SSR Journal of Engineering and Technology (SSRJET)

Homepage: https://ssrpublisher.com/ssrjet/

Volume 2, Issue 12, 2025 ISSN: 3049-0383

OPEN ACCESS

Ransomware Detection Model Using Deep Learning Algorithms

Tajudeen Olanrewaju Toyyib[1,] Aro O. Taye[2], Saka Kayode Kamil[3]

[1,2,3]Department of Computer Science, Al-Hikmah University Ilorin, Nigeria

Received: *07.12.2025* | ***Accepted:*** *16.12.2025* | ***Published:*** *31.12.2025*

***Corresponding author:** Tajudeen Olanrewaju Toyyib

***Corresponding author's Email: toyyibolanrewajutajudeen@gmail.com**

DOI: *10.5281/zenodo.18107249*

Abstract **Original Research Article**

Ransomware has emerged as one of the most destructive types of cyberattacks, harming people's and companies' reputations, disrupting operations, and resulting in significant financial losses. In order to increase the classification accuracy and resilience, this study created an improved ransomware detection model by combining two deep learning models with a meta-algorithm. Two benchmark datasets, the Windows PE File Analysis Dataset and the Resilient Information Systems Security Group (RISSG) ransomware dataset, were employed. After that, an autoencoder was used to optimize the features. Artificial Neural Networks (ANN) and Recurrent Neural Networks were used to classify the optimized features. The model was evaluated with performance metrics, including accuracy, precision, recall, F1-score, and time-taken. The experimental results for the two datasets showed that the best accuracy of 94.05% was obtained in RNN, the highest precision value of 0.9200 was obtained in RNN, the highest recall value of 0.9402, the highest F1-score value of 0.7855; all these best valuation metrics were recorded in RNN for the PE File Analysis Dataset. The lowest time of 0.11 was obtained in RNN for the Resilient Information Systems Security Group Dataset (RISSG Group rissggrouphubransomware dataset), when auto-encoder feature selection was used. The findings demonstrated the ransomware detection model's capacity to successfully identify complicated ransomware variants by achieving the highest accuracy and lowering misclassification rates when compared to traditional detection techniques. The work advanced the field of cybersecurity by introducing a scalable and intelligent ransomware detection model that integrated boosting, feature, and deep learning techniques.

Keywords: Ransomware, Cyberattacks, Autoencoder, Resilient Information Systems Security Group, Artificial Neural Network, Recurrent Neural Network, Cybersecurity.

1. INTRODUCTION

The development and widespread availability of computer and internet technology have made network security susceptible to cyberattacks. Ransomware is a common type of malware used in cyberattacks to deceive victims into giving the attackers access to private and sensitive data (Cremer et al., 2022). As a result, unless victims pay a ransom for stolen files or data, they might no longer be able to access their data. To address these problems, various strategies have been developed (Alshaikh et al., 2020).. An exhaustive analysis of the literature makes it clear that some linguistic criteria are not always adequate to identify groups of dangerous URLs.

Ransomware attacks have consequences that extend beyond the money lost once the ransom is paid. For businesses, operational disruptions can result in significant productivity losses, especially in critical sectors like healthcare.

The advent of the internet has altered our method of connecting, talking, and getting information but along with its incredible advancements come unanticipated obstacles, notably in cybersecurity. Cybercriminals seized the chance to benefit from the internet's accessibility and anonymity as it gained popularity and spread its virtual network globally (Amjad et al, 2023).

Malware is any code that is added, altered, or deleted from a software system with the goal to harm the system or interfere with its normal operation (Namanya et al., 2018). Malware is a serious threat to contemporary computer systems, which are essential for a variety of societal operations. Recent years have seen a number of notable cyberattacks against multinational companies that have caused significant financial losses, business interruptions, and reputational damage (Cremer et al., 2022). Ransomware, which preys on untraceable payment methods like Bitcoin, has emerged as a prominent threat among the various types of malware.

Attackers use this type of malicious software to encrypt individual files and then demand a ransom to unlock them (Cen et al., 2024). Additionally, malware frequently poses as trustworthy programs, making it even harder for authorized people to access data. These threats jeopardize user data and other system assets' privacy, dependability, legitimacy, and accessibility. As a result, confidential information may be revealed, changed, or made unavailable to authorized individuals (Alshaikh et al., 2020). The seriousness of the malware problem in cyberspace security cannot be ignored.

One of the most destructive types of cyberattacks nowadays is ransomware, which harms people's and businesses' reputations globally while generating significant financial losses and operational disruption (Cremer et al., 2022). The efficiency of conventional signature-based and heuristic detection methods has been greatly diminished by its quick evolution through strategies like polymorphism, obfuscation, and zero-day attacks (Sallout et al., 2022). As such, there is an urgent need for powerful, intelligent, and adaptable detection systems that can recognize both established and new ransomware strains.

Users may suffer severe consequences from ransomware malware, such as financial loss, private breaches, and data theft (Manju Bhargava et al., 2022). In 2023, ransomware assaults increased more than ever before, which is why cyber experts should take proactive measures to improve the speed of the detection methods (Kritika, 2025). Numerous researchers have worked hard to create practical and efficient techniques for identifying ransomware threats (Naseer et al., 2021). These techniques cover a variety of strategies, including as pattern-matching (classification) and feature reduction (Jameel & Jawhar, 2023). Nonetheless, there has been a noticeable trend in recent research toward the use of feature selection and other methods.

