Indonesian Journal of Electrical Engineering and Informatics (IJEEI)
Vol. 12, No. 4, December 2024, pp. 1111~1133
ISSN: 2089-3272, DOI: 10.52549/ijeei.v12i4.5648

Cyber Security Threat Prediction Using Time-Series Data with LSTM Algorithms

Lukman Hakim[1], Lili Ayu Wulandhari[2]
[1,2]Department of Computer Science, BINUS Graduate Program, Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia

Article Info

Article history:

Received Jun 13, 2024
Revised Dec 28, 2024
Accepted Dec 31, 2024

Keywords:

Forecasting
Machine Learning
Cyber Security
Threat
Long Short-Term Memory
Time-Series

ABSTRACT

Cybersecurity remains a paramount concern in the digital era, with organizations and individuals increasingly vulnerable to sophisticated cyberattacks. In Indonesia alone, 2022 recorded over 976 million cyberattacks, highlighting the critical need for proactive security measures. Traditional reactive security systems often fail to anticipate emerging threats, leading to substantial financial losses and data breaches. The complexity and volume of modern cyberattacks make it increasingly difficult for security teams to manually analyze and predict potential threats in real time. This study aims to address these challenges by developing and evaluating Long Short-Term Memory (LSTM) regression models to predict three types of cyber attacks: flood, spyware, and vulnerability. The experiments demonstrate that preprocessing techniques such as normalization and standardization can positively impact model performance. The results show promising outcomes, particularly for flood attacks (RMSE: 59.8810, R-squared: 0.9214) and spyware attacks (RMSE: 133.9567, R-squared: 0.7685) after standardization, while vulnerability attack predictions showed more limited improvement (RMSE: 503.5521, R-squared: 0.2358). These findings suggest the potential of LSTM-based approaches in enhancing cybersecurity threat prediction capabilities

.

Corresponding Author:

Lukman Hakim,
Department of Computer Science,
Bina Nusantara University,
Jl. Raya Kb. Jeruk No.27, RT.1/RW.9, Kemanggisan, Kec. Palmerah, Kota Jakarta Barat, Daerah Khusus Ibukota Jakarta 11530, Indonesia.
Email: lukman.hakim002@binus.ac.id

1. INTRODUCTION

Despite advances in cybersecurity prediction methods, current approaches face significant limitations in accurately forecasting different types of cyberattacks. While existing studies have demonstrated the potential of machine learning for attack prediction, they often focus on single attack types and lack comprehensive evaluation of preprocessing techniques' impact on prediction accuracy. This gap is particularly critical given Indonesia's escalating cybersecurity challenges, with 976,429,996 recorded incidents in 2022 spanning multiple attack categories, including malware (56.84%), data breaches (14.75%), and trojan activities (10.9%).

As the application of information technology advances, so do the threats of cyberattacks. These attacks can disrupt government operations, steal sensitive data, or even damage critical infrastructure. Firewall logs from an institution reported that in September, there were 393 cyberattacks per hour, totaling 273,595 attacks. Thus, protecting government infrastructure and data is crucial, and increasing vigilance is necessary to identify and address potential cyber threats.

.

Cyberattacks refer to threats and attacks that can damage, steal, or disrupt data and information stored and transmitted through information technology [1]. These attacks can take various forms, such as malware attacks, phishing attacks, DDoS (Distributed Denial of Service) attacks, ransomware attacks, and others. Cyberattacks are considered a serious threat because they can cause significant damage to national security and national interests of Indonesia. Cyber-attacks are carried out through computer networks and information systems with the aim of accessing, damaging, or stealing sensitive data, disrupting system operations, or causing losses to critical infrastructure [2].

To counteract cyber security threats, early detection and rapid response are key. Cybersecurity threat prediction using machine learning techniques is one approach that has been explored by previous researchers. Historical cyber-attack data can be a valuable source of information for developing predictive models that can identify attack patterns and forecast potential future attacks. Time series analysis is a powerful tool aimed at identifying patterns and studying the nature and structure of data representing technological processes, economic indicators, information signals, and more [3].

Time series data are a type of data collected at regular time intervals [4]. These data include observations taken at different points in time and are ordered chronologically. Each observation in time series data is associated with the time at which the data was collected. Time series data have a special structure that allows for the analysis and prediction of trends, seasonal patterns, and random fluctuations in the data. Time series analysis is used to understand patterns and trends in the data, make future predictions, and make decisions based on the historical patterns found in time series data.

Machine learning is a subfield of artificial intelligence that focuses on developing techniques and computer models that enable systems to learn from data and improve their performance over time without being explicitly reprogrammed. In machine learning, computers are used to analyze data, identify patterns, and make decisions or predictions based on the provided data. Machine learning as the process of solving practical problems by collecting data and algorithmically building statistical models based on that dataset [5].

One of the prominent techniques within machine learning is the Recurrent Neural Network (RNN). RNNs are a class of artificial neural networks specifically designed to handle sequential data and time-series information. Unlike traditional neural networks, which assume that all inputs are independent of each other, RNNs have connections that form directed cycles, allowing information to persist. This capability makes RNNs particularly well-suited for tasks where the context or previous elements in the sequence influence the outcome, such as language modeling, speech recognition, and time-series prediction. By leveraging their internal memory, RNNs can capture temporal dependencies and patterns, making them a powerful tool for analyzing and predicting data that unfolds over time. This is a critical feature that distinguishes RNNs from traditional neural networks[6].

Many forecasting models have been proposed to find effective methods applicable in practical situations. However, the primary limitation in time series forecasting is the lack of deterministic causality [7], making it difficult to pinpoint the exact causes or cause-and-effect relationships of observed events. In some cases, we can predict what will happen based on historical data, but it is challenging to understand why it happens or what causes it, leading to uncertainty or unreliability in forecasting. To address this limitation, model development usually relies on large amounts of input or random events.

Previous research demonstrated that LSTM models consistently produced lower Root-Mean-Square Error (RMSE) compared to ARIMA models for all tested time series data [8]. The average error reduction achieved by LSTM models was between 84% and 87% compared to ARIMA models. In conclusion, ARIMA and LSTM have different approaches to time series modeling. LSTM excels in long-term memory capability and capturing complex patterns but requires more parameters and longer computation times. Conversely, ARIMA is simpler and faster but has limitations in capturing long-term patterns and non-linear relationships. There is a study that discusses the application of time series forecasting techniques to predict the intensity of cyber attacks[9]. The primary goal of this research is to identify patterns of cyber attacks based on the temporal correlation between the number of attacks per day and to use this information to predict future attack intensity. The results of this study show that the forecasting system using the ARIMA model to predict cyber attack intensity outperforms the naive forecasting method by 14.1% when predicting attacks of all types and by up to 21.2% when predicting attacks of specific types. In the study "Forecasting Network Intrusions from Security Logs Using LSTMs,"[10] three models are discussed: Naive Baseline, ARIMA Baseline, and Long Short-Term Memory (LSTM). The evaluation is conducted using the Mean Absolute Error (MAE) metric. The evaluation results show that the LSTM model has a lower MAE compared to ARIMA and Naive for most types of IDS alerts. This indicates that LSTM provides more accurate predictions in estimating the value of IDS alerts. However, there are some types of alerts where ARIMA or Naive have a lower MAE than LSTM.

Based on previous studies [8, 9, 10], LSTM has demonstrated significant advantages for cyber attack prediction through multiple performance indicators: achieving 84-87% error reduction compared to ARIMA models across all tested time series data, excelling in capturing complex patterns and long-term dependencies

in the data despite requiring more computational resources. While traditional ARIMA models showed some effectiveness by outperforming naive forecasting methods by 14.1% for general attacks and 21.2% for specific attack types, LSTM demonstrated superior performance in network intrusion detection by achieving lower MAE scores across most IDS alert types. LSTM's ability to handle non-linear relationships and maintain long-term memory capabilities makes it particularly well-suited for cyber attack prediction, despite the trade-off of increased computational complexity compared to simpler ARIMA models. However, it's important to note that for some specific types of alerts, traditional methods like ARIMA or Naive approaches may still perform better, suggesting the need for careful evaluation of model selection based on the specific characteristics of the cyber attack patterns being analyzed.

