A Mini Project Report (CS755PC)on

Ransomware Detection using Explainable AI

Submitted

in partial fulfillment of the requirements for

the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

(Artificial Intelligence and Machine Learning)

by

Ms. C Sai Sirisha
(21261A6613)

Under the guidance of

Dr. D. Koteswara Rao
Assistant Professor

Department of Emerging Technologies

Mahatma Gandhi Institute of Technology (Autonomous)

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)
Kokapet(V), Gandipet(M), Hyderabad.
Telangana - 500 075.

2024-25

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

GANDIPET, HYDERABAD – 500075, Telangana

CERTIFICATE

This is to certify that the mini project entitled **“Ransomware Detection using Explainable AI”** is being submitted by **C Sai Sirisha (21261A6613)** in partial fulfillment of the requirements for the Mini Project (CS755PC) in **COMPUTER SCIENCE AND ENGINEERING** (Artificial Intelligence and Machine Learning) is a record of Bonafide work carried out by her.

The results of the investigation enclosed in this report have been verified and found satisfactory.

Supervisor	Head of the Department
Dr. D. Koteswara Rao	**Dr. M. Rama Bai**
Assistant Professor	Professor

External Examiner

DECLARATION

This is to certify that the work reported in this project titled **“RANSOMWARE DETECTION USING EXPLAINABALE AI”** is a record of work done by me in the Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Hyderabad.

No part of the work is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the work done entirely by me and not copied from any other source.

C SAI SIRISHA (21261A6613)

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to **Dr. G. Chandramohan Reddy**, Principal MGIT, for providing the working facilities in college.

I wish to express my sincere thanks and gratitude to **Dr. M Ramabai**, Professor and HoD, Department of ET, MGIT, for all the timely support and valuable suggestions during the period of project.

I am extremely thankful to, **Mrs. Mahammad Nazini**, Assistant Professor, and **Mr. Mallela Srikanth**, Associate Professor, Department of ET, MGIT, mini project coordinators for their encouragement and support throughout the project.

I am extremely thankful and indebted to my internal guide **Dr. D. Koteswara Rao**, Assistant Professor, Department of ET, for his constant guidance, encouragement and moral support throughout the project.

Finally, I would also like to thank all the faculty and staff of ET Department who helped me directly or indirectly, for completing this project.

C SAI SIRISHA (21261A6613)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

XAI	Explainable Artificial Intelligence
LIME	Local Interpretable Model-Agnostic Explanations
SHAP	Shapely Additive Explanations
ANN	Artificial Neural Network
DNN	Deep Neural Network

ABSTRACT

Ransomware attacks have recently increased in number and sophistication, posing serious hazards to individuals and organizations. While conventional signature-based anti-ransomware systems are effective in the detection of known threats, they struggle to identify new ransomware samples. Numerous academics have concentrated on studying the behaviour and operations of executables in order to overcome this constraint. As a result of this dynamic analysis procedure, a number of dynamic-based attributes that provide diverse insights about the behaviour of the executable surface. This project uses features such as ImageBase, VersionInformationSize, SubSystem, Legitimate etc. These features are trained to the ANN(Artificial Neural Network) model. In addition, we provide an Explainable Artificial Intelligence (XAI) supported ransomware detection model, enhanced with LIME (Local Interpretable Model-agnostic Explanations) and SHAP (Shapley Additive explanations) to provide local and global explanation for detection. The model aims to accurately classify and predict ransomware attacks, providing insights into feature importance and decision-making processes achieving an accuracy of 98.55%.

CHAPTER-1

INTRODUCTION

Ransomware, a sort of malicious software that encrypts a victim's files and demands a ransom for their decryption, initially appeared in the late 1980s. The "AIDS Trojan" or "PC Cyborg" from 1989 is one of the earliest recorded instances. Ransomware has grown throughout time, becoming increasingly complex and ubiquitous. Major outbreaks in 2017, such as WannaCry and NotPetya, emphasised the global threat, prompting substantial advances in detection and protection measures.

1.1Introduction to the project

Ransomware detection is a very critical area of cybersecurity, given the increased frequency and complexity of attacks. Detection was initially based on signature-based methods, which matched known malware patterns from databases. These approaches are effective against known threats but fail to identify novel or polymorphic ransomware that changes its code to avoid detection. Modern methods include machine learning (ML) and artificial intelligence (AI), analyzing behavioral patterns to proactively identify ransomware-like activities. Techniques like Explainable AI (XAI), LIME (Local Interpretable Model-agnostic Explanations), and SHAP (SHapley Additive Explanations) enhance transparency, enabling the user to understand model decisions and builds trust in these advanced detection systems.

Emerging new research in ransomware detection is dynamic analysis, within which malware executes in controlled surroundings such as a sandbox, not directly affected by the limitations created by static analysis, reliant on source code examination which is prone to obfuscation techniques. After undergoing this phase of dynamic analysis, identifying ransomware becomes a straightforward case of classification, usually done between two classes: ransomware and benign. Deep learning and machine learning dominate this field, dividing the detection technologies into real-time monitors and multi-feature analysis. Real-time monitors have the capability of detecting abnormal system behavior while missing important sequential or cross-perspective relationships, so there is room for further improvement in detection methodology.

1.1.1What is Ransomware?

Ransomware can refer to any software that hinders a user's access to his or her files or the operating system and demands a sum of money from the victim to regain access. It is often mentioned in Bitcoins to ensure that the attacker remains anonymous when seeking for a ransom. Ransomware is defined as originating from phishing emails, website activities or taking advantage of weaknesses in applications. The decision on paying the ransom does not guarantee the data to be recovered because the attackers may not release the decryption key or may ask for more money.

1.1.2How does Ransomware occur?

More specifically ransomware mostly happens through several methods that are as follows: they are all means that target certain weaknesses of human values as well as of the programs.

Here's a detailed look at the primary ways ransomware infections occur: Here's a detailed look at the primary ways ransomware infections occur:

Phishing Emails

Spoofing, as one of the most common forms of deceptive emails, directs a severe threat at people as the messages look like they have been sent by contacts, co-workers, banks, credit unions, or well-recognized companies. These are often accompanied with links to a webpage or contain an attachment which is infected. Opening these attachments or links can result to transferring of ransomware to the computer or the desired device.

Malicious Websites

Having the HTML code injected is another way as users access websites that contain injected code or contain malicious codes that take advantage of the browser or the plugins. Additionally, The WebAds present on reputable sites can make the visitor redirect to sites that then 'deliver' ransomware.

Removable Media and Network Shares

Ransomware can be spread through infected USB sticks or other similar mediums such as CDs, DVDs, and so forth that can be connected to a computer. Ransomware in a network spreads through shared network drives that cause the spread of the virus to other systems.

Software Downloads and Piracy

Cracked software, obtained by downloading or acquiring it from illegitimate sources, poses a significant risk of infecting your system with ransomware. This software is often bundled with malware and may be easily downloaded from fake websites that resemble official ones or as additional files to software from seemingly reliable but unsafe sources.

1.1.3Process of a Ransomware Attack

The entry point of ransomware is when it penetrates and installs itself on the client's computer through any of the aforementioned techniques. Inside the system, the ransomware launches its task and can easily evade usual security system protocols and procedures. It then either encrypts the data so they cannot access it or completely locks the system, and then asks for a ransom once they provide you with the decryption key. A communication is usually made to the victim demanding a certain amount of money in exchange for the release of the victim's information or to correct a malicious change that the hacker made on the victim's system. Specific information is given about the manner of paying the amount, sometimes with an indication of anonymity such as the use of Bitcoins.

1.1.4 Explainable AI (XAI)

Explainable AI is centered on the creation of an AI system or model whose decisions and processes are intelligible to humans. This means improving the explanatory components of AI to make it possible for users to understand its decision-making process. Such transparency would help determine accountability and responsibility when ethical dilemmas arise from usage of AI.

XAI methods improve trust, understanding, and accountability in AI systems. As AI becomes integral to fields like medicine, banking, and justice, it is essential to clarify how these systems arrive at their conclusions. Providing natural language explanations for AI outcomes not only builds user trust but also ensures compliance with legal requirements for fairness and accountability.

1.1.5 Machine Learning

Before delving deeper into how numerous strategies of machine learning work and the capabilities and drawbacks of each form, it is helpful to start with a clear understanding of the definition. While it is often regarded as a subfield of AI, machine learning was developed out of artificial intelligence research; specifically, the application of machine learning in data science is centered on the idea of utilizing it as a method to build data models.

Machine learning also involves developing models with the help of which the computer will be interpreting the data. The term 'learning' is used when parameters of these models are made to be adjustable for adjusting the value in conforming to the data it comes across. Once such models are learned from past data they gain the ability to learn new data. Discussing the philosophical aspect of comparing this version of learning as a mathematics based model as opposed to human learning, is a discussion for another time.

1.1.6 Features of Machine Learning

Currently, machine learning is one of the most promising technologies among the modern ones and it forms a massive influence on a vast amount of different fields and tasks. Its main features are:

Automated Learning: Artificial neural networks do not require programmers to set the exact details of how the machine will get to the answer. These learn from data and get better over time as new data is given to them or as they get used to the patterns and relationships in the data.

Data-Driven: Machine learning is data-driven which is an aspect that is well embraced in the technological advancement. The characteristics of the data collected inevitably affect the efficiency and effectiveness of the models generated.

Adaptability: It is possible to state that models can learn from new, unsuspected data. Due to that, they are always refined and developed as they continue to encounter more data in the culture.

1.1.7 Deep Learning

Artificial Neural Networks is an algorithmic design strongly inspired by the structure and functions of the brain. Deep learning is referred to this branch of machine learning. These neural networks need that huge amounts of data be gathered to train them on complicated tasks such as audio and picture recognition and natural language processing and game playing. Features of Deep Learning. Deep learning is a class of machine learning that uses neural network with more than three layers. Here are the key features of deep learning:

Automated Extraction: There are fundamentally one key benefit when it comes to the use of deep learning, and it is that feature selection is automate they do not have to look for features themselves which used to take a lot of time.

Deep Structures: Great depth learning models are those models that contain a large number of layers in the models; the HDMs are the most basic models among the DNNs. These layers allow the network to comprehend the information and details which are enclosed in the data.

Scalability: Having said this, deep learning can likely develop in to many large data sets and many large numbers of models. If shareholders have a strong computer and quality data at their disposal, it can execute a large number of activities effectively and promptly.

Non-linear Processing: Personally, there are tremendous differences between such deep learning models and simple linear models as the former could search for nonlinear relationships in the data with help of nonlinear activation functions such as sigmoid, tanh, and ReLU.