By removing features that are useless or have little to no predictive information, feature selection yields a subset of input variables (Pudjihartono et al., 2022). This method can greatly increase the classifier models' comprehensibility and frequently creates models that more effectively generalize to unknown points (Krishna et al., 2024). The study choosed ransomware datasets using specific learning algorithms and apply a novel nature-inspired optimization technique called the lion optimization algorithm for attributes: Artificial Neural Network (ANN), and **Recurrent Neural Network (RNN)** for classification.

2. LITERATURE REVIEW

Nazma et al (2025) created a new method for detecting ransomware using deep learning that prioritized system behavior analysis above static file signatures. Support Vector Machines (SVM) and Random Forest classifiers were trained using the system's primary behavioral variables, which included file access patterns, encryption rates, system resource usage, and registry updates. An evaluation using a publicly accessible dataset enhanced with ransomware-like behaviors revealed an alarm latency of less than two seconds and a complete detection accuracy of over 96%. This study lays the groundwork for subsequent research that integrates cross-platform security measures, cloud-based response systems, and adaptive learning.

Farhan and Salman (2024) created a technique to improve security against these malevolent threats by analyzing and detecting ransomware behavior on Android devices using deep learning algorithms. A feedforward neural network more precisely, a Keras Sequential model was used in the model. There were three layers of closely spaced neurons in the model. There are 64 units in the first layer, 64 units in the second, and 2 units in the third and last layer. The proposed model recorded an accuracy value of 98.9%.

Singh et al (2024) created an innovative method that combined a number of deep learning techniques to detect Ransomware-as-a-Service (RaaS) attacks. The Ransom Detect Fusion ensemble model was established by combining the three prediction powers of Multilayers Perceptron. The suggested ensemble approach outperformed the current models with strong performance metrics such as an accuracy value of 98.79%, a recall value and precision value of 98.85%, and an F1-score value of 98.80%. The empirical results demonstrated that the ensemble model outperformed individual MLP models, hence validating its potential to enhance cybersecurity defenses.

Davidian et al (2024) conducted a methodical evaluation of several Deep Learning techniques, context-window sizes, and subsets of API call data in order to identify the best ransomware classifier. The results indicated that CNN and LSTM were the most effective deep learning techniques, and that a context window size of seven is optimal. Additionally, the precision of the classifier was much increased by appending the operation result to the API call name. The results of performance investigation indicated that this classifier can be used successfully in real-time situations.

Jemal (2023) design a multi-variant classifier to identify I/O operations from ransomware in programs that aren't malicious. Two deep learning models were utilized in the study: Bi-directional Long Short-Term Memory (Bi-LSTM) and Convolutional Neural Networks (CNN). Classic learning techniques such as Random Forest (RF), Support Vector Machine (SVM), and Logistic Regression (LR) were compared with the deep learning models. During the encryption of a large network shared directory, 70 binaries from 30 distinct ransomware strains were taken and included in the ransomware samples. Additionally, zero-day ransomware samples were used to test the deep learning models. Both Bi-LSTM and CNN achieved above 98% in accurately classifying ransomware and benign samples.

Mijwil et al (2023) employed cybersecurity procedures to defend computer networks against intrusions, hacking, and data theft, and examined the function of artificial intelligence in this field. The most significant literature that examined the functions and impacts of machine learning and deep learning approaches in cybersecurity was also compiled in the study. The findings shown that by anticipating and comprehending the behavior and traffic of malicious software, machine learning and deep learning approaches significantly contribute to the defense of computer systems against unwanted access and to the management of system penetration.

METHODOLOGY

3.1 Developed Ransomware Detection Model

The ransomware detection methodology included multiple steps to follow in order to accomplish the study's goal. The ransomware datasets, the Windows PE File Analysis Dataset and the Resilient Information Systems Security Group Dataset, were sourced from publicly accessible ransomware datasets (RISSG Group rissggrouphubransomware dataset 2016). The datasets undergone the preprocessing stage. After pre-processing, an auto-encoder was used to optimize the data. A recurrent neural network (RNN) and an artificial neural network (ANN) were used to classify the optimized characteristics. Ultimately, the built ransomware detection model was assessed using a few chosen performance evaluation measures. Figure 1 showed the system framework.