"*Phishcasting*" —*an* innovative cybersecurity approach using CoT-Net, which combines CNN and LSTM to predict phishing attack volumes. The research utilized a dataset of 1.5 million phishing attacks over 10 years, focusing on five major brands. Experimental results demonstrated CoT-Net's superiority compared to FB Prophet and LSTM, with an RMSE of 396.07 for regression and an accuracy of 0.64 for classification [11]. A comprehensive approach for anomaly detection and trend prediction in intelligent operations based on KPIs using the methodology employed by the S-ESD algorithm for anomaly detection and the Prophet model for trend prediction. Using a base station operator dataset over 29 days, the Prophet model achieved a MAPE of 0.0783 after parameter optimization, highlighting the effectiveness of the approach in operational data analysis [12]. An innovative approach for cloud computing intrusion detection using time series anomaly analysis and machine learning has been developed. The research used the CSE-CIC-IDS2018 dataset focusing on the Botnet attack subset, developing a collaborative feature selection and Prophet-based prediction model. Results were significant: predictors reduced from 70 to 10, with performance metrics decreasing, such as MAE from 0.216 to 0.122, MSE from 0.081 to 0.032, and RMSE from 0.270 to 0.171. Training, prediction, and cross-validation times decreased by 85%, 15%, and 97% respectively, demonstrating potential for real-time attack detection with fewer false alerts [13].

This study makes three key contributions to the field of cyber attack prediction: 1). A comparative analysis of LSTM and FB Prophet performance across three distinct attack types (flood, spyware, and vulnerabilities), providing insights into the model's effectiveness for different threat categories, 2). A systematic evaluation of data preprocessing techniques' impact on LSTM prediction accuracy, addressing a critical gap in existing literature, 3) Development of an optimized LSTM-based prediction framework that enables hour-ahead forecasting of attack frequencies, incorporating various training parameters for enhanced accuracy. We hypothesize that LSTM models will demonstrate varying levels of prediction accuracy across different attack types, with more consistent patterns (like flood attacks) showing higher prediction accuracy than more irregular patterns, and appropriate data preprocessing techniques will significantly improve LSTM prediction performance, with normalized data expected to yield better results than raw data. The paper is structured as follows: The Introduction provides an overview of the importance of cyber security threat prediction, the relevance of LSTM algorithms, and identifies the existing gaps in the literature. The Research Method details the experimental setup through several stages: Research Process Stages outlines the overall workflow, including model development and evaluation phases; Data Collection describes the sources and methods for obtaining cyber attack data; Data Transformation explains how raw data is processed into a suitable format for analysis; Data Scaling discusses normalization and standardization techniques applied to the data; Regression Model elaborates on the LSTM model architecture and training parameters; and Forecasting Performance Evaluation Metrics outlines the metrics used to assess the prediction accuracy of the model. The Result and Discussion section presents the findings from the experiments, comparing the prediction accuracy of the LSTM model for the different types of attacks, and analyzes the impact of preprocessing techniques such as normalization and standardization on the model's performance. Finally, the Conclusion summarizes the key insights from the study, highlighting the effectiveness of LSTM in predicting cyber attacks and suggesting potential areas for future research.

2. RESEARCH METHOD

This research phase is where data analysis and regression model development for predicting cybersecurity threats will be conducted. The study will utilize firewall log datasets collected over three months, from July to September. It aims to uncover information about cyber attack patterns, trends, and potential attack characteristics. Additionally, feature extraction from the dataset will be involved to identify features that may influence cyber attacks. During this phase, predictive models will be built for three types of cyber attacks: spyware, flood, and vulnerability. The research phase is illustrated in a flowchart as shown in Figure 1.

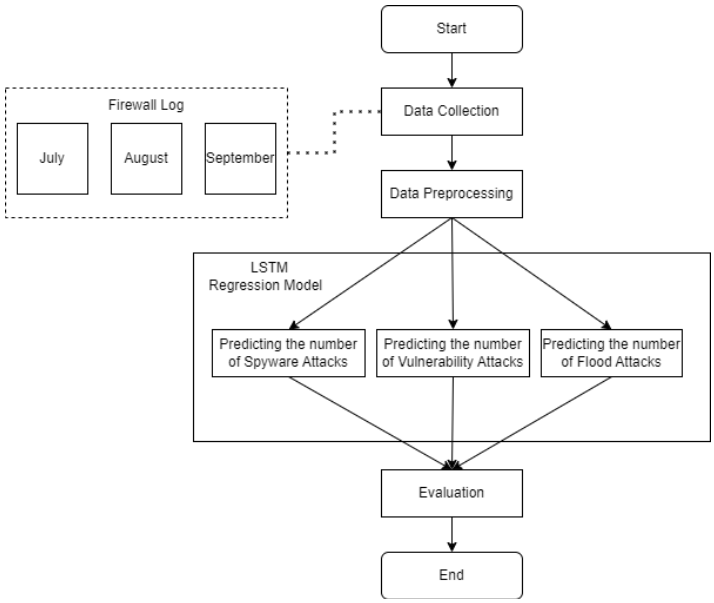

Figure 1. Research Process Stages

The research process proceeds through several key stages. Firstly, data collection involves gathering logs from the firewall spanning the period from July to September 2023. Subsequently, in the data processing phase, these logs undergo preprocessing to handle any inconsistencies and prepare them for analysis. Following this, the research employs the Long Short-Term Memory (LSTM) regression model. This model is utilized to forecast the occurrences of flood, spyware, and vulnerability attacks individually. Lastly, the evaluation phase assesses the performance and accuracy of the predictive model, providing insights into its efficacy in anticipating cybersecurity threats.

2.1. Data Collection Process

The dataset used in this research is the firewall log installed at an institution. This dataset includes comprehensive information about the network traffic passing through the firewall device, including source and destination addresses, used ports, protocols, and details about applied firewall rules. Additionally, this dataset also records security-related information regarding attempted attacks and suspicious activities, as well as time and date information associated with each entry in the firewall log. This data will serve as the basis for analyzing cybersecurity threats and developing prediction models using machine learning techniques and time-series analysis.

The dataset consists of 719,270 records combining information from three months, namely from July to September. The breakdown for each month is as follows: 327 cyber attacks occurred per hour in July, totaling 243,292 attacks; 273 cyber attacks occurred per hour in August, totaling 202,383 attacks; 393 cyber attacks occurred per hour in September, totaling 273,595 attacks. Analysis of the attack distribution shows a relatively balanced representation across different attack types, with spyware attacks accounting for 43.7% (314,282 cases), flood attacks representing 38% (273,322 cases), and vulnerability attacks comprising 17.2% (123,666 cases) of the total recorded incidents. While there are differences in the distribution, the substantial sample size for each attack type (>100,000 cases) and the moderate ratio between the largest and smallest classes (2.5:1) indicate that no specific class balancing techniques were necessary for this dataset. This natural distribution was maintained to preserve the authentic patterns and relationships within the cyber attack data.

Dataset consists of logs from a firewall, meticulously recording activities every second. Within these logs, various types of threats and content are logged. The presence of these various types underscores the need for preprocessing steps, as the data patterns associated with each type of threat vary significantly. Therefore, in Subsection 2.2 Data Preprocessing Steps, the specific patterns and trends of three particular types of threats will be elucidated through visual representations, providing a basis for the importance of developing distinct forecasting models for each type of threat. Figure 2 shows the capture results in the firewall log in Excel format.