1.1.8 Libraries

The libraries used in the proposed method are as follows:

Pandas: It is a library intended for manipulation and analysis of data. In this case, it is used to read the dataset into a DataFrame for easier handling and preprocessing of data. Precisely, this dataset is Ransomware.csv.

NumPy: It supports big, multi-dimensional arrays and matrices, together with an enormous collection of mathematical functions to carry out operations on these arrays. Hence, this is applied in this case for array operations.

TensorFlow: This is a Google-developed open-source machine learning library. In the script, it was used for Artificial Neural Network model construction. In the script, a tf.keras.Sequential model with many dense layers and dropout layers is created to avoid overfitting. Afterwards, this model was trained with the Adam optimizer and a binary cross-entropy loss function for the binary classification task and was then trained and validated from the dataset.

Sklearn: It is a very basic, lightweight, and powerful library used in mining and for data analysis. Applied here for normalizing the data using the MinmaxScaler and division of data by train_test_split into training, validation, and test sets.

Matplotlib: It is a plotting library. Here it is used for training and validation accuracy and loss over the epochs plot to show how the model has performed.

LIME: A library for explaining the predictions of machine learning classifiers. It is used in this code for explaining how the ANN model makes its predictions. This uses LimeTabularExplainer to interpret a single instance of test data and shows which features contributed the most toward the model's prediction.

SHAP: SHAP is another library for explainability in machines. It explains the output of any machine learning model using Shapley values from cooperative game theory. Next, an application of SHAP is applied to compute the SHAP values for a subset of test data, providing an integrated all-in-one view of 'feature importance' and how each feature attribution impacts in prediction.

1.1.9 Algorithms

The algorithms used in the proposed method are as follows:

MinMaxScaler: The above specified algorithm is used in the feature normalization of a dataset. In other words, it standardizes each of the selected feature to some pre-designated standard range. This may be useful to feed inputs to a neural network without distorting them—primarily due to a certain dimension outcompeting the others because of its size.

Train-Test Split: This will enable the data set to be split into the training, validation and the test data set by this function known as the train_test_split from scikit-learn. This is to ensure that one can train a model on one regime/subset of the data, then validate or test to show how the one is able to generalize to other regimes/subsets.

Artificial Neural Network: An example of this is the obtaining and installation of the Keras API of TensorFlow. It will therefore consist of a couple of iceberg-like luxurious layers with interconnection in which ReLU shall be adopted by the model in the activation process and dropout in the over-learning prevention. A dense layer is used to extract more on the kinds of features of data provided while a dropout layer is used in reducing dependency and sensitivity of the neural network during training by setting some of the neurons to zero.

AdamOptimizer: It is a first order algorithm used in optimization and learning rates are adjusted in the processes of learning with the aim of minimizing the loss in the defined function. It was just the next step further from stochastic gradient descent in the respect that it determines the individual delta learning rates for the parameters. Inheritfrom two other extensions of stochastic gradient descent, it inherits their advantages: To the purpose of this work we used for optimization AdaGrad and RMSProp.

LIME: Local Interpretable Model-agnostic Explanations uses LIME to explain what the neural network is predicting. It applies an interpretable model, such as linear regression, to approximate the model locally in order to understand each feature's contribution in a specific prediction. This is useful to see how the model is making its decisions and which of the features are most influential.

SHAP: SHAP values are a unified measure of feature importance for any machine learning model. The SHAP algorithm explains the output of machine learning models via game theory, computing the contribution of each feature to the prediction. We will use KernelExplainer with background data summarized by k-means clustering in order to explain the model's predictions.

1.2Objective of the Project

- Integrating Explainable Artificial Intelligence (XAI) techniques, specifically LIME and SHAP to enhance the interpretability of the ransomware detection model.
- Providing both local and global explanations for the model's predictions to increase transparency.
- Create a machine learning model to reliably identify ransomware outbreaks.

The objective of this project is to use a machine learning-based model for detecting ransomware assaults, with high accuracy and dependability. In addition to detection, the project intends to address the black-box nature of machine learning models by combining explainable AI techniques, notably LIME (Local Interpretable Model-agnostic Explanations) and SHAP.

Use LIME (Local Interpretable Model-agnostic Explanations) to develop local explanations for specific model predictions. This aids in understanding how specific input features contribute to a specific prediction, hence making the model's behaviour more apparent on a per-case basis. SHAP (Shapley Additive explanations) is used to provide global explanations, demonstrating the total importance of each attribute across all forecasts. This method provides a comprehensive view of feature contributions, which aids in the detection of crucial signs of ransomware behaviour. Create visual representations of key components of the model, such as training loss over time, feature importance rankings, and individual prediction explanations from LIME and SHAP. These visual tools help stakeholders understand and trust the model.

1.3 Methodology adopted to satisfy the objective

The approach for constructing the ransomware detection model requires multiple stages, which are:

1.3.1 Data collection and preprocessing

Observing several features of the network traffic that points to ransomware, Wireshark is used to gather a ransomware dataset from the nodes in the network. As mentioned earlier, the main parameters that can be identified on the network traffic level are the originated port, the destination ports, the packet duration, and the sequence number. These elements are then put together to produce sequences that portray the functioning of links.

1.3.2. Feature Combination

The goal is to suppose the feature level of the feature matching process to obtain a process of feature integration, where the features are aggregated into a single vector or sequence covering all instances of the data. This entails integrating features like ImageBase, versionInformation, Size, SubSystem as well as the Legitimate features into a vector upon which distances for each instance in the set can be computed. This transformation puts the ransomware samples into the vector in the multi-dimensional space. When the feature space is rather large and includes tens or hundreds of thousands of features, then PCA with mean capturing most of the feature space variance is preferably used to use only important features.

1.3.3 ANN

ANN is composed of artificial neurons, which are referred to as units. These units are described in a set of oriented or stacked layers to form a complete Artificial Neural Network system. A layer can be contained in as few as one dozen and as many as millions because of the architecture of these deep neural networks must be designed to uncover the necessary features in the data set. Generally, Artificial Neural Network has an input layer; it also must contain an output layer and one or more hidden layers as shown in figure 1. Categorized broadly, the layers in a neural network are the input layer which takes in data from the outside world that the network is required to learn about or analyze. Next, this data goes through one or even several concealed layers, which transform the information obtained into data useful for the output layer. Last but not the least the output layer of the Artificial Neural Networks gives out an output or response in form of a response of ANNs to inputs given.

In most cases, the units in the neural network are connected layer by layer, that is, connections between layers are made. Of course, each relation has weights which define how one unit affects another unit in the given network. It means that during the transfers of data from one unit to the other the neural network learns more and more about the data and toward the end of the process an output from the output layer is given.

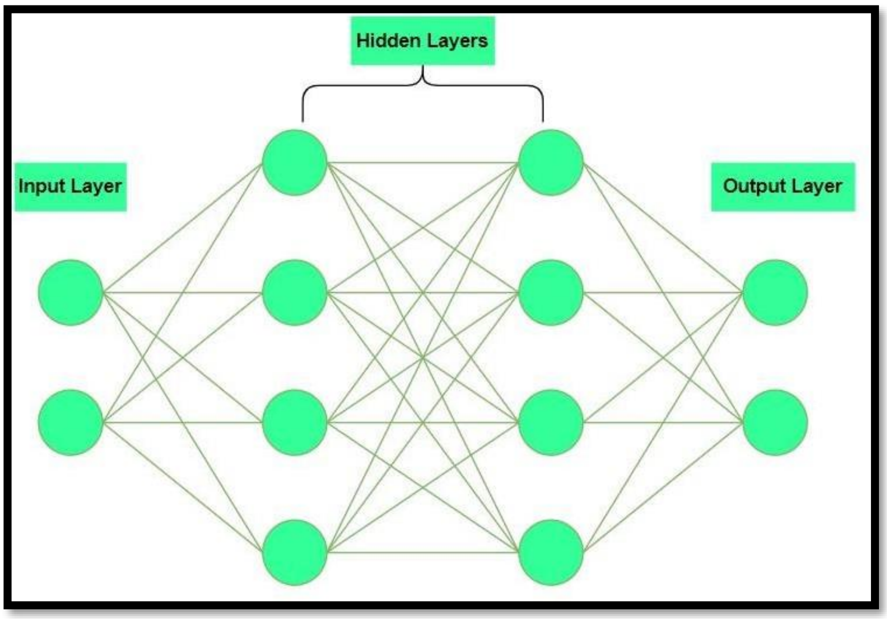


Figure 1 Artificial Neural Network

1.3.4 Operations of explainability

LIME (Local Interpretable Model-agnostic Explanations) is used to generate local explanations for individual forecasts. LIME facilitates comprehension of how particular features affect the model's choice in each case. Global explanations are provided via SHAP (Shapley Additive explanations), which illustrates the overall significance of each attribute across all predictions. Key markers of ransomware behaviour can be identified with the help of SHAP values, which provide an extensive perspective of feature contributions.

1.4 Architecture Diagram

The architecture diagram contains three phases as shown in figure 2. The first phase says about feature extraction it states the features are extracted from the dataset named ransomware dataset we have legitimate as the target variable. The second phase is Data pre-processing in which the features are combined in a particular sequence and then they are split into train and test data, here we use ANN (Artificial Neural Network) as the algorithm we use for predicting the accuracy of ransomware detection. In the third phase, we perform data explainability which means it specifies the reason behind the output through XAI operations using LIME and SHAP. The LIME operation provides local information for all individual predictions and SHAP operations provide global information to demonstrate the overall importance of each attribute.