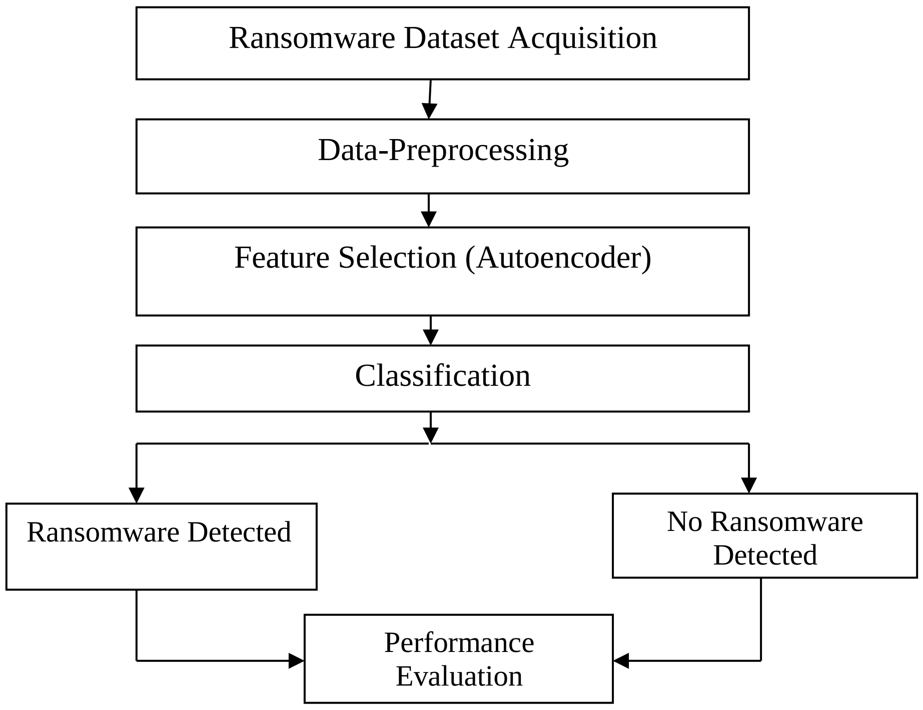


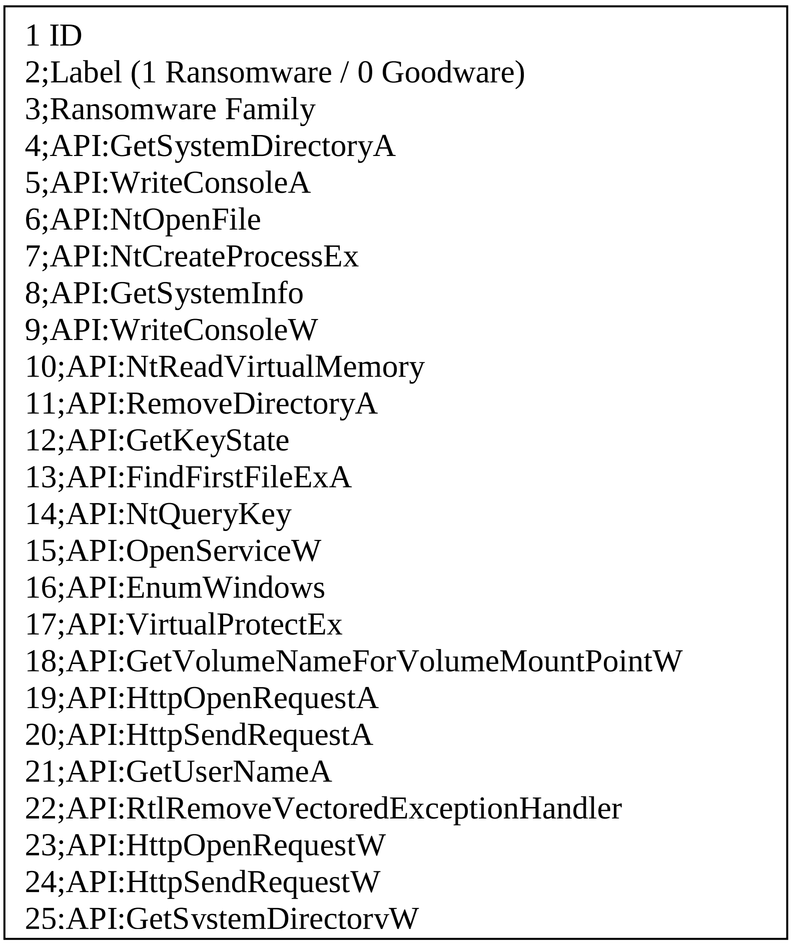
Figure 1: System Framework.

3.2 Acquisition and Description of the Dataset

Training and validating any established prediction or detection model requires the use of datasets. The ransomware detection model acquired its datasets from two ransomware datasets: the Windows PE File Analysis Dataset and the Resilient Information Systems Security Group Dataset (RISSG Group rissggrouphubransomware dataset 2016).

3.2.1 Description of Ransomware Datasets

The dataset is made up of PE file properties that were taken from a group of DLL files and Windows executables. With different attributes taken from its PE header and structure, each entry represents a distinct file. Both benign software samples and known malware samples identified by VirusShare hashes are included in the datasetThe 2016 Ransomware dataset includes a dynamic analysis of 582 ransomware samples and 942 goodware samples, for a total of 1524 samples. Cuckoo Sandbox was used to retrieve and analyze the dataset at the end of February 2016. Figure 2 displays the attributes sample.