2.2. Data Preprocessing Steps

The initial preprocessing began with comprehensive data cleaning procedures, where incomplete log entries were identified and systematically removed to maintain data integrity. Missing values within the dataset were addressed through appropriate handling mechanisms, while timestamp consistency was validated to

ensure the temporal accuracy of the security event recordings. The cleaning process also included verification of data types and formats across all fields to maintain uniformity in the dataset structure. Following the cleaning phase, feature selection was performed to extract relevant attributes from the firewall logs, including crucial parameters such as timestamps, source and destination IP addresses, port numbers, protocol types, attack categories, and firewall rule identifiers. The temporal aspect of the data was then restructured through aggregation, where individual security events were grouped into hourly intervals to facilitate time-series analysis. This aggregation process involved computing attack frequencies for each hour, with separate tracking mechanisms implemented for three distinct attack categories: spyware, flood attacks, and vulnerability exploits. The preprocessing workflow also incorporated validation checks at each stage to ensure the transformed data maintained its representational accuracy of the original security events while being optimized for subsequent modeling phases.

	A	B	C
1	Receive Time	Type	Threat/Content Type
2	9/22/2023 11:56	THREAT	vulnerability
3	9/22/2023 11:55	THREAT	flood
4	9/22/2023 11:55	THREAT	flood
5	9/22/2023 11:54	THREAT	vulnerability
6	9/22/2023 11:53	THREAT	vulnerability
7	9/22/2023 11:53	THREAT	vulnerability
8	9/22/2023 11:53	THREAT	vulnerability
9	9/22/2023 11:53	THREAT	vulnerability
10	9/22/2023 11:53	THREAT	vulnerability
11	9/22/2023 11:53	THREAT	vulnerability
12	9/22/2023 11:53	THREAT	vulnerability
13	9/22/2023 11:53	THREAT	vulnerability
14	9/22/2023 11:53	THREAT	vulnerability
15	9/22/2023 11:53	THREAT	vulnerability
16	9/22/2023 11:53	THREAT	vulnerability
17	9/22/2023 11:53	THREAT	vulnerability
18	9/22/2023 11:53	THREAT	vulnerability
19	9/22/2023 11:53	THREAT	vulnerability
20	9/22/2023 11:53	THREAT	vulnerability
21	9/22/2023 11:53	THREAT	vulnerability
22	9/22/2023 11:53	THREAT	vulnerability
23	9/22/2023 11:53	THREAT	vulnerability
24	9/22/2023 11:53	THREAT	vulnerability
25	9/22/2023 11:52	THREAT	vulnerability

Figure 2. Firewall Log

2.3. Data Transformation

Data transformation in the context of data analysis is an important process to yield more meaningful insights and easier interpretation. This transformation allows for a better understanding of network activity patterns and identification of trends or patterns that may be hidden within the original data. The first step in this process is to extract the initial firewall log data from its source, in this case, the firewall log data at a certain institution during the July-September 2023 timeframe.

Next, the initial firewall log data will be grouped based on the hour, where each entry in the data will be grouped into one-hour time intervals. Then, aggregate calculations will be performed for each category of attacks that occur within each hour. This means there will be three values reflecting the total number of vulnerabilities, spyware, and flood attacks that occurred within each hour. Figure 3 illustrates the data transformation results in the form of hourly aggregates.

Threat/Content Type	flood	spyware	vulnerability
Receive Time			
2023-07-01 12:00:00	8	7	3
2023-07-01 13:00:00	10	0	3
2023-07-01 14:00:00	7	5	18
2023-07-01 15:00:00	6	2	10
2023-07-01 16:00:00	0	5	7
...	...	...	...
2023-09-30 19:00:00	7	2	895
2023-09-30 20:00:00	0	5	10
2023-09-30 21:00:00	0	2	4
2023-09-30 22:00:00	0	11	10
2023-09-30 23:00:00	0	5	6

Figure 3. Aggregate Data in Hours

The generated figure illustrates the outcome of the data transformation process, depicting the computation of the number of cyber attacks recorded per hour. This visualization provides a clear overview of the temporal distribution of cyber threats over the specified time period. Each data point represents the count of cyber attacks within a one-hour interval, offering insights into the fluctuation and intensity of security threats throughout the day. By aggregating the attack data into hourly counts, patterns and trends in cyber threat activity can be identified, enabling a more targeted and proactive approach to cybersecurity management.

Furthermore, leveraging the aggregated data from the hourly counts of cyber attacks, the subsequent step involves visualizing the patterns and trends of each type of cyber threat. By plotting the data on a graph, distinct patterns may emerge, indicating periods of heightened or reduced threat activity. This visualization enables analysts to identify recurring trends, such as spikes in attack frequency during specific times of the day or week, as well as any long-term trends over the duration of the dataset. Understanding these patterns is crucial for developing effective mitigation strategies and allocating resources appropriately to combat cyber threats in a timely manner. Figure 4, Figure 5, and Figure 6 will show graphic visualization forms for each type of cyber attack.

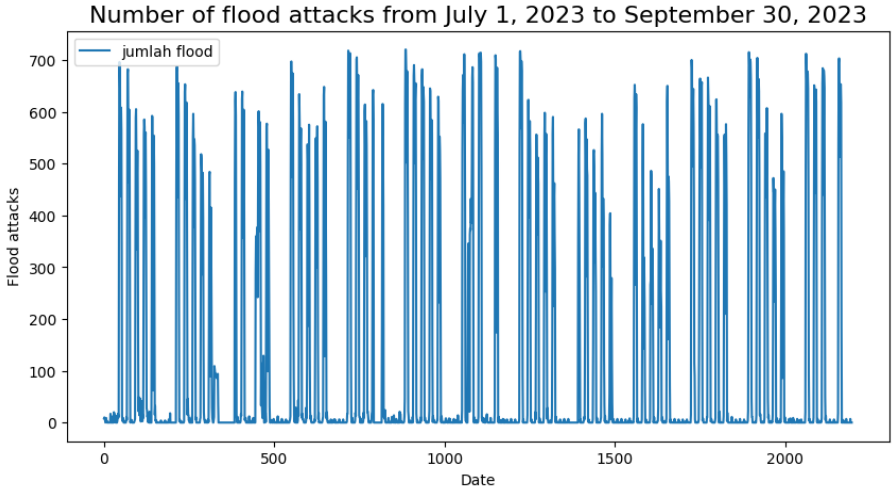

Figure 4. Flood Attack Pattern Graph

The visualization graph of flood cyber attacks depicts insightful information derived from the dataset, consisting of 2196 data points. The graph showcases the distribution of attack occurrences over time, revealing a nuanced understanding of the attack patterns. The data statistics further highlight key characteristics of the flood attacks, including an average attack rate of 124.076047 attacks per hour, with a standard deviation of 222.418445. The minimum number of attacks recorded is 0, indicating periods of no activity, while the maximum recorded attack count peaks at 720. Additionally, the quartile values provide additional context, with the median (50th percentile) attack count at 1 and the 25th and 75th percentiles at 0 and 122.5, respectively.