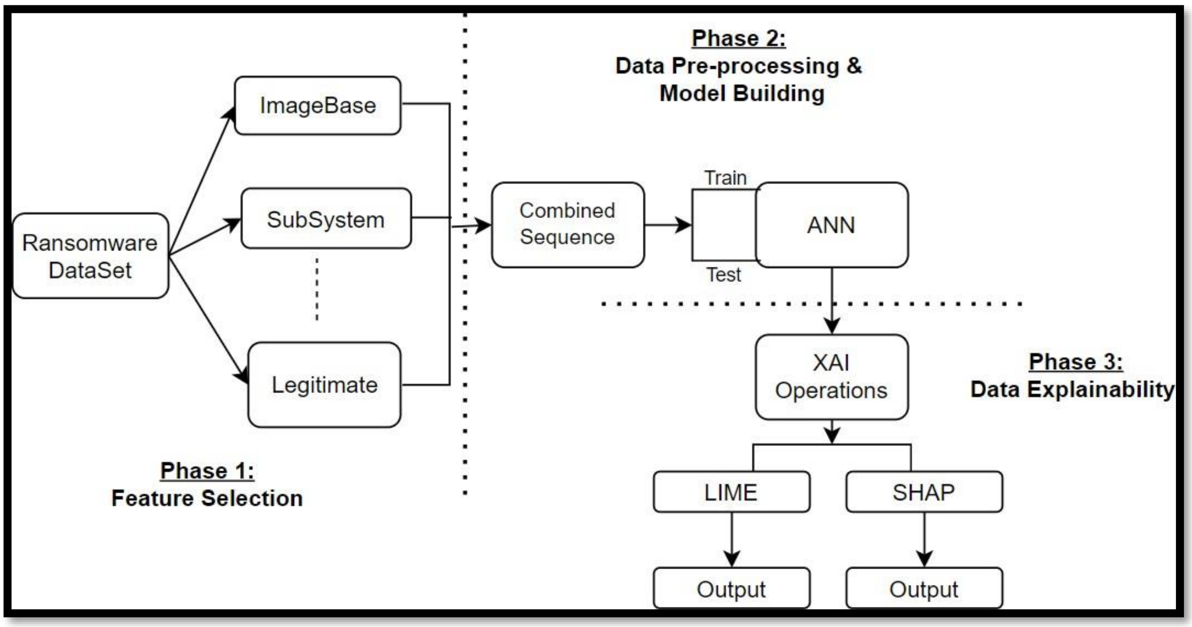


Figure 2 Architecture Diagram

CHAPTER-2

LITERATURE SURVEY

The chapter comprises existing approaches and the drawbacks of existing approaches

2.1 Existing approaches

S. Gulmez, A. Gorgulu Karkisim and I. Sogukpinar [1] have proposed a paper called "XRan: Original Research: A hybrid model of deep learning for ransomware detection using dynamic analysis with explainable artificial intelligence." This work aims at developing XRan, an innovative deep learning-based system for ransomware detection by means of dynamic analysis with ascertainable improvement in interpretability through XAI models. The methods that the researchers employed included dynamic analytical approaches to the training of Xavier's XRan model and specifically, the XAI interpretable models – LIME and SHAP – toexplain the importance of features and abstraction level in the model.

Kavitha Kunku, ANK Zaman, and Kaushik Roy [2] published a work labelled "Ransomware Detection and Classification Using Machine Learning." This report elaborates on the primarily malevolent ramifications of ransomware in the field of cybersecurity since ransomware is bent on causing massive disruption to computer systems, servers, and mobile and web applications across various industries. The proposed research work will therefore be focusing on the development of a machine learning system which will be able to assess ransomware attacks as and when they occur and at the same time categorize them.

In 2021, **Loubna Moujoud, Abdelhamid Belmekki, Meryeme Ayache**, and others proposed the following study: "A state-of-the-art Survey on Ransomware Detection Using Machine Learning & Deep Learning". A Cutting-Edge Survey on Deep Learning and Machine Learning for Ransomware Identification [3]. This study classifies various ransomware detection methods and assesses how feasible they are in real-world scenarios. They discuss the advantages of these technologies for detection, such as their capacity to recognize ransomware that is both new and old, as well as unexpected threats, which the traditional signature-based detection method occasionally fails to detect.

Wael M. S. Yafooz, Hashem Alolofi, Ghilan Al-Madhagy Taufiq-Hail, Abdel-Hamid M. Emara, Ahmed Abdel-Wahab, and Ramadhan A. M. Alsaidi have developed a paper on the issue of "Ransomware Detection Using Machine and Deep Learning Approaches" [4]. The essay examines ransomware, a malicious software that locks victims out, which is one of the worst types of software employed in cyberattacks.

Yahya Nasser, Fadi Dornaika, Victor Gil-Garcia, and Chaker Larabi authored a different paper titled "Android Ransomware Detection Using Supervised Machine Learning Techniques Based on Traffic Analysis" [5]. This paper focuses on the type of analysis that looks for malware on Android devices using ML and DL. It should cover a Kaggle dataset containing various types of Android ransomware in addition to safe traffic. The paper identifies traffic factors that are critical for ransomware detection and proposes an ensemble model composedof decision tree, support vector machine, and k-nearest neighbor models to enhance traffic detection.

Farnoush Manavi and Ali Hamzeh released an article titled "Ransomware Detection Based onPE Header Using Convolutional Neural Networks" [6]. This study aims to present a new approach to early ransomware detection by using PE headers, an essential part of all executable files. Using data mining, this technique takes features out of the PE header and turns them into grayscale images that CNN analyzes.

Daniel Morato, Eduardo Berrueta, Eduardo Magaña, Mikel Izal with others developed a paper with the title "Crypto-ransomware detection using machine learning models in file-sharing network scenario with encrypted traffic" [7]. This enriched review discusses different approaches in ransomware identification and more particularly how machine learning approaches can be used in file-sharing networks with encrypted communications.

Rawshan Ara Mowri, Madhuri Siddula, and Kaushik Roy [8] presented one paper which is "Application of Explainable Machine Learning in Detecting and Classifying Ransomware Families Based on API Call Analysis." This paper presents an original method for ransomware identification and categorization through data mining and XAI. It employs an automated web crawler for ransomware samples collection, analyzes the API call frequency, and constructs a relevant corpus through feature engineering. Six supervised machine learning models are tested and logistic regression comes up with better test accuracy of 99.15%.

Ahmed Dib, Sabri Ghazi, and Mohamed Said Mehdi [9] developed the paper "Ransomware Attack Detection based on Pertinent System Calls Using Machine Learning Techniques." Thispaper identified the following research gaps: it provides a set of procedures for system calls collection from ransomware and benign applications, including important preprocessing techniques to reduce data dimensionality and select prominent features.

Subash Poudyal, Dipankar Dasgupta [10] developed the paper “AI-Powered Ransomware Detection Framework.” The paper holds the prospect of introducing a novel approach to ransomware classification and recognition via data mining and explainable artificial intelligence (XAI). The approach used is a combination of the static and dynamic reverse engineering where both are used to isolate the malware. It employs multi- level analysis and natural language processing (NLP) for the extraction of behavioral features.

Umme Zahoora, Asifullah Khan, Muttukrishnan Rajarajan with others developed a research paper title “Ransomware detection using deep learning based unsupervised feature extractionand a cost sensitive Pareto Ensemble classifier” [11]. This paper holds developing the core features are obtained utilizing CFH before training various base learners on the data reduced features.

Table 1 states the summary of the existing approaches specifying the objective, methodology, significance and drawbacks of the approaches

Table 1: Summary of Existing Approaches

SNo	Objective	Methodology	Significance	Drawbacks
[1]	A Method to develop deep learning based ransomdetection usingdynamic analysis and xai.	It explores dynamic analysis techniques to train xtrain model,and they Lime and Shapto explain featureimportance.	The usage of Limeand shap increases interpreability of Xran decision,xran scored high accuracy in detecting ransomware across various datasets.	High computational Complexity and lacking of diverse datasets.

[2]	A method to quickly identifying and neutralizing the ransomware threats.	Using XGBoost classifier and Random Forest algorithms to detect ransomware behaviour.	By using XGboost classifier and Random Forest areeffective in identifying and categorizing ransomware attacks.	The effectiveness of these algorithms depends highly on quality and quantity of data.
[3]	A method to compare various machinelearning and deep learning techniques for ransomware detection by using dynamically.	The methods used are deep convolutional neural networks, LSTM networks andAutoencoder.	This paper highlights effectiveness of using advanced techniques in detecting and identifying types of ransomware.	Dataset limitations, computational complexities and unsatisfactory experiment results.
[4]	A method to detect malicious URLs associated with ransomware attacks.	The hyperparameters tooptimize performanceof classification methods.	This uses ML and deep Learning techniques effectively by accurately identifying and blocking malicious ransomwarethreats	High computational complexity.

[5]	To address data imbalance issues and to use ensemble learning and deep learning models for improved detection.	It explores using dataset, pre-preprocessing it to extract the features, training models and evaluating the models accuracy and other metrics.	By analyzing the decision tree out performed other models with accuracy 97.24%, SVM excelled with 100% recallrate, addressing data imbalance improves models performance.	The primary drawback of this project is which supports only lightweight and efficient models such as android which limits device performance.
[6]	To develop a static method for ransomware detection based on feature extraction from PE header of executable files.	This method involves converting executable files into images extract more effective features then CNN to extract the images and classify them.	The proposed methods achieves higher accuracy of 93.3% in detecting ransomware by extracting features from PE header to executable files.	The method have limitations in identifying and evolving forms of ransom threats have different structural characteristics, CNN model heavily depends quality and diversity of dataset.

CHAPTER 3
Proposed Method

3.1 Problem statement and Objectives of the project work:

A problem statement assists a learner in developing the consensus of what part of a given concern needs to be solved, what needs to be achieved, and in what areas the learner has to work. Project goals are among the most critical components of any project because they define what you intend to accomplish at the end of a project.

3.1.1 Problem Statement:

This research addresses a significant security issue posed by machine learning models, which, when properly built, may detect ransomware quickly. First and foremost, it must offer a high degree of detection by developing an algorithm that allows separating ransomware files from genuine files with virtually no (or extremely few) false positives or false negatives. When incorporating them into real-world cybersecurity systems, this becomes crucial to preventing scenarios in which numerous false alarms are set off or significant threats go undetected entirely.

The second step is feature extraction, which involves choosing important features from the project's feature set to use as training data for the model. This involves extracting features from the provided data or feature selection is the process of identifying, from the multitude of features observed in the data, the most pertinent features that are important to the classification task. A model's future efficacy is contingent upon the selection of features that are extracted throughout the construction process.

Thirdly, the project's explanation of the models it utilizes is its main goal. By using Intelligent AI, which makes use of methods like SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations), you ought to be able to follow the model's decision-making process. These methods provide proofs at the local and global levels, which aids a cybersecurity specialist in understanding the rationale behind a model's particular conclusion. Lastly, the project's main objective is to apply the suggested approach in real time.

3.1.2 Objectives

- Integrating Explainable Artificial Intelligence (XAI) techniques, specifically LIME and SHAP to enhance the interpretability of the ransomware detection model.
- Providing both local and global explanations for the model's predictions to

increase transparency.

- Create a machine learning model to reliably identify ransomware outbreaks.

The objective of this research is to build a machine learning model that can predict ransomware outbreaks reliably and provide high interpretability. The model will detect methods and give logical explanations for its predictions using Intelligible AI techniques, specifically SHAP (SHapley Additive ExPlanations) and LIME (Local Interpretable Model-agnostic Explanations). These tools improve transparency and reliability by highlighting key features necessary for model assessments.