1 ID
2;Label (1 Ransomware / 0 Goodware)
3;Ransomware Family
4;API:GetSystemDirectoryA
5;API:WriteConsoleA
6;API:NtOpenFile
7;API:NtCreateProcessEx
8;API:GetSystemInfo
9;API:WriteConsoleW
10;API:NtReadVirtualMemory
11;API:RemoveDirectoryA
12;API:GetKeyState
13;API:FindFirstFileExA
14;API:NtQueryKey
15;API:OpenServiceW
16;API:EnumWindows
17;API:VirtualProtectEx
18;API:GetVolumeNameForVolumeMountPointW
19;API:HttpOpenRequestA
20;API:HttpSendRequestA
21;API:GetUserNameA
22;API:RtlRemoveVectoredExceptionHandler
23;API:HttpOpenRequestW
24;API:HttpSendRequestW
25:API:GetSvstemDirectorvW

Figure 2: Sample of Attributes for Ransomware Dataset

3.3 Pre-processing of the Dataset

In data science, data preprocessing is a crucial step that turns unstructured data into a clean format for study. It includes things like encoding variables, addressing missing values, and normalizing data. Gaining proficiency in Python preprocessing guarantees trustworthy insights for precise forecasts and efficient decision-making. When data is transformed before being fed into an algorithm, this is referred to as pre-processing. This stage involves creating the entire feature-rich CSV dataset and splitting the application into two CSV files: one for malicious apps and another for secure apps, both of which were manually completed and integrated. The step-by-step is shown in Figure 3

```
Step 1: Start
Step 2: Load data in Pandas.
Step 3: Drop columns that aren't useful.
Step 4: Drop rows with missing values
Step 5: Create dummy variables.
Step 6: Take care of missing data.
Step 7: Convert the data frame to NumPy.
Step 8: Divide the data set into training data and test data.
Step 9: End
```

Figure 3: Algorithm for Data Preprocessing in Python

3.4 Algorithms for Feature Selection Using Auto-Encoder

An autoencoder will be used in the feature selection methods Figure 4 represents the standard auto-encoding framework, while Figure 5 depicts the algorithm.

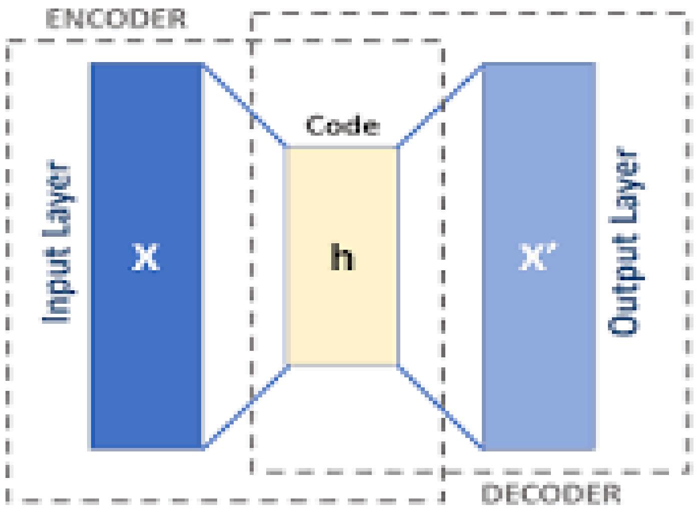


Figure 4: Framework of Auto-encoder

```
Step 1: Start
Step 2: Import necessary libraries
Step 3: Load the MNIST dataset.
Step 4: Define a basic Auto-encoder.
Step 5: Compiling and Fitting Auto-encoder.
Step 6: Visualize original and reconstructed data.
Step 7: End
```

Figure 5: Algorithm of Auto-encoder

3.5 Algorithms for Classification Using Learning Algorithms

Two algorithms for learning were used: ANN and RNN for the task. The following subsections provide the algorithms or detailed instructions for each algorithm.

3.5.1 Algorithm of Artificial Neural Network (ANN)

There are a number of crucial processes involved in designing an ANN, ranging from problem conceptualization and data preparation to model training, evaluation, and deployment. The step-by-step process for ANN is shown in Figure 6.

```
Step 1: Start
Step 2: Define the Problem and Collect Data.
Step 3: Prepare Data for Training.
Step 4: Design Neural Network Architecture.
Step 5: Initialize Parameters.
Step 6: Implement Forward Propagation.
Step 7: Define and Compute Cost Function.
Step 8: Training the Neural Network.
Step 9: Define the Problem and Collect Data.
Step 10: End
```

Figure 6: Algorithm of ANN

3.6 Algorithm of Recurrent Neural Networks

An artificial neural network version called a Recurrent Neural Network (RNN) was created to reorganize consecutive data features and use patterns to forecast the next potential state or circumstance. The following is the RNN algorithm:

Step 1: Start
Step 2. State Update:

$$\boldsymbol{ht = f(ht-1, xt)ht = f(ht-1, xt)}$$

Where:

- htht is the current state
- ht−1ht−1 is the previous state
- xtxt is the input at the current time step

Step 3. Activation Function Application:

$$\boldsymbol{ht = tanh(Whh \cdot ht-1 + Wxh \cdot xt)h = tanh(Whh \cdot ht-1 + Wxh \cdot xt)}$$

Here, WhhWhh is the weight matrix for the recurrent neuron, and WxhWxh is the weight matrix for the input neuron.
Step 4. Output Calculation:

$$\boldsymbol{yt = Why \cdot htyt = Why \cdot ht}$$

where $ytyt$ is the output and $WhyWhy$ is the weight at the output layer.

To update these parameters, backpropagation is used. But since RNN operates on sequential data, we employ backpropagation via time, which is an updated kind of backpropagation.