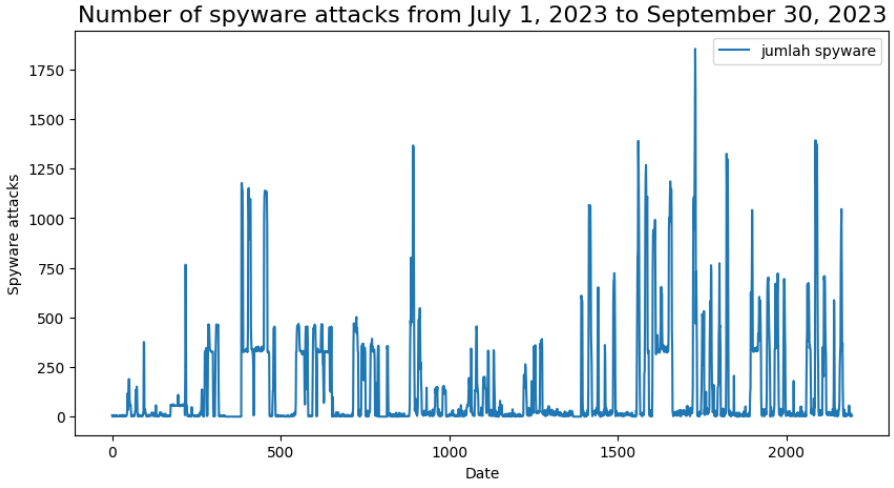

Figure 5. Spyware Attack Pattern Graph

The visualization graph of spyware cyber attacks provides valuable insights derived from a dataset comprising 2196 data points. It presents a comprehensive overview of the distribution of spyware attack occurrences over time, revealing discernible patterns and trends. The data statistics further elucidate key characteristics of spyware attacks, including an average attack rate of 143.116120 attacks per hour, with a standard deviation of 253.925969. The analysis indicates that the minimum number of spyware attacks recorded is 0, signifying periods of no activity, while the maximum attack count peaks at 1854. Additionally, quartile values offer additional context, with the median (50th percentile) attack count at 16 and the 25th and 75th percentiles at 4.75 and 197.5, respectively.

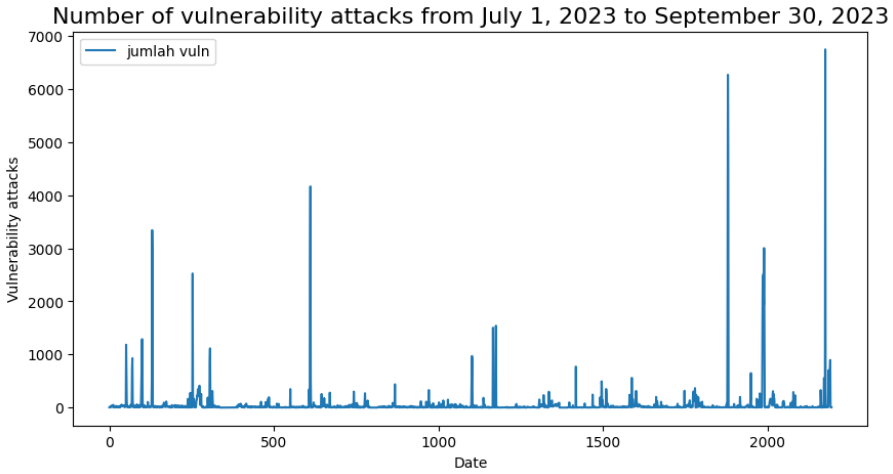

Figure 6. Vulnerability Attack Pattern Graph

The visualization graph depicting vulnerability cyber attacks presents a comprehensive overview derived from a dataset consisting of 2196 data points. It provides valuable insights into the distribution and frequency of vulnerability attacks over the observed period. The data statistics reveal essential characteristics of vulnerability attacks, including an average attack rate of 56.305100 attacks per hour, with a standard deviation of 314.855266. The analysis indicates that the minimum number of vulnerability attacks recorded is 0, suggesting periods of no activity, while the maximum attack count peaks at 6745. Quartile values offer additional context, with the median (50th percentile) attack count at 9 and the 25th and 75th percentiles at 3 and 20, respectively.

2.4. Data Scaling

Scaling data is necessary to produce a balanced distribution of values and ensure that all features have proportional weights in data analysis. This scaling process is an important step in data preparation before further modeling or analysis processes are carried out.

The scaling method used is data normalization using MinMaxScaler. Data normalization involves scaling the data to a range typically between 0 and 1. The process of data normalization is performed by subtracting the value of each data point by the minimum value in the dataset, then dividing it by the difference between the maximum and minimum values of the dataset. Data normalization is useful when the data distribution is not very normal and the range of values for each feature can vary significantly. Figure 4 shows an example result of the data normalization process.

```
array([[0.01388889],
       [0.02361111],
       [0.01527778],
       [0.01388889],
       [0.6458334 ],
       [0.9666667 ],
       [0.9638889 ],
       [0.8972222 ],
       [0.60694444],
       [0.7986111 ],
       [0.84444445],
       [0.78472227],
       [0.47777778],
       [0.01805556],
       [0.00416667]], dtype=float32)
```

Figure 7. Data Normalization using MinMaxScaler

The visualization of the data scaling process utilizing MinMaxScaler illustrates the normalization of the dataset, ensuring uniformity and consistency in the numerical ranges of the features. By applying MinMaxScaler, the data values are transformed to fall within a specified range, typically between 0 and 1, preserving the relative differences between the data points. This normalization facilitates the comparison and interpretation of the features, particularly in machine learning models where consistency in scale is crucial for accurate predictions. The MinMaxScaler effectively scales the data, enhancing model performance and reliability by mitigating the impact of varying feature scales.

2.5. Regression Model

The regression process is the next stage in this research, where the dataset that has undergone pre-processing will be used to develop predictive models of cybersecurity threats. In this stage, three regression models for spyware, flood, and vulnerability will be developed using LSTM to evaluate their performance. The process of building these regression models is illustrated in Figure 7.

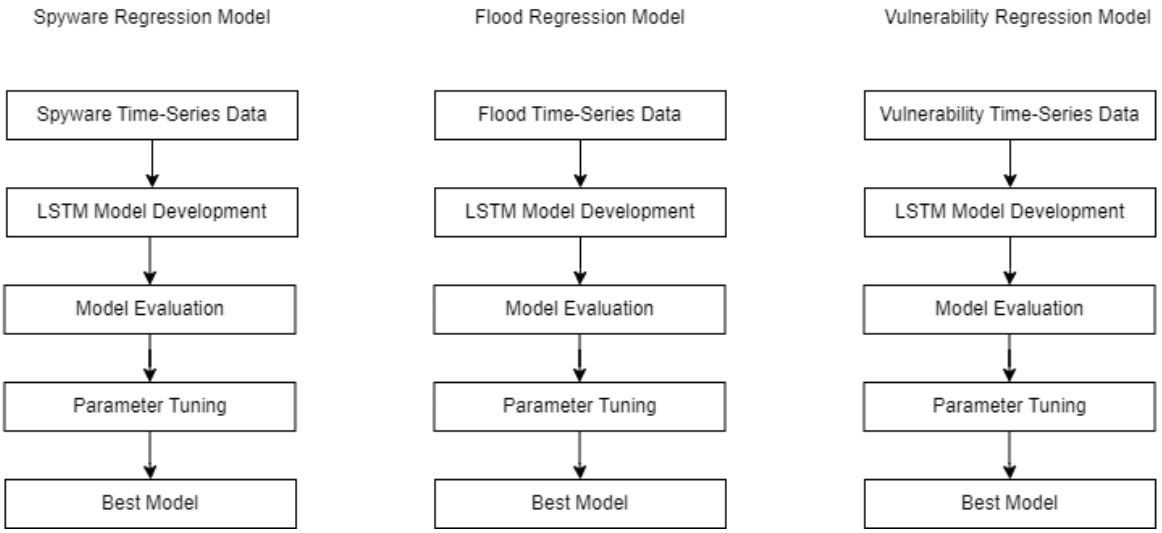

Figure 8. Regression Model

In the regression stage, the focus shifts to developing predictive models for cybersecurity threats using the pre-processed dataset. Specifically, the LSTM algorithm is employed to construct regression models for spyware, flood, and vulnerability threats. LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) known for its effectiveness in capturing long-term dependencies in sequential data, making it well-suited for analyzing time-series data like cybersecurity logs. By employing the LSTM approach, the aim is to evaluate its performance in predicting the occurrences of spyware, flood, and vulnerability threats within the dataset.