Various multiple machine learning strategies would be identified at the project level. From these, the most apt approach to detect ransomware would be identified, thus dividing the carefully curated set of data into two subset training and testing levels; depending on the fine knowledge of the dataset regarding a sample's composition either an unsupervised or even supervised training method is provided whereby the model aligns towards fine-grained knowledge for sample composition.

LIME will be used for local explanation of predictions which illustrates how input elements affected particular outcomes, hence performing instance-by-instance analysis. In this manner, the model shall become more transparent to a human auditor under chosen conditions. SHAP provides a general overview by quantifying feature contribution to overall predictions as well as specific outputs through critical indicators for ransomware detection.

Visualization tools are integral to the project. It includes tracking training loss to keep track of how the model is performing, feature importance charts for prioritizing influential attributes, and visual interpretations of predictions via LIME and SHAP. These tools can help cybersecurity experts gain better insights and confidence in the model's outcomes, making it easier to bridge the gap between technical complexities and actionable decision-making.

This project integrates robust assessment methods with state-of-the-art interpretability techniques and therefore delivers an efficient ransomware detection system while contributing to advancing intelligible AI research. This dual approach will make the model both accurate and credible, as the opaque decision-making in machine learning is a common challenge. The project will, in the end, arm cybersecurity professionals with clear analyses and actionable recommendations to fight against ransomware threats.

3.2 Explanation of Architecture Diagram

Architecture diagram is the act of creating visual representation of what comprises a given software system. Architecture in the context of a software system is mainly used to describe different functions as well as their instances with corresponding purposes and further interaction. The phases of architecture diagram are as follows:

Phase 1: Feature Selection

This phase tries to select only the most important and appropriate features from the ransomware dataset to improve the performance of the machine learning model.

Ransomware Dataset: The dataset would contain an array of attributes collected from a set of samples of ransomware.

File Metadata: File size, creation date, last date modified, and other properties.

Binary Features: These equations are derived from characteristics in the binary content of the file, which include explicit byte sequences, entropy measures, and other low-level features.

Behavioral Indicators: Observations on how the ransomware behaves at runtime, which among others include the network connections that it tries to establish, those files it touches or changes, and the system calls and various library functions that it invokes to be able to achieve its goal.

Identification and Selection of features:

ImageBase: It is normally the base address within memory where the executable file is loaded. This attribute is important to be analyzed and helps to understand how ransomware would execute in a system. This is used to specify the size of the version information within the .exe file, which becomes helpful for distinguishing between ransomware and clean software from different versions, as the versioning details can be analyzed.

It specifies the sub-system of the operating system that the executable requires to run. This would include, for example, Windows GUI or console. Knowing the targeted environment makes it possible to learn the context in which this ransomware should be executed.

Legitimate: It tells if the software is good. This binary label forms the basis of supervised learning where the model learns how to classify off these tagged examples.

Phase 2: Data Preprocessing

This entails formatting and integrating the desired functionalities so they make up a sequential structure adequate for training an artificial model. Important features include a base image, version info size, subsystem, and legitimacy status. All are put together in a serial structure ready for input data.

An ANN is a kind of machine learning model, inspired by the structure and function of the brain, which processes data through multilayered networks. It identifies patterns within input data. During training, the ANN adjusts its internal parameters, known as weights, to minimize the difference between its predictions and actual labels, which are legitimate or non-legitimate. Multiple passes over the data, in epochs, refine the accuracy of the model.

Phase 3: Data Explainability

Apply explainable AI techniques so that decisions of the trained ANN model are interpreted and understood, making its predictions transparent and trustworthy.

XAI Operations (Explainable AI): The ANN model is considered to have black-box behavior due to which it can be interpreted with Explainable AI techniques. Explainability in the model dictates how important it is to make decisions during deployment.

LIME(Local Interpretable Model-agnostic Explanations): It explains individual predictions – given input data – by approximating the complex model locally with an interpretable model. LIME generates explanations by perturbing the input data and observing the changes in the model's output.

Output: The output of LIME is essentially explanations for individual predictions, showing in a manner of speaking which features were most influential in making a particular decision—it may result that a high value of, e.g., the `ImageBase` feature contributed heavily to marking a file as ransomware. SHAP (SHapley Additive exPlanations)

SHAP: SHAP assigns an importance value to each feature for a particular prediction by Shapley values from cooperative game theory.

Output: The SHAP output is the contribution of each feature to the predictions of the model. It tries to show an aggregate understanding regarding the impact of the different features on decisions taken by the model. Example can be of "VersionInformationSize" being the major positive influencer into prediction of Ransomware in some particular sample.

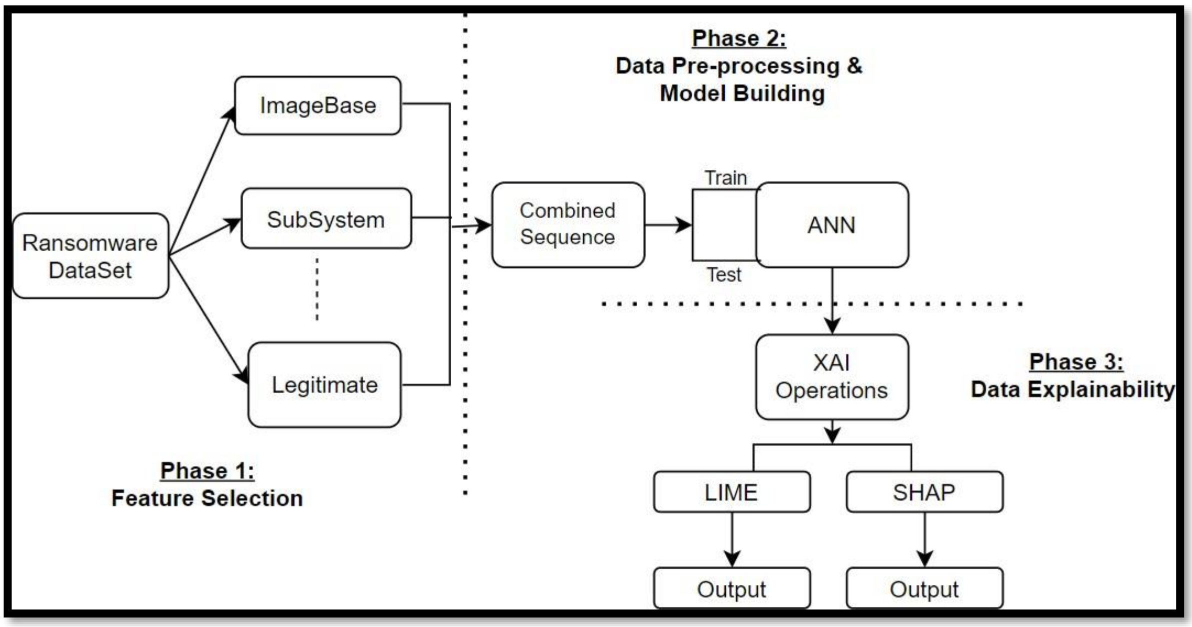


Figure 3 Architecture Diagram

3.2.2 Modules - Connectivity diagram:

Dataset: A dataset can also be stated as an orderly arranged body of data that constitute a database records may compose the rows while an attribute or a field may form columns. In the context of ML, datasets are, in fact, significant because these collections contain the data used in the training or testing of the model. Modeling should be done using dataset that is reflective of problem domain and to be able to do this it must be able to predict unseen set of data.

Feature Selection: Feature selection can be defined of the process of feature selection, which is a core machine learning, where selection of the right features from a provided list is done. Furthermore, when dwelling on such considerable traits as such, it helps advance the size of improvements and reliability together and likewise keeps away from over-learning. On the topic of overfitting, it occurs when a model learns noise rather than making a proper and direct map of the training data and cannot generalize in a new environment.

Data Preprocessing: Data pre-processing encompasses procedures, which are inevitable to prepare and format the data in a way that will be suitable for use/analysis in a model of machine learning or otherwise. Some of the activities performed here include error corrections and feature preprocessing which is feature cleaning options that include handling of features with errors and missing values besides scaling which is the process of bringing feature values to the nearest range. Cleaning data make the data more relevant thus expanding the accuracy and efficacy of the machine learning model.

ANNs, or artificial neural networks: Artificial neural networks are computer-controlled mathematical models is adapted from biological neural networks seen in the human brain. In order to facilitate pattern recognition and learning, these devices are fundamentally composed of layers of artificial neurons, or nodes, connected in some manner.

LIME, which stands for Local Interpretable Model-agnostic Explanations, is an acronym for the previously mentioned idea. Its goal was to provide a thorough explanation of the predictions generated by a specific machine learning model. This is achieved by comparing or calculating the impacts of the significant variables on the research variable's value on the intricate model.

SHAP (SHapley Additive exPlanations) is a tool for teaching learned patterns to users operating at levels above machine learning models. SHAP, which was initially inspired by concepts from loose cooperative game theory, enables an attribute to be given a specific value that represents its sensitivity to predictions made by a model.

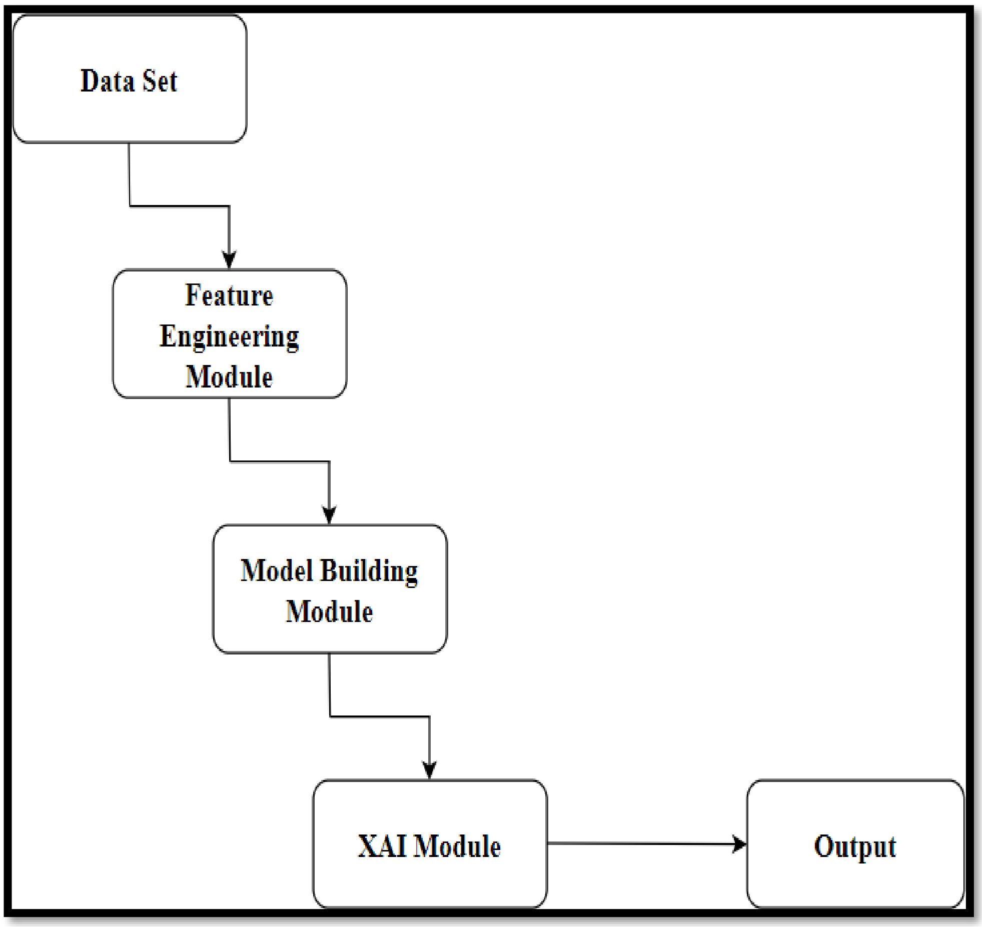


Figure 4 Module Connectivity Diagram

3.2.1 Software and Hardware requirements

To implement the proposed method the following software and hardware are required.