3.7 Performance Evaluation Metrics

Measurable values known as performance assessment metrics are employed to assess the prediction or defection model's efficacy, efficiency, and quality. The following metrics will be employed in the proposed study:

3.7.1 Accuracy

This is the percentage of test instances that were correctly and incorrectly classified. The correctly classified instances give the accuracy, while the incorrectly classified instances can be computed by subtracting the correctly classified instances from 100, as shown in Equation (1).

$$\textbf{Accuracy} = \frac{\textbf{TP} + \textbf{TN}}{\textbf{TP} + \textbf{FP} + \textbf{TN} + \textbf{FN}} \times 100\% \qquad \textbf{1}$$

Where TP = True positive, FP = False positive, TN = True negative, FN = False negative

3.7.2 Recall

The proportion of actual positives (e.g., people with a disease) that were correctly identified. It is also referred to as the true positive rate or recall, as shown in Equation (2).

$$\textbf{Recall} = \frac{\textbf{TP}}{\textbf{TP} + \textbf{FN}} \times 100\% \qquad \textbf{2}$$

3.7.3 Precision

The ratio of correctly predicted positive observations to the total predicted positive observations. It is computed by dividing the number of true positives by the sum of true positives and false positives, as shown in Equation 3.

$$\textbf{Precision} = \frac{\textbf{TP}}{\textbf{TP} + \textbf{FP}} \times 100\% \qquad \textbf{3.}$$

3.7.4 F1-Measure

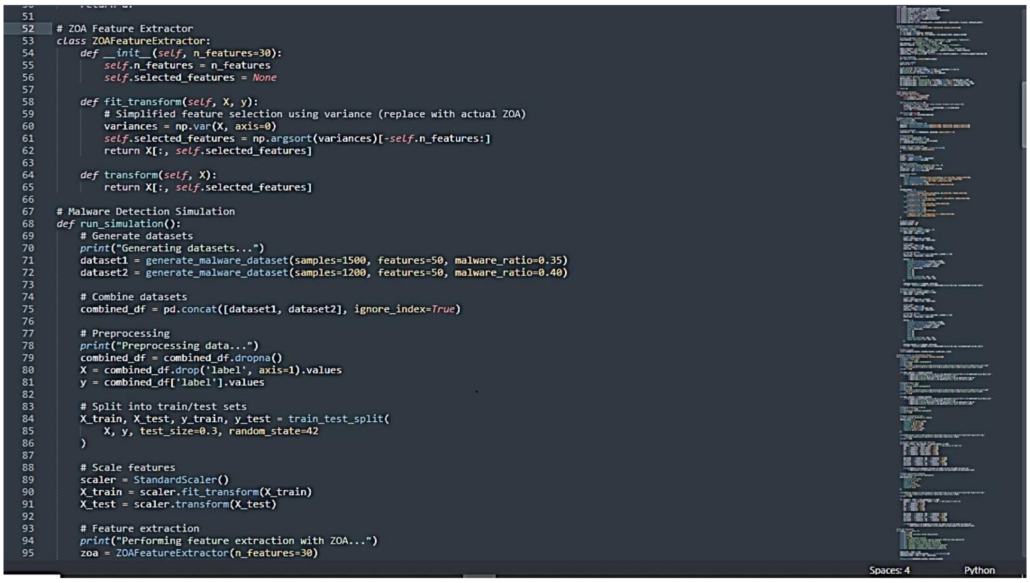
The harmonic mean of precision and recall. It provides a balance between precision and recall, especially when both false positives and false negatives are important. The formula is shown in Equation 4.

$$\mathbf{F1} = \mathbf{2} * \frac{\mathbf{precision} * \mathbf{Recall}}{\mathbf{Precision} \ + \mathbf{Recall}} \qquad \mathbf{4.}$$