The existing data will be processed by dividing the dataset into two separate subsets: training data and testing data. The data-splitting process is done by separating the features from the labels or targets to be predicted. These features become independent variables (X), while the labels become dependent variables (y). The dataset is then split such that 80% of the data is allocated to training and 20% to testing.

2.6. LSTM Model Architecture and Hyperparameter Configuration

The implemented Long Short-Term Memory (LSTM) architecture, a specialized variant of Recurrent Neural Network (RNN), was meticulously designed to effectively capture temporal patterns in cyber attack occurrences while maintaining long-term dependencies in the time series data. The network architecture consists of multiple layers strategically structured to optimize predictive performance. The input layer processes sequences determined by a carefully selected windowing/timestep parameter, which was experimentally optimized for each attack type to capture relevant historical patterns while avoiding the inclusion of unnecessary noise in the predictions. The model's deep architecture incorporates multiple LSTM layers, with experimental configurations testing both single- and dual-layer implementations (1 and 2 layers) to evaluate depth requirements for pattern recognition. Each LSTM layer was configured with variable hidden sizes of 50 and 75 units, allowing for comprehensive testing of the model's capacity to capture complex temporal patterns in the attack data. The model's hyperparameters underwent extensive experimental tuning:

batch sizes of 8 and 16 were evaluated to optimize the trade-off between training stability and computational efficiency, while learning rates of 0.001 and 0.0001 were tested to ensure effective gradient descent during training without compromising convergence. The training process was configured to run for 100 epochs with an implemented early stopping mechanism monitoring the validation loss to prevent overfitting. Figure 12 shows the architecture of the LSTM model.

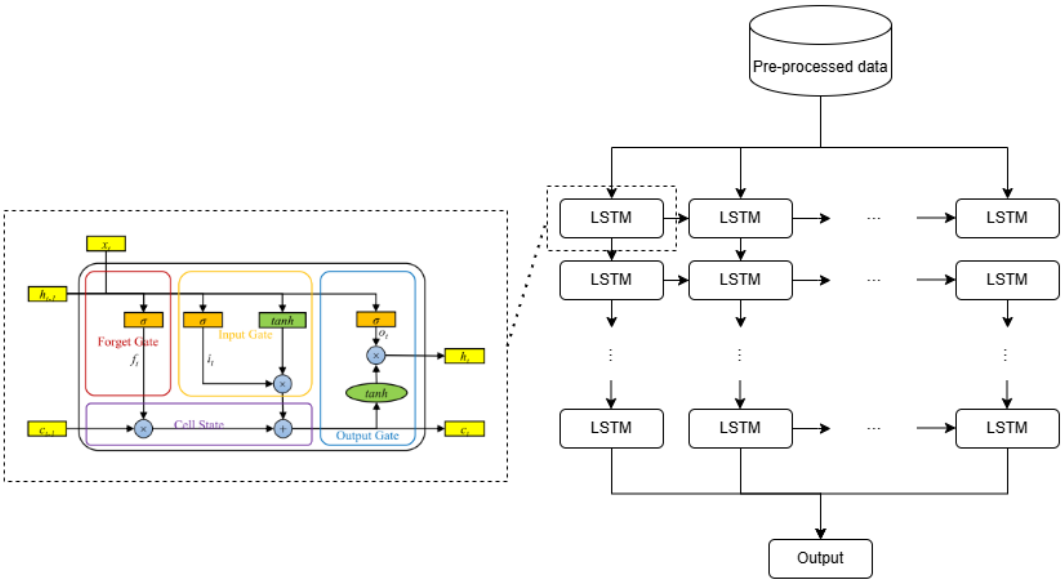

Figure 9 Architecture of LSTM Model

2.7. Forecasting Performance Evaluation Metrics

Forecasting metric evaluation refers to the process of measuring how accurate a forecasting model is in predicting future values. In forecasting metric evaluation, various metrics and measures are used to assess the model's performance. Measurements based on absolute or squared errors are also known as scale-dependent measurements because their scale depends on the data scale [14]. Common metrics used in scale-dependent measurement include Root Mean Squared Error (RMSE) and R Squared (R2).

The Root Mean Squared Error (RMSE) measures the average magnitude of the errors between the predicted values and the actual observed values. RMSE calculates the square root of the average of the squared differences between predicted and observed values. This metric provides a measure of the model's predictive performance, with lower RMSE values indicating a better fit between the predicted and actual values.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n}} \quad (1)$$

Where:
x_i = actual value
$\bar{x}$ = predicted value
n = number of observation

R-squared (R2) is a statistical measure used to evaluate the goodness of fit of a regression model. It represents the proportion of the variance in the dependent variable that is explained by the independent variables in the model. In other words, R-squared indicates how well the independent variables explain the variability of the dependent variable. The value of R-squared ranges from 0 to 1, where 0 indicates that the model does not explain any of the variability of the dependent variable, and 1 indicates that the model perfectly explains all the variability.

$$\text{R2} = 1 - \frac{\text{RSS}}{\text{TSS}} \quad (2)$$

Where:
RSS = sum of squared residuals
TSS = total sum of squares

3. RESULTS AND DISCUSSION

The experimental results demonstrate varying degrees of success in predicting different types of cyber attacks using LSTM models. For flood attacks, the model achieved its highest performance with an RMSE of 59.8810, R-squared of 0.9169, and MAE of 38.9704 after standardization, indicating strong predictive capability with approximately 91% of variance explained. The spyware attack predictions showed moderate performance, achieving an RMSE of 133.9567, an R-squared of 0.7719, and MAE of 72.0125 after standardization, suggesting the model captures general trends but with less accuracy compared to flood attack predictions. In contrast, vulnerability attack predictions showed limited capability, with an RMSE of 503.5521, R-squared of 0.2358 and MAE 110.3733 after standardization, indicating significant challenges in capturing these attack patterns.

Data preprocessing techniques, particularly standardization, consistently improved model performance across all attack types compared to raw data, though the degree of improvement varied significantly. Flood attacks showed the most substantial improvement from preprocessing, while vulnerability predictions showed minimal enhancement despite the same preprocessing techniques being applied. This variance in improvement suggests that the underlying patterns and characteristics of different attack types significantly influence the effectiveness of preprocessing methods.

3.1 LSTM

The LSTM model developed will compare the results for three types of cyberattacks, namely predicting the number of flood, spyware, and vulnerability attacks. The variable to be predicted is the number of attacks that will occur in the next hour. The LSTM model will utilize several training parameters that refer to a number of variables adjusted during the model training process. This allows the model to learn from the provided data and improve its ability to make accurate predictions. Some parameters used in model training include hidden size, layer, batch size, and learning rate.

Each model will be evaluated using performance evaluation metrics, namely Root Mean Squared Error (RMSE), R-squared, and Mean Absolute Error (MAE). RMSE measures the average magnitude of error between predicted values and actual values, with lower values indicating better model performance. On the other hand, R-squared represents the proportion of the dependent variable's variance that can be predicted from the independent variable, with values closer to 1 indicating a better fit of the model to the data. Mean Absolute Error (MAE) provides an additional measure of model accuracy by calculating the average absolute difference between predicted and actual values. In the RMSE metric, the best value is the lowest error rate, while in the R-squared metric, the best value is close to 1, and for MAE, lower values indicate more accurate predictions.

3.1.1 LSTM - Spyware

The LSTM architecture employed for modeling the number of spyware attacks adopts a specific configuration, where the number of timesteps is set to 3. In this setup, both the feature and target data are defined as follows: The feature data comprises slices of the dataset with a length of 3 timesteps, denoted as dataset[i:i+ timesteps]. For example, if the timestep is set to 3 and the index i i is equal to 1, then the feature is obtained by selecting the data points from index 1 to index 4 (inclusive). On the other hand, the target corresponds to the data point immediately following the end of the feature sequence, indicated as dataset[i+ timesteps:i+ timesteps +1]. Continuing the example with i=3 and a step size of 3, the target data would correspond to the values at indices 4 to 5 in the dataset. This configuration allows the LSTM model to process sequential data effectively, capturing patterns and dependencies within the data over the specified timesteps.