Software Requirements:

Software product	Specifications	Purpose
IDE Integration	VS Code Extension/ Jupyter Notebooks	A web-based interactive development environment
Version Control	Git and GitHub	GitHub is a web-based platform that uses Git for version control
Programming Language	Python	it is a versatile, high-level programming language
Libraries	TensorFlow, scikit-learn, Keras, pandas, NumPy, LIME, SHAP	Provides a comprehensive ecosystem environment

Table 2 Software Requirements

Hardware Requirements:

Hardware Device	Specification	Purpose
Operating System	Windows 10/Windows 11	Provides a user-friendly environment for personal and professional use.
CPU	Intel core i5 or equivalent	Provides the central processing power to a computer
RAM	Minimum of 8GB	Provides temporary storage for data that the CPU needs to access quickly.
Hard Disk	256GB	Provides long-term storage of data.

Table 3 Hardware Requirements

3.3 Modules and its Description

A module is a collection of source files and build settings that let you divide your project into discrete units of functionality.

3.3.1 Data Pre-processing

The stages of data pre-processing module are as follows:

Loading the Dataset: A Pandas Data Frame is a primary structure for working with data in Python and is used in almost every data science project. Pandas is used to analyze a CSV file which is structured data and this package is very good at handling such data. The Data Frame is a convenient data structure to work with after the data has been loaded into it, and providesa good interface for data exploration.

Selecting Features and Target: In data science, one of the most vital and fundamental aspects is to find which features (independent variables) or which variable (y, dependent variable) should be predicted. The independent variables are chosen out of all the available variables according to the nature of the specific problem being solved, while the dependent variable is the variable whose values are modeled by the data.

Normalizing the Features: Standardization might also be useful in making sure that all the features are on the same scale, and this is important because if some features are very large in comparison to other features it is very easy for the large features to control the learning process in a model. The MinMaxScaler function from scikit-learn in this case applies the transformation to the feature values into the range of 0 to 1.

Data Splitting: The data set is now separated into training and test data sets in order to precisely evaluate the effectiveness of a trained machine learning model. The estimating set is made up of a portion of the total data that is utilized to provide input to the model during training in order to make corrections or enhancements. The train_test_split function from scikit-learn makes this split simple.

3.3.2 Model Building

The stages of model building module are as follows:

Building Artificial Neural Network (ANN): ANNs are adaptive models that mimic the structural aspect of the human brain and are widely used in different fields. As for the architecture of ANN in this module, the dense (fully connected layers) is used in the layers.

Compiling the Model: The first set of parameters when compiling the model are three, the Adam optimizer is chosen because when used jointly with the learning rate, it allows for the optimization of the model parameters concerning the gradient of their position.

Training the Model: Training means passing the training data through the compiled model for a certain number of epochs (complete passes of the data set through the model) and in a certain batch size (the number of samples to pass through the model before updating the model's parameters).

3.3.3 Model Evaluation and Visualization

Model evaluation is the entire process by which an analyst uses some of the gathered metrics to determine how well the model is working. It has very vital to evaluate a model due to the fact that, it facilitates our judging of the performance of our model.

Visualizing Training Performance: It is critical to test the model after training it so that the model's learning curve can be evaluated and its strengths over epochs can be identified. The percentages of accuracy and loss, as well as epochs, are plotted using Matplotlib that is one of the most widely used data plotting tool in Python programming language. These plots provide insights into how well the model is learning: In effective learning, accuracy and loss should increase and decrease respectively, whereas diverging trends, such as low training loss and high testing loss point to overfitting, high training as well as testing loss point to underfitting.

3.4Analysis through UML diagrams

3.4.1 Class Diagram

Class diagrams are a sort of UML (Unified Modelling Language) diagram that is used in software engineering to visually express the structure and interactions of classes inside a system. UML is a standardised modelling language used to design and document software systems. UML is a standardised modelling language used to design and document software systems. They are an important element of the software development process.

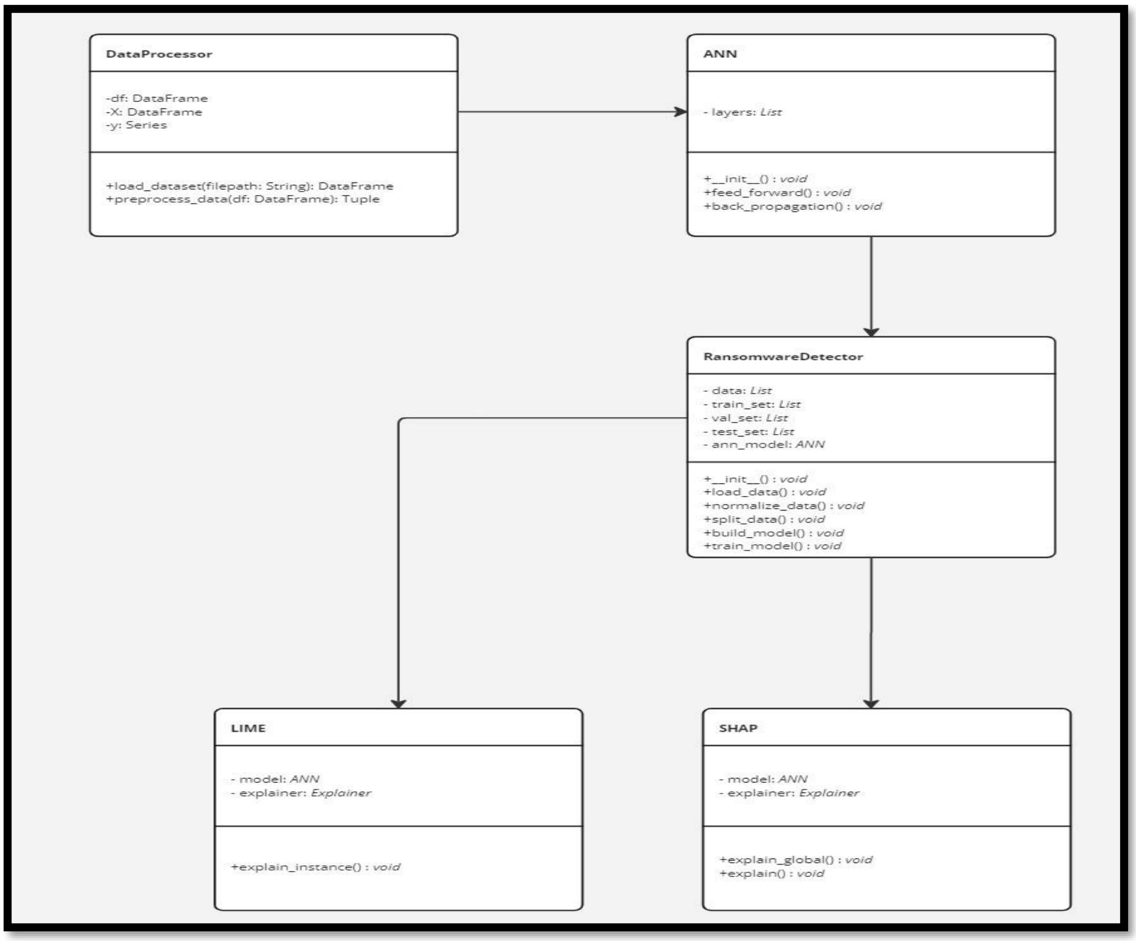


Figure 5 Class Diagram

3.4.2 Sequence Diagram

Building a robust system that can distinguish between content that is ransomware and non-ransomware would be necessary to complete this categorization assignment. In order to do this, the proposal combines two essential elements: It draws attention to two factors that are vital to the validity and repeatability of the classification model to be classified.

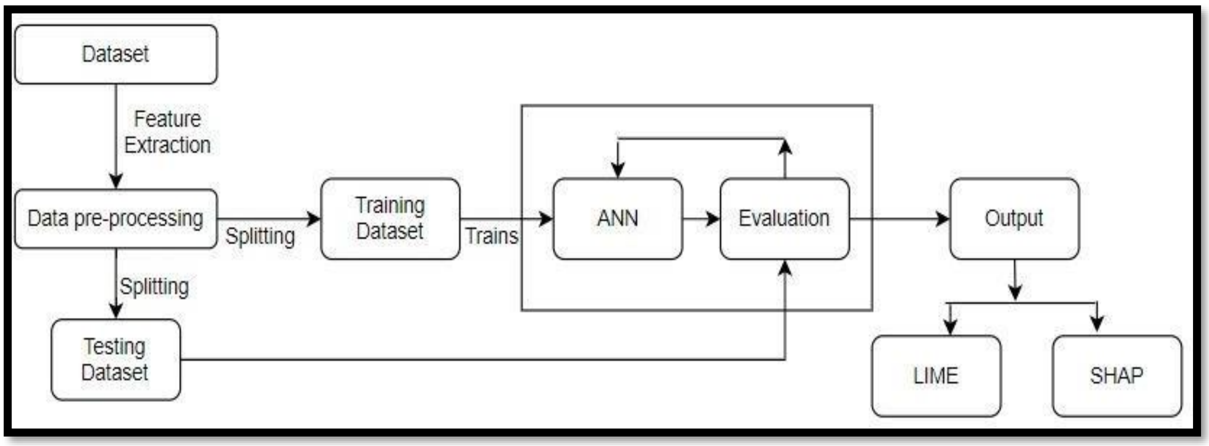


Figure 6 Sequence Diagram

3.4.3 Activity Diagram

The image figure 7 specifies the activity diagram of ransomware detection and finds the results of both XAI techniques LIME and SHAP.