4. RESULT

4.1 Experimental System Setup

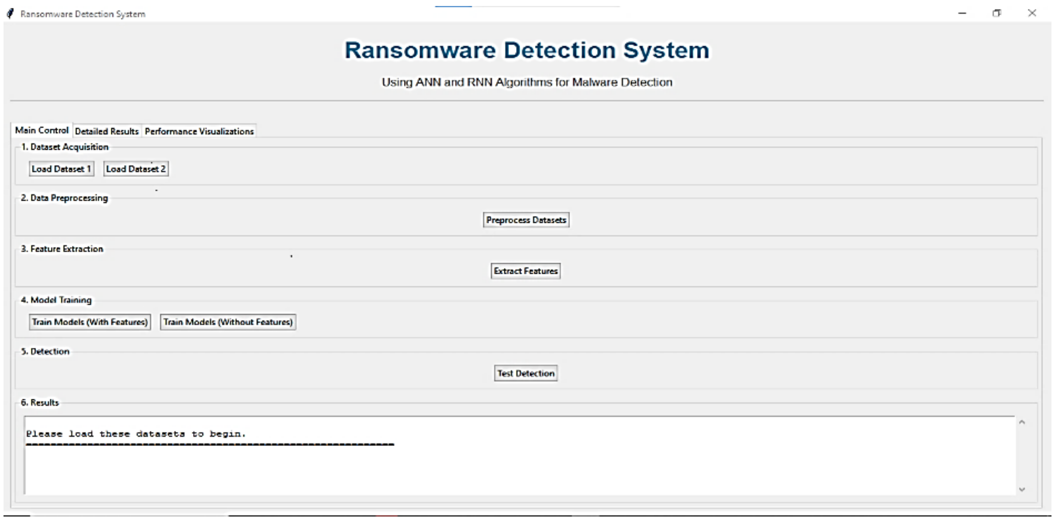
Python, which provided an extensive collection of libraries designed for data science applications, was used to develop the experimental environment. Important libraries include Scikit-learn for putting machine learning methods into practice, Matplotlib and Seaborn for insightful data visualization, Pandas for efficient data manipulation and analysis, and NumPy for numerical computations. This robust tool ecosystem speeds up development and makes managing difficult tasks much easier. Figure 7 and 8 show how the created ransomware detection model uses a variety of Python components to process data and display performance evaluation findings in an easy-to-use interface.



```python

# ZOA Feature Extractor
class ZOAFeatureExtractor:
    def __init__(self, n_features=30):
        self.n_features = n_features
        self.selected_features = None

    def fit_transform(self, X, y):
        # Simplified feature selection using variance (replace with actual ZOA)
        variances = np.var(X, axis=0)
        self.selected_features = np.argsort(variances)[-self.n_features:]
        return X[:, self.selected_features]

    def transform(self, X):
        return X[:, self.selected_features]

# Malware Detection Simulation
def run_simulation():
    # Generate datasets
    print("Generating datasets...")
    dataset1 = generate_malware_dataset(samples=1500, features=50, malware_ratio=0.35)
    dataset2 = generate_malware_dataset(samples=1200, features=50, malware_ratio=0.40)

    # Combine datasets
    combined_df = pd.concat([dataset1, dataset2], ignore_index=True)

    # Preprocessing
    print("Preprocessing data...")
    combined_df = combined_df.dropna()
    X = combined_df.drop('label', axis=1).values
    y = combined_df['label'].values

    # Split into train/test sets
    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.3, random_state=42
    )

    # Scale features
    scaler = StandardScaler()
    X_train = scaler.fit_transform(X_train)
    X_test = scaler.transform(X_test)

    # Feature extraction
    print("Performing feature extraction with ZOA...")
    zoa = ZOAFeatureExtractor(n_features=30)
```

Figure 7: Interface for Model Running Environment

The ransomware detection model's operating environment interface is shown in Figure 7; this interface preceded any other interfaces.



```
Please load these datasets to begin.
------------------------------------------------------------
```

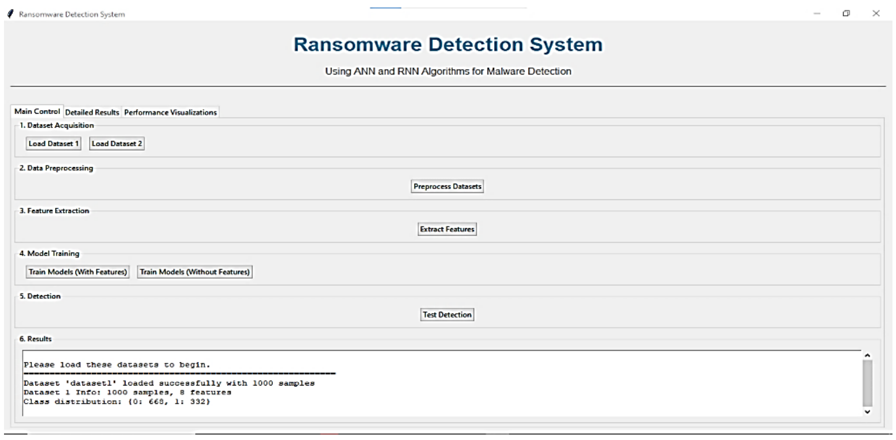
Figure 8: Interface of Malware Detection Model

The interface that fully displays the many functions of the created ransomware detection model is displayed in Figure 8. The interface includes the configuration button, with two buttons for dataset loading. Status and time taken are also considered.

4.2 Results of Pre-Processing Phase

The ransomware dataset raw data were pre-processed and standardized, as shown in

Figure 9.



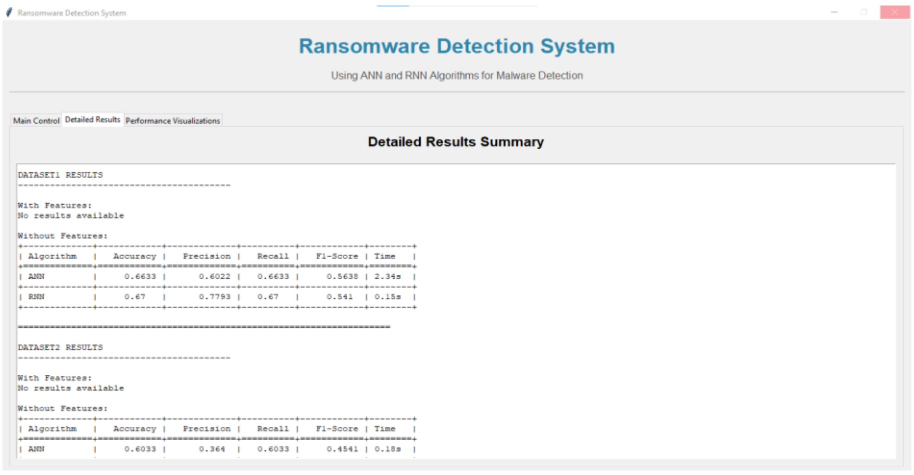
```
Please load these datasets to begin.
------------------------------------------------------------
Dataset 'dataset1' loaded successfully with 1000 samples
Dataset 1 Info: 1000 samples, 8 features
Class distribution: {0: 668, 1: 332}
```

Figure 9: Interface for Preprocessed Ransomware Datasets

The pre-process stage of the created ransomware detection model, depicted in Figure 9, involved fully normalizing the dataset. Pre-processing makes it possible for the deep learning algorithms to identify patterns in the data, identify abnormalities, and accurately categorize risks. It establishes the framework for creating a detection model that is intelligent, scalable, and reliable.