The initial evaluation of spyware attack modeling based on the LSTM architecture, as described earlier, is presented in Table 1. The performance metrics, such as RMSE and R-squared, are computed using the parameters outlined in Table 2.

Table 1. Preliminary evaluation results for Spyware Model

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	120.8641	615.964.672	Training	105.5611
			Testing	136.9093
R Squared			Training	0.8293
			Testing	0.7433
MAE			Training	48.5201
			Testing	69.2118

Table 2. List of Parameters Used in Training Spyware Model

Parameter	Value
Epoch	250
Hidden Size	75
Layer	1
Batch Size	8
Learning Rate	0.001

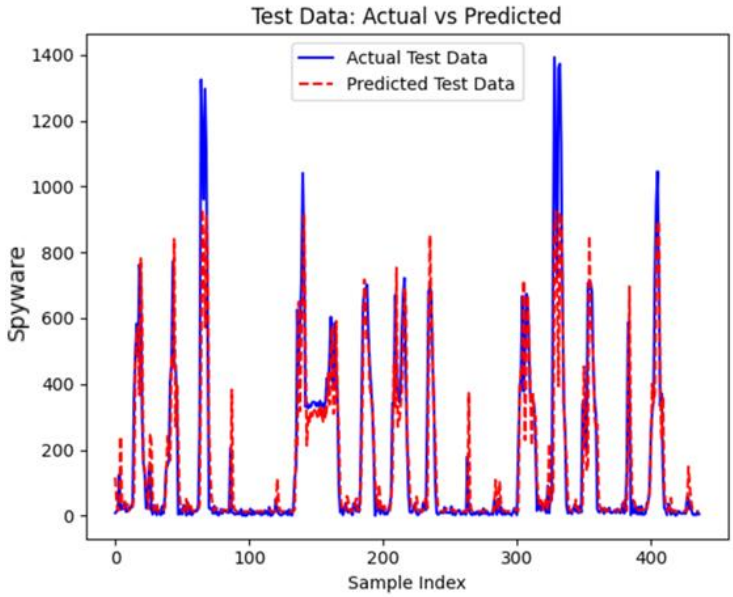

Figure 10. Comparison of Actual and Predicted Preliminary Spyware Data

The evaluation results from Table 1, which utilized the parameters outlined in Table 2, have not shown satisfactory performance for predicting spyware attacks. However, this outcome does not discount the potential of achieving better results. There are still opportunities to fine-tune the model by exploring alternative parameter configurations or adjusting the architecture to enhance its predictive capabilities.

Since the initial evaluation results suggest the potential for improvement, normalization of the training and testing data was performed using the MinMaxScaler function. This normalization process aims to scale the data to a uniform range, which can aid in optimizing model performance. The evaluation results following this normalization process are presented in Table 3, while the parameters used are outlined in Table 4.

Table 3. Evaluation Results After Data Normalization

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	201.2323	619.315.200	Training	95.7142
			Testing	138.7175
R Squared			Training	0.8498
			Testing	0.7545
MAE			Training	50.6956
			Testing	71.1422

Table 4. List of Parameters Used in Training Spyware Model

Parameter	Value
Epoch	250
Hidden Size	75
Layer	2
Batch Size	8
Learning Rate	0.001

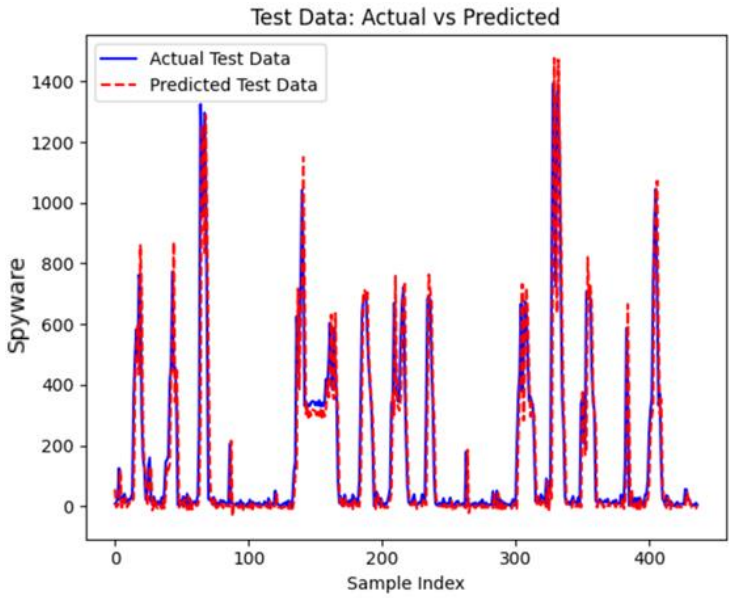

Figure 11. Comparison of Actual and Predicted Normalized Spyware Data

The evaluation of the spyware model before and after data normalization reveals notable differences in performance metrics. Before normalization, the model exhibited certain scores, while after normalization, these scores showed variations. Specifically, the RMSE scores for both the training and testing sets experienced changes, as did the R-squared scores, reflecting shifts in predictive accuracy. Additionally, the parameter configuration, particularly the number of layers in the model architecture, also underwent alteration. These adjustments aimed to refine the model's learning process and enhance its ability to capture underlying patterns in the data. Overall, the normalization of the data contributed to improvements in the spyware model's performance, as evidenced by the shifts in evaluation metrics and parameter values.

In addition to data normalization using MinMaxScaler, another approach to preprocessing involves data standardization using the StandardScaler function. This alternative preprocessing technique aims to standardize the distribution of the features, ensuring that they exhibit a mean of zero and a standard deviation of one. By applying this method, the spyware model undergoes further refinement, allowing for improved convergence during training and potentially enhancing its predictive performance. Following data normalization, the evaluation process yielded results as depicted in Table 5, and the model was trained with parameters outlined in Table 6.

Table 5. Evaluation Results After Data Standardization

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	193.6512	632.823.808	Training	93.8280
			Testing	133.9567
R Squared			Training	0.8573
			Testing	0.7719
MAE			Training	50.8908
			Testing	72.0125

Table 6. List of Parameters Used in Training Spyware Model

Parameter	**Value**
Epoch	250
Hidden Size	50
Layer	2
Batch Size	8
Learning Rate	0.001

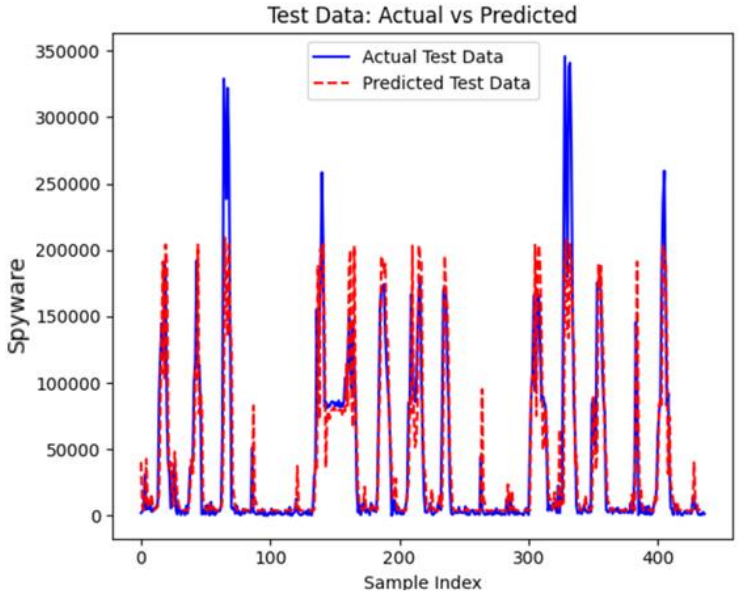

Figure 12. Comparison of Actual and Predicted Standardized Spyware Data

The comparison between the evaluation results of the spyware model before and after data standardization reveals notable differences. Before standardization, the spyware model exhibited slightly higher RMSE scores for both training and testing datasets, along with marginally higher R-squared scores for training data. Conversely, after standardization, there was a decrease in the RMSE scores for both training and testing datasets, indicating a reduction in prediction errors. Additionally, the R-squared scores remained relatively consistent, suggesting that the model's explanatory power was maintained even after data standardization. Notably, the parameter Hidden Size underwent a change, decreasing from 75 before standardization to 50 after standardization. This adjustment reflects the impact of data standardization on the model's architecture, potentially optimizing its performance by adapting to the standardized input data distribution.