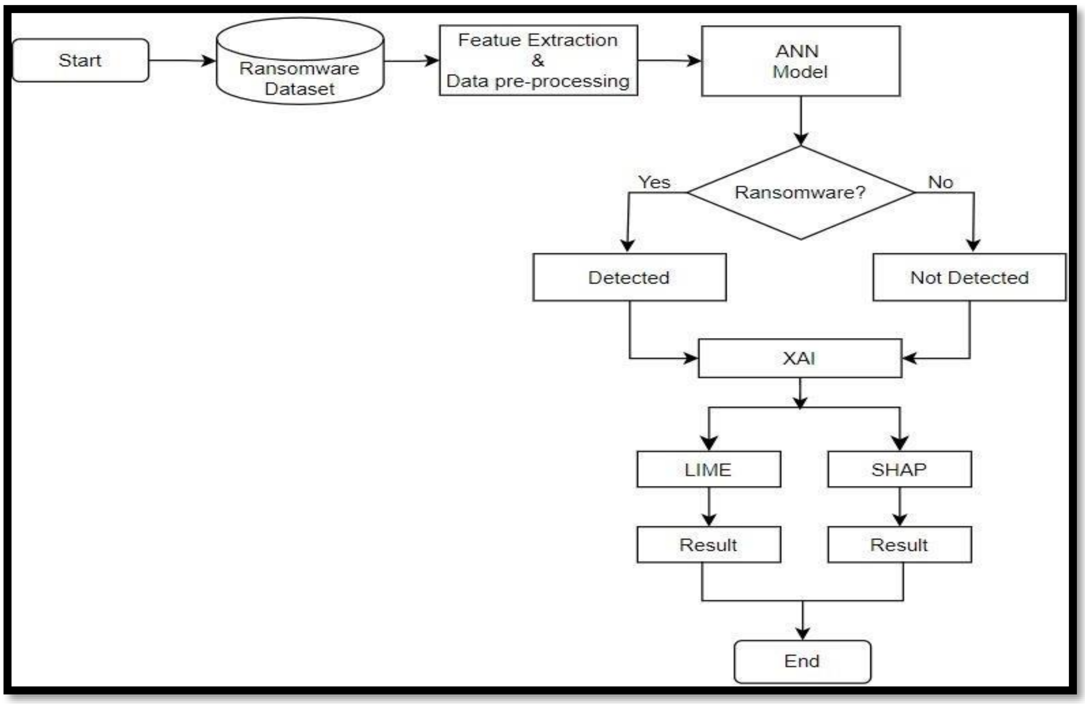


Figure 7 Activity Diagram

CHAPTER-4
Results and Discussion

4.1 Description of Dataset

The data set, titled "Ransomware," has been provided in CSV format. The specific data set is in the following format: There are 1,38, 047 columns in the.csv file, and each one contains unique file data. A. This dataset appears to be under the category of ransomware analysis and identification, which is an essential part of cybersecurity given the topics covered. The dataset presents each row of data as a string with various data fields merged into a single record, separated by the pipe (|) delimiter. Upon initially perusing the entries, it becomes evident that ever one is composed of a filename, a hash value, and many digits at the conclusion. These attributes most likely contain specifics about the file, such as its size or memory consumption, execution timings, and other file-related data. This is inferred from sample sets' form and organization, which are identified by a data delimitation on presented properties.

Drawing out particular columns to be utilized as features for the project's model is one of the following tasks. Characteristics, Size of initialized data, Major Subsystem version, Number of resources, Subsystem, Minimum entropy of resources, Base of data, ImageBase, Size of Version Information, Maximum entropy of sections, Major Operating System version, Minimum size of resources, Size of stack reserve, and so on are among its attributes. Overall dimensions image, as well as the major linker version. The genuine column is the target variable for the classification task, and its purpose is to determine whether a file is malicious or legitimate. Thus, the collection of traits that are thought to be helpful in categorization is identified in the current paper by choosing the project's columns. The project goes on to store the name of selected columns, probably, for subsequent use during the training and/or testing phase of the model. As the project applies TensorFlow, one would assume, the project developed an ANN to classify files based on given features. ANNs, especially, are suitable for this task since they emphasize more on capturing diverse patterns and interactions within the data and therefore they are useful in cybersecurity fields like ransomware detection.

4.2 Detailed Explanation of the experimental results

An experiment is a scientific test which is carried out in order to establish when certain things occur or the effect that is likely to occur as the result fixed treatment on a particular material.

4.2.1 Epochs vs Accuracy

This is done through the creation of a line graph that shows the learning progress of a machine learning model by using different epochs. Figure 8 shows, the value on the x-axis as the number of epochs or iterations over the whole training data set and the y-axis represents the accuracy or the percentage correct classification. The graph features two lines: The one on the left is usually in blue and is used to track training accuracy while the one on the right, usually in orange, follows validation accuracy. The training accuracy line portrays a progressive rise with the epochs crossing the test/validator accuracy line depicting the effective learning taking place on the training data. On the other hand, from training accuracy line, there is a smooth slope showing that the accuracy of a model against training data and the model has pits during certain epochs and peaks at other epochs which represents the validation accuracy line that tell us the performance of the model on new data and whether the model is overfitting or not.

Self training case follows the same trend as the base case, where both accuracies start at the same value and then as training goes on; there is a significant difference. Moreover, the low points are visible at the begining of the second epoch and then the accuracy starts to increase. Towards the latter phases of the training process, both of these accuracies levelling with one another, with the validation accuracy occasionally exceeding the training accuracy. This indicates that the model is overfitting because the epochs reach 20, while the validation loss is still decreasing, meaning the model is generalising well well to new data. The graph to the right represents the training and validation metrics showing the training accuracy increasing gradually graph while validation accuracy tends to be erratic and stabilizing in the highest level; These aspects justify the need to pay attention to the two metrics to enhance learning at each phase.

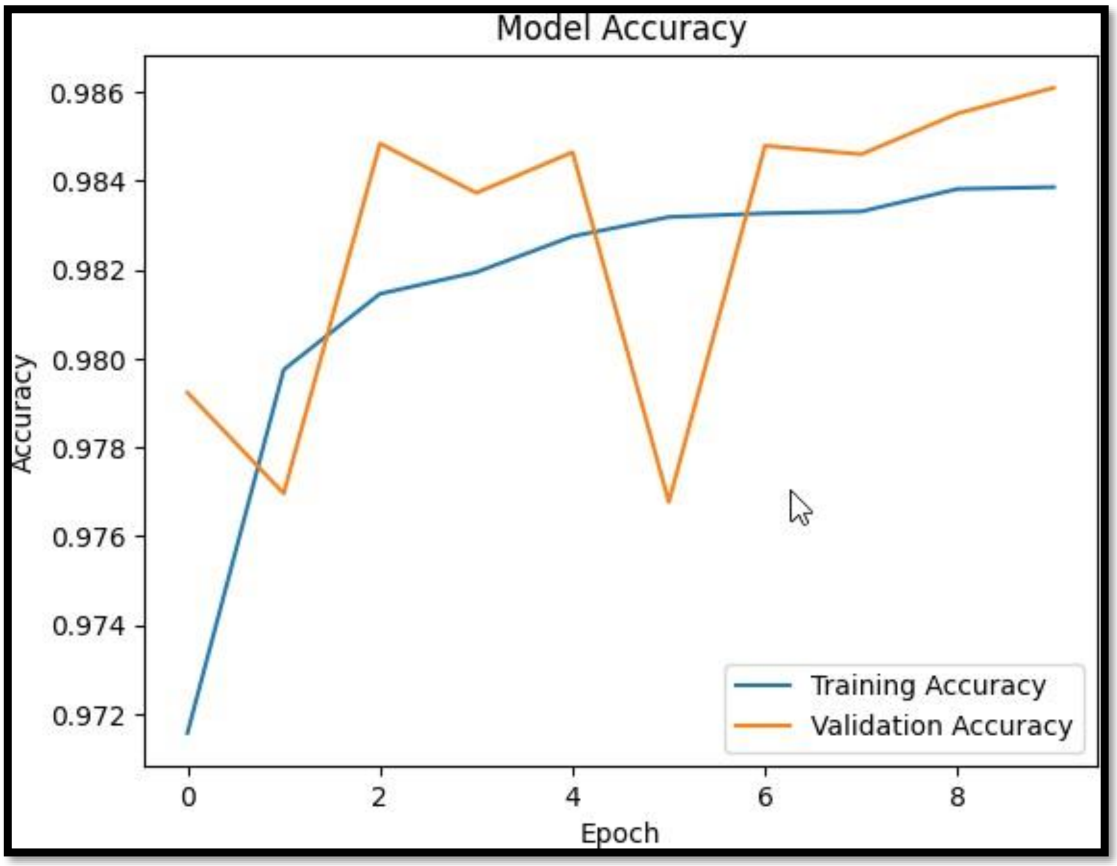


Figure 8 Epoch vs Accuracy

4.2.3 Epochs vs loss

The shown line graph defines the correlation between epochs and loss at the machine learning training. Figure 9 shows, the x-axis is the number of epochs starting from 0 to 9, while the y-axis represents a loss value within the approximately 0. 045 – 0. First of all, both the lines rise from certain loss values: 3. 8 and 4. 3, respectively, and then dramatically decline during the first epochs, thus indicating that the networks start to learn very quickly. After this period, the rate of loss reduction decreases slowly, and the fact that the models have started to reach the lowest error rate indicates that the process is slowing down. However, the orange line seems to oscillate more frequently than the blue line, which could be due to overfitting or have higher variance in that model or in the collected data.

