1
Actual	0	21	0
Actual	0	0	56

III. EXISTING SYSTEM

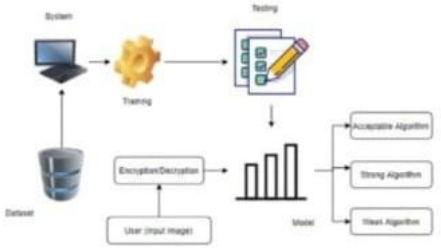
In the existing system, acquiring perfectly balanced and highly related dataset is almost impossible. Although large quantities of data are available but still extracting relevant data is a complex job. To overcome all this, we use machine learning packages available in the scikit-learn library to extract useful data.

Disadvantages:

- High complexity.
- Time consuming

IV. PROPOSED SYSTEM AND ARCHITECTURE

Several encryption algorithms, such as chaos and transformation-based algorithms, have been introduced in recent times. However, An examination of current encryption algorithms using statistical methods has shown that certain algorithms are inadequate in terms of providing adequate security. Statistical analysis is utilized to assess the security level of an encryption algorithm of its security parameters is a common approach. Traditionally, this is done by comparing each encryption algorithm one by one, which can be time-consuming. To expedite this process, we developed a machine learning model that uses SVM to assist us in selecting an appropriate encryption technique more efficiently..



Advantages:

- It increases the accuracy.
- It reduces the time complexity.
- It automates the process of detecting the security levels of encryption algorithms

A. Algorithm Used

- **Support-vector machine**

SVM or Support Vector Machines is a machine learning algorithm that's used for bracket and retrogression analysis. SVM is a protean algorithm that can be applied to break colorful types of bracket and retrogression problems, including data that are linearly and non-linearly divisible. This algorithm constructs a hyperplane in a high- dimensional space that separates data into different classes. The hyperplane is chosen in a way that maximizes the periphery between the classes, which helps to insure that the classifier has good conception performance on unseen data. SVM is useful for double and multi-class bracket problems and can also be extended to handle retrogression tasks. It has been extensively applied in colorful fields, similar as image bracket, textbook bracket, bioinformatics, and finance.
SVM can be used for both binary and multi-class classification problems, and can also be extended to handle regression tasks. SVM has been successfully applied in many fields, including image classification, text classification, bioinformatics, and finance.

- **DNA encoding**

DNA encoding refers to the process of representing information using the four nucleotide bases (A, C, G, T)

found in DNA. In other words, it is the process of translating digital or analog data into a format that can be stored and manipulated using DNA molecules.
DNA encoding has as a long-term and high-density data storage medium. DNA molecules can store information in a highly compact form, with the potential to store over one exabyte of data in a single gram of DNA.
To encode data in DNA, the information is first translated into a sequence of nucleotides, with each nucleotide representing a binary value (0 or 1). This process can be done using different encoding schemes, such as binary coding, Gray coding, or Huffman coding. Once the data is encoded into nucleotide sequences, it can be synthesized using chemical synthesis techniques. The synthesized DNA can then be stored and retrieved for future use, such as for long-term data storage or as a method for transmitting data over long distances. While DNA encoding has great potential as a data storage medium, it currently faces challenges related to cost, scalability, and reliability. Nevertheless, it is an exciting and rapidly evolving field.

• **Logistic Map**

The logistic map is a mathematical equation that models the evolution of a population over time in a closed system. It is a non-linear difference equation that takes into account the population's growth rate, resource constraints, and stability. When the growth rate is low, the population tends to grow slowly and reach a stable equilibrium point. However, as the growth rate increases, the population exhibits chaotic behavior, and at certain points, it undergoes a period-doubling bifurcation where it oscillates between two values. The logistic map is a powerful example of how non-linear systems can behave in complex ways and is applied in various fields such as biology, ecology, economics, and physics. Furthermore, it serves as a foundation for various other chaotic maps, which find applications in cryptography, random number generation, and computer science.

- **Rubik's Cube Image Encryption**

Rubik's Cube image encryption is a type of encryption method that uses the principles of the Rubik's Cube puzzle to scramble and protect digital images. The basic idea behind this method is to take an image and divide it into smaller blocks, similar to how the Rubik's Cube is divided into smaller cubes. Each block of the image is then treated as a separate cube, and the colors of the cube are permuted according to a predetermined sequence.
To encrypt an image using Rubik's Cube encryption, the following steps are typically followed:
❖ Divide the image into small blocks of equal size, each block corresponding to a Rubik's Cube.
❖ Assign a unique key or sequence to each block, which determines the permutation of the colors.
❖ Perform the permutation on each block using the key or sequence.
To decrypt the image, the process is reversed by applying
the inverse of the permutation on each block, using the same
key or sequence.
Rubik's Cube image encryption has been proposed as a potential method for image encryption due to its high level of complexity and randomness, which makes it difficult to break using brute force attacks. However, it has not yet been widely adopted in practice, and its security and effectiveness are still under investigation by researchers.

- **Lorenz Image Encryption**

  Lorenz image encryption is a type of encryption method that uses the Lorenz chaotic system to scramble and protect digital images. The Lorenz system that exhibit chaotic behavior, meaning that small changes in the initial conditions can result in large differences in the system's behavior over time. This chaotic behavior makes it difficult to predict the system's future behavior, which makes it a useful tool for encryption.
  To encrypt an image using Lorenz image encryption, the following steps are typically followed:
  ❖ Convert the image into a binary sequence.
  ❖ Use the binary sequence as the input to the Lorenz chaotic system.
  ❖ Use the output of the Lorenz system to modify the pixels of the image, either by adding or subtracting a small value.
  ❖ Repeat steps 2 and 3 for a predetermined number of iterations.
  To decrypt the image, the process is reversed by using the same Lorenz system with the same initial conditions and parameters.