When comparing our results with the study by [10], several insights emerge. Their research evaluated various network intrusion types using mean absolute error (MAE) as the primary metric. For attacks related to information security and privacy violations, which are most comparable to our spyware analysis, their LSTM model achieved the following results:

- Attempted information leak: MAE of 0.519.
- Potential corporate privacy violation: MAE of 0.699.
- Executable code detection: MAE of 0.111.

The difference in MAE values between our study and [10], despite both utilizing normalization techniques, can be attributed to several fundamental methodological differences. The primary distinction lies in the nature of attack detection and prediction granularity. While [10] focused on binary detection of intrusion events - essentially predicting whether an attack occurred or not - our research aims to predict the specific frequency and volume of spyware attacks per hour. This quantitative prediction approach inherently involves more complex patterns and higher potential for variation in predictions, leading to larger MAE values. The temporal resolution of predictions also plays a crucial role in these differences. Our model's requirement to make precise hourly predictions introduces additional complexity compared to potentially coarser time intervals used in [10], as shorter prediction windows typically demand higher precision and are more susceptible to fluctuations in attack patterns.

3.1.2 LSTM - Flood

In flood modeling, the architecture of Long Short-Term Memory (LSTM) networks differs from that used in spyware modeling, particularly in terms of the timesteps employed. In flood modeling, LSTM networks typically utilize a timestep value of 8. This means that the input sequences, or features, are constructed by selecting consecutive sets of 8 data points from the dataset. For instance, if the timestep is set to 8 and the starting index is 1, then the feature would encompass data points from 1 to 9. The target, on the other hand, is determined by selecting the data point immediately following the last point in the feature sequence. For example, if the feature sequence spans from 1 to 9, the target would be the data point at index 9+1, thus ranging

from 9 to 10. This approach allows the LSTM network to capture temporal dependencies and patterns within the flood data, enabling effective modeling and prediction of flood occurrences.

The initial evaluation results of the flood attack model, as presented in Table 7, and the parameter settings used for this evaluation are outlined in Table 8.

Table 7. Preliminary evaluation results for Flood Model

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	214.4665	630.755.328	Training	67.8968
			Testing	68.6309
R Squared			Training	0.8913
			Testing	0.8847
MAE			Training	33.9452
			Testing	34.8665

Table 8. List of Parameters Used in Training Flood Model

Parameter	Value
Epoch	250
Hidden Size	50
Layer	2
Batch Size	8
Learning Rate	0.001

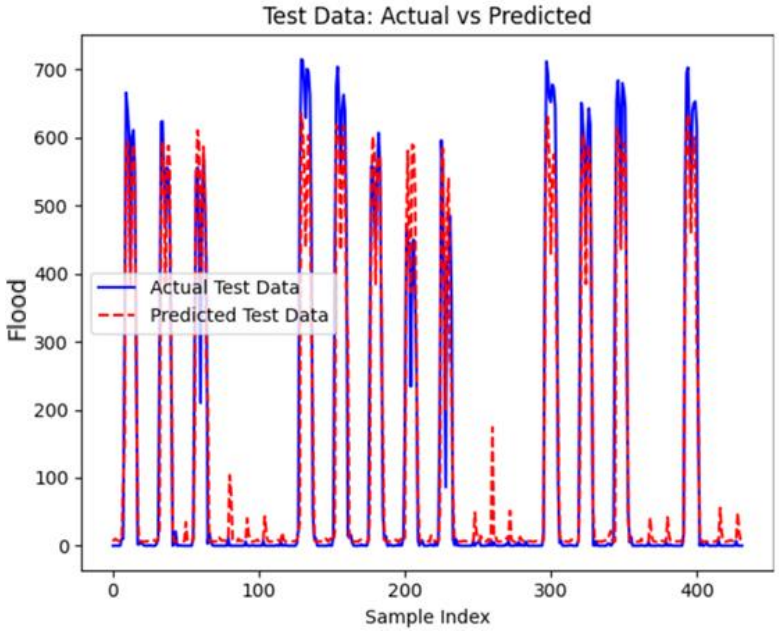

Figure 13. Comparison of Actual and Predicted Preliminary Flood Data

The evaluation results from Table 7, utilizing the parameters outlined in Table 8, have demonstrated promising performance, indicating the model's ability to effectively capture patterns and predict flood attacks. However, there remains room for further improvement. To enhance the model's performance, several additional approaches will be explored. These include data normalization and standardization techniques, which aim to preprocess the data to a standardized scale, facilitating more effective model training. Additionally, parameter tuning will be conducted to optimize the model's hyperparameters, further refining its predictive capabilities.

The first experiment aims to improve the model's performance by normalizing both the training and testing data using the MinMaxScaler function. This normalization process ensures that the data features are transformed to have a mean of 0 and a standard deviation of 1, enhancing the model's ability to learn from the input data effectively. The evaluation results obtained from this experiment are presented in Table 9, while the parameters used for this experiment are detailed in Table 10.

Table 9. Evaluation Results After Data Normalization

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	209.5648	620.834.816	Training	62.7192
			Testing	58.3243
R Squared			Training	0.9207
			Testing	0.9322
MAE			Training	33.4012
			Testing	31.5064

Table 10. List of Parameters Used in Training Flood Model

Parameter	Value
Epoch	250
Hidden Size	50
Layer	2
Batch Size	8
Learning Rate	0.001

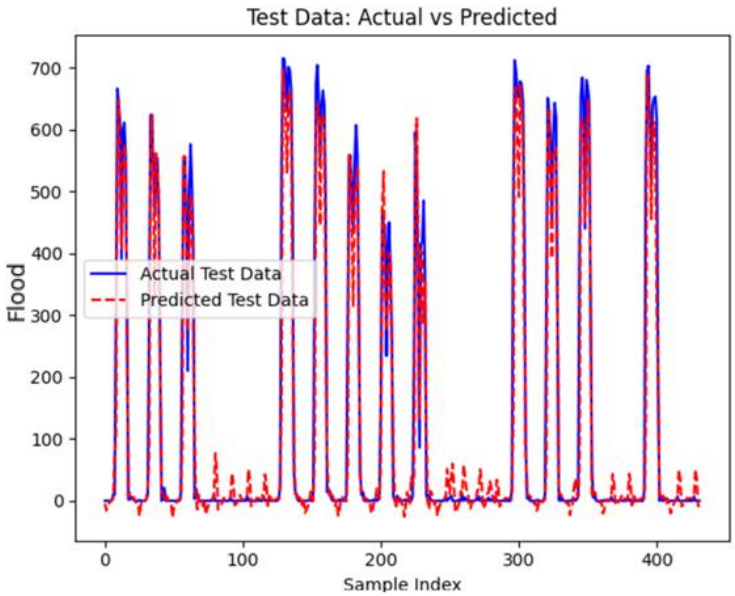

Figure 14 Comparison of Actual and Predicted Normalized Flood Data

The evaluation of the flood model before and after data normalization reveals notable improvements in performance metrics. Prior to normalization, the RMSE scores for both the training and testing sets were higher compared to the scores obtained after normalization. This indicates a reduction in prediction errors, with the RMSE scores showing a decrease after normalization. Similarly, the R-squared scores remained consistent before and after normalization, suggesting that the model's ability to explain the variance in the data was unaffected by the normalization process.