4.3 Results of the Ransomware Detection Model

The evaluation metrics for the ransomware detection model that was created utilizing the two datasets were as follows: Windows PE File Analysis Dataset and the Resilient Information Systems Security Group Dataset (RISSG Group rissggrouphubransomware dataset 2016), are accuracy, precision, recall, F1-score, and time taken to build the ransomware detection model, as shown in Figure 10.


Ransomware Detection System
Ransomware Detection System
Using ANN and RNN Algorithms for Malware Detection
Main Control
Detailed Results
Performance Visualizations
Detailed Results Summary

```
DATASET1 RESULTS
----------------------------------------

With Features:
No results available

Without Features:
+-------------+------------+-------------+-----------+------------+--------+
| Algorithm   |   Accuracy |   Precision |    Recall |   F1-Score | Time   |
+=============+============+=============+===========+============+========+
| ANN         |     0.6633 |      0.6022 |    0.6633 |     0.5638 | 2.34s  |
+-------------+------------+-------------+-----------+------------+--------+
| RNN         |     0.67   |      0.7793 |    0.67   |     0.541  | 0.15s  |
+-------------+------------+-------------+-----------+------------+--------+

======================================================================

DATASET2 RESULTS
----------------------------------------

With Features:
No results available

Without Features:
+-------------+------------+-------------+-----------+------------+--------+
| Algorithm   |   Accuracy |   Precision |    Recall |   F1-Score | Time   |
+=============+============+=============+===========+============+========+
| ANN         |     0.6033 |      0.364  |    0.6033 |     0.4541 | 0.18s  |
```

Figure 10: Interface for Results Display

The results of the created ransomware detection model were displayed in relation to several evaluation criteria, as shown in Figure 10. The following subsections provide additional examples of the outcomes for each performance evaluation metric:

4.4.1 Ransomware Detection Model

This section displays the outcomes of feature selection using an auto-encoder for the ransomware detection model and feature selection not being used for the Windows PE File Analysis Dataset, as indicated in Tables 1 and 2.

Table 1: Ransomware Detection (without feature selection)

Algorithm	Accuracy(%)	Precision	Recall	F1-Score	Time-Taken
ANN	90.33	0.8903	0.8900	0.7552	0.24s
RNN	92.45	0.9102	0.9300	0.7705	0.18s

Table 1 shows that the ANN had the lowest accuracy value of 90.33%, the RNN had the highest accuracy value of 92.45%, the ANN had the lowest precision value of 0.8903, the RNN had the highest precision value of 0.9102, the ANN had the lowest recall value of 0.8900, the RNN had the highest recall value of 0.9300, the ANN had the lowest F1-score value of 0.7052, the RNN had the highest F1-score value of 0.7205, the lowest time of 0.18s was spent building, and the ANN had the highest time of 0.24s.

Table 2: Ransomware Detection (with auto-encoder)

Algorithm	Accuracy (%)	Precision	Recall	F1-Score	Time-Taken (s)
ANN	93.05	0.8803	0.9308	0.7823	0.22s
RNN	94.05	0.9200	0.9402	0.7855	0.16s

Table 2 shows that the ANN had the lowest accuracy value of 93.05%, the RNN had the highest accuracy value of 94.05%, the ANN had the lowest precision value of 0.0.8803, the RNN had the highest precision value of 0.8906, the ANN had the lowest recall value of 0.9308, the

RNN had the highest recall value of 0.9402, the ANN had the lowest F1-score value of 0.7823, the RNN had the highest F1-score value of 0.7855, the lowest time of 0.18s was spent building, and the ANN had the highest time of 0.18s.

4.4.2 Ransomware Detection Model (Resilient Information Systems Security Group Dataset)

The outcomes of using an auto-encoder for feature selection in the ransomware detection model and without using the feature selection approach on the Resilient Information Systems Security Group Dataset are displayed in this section, as shown in Table 3 and Table 4.

Table 3: Ransomware Detection Model (without feature selection)

Algorithm	Accuracy (%)	Precision	Recall	F1-Score	Time-Taken (s)
ANN	88.05	0.8202	0.8820	0.7202	0.16
RNN	89.56	0.8405	0.8950	0.7400	0.11

According to Table 3, ANN had the lowest accuracy value of 88.05%, RNN had the highest accuracy value of 89.56%, ANN had the lowest precision value of 0.8202, RNN had the highest precision value of 0.8405, ANN had the lowest recall value of 0.8820, RNN had the highest recall value of 0.8950, ANN had the lowest F1-score value of 0.7202, RNN had the highest F1-score value of 0.7400, RNN had the longest build time of 0.11 seconds, while ANN had the longest build time of 0.16 seconds.