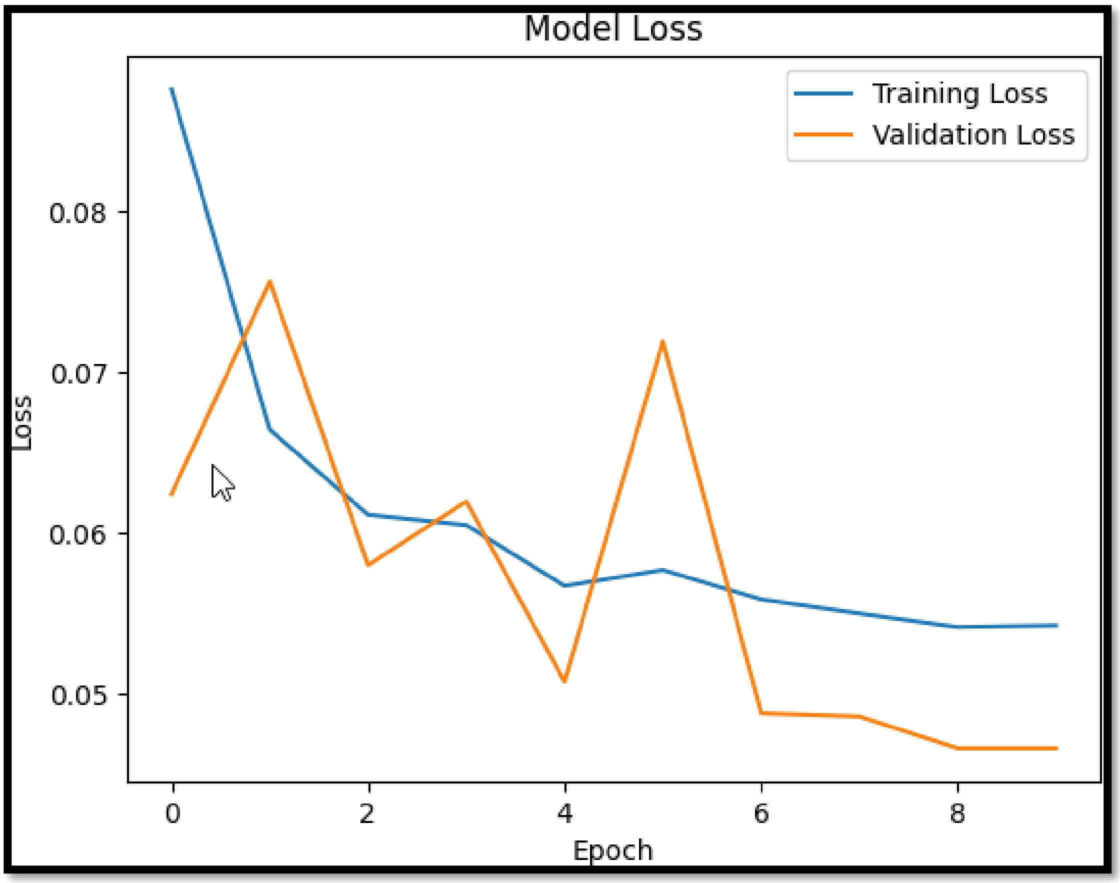
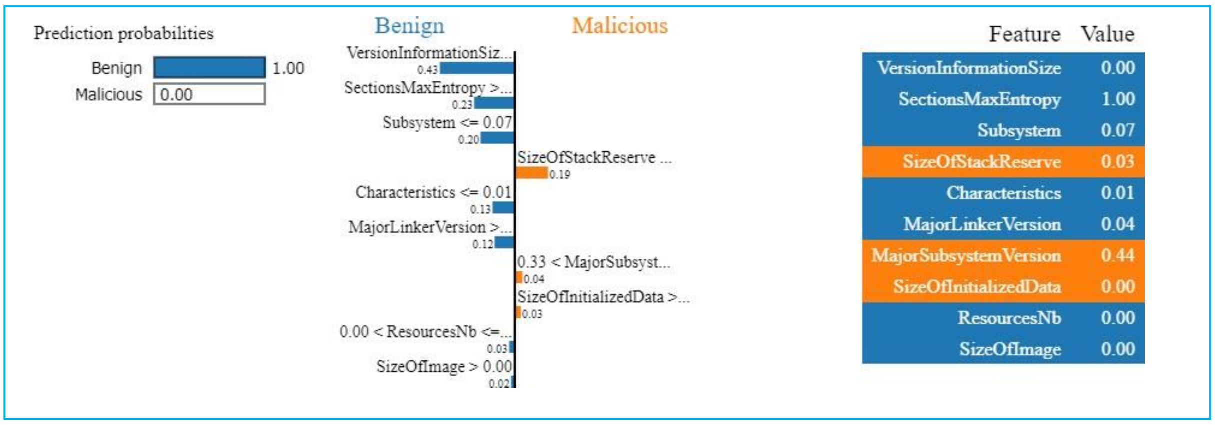


Figure 9 Epoch vs Loss

4.2.4 LIME

It allows offering detailed insights about the performance of a machine learning model and the importance of features that the model used to distinguish between benign and malicious entities. They specifically can be distinguished into three large sub-categories. In Figure 10, the far-left probability bar shows that the model gave a probability of 1 meaning it is fully blue that the entity is benign and 0 meaning that the bar is fully empty which indicates that the entity is not malicious. The second section identifies the importance of features utilized in predicting congestion and offers a breakdown of the results from the features. Benign CLASS: VIII The features which contribute towards the benign classification includes VersionInformationSize with contributing value of 0. 43, SectionsMaxEntropy, with a contributing value of 0. 25 and Subsystem with contributing value of 0. 07 as depicted in blue color. On the other hand, features promoting the malicious class identified with a feature name in orange: SizeOfStackReserve which contributes 0. 19 and MajorSubsystemVersion which contributes 0. 44 to the classification. In the right section, it gives the real values of these features including VersionInformationSize, SectionsMaxEntropy, Subsystem, and SizeOfStackReserve that are valued 0,00, 1,00, 0,07, 0,03 respectively. This comprehensive visualization provides a convenient way to explain how features and their different values now affect the model's prediction and how confident the model is regarding the benign classification, as well as the level of importance of each feature during the entire decision-making process.



Feature	Value
VersionInformationSize	0.00
SectionsMaxEntropy	1.00
Subsystem	0.07
SizeOfStackReserve	0.03
Characteristics	0.01
MajorLinkerVersion	0.04
MajorSubsystemVersion	0.44
SizeOfInitializedData	0.00
ResourcesNb	0.00
SizeOfImage	0.00

Figure 10 Result of Lime

4.2.5 SHAP

It looks like it reflects a kind of visual analysis called SHAP plot, the tool commonly used for explaining the results of machine learning on the basis of the features' contributions. Equation (8-3) Pivotal to the story is the base value, which is estimated to be around 0. This is 2878 in this case but generally, it represents the average of the estimated model output on an entire training dataset. This base value is subtracted or added depending on the degree to which the feature added to or subtracted from the learning process. Every feature in the SHAP plot is represented as an arrow as shown in figure 11, which characterizes the effect of the feature on the model's prediction regarding a certain instance. Such contributions can be exhibiting benefits or losses, though the former are highlighted pink and the latter are highlighted blue. The length of each arrow describes how strong the impact of the designated feature is and the position of each of the arrows indicates where the feature contributes to. This causes features that are positioned to the right of the base value to contribute to increase on the output of the model while those that are positioned to the left to decrease the output of the model.

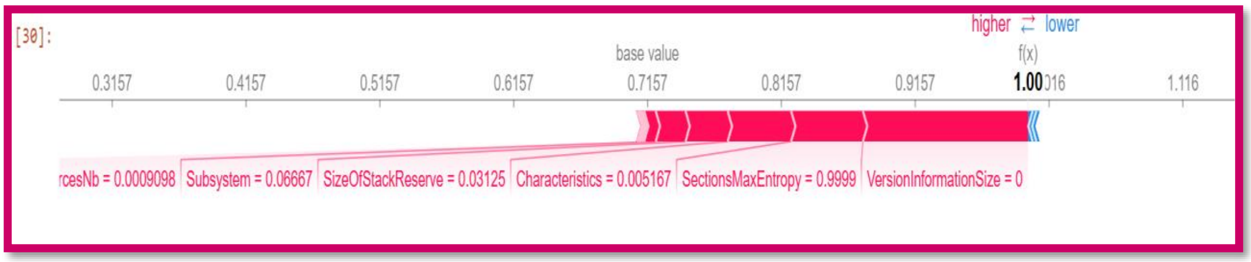


Figure 11 Result of SHAP

4.2.6 Accuracy Table

Model	**Validation Accuracy**	**Test Accuracy**
ANN (Proposed Method)	98.55%	98.40%
Decision Tree	97.24%	96.98%
KNN	88.73%	88.43%

Table 4 Accuracy Table

4.3 Significance of the proposed methods with it's advantages

This research approach involves both qualitative and quantitative views and code of approaches like literature studies, theoretical, focus group and content validated approaches.

Significance of the Methodology

The proposed methodology of ransomware classification and of explaining its functioning is useful for several reasons. This section elaborates the significance of every part of the methodology and describes why it is essential when it comes to cybersecurity or ransomware attacks.

Feature scaling with MinMaxScaler is a crucial preprocessing step in preparing data for model training, ensuring that all features contribute equally to the process. This addresses the problem of scale invariance, where the learning algorithm might emphasize features with larger scales rather than their relative importance. MinMaxScaler normalizes predictor variables to a range between 0 and 1, increasing the model's efficiency by ensuring each variable contributes equally.

Local Interpretable Model-agnostic Explanations (LIME) and SHapley Additive exPlanations (SHAP) are essential tools for interpreting model predictions. LIME provides localized explanations for specific predictions, enhancing understanding and trust in the model's decisions, particularly in cybersecurity, where it is crucial to understand why a file is deemed malicious or safe. SHAP, based on cooperative game theory, offers both global and local explanations of model predictions, ensuring that all features contribute proportionally and consistently. This dual-level interpretation helps identify the most influential characteristics affecting predictions, improving model interpretability.

Performance metrics like accuracy, precision, recall, and F1-score are vital for assessing model performance comprehensively, avoiding overemphasis on a single metric. For instance, high precision in a skewed dataset might result from the model consistently predicting the majority class. Combining specificity and sensitivity with F1-score provides a balanced evaluation of true and false positives/negatives, crucial for distinguishing between benign and malicious files. Cross-validation further enhances model robustness, minimizing overfitting and providing a better approximation of testing accuracy on unseen data.

Integration with real-time monitoring systems and continuous learning and adaptation are critical for effective ransomware detection and response. Real-time capabilities allow for prompt action against ransomware threats, crucial for protecting valuable data before encryption occurs. Continuous learning ensures the model stays updated with new ransomware variants and evolving threats, maintaining its functionality and efficiency. Enhanced usability, broader feature sets, and collaborative detection networks further improve model accuracy and robustness, while adhering to regulatory and compliance considerations ensures the model's practical applicability and legal compliance across industries.