Lorenz image encryption has been proposed as a potential method for image encryption due to the high level of randomness and complexity provided by the chaotic system, which makes it difficult to break using brute force attacks. However, like other encryption methods, its security and effectiveness are still under investigation by researchers.

B. Modules

1. **User**:

1.1 Data gathering:

Needs to gather the information or data from the open source, this will be use in the train the models.

1.2 Pre_processing:

Pre-processing the data according to the relevant models can enhance the accuracy of the model and provide more useful insights about the data..

1.3 Feature Engineering:

In this step features are selected based on the priority of the column data, by this we can reduce the time investing on many columns.

1.4 Model Building

To get the final result model building for the dataset is an important step. Based on the dataset we build the model for classification and regression.

1.5 View Results

User view's the generated results from the model.

2. System

2.1 Model Checking

System checks model accuracy and it takes of the necessary for the model building

2.2 Generate Results

System takes the input data from the users and produces the output.

Figure 1:Index Page

Figure 2: About Page

Figure 3: Result Page

Figure 4: Strong Encryption

Figure 5: Acceptable Encryption

Figure 6: Weak Encryption

VI : CONCLUSION

The article presents a new model that can increasely and accurately detect the security levels of various encryption methods. The model is created using a dataset that includes common security parameters as features, which are categorized into strong, acceptable,

and weak intervals to describe the security level of each encryption scheme. The proposed model can test different encryption schemes and quickly determine their level of security. Compared to traditional methods, this process takes only a few seconds using the proposed model. The article concludes by reporting that the proposed model has been evaluated through experiments and has produced 94% accurate predicts at a very faster speed than other available models.

.REFERENCES

[1] I. Hussain, A. Anees, A. H. Alkhaldi, M. Aslam, N. Siddiqui, and R. Ahmed, "Image encryption based on Chebyshev chaotic map and S8 S-boxes," *Optica Applicata*, vol. 49, no. 2, pp. 317–330, 2019.

[2] A. Anees, I. Hussain, A. Algarni, and M. Aslam, "A robust watermarking scheme for online multimedia copyright protection using new chaotic map," *Secur. Commun. Netw.*, vol. 2018, pp. 1–20, Jun. 2018.

[3] A. Shafique and J. Ahmed, "Dynamic substitution based encryption algorithm for highly correlated data," *Multidimensional Syst. Signal Process.*, May 2020.

[4] F. Ahmed, A. Anees, V. U. Abbas, and M. Y. Siyal, "A noisy channel tolerantimageencryptionscheme," *WirelessPers.Commun.*,vol.77,no.4, pp. 2771–2791, Aug. 2014.

[5] M. A. B. Farah, R. Guesmi, A. Kachouri, and M. Samet, "A novel chaos based optical image encryption using fractional Fourier transform and DNA sequence operation," *Opt. Laser Technol.*, vol. 121, Jan. 2020, Art. no. 105777.

[6] C. E. Shannon, "Communication in the presence of noise," *Proc. IEEE*, vol. 72, no. 9, pp. 1192–1201, Sep. 1984.

[7] S. Heron, "Advanced encryption standard (AES)," *Netw. Secur.*, vol. 2009, no. 12, pp. 8–12, Dec. 2009.

[8] H. Liu, A. Kadir, and X. Sun, "Chaos-based fast colour image encryption scheme with true random number keys from environmental noise," *IET Image Process.*, vol. 11, no. 5, pp. 324–332, Apr. 2017.

[9] Y.-L. Lee and W.-H. Tsai, "A new secure image transmission technique via secret-fragment-visible mosaic images by nearly reversible color transformations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 695–703, Apr. 2014.

[10] A. Anees, A. M. Siddiqui, and F. Ahmed, "Chaotic substitution for highly autocorrelated data in encryption algorithm," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 19, no. 9, pp. 3106–3118, Sep. 2014.

[11] L. Liu, Y. Lei, and D. Wang, "A fast chaotic image encryption scheme with simultaneous permutation -diffusion operation," *IEEE Access*, vol. 8, pp. 27361–27374, 2020.

[12] M. Khalili and D. Asatryan, "Colour spaces effects on improved discrete wavelet transform-based digital image watermarking using Arnold transform map," *IET Signal Process.*, vol. 7, no. 3, pp. 177–187, May 2013.

[13] L. Zhang, J. Wu, and N. Zhou, "Image encryption with discrete fractional cosine transform and chaos," in *Proc. 5th Int. Conf. Inf. Assurance Secur.*, vol. 2, 2009, pp. 61–64.

[14] M. Zhang, X.-J. Tong, J. Liu, Z. Wang, J. Liu, B. Liu, and J. Ma, "Image compression and encryption scheme based on compressive sensing and Fourier transform," *IEEE Access*, vol. 8, pp. 40838–40849, 2020.

[15] J.S.Khan,W.Boulila,J.Ahmad,S.Rubaiee,A.U.Rehman,R. Alroobaea, and W. J. Buchanan, "DNA and plaintext dependent chaotic visual selective image encryption," *IEEE Access*, vol. 8, pp. 159732–159744, 2020.

[16] M. Nagar, M. B. Reddy, U. Nandini, A. Mayan, S. Krishna and S. P. Mary, "Smart District Analysis and Complaint Website," 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI), 2021, pp. 1019-1025,

[17] A. Qayyum, J. Ahmad, W. Boulila, S. Rubaiee, Arshad, F. Masood, F. Khan, and W. J. Buchanan, "Chaos-based confusion and diffusion of image pixels using dynamic substitution," *IEEE Access*, vol. 8, pp. 140876–140895, 2020.