Table 4: Ransomware Detection Model (with auto-encoder)

Algorithm	Accuracy (%)	Precision	Recall	F1-Score	Time-Taken (s)
ANN	90.01	0.8604	0.8956	0.6874	0.30s
RNN	91.23	0.8701	0.9103	0.6903	0.12s

According to Table 4, ANN had the lowest accuracy value of 90.01%, RNN had the highest accuracy value of 91.23%, ANN had the lowest precision value of 0.8604, RNN had the highest precision value of 0.8701, ANN had the lowest recall value of 0.8956, RNN had the highest recall value of 0.9103, ANN had the lowest F1-score value of 0.6874, RNN had the highest F1-score value of 0.6903, RNN had the shortest build time of 0.12 s, while ANN had the longest build time of 0.30s.

5. CONCLUSION

In this study, deep learning algorithms were used to create a ransomware detection model. Two datasets such as Windows PE File Analysis Dataset and the Resilient Information Systems Security Group Dataset (RISSG Group rissggrouphubransomware dataset 2016) were used for performance evaluation of the model. Python functions were used to pre-process the datasets. They used an autoencoder algorithm to optimize the preprocessed features. The RNN and CNNs were fed the optimal features in order to classify them. The evaluation metrics used were accuracy, precision, recall, F1-score, and time taken to build a model. According to the experimental results for the two datasets, RNN produced the best valuation metrics for the PE File Analysis Dataset, including the highest accuracy of 94.05%, the highest precision value of 0.9200, the highest recall value of 0.9402, and the highest F1-score value of 0.7855. The lowest time of 0.11 was obtained in RNN for the Resilient Information Systems Security Group Dataset (RISSG Group rissggrouphubransomware dataset 2016).

5.1 Recommendations

Based on the results of this research on ransomware detection using deep learning algorithms, the following recommendations are proposed:

(i) **Implementation of intelligent Ransomware Detection Models:** Organizations and cybersecurity practitioners should adopt advanced machine learning and deep learning models for ransomware detection rather than depending solely on methods like signature-based or heuristic approaches. This will improve the detection of zero-day and polymorphic ransomware attacks.

(i) **Ensemble methods should be explored:** Future studies can incorporate advanced ensemble techniques such as Bagging, Random Forests, Gradient Boosting, and Stacking.

References

Cremer, F., Sheehan, B., Fortmann, M., Kia, A. N., Mullins, M., Murphy, F., & Materne, S. (2022). Cyber risk and cybersecurity: a systematic review of data availability. *Geneva Papers on Risk and Insurance: Issues and Practice*, *47*(3), 698–736. https://doi.org/10.1057/s41288-022-00266-

Amjad H., Ayesha S., Musaed A., Ammara G. & Khursheed A (2023). Enhancing ransomware defense: deep learning-based detection and family-wise classification of evolving threats. Journal of Center for Biotechnology Information.

Davidian, M., Kiperberg, M., & Vanetik, N. (2024). Early Ransomware Detection with Deep Learning Models. *Future Internet*, *16*(8). https://doi.org/10.3390/fi16080291

Farhan, R. I., & Salman, R. H. (2024). An Approach to Android Ransomware Detection Using Deep Learning. *Wasit Journal for Pure Science*, *3*(1), 90–94.

Jemal, M. (2023). *Detection of Crypto-Ransomware Attack Using Deep Learning*.

Krishna, R., Jayanthi, D., Shylu Sam, D. S., Kavitha, K., Maurya, N. K., & Benil, T. (2024). Application of machine learning techniques for churn prediction in the telecom business. *Results in Engineering*, *24*(August), 103165. https://doi.org/10.1016/j.rineng.2024.103165

Kritika, E. (2025). Cyber Security and Applications A comprehensive literature review on ransomware detection using deep learning. *Cyber Security and Applications*, *3*(October 2024), 100078. https://doi.org/10.1016/j.csa.2024.100078

Malashin, I., Tynchenko, V., Gantimurov, A., Nelyub, V., & Borodulin, A. (2024). Applications of Long Short-Term Memory (LSTM) Networks in Polymeric Sciences: A Review. *Polymers*, *16*(18), 1–44. https://doi.org/10.3390/polym16182607

Manju Bargavi, M.Senbagavalli, Tejashwini K.R., & Tejashvar. K.R. (2022). Data Breach – Its Effects on Industry. *International Journal of Data Informatics and Intelligent Computing*, *1*(2), 51–57. https://doi.org/10.59461/ijdiic.v1i2.31

Naseer, M., Rusdi, J. F., Shanono, N. M., Salam, S., Muslim, Z. Bin, Abu, N. A., & Abadi, I. (2021). Malware Detection: Issues and Challenges. *Journal of Physics: Conference Series*, *1807*(1), 1–7.

https://doi.org/10.1088/1742-6596/1807/1/012011

Sigh, H., Zhang, Y., Liu, H., Li, X., Chang, J., & Zheng, H. (2024). Light Recurrent Unit: Towards an Interpretable Recurrent Neural Network for Modeling Long-Range Dependency. *Electronics (Switzerland)*, *13*(16), 1–18. https://doi.org/10.3390/electronics13163204