CHAPTER-5
Conclusion and Future Enhancements

5.1. Conclusion

The ransomware detection project based on machine learning models focuses on several critical phases to enhance their effectiveness. Of these critical phases:

The ransomware detection project, and which centers around machine learning models, sees the need to highlight some of the critical phases in a bid to enhance their effectiveness. Feature extraction is another important phase where the practitioner has the task of choosing which features to extract from the dataset. The improvement of such a function not only increases reliability in terms of identifying ransomware and distinguishing it from legitimate files but also increases the model's resilience to different types of ransomware. Due to concentration on feature extraction, the project enhances the detection of the program and differentiates between the malicious and non-malicious software thus increasing the detection ratio.

Another equally important feature of the project is the incorporation of the use of explainability techniques such as SHAP (Shapley Additive Explanations) and LIME (Local Interpretable Model-agnostic Explanations). These methods explain the decision making of the model and offers solution at global and local level. This enhances trust in the detection process more especially in cybersecurity applications as it expounds on how a given choice is arrived at. The post deciduous of SHAP and LIME enable the comprehension and the interpretation of the simulation's outcome by exposing the explanatory variables, allowing cybersecurity experts to make the right decisions.

Finally, the project's focus on the real-time use of the developed application can be considered very relevant to practical reality. The incorporation of the developed model, in the operational cyber security systems helps in the identification of ransomware threats as they happen thus enhancing the formulation of proper responses. Therefore, the performance indicators of the project were as follows: a test accuracy of the logistic regression of 99. 5% where the total training set was the input to the committee of N-nearest neighbors and its output was split into 15% for training, validation and testing at 98. 55% and 98. It also reveales that the accuracy of the proposed method is around 92%, and the time of testing is 40% respectively, which enhances the efficiency and effectiveness of the proposed method. Other computed tools based on LIME and SHAP are also implemented to gain more understanding and help cybersecurity specialists in case of appropriate actions. Thus, the project has a rather narrow approach that covers both technological and practical sides of the problem, making it possible to apply the obtained results in practice.

5.2. Future Enhancements

To conclude the study, following enhancements for the improved performance and stability of the ransomware detection framework have been recommended. The integration capability with the real-time monitoring systems is mandatory for the efficient identification and response to threats. This one entails how methods should be arranged for processing data feeds from different sources in real-time, thus achieving instantaneous identification and removal of ransomware threats. Continuous learning and adaptation also ought to be performed; the model ought to be retrained at set time intervals to the new samples and new kinds of ransomware. This is good because the tutoring/teaching process is adaptive; probabilistic models and distance education might be researched to incorporate the new data into models.

Thus, expanding the usage of ensemble methods and introducing superior algorithms is another significant opportunity. The incorporation of such complex models as gradient boosting and, especially, deep learning will improve the detectors' performance and robustness. Analysis of the separate models' performance, as well as the possible formation of the best ensembles, is also required. Also, updating the features and using network traffic, behavioral analysis, and system logs to enhance the model's stability will do wonders. Another is operations and the application of the methods of feature engineering is also still a suitable way of extending the feature set. Optimizing the user interfaces and visualization instruments, including developing clear graphical user interfaces for SHAP and LIME explanation, will help cybersecurity staff utilize the models' predictions more effectively. Regular cooperation with other institutions, which are performing similar activities, following the legal requirements that apply to this matter and regular audits of the executed activities will help to maintain the effectiveness and compliance with the recognized standards of the presented framework.

CHAPTER-6
APPENDICES

```
import tensorflow as tf #ANN's
import pandas as pd #dataset
import numpy as np #arrays
df = pd.read_csv("Ransomware.csv",sep="|")#Fileread

X=df[['ImageBase','VersionInformationSize','SectionsMaxEntropy','MajorOperatingSystem
Version','ResourcesMinSize','SizeOfStackReserve','Characteristics','SizeOfInitializedData','
MajorSubsystemVersion','ResourcesNb','Subsystem','ResourcesMinEntropy','BaseOfData','S
izeOfImage','MajorLinkerVersion']]
y=df['legitimate']#features and target class.

cols = X.columns#storing the columns names
cols

from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler() #Normalizing
X = sc.fit_transform(X)# Fitting

from sklearn.model_selection import train_test_split#splitting
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5,
random_state=42)
X_train[:10]

model = tf.keras.Sequential([
   tf.keras.layers.Dense(1024, activation="relu"),
   tf.keras.layers.Dropout(0.2),
   tf.keras.layers.Dense(512, activation="relu"),
   tf.keras.layers.Dropout(0.2),
   tf.keras.layers.Dense(256, activation="relu"),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(128, activation="relu"),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(64, activation="relu"),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(32, activation="relu"),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(1, activation="sigmoid"),
])
model.compile(optimizer="adam",metrics=["accuracy"], loss="binary_crossentropy")#ANN
structure

model_history                                                    =
model.fit(X_train,y_train,epochs=10,batch_size=32,validation_data=(X_val,
y_val))#Model_train

predict=model.predict(X_test)
model.evaluate(X_test,y_test)
model_history.history#Accuracy_check
print(model_history.history["val_accuracy"])

import matplotlib.pyplot as plt#Plotting

plt.plot(model_history.history["accuracy"])
plt.plot(model_history.history["val_accuracy"])
plt.show()#Accuracy_plot

plt.plot(model_history.history["loss"])
plt.plot(model_history.history["val_loss"])
plt.show()#loss_plot

import lime
import lime.lime_tabular
import numpy as np
```

```
# Assuming X_train, X_test, y_train, y_test, cols, and model are already defined and the model
is trained
explainer = lime.lime_tabular.LimeTabularExplainer(
    training_data=np.array(X_train),
    feature_names=cols,
    class_names=['Benign', 'Malicious'],
    mode='classification'
)
# Adjusted predict function to ensure output is a 2D array with probabilities for both classes
def predict_proba(x):
    proba = model.predict(x)
    if proba.shape[1] == 1:
        proba = np.hstack((1 - proba, proba))
    return proba
# Define the predict function
predict_fn = predict_proba
# Verify the predict function
print("Prediction for a single instance:", predict_fn(np.array([X_test[0]])))
# Explain a single instance
exp = explainer.explain_instance(
    data_row=X_test[0],
    predict_fn=predict_fn
)
exp.show_in_notebook(show_all=False)

import shap
import numpy as np
# Define a function to return class probabilities
predict_proba_fn = lambda x: model.predict_proba(x)
# Summarize the background with shap.kmeans
background = shap.kmeans(X_train, 10)  # Adjust k as necessary
# Create a SHAP explainer with the summarized background
explainer = shap.KernelExplainer(predict_proba, background)
# Compute SHAP values for a subset of test data
```

```
shap_values = explainer.shap_values(X_test[:10])
import shap
background = shap.kmeans(X_train, 10)
explainer = shap.KernelExplainer(model.predict, background)
shap_values = explainer.shap_values(X_test[:10])
shap.force_plot(explainer.expected_value,shap_values.reshape((10,15))[0],
X_test[0],feature_names=['ImageBase','VersionInformationSize','SectionsMaxEntropy','Maj
orOperatingSystemVersion','ResourcesMinSize','SizeOfStackReserve','Characteristics','Size
OfInitializedData','MajorSubsystemVersion','ResourcesNb','Subsystem','ResourcesMinEntro
py','BaseOfData','SizeOfImage','MajorLinkerVersion'])
```

References

[1] A. Gulmez, S., A. Gorgulu Kakisim, and I. Sogukpinar, "XRan: Explainable deep learning-based ransomware detection using dynamic analysis," .

[2] K. Kunku, A. Zaman, and K. Roy, "Ransomware Detection and Classification using Machine Learning," arXiv:2311.16143v1 [cs.CR], Nov. 5, 2023. [Online]. Available: https://arxiv.org/pdf/2311.16143.pdf

[3] "A State-of-the-Art Survey on Ransomware Detection Using Machine Learning & Deep ResearchGate, [online]. Available: https://www.researchgate.net/publication/373415995_A_State-of-the-Art_Survey_on_Ransomware_Detection_using_Machine_Learning_and_Deep_Learning

[4] "Ransomware Detection using Machine and Deep Learning Approaches," International Journal of Advanced Computer Science and Applications, vol. 13, no. 11, 2022. [Online]. https://thesai.org/Downloads/Volume13No11/Paper_12-Ransomware_Detection_Using_Machine_and_Deep_Learning_Approaches.pdf

[5] A. Albin Ahmed, A. Shaahid, F. Alnasser, and D. Alqahtani, "Android Ransomware Detection Using Supervised Machine Learning Techniques Based on Traffic Analysis," arXiv, [online]. Available: https://arxiv.org

[6] I. M. Ogiriki, "Machine Learning Models Interpretability for Malware Detection Using Model Agnostic Language for Exploration and Explanation," Rowan University.

[7] E. Berrueta, D. Morato, E. Magaña, and M. Izala, "Crypto-Ransomware Detection Using Machine Learning Models in File-Sharing Network Scenarios with Encrypted Traffic," arXiv, [online]. Available: https://arxiv.org

[8] R. A. Mowri, M. Siddula, and K. Roy, "Application of Explainable Machine Learning in Detecting and Classifying Ransomware Families Based on API Call Analysis," *arXiv preprint arXiv:2210.11235v3*, Nov. 2022.

[9] A. Dib, G. Sabri, and M. S. M. Mendjel, "Ransomware Attack Detection Based on Pertinent System Calls Using Machine Learning Techniques," *International Journal of Computer Networks and Communications (IJCNC)*, vol. 15, no. 4, pp. 123-135, July 2023, doi: 10.5121/ijcnc.2023.15408.

[10] S. Poudyal and D. Dasgupta, “AI-Powered Ransomware Detection Framework,” ResearchGate,[online].Available: https://www.researchgate.net/publication/346577369 _AIPowered_Ransomware_Detection_Framework

[11] U. Zahoora, A. Khan, M. Rajarajan, S. Khan, M. Asam and T. Jamal, ” Ransomware detection using deep learning based unsupervised feature extraction and a cost sensitive Pareto Ensemble classifier,”.

[12] B. M. Khammas,” RansomwareDetection using RandomForestTechnique,”ScienceDirect, doi: https://doi.org/10.1016/j.icte.2020.11.001.

[13] M. I. Jaya and M. F. Ab. Razak, ” Dynamic Ransomware Detection for Windows Platform Using Machine Learning Classifiers”.

[14] B. Marais,T. Quertier and S. Morucci, ” AI-based Malware and Ransomware Detection Models,”.

[15] E. Ketzaki, P. Toupas, K. M. Giannoutakis, A. Drosou, D. Tzovaras, ”A Behaviour based Ransomware Detection using Neural Network Models,”.