In the second experiment, further efforts were made to enhance the flood model's performance by standardizing both the training and testing data using the StandardScaler function. Standardization ensures that the data features are transformed to have a mean of 0 and a standard deviation of 1, which can facilitate more effective model training and convergence. The evaluation results obtained from this experiment are presented in Table 11, while the parameters used remain consistent with those outlined in Table 12.

Table 11. Evaluation Results After Data Standardization

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	232.2856	623.083.520	Training	61.3430
			Testing	59.8810
R Squared			Training	0.9071
			Testing	0.9169
MAE			Training	35.5937
			Testing	38.9704

Table 12. List of Parameters Used in Training Flood Model

Parameter	Value
Epoch	250
Hidden Size	75
Layer	2
Batch Size	8
Learning Rate	0.001

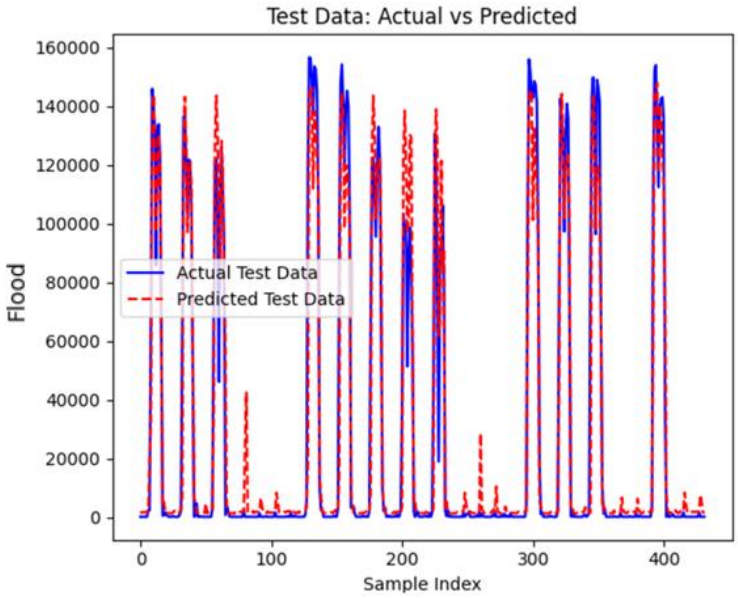

Figure 15. Comparison of Actual and Predicted Standardized Flood Data

Following the standardization of data in the flood model, there were discernible changes in the evaluation metrics. Notably, there was a reduction in both RMSE scores for the training and testing sets after standardization, indicating a decrease in prediction errors. This reduction in error rates demonstrates an improvement in the model's accuracy in forecasting flood occurrences. However, there was a slight decrease in the R-squared scores for both training and testing datasets post-standardization. While this decrease suggests a slight reduction in the model's ability to explain the variance in the data, the overall impact on predictive performance remains positive due to the more significant reduction in RMSE. It's important to note that alongside data standardization, the hidden size parameter was also adjusted, increasing from 50 to 75.

When comparing our results with those reported by [8] in their comparative study of ARIMA and LSTM models, we observe similar patterns of LSTM's superiority in time series prediction. Their study demonstrated significant RMSE reductions when using LSTM compared to ARIMA across various financial datasets, with an average reduction of 87.445% for stock indices and 84.394% for individual stocks. Our flood prediction model achieved an RMSE of 68.6309 for the testing dataset, which shows comparable effectiveness when considering the different nature of the data.

These results suggest that while our LSTM implementation for flood prediction may not show the same magnitude of improvement as seen in financial time series prediction, it still demonstrates significant predictive capability, particularly after data normalization. The different scales of improvement can be attributed to the distinct characteristics of flood data compared to financial time series, highlighting the importance of domain-specific model optimization.

3.1.3 LSTM - Vulnerability

In vulnerability attack modeling, the use of LSTM with different architectures across various timesteps is crucial.Unlike the flood model, which utilizes a timestep of 8, the LSTM architecture employed for vulnerability modeling follows a different approach. With a timestep of 3, similar to the spyware model, it focuses on capturing intricate patterns within shorter sequences of data.

The initial evaluation of the vulnerability attack model, as depicted in Table 13, reveals the performance metrics of Root Mean Square Error (RMSE) and R-squared. These metrics provide insights into the model's predictive accuracy and explanatory capability based on the parameters outlined in Table 14.

Table 13. Preliminary evaluation results for Vulnerability Model

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	122.4572	594.210.816	Training	175.3885
			Testing	552.9678
R Squared			Training	0.5555
			Testing	0.2358
MAE			Training	40.3189
			Testing	113.2646

Table 14. List of Parameters Used in Training Vulnerability Model

Parameter	Value
Epoch	250
Hidden Size	75
Layer	2
Batch Size	16
Learning Rate	0.001

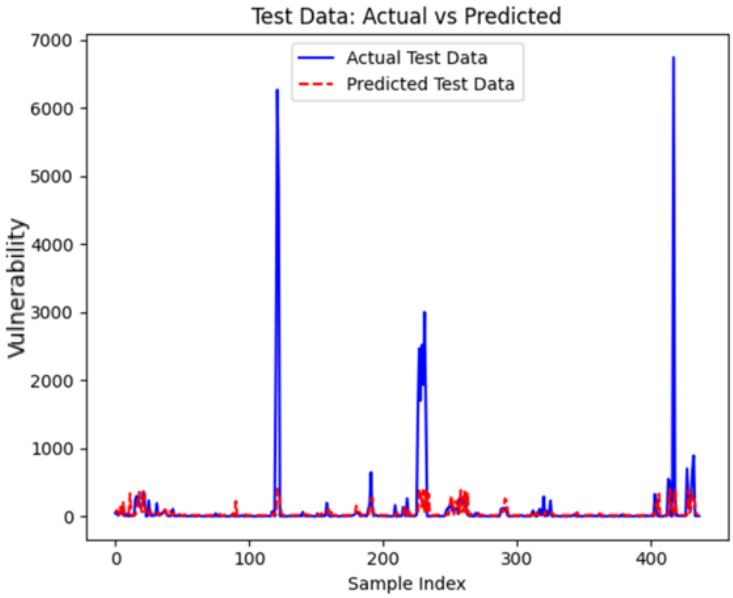

Figure 16. Comparison of Actual and Predicted Preliminary Vulnerability Data

The evaluation results from Table 13, utilizing the parameters outlined in Table 14, indicate suboptimal performance due to high error rates reflected in the RMSE measurements and significantly low accuracy in R-squared measurements. To enhance these evaluations, several strategies will be employed. Firstly, data normalization will be conducted to ensure consistent scales across features, aiding the model in capturing underlying patterns effectively. Secondly, data standardization will be implemented to adjust the distribution of features, potentially improving the model's convergence and performance. Additionally, parameter tuning will be carried out to optimize the model's configuration for better predictive accuracy and explanatory power.

In the quest for improved results, the first experiment entails normalizing both the training and testing data using the MinMaxScaler function. This preprocessing step aims to scale the data to a uniform range, facilitating better convergence and performance of the model. The evaluation outcomes are then documented in Table 15, with the model parameters aligned with those specified in Table 16.

Table 15. Evaluation Results After Data Normalization

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	139.6383	617.476.096	Training	161.4050
			Testing	567.1208
R Squared			Training	0.4954
			Testing	0.2106
MAE			Training	39.7142
			Testing	114.4881

Table 16. List of Parameters Used in Training Vulnerability Model

Parameter	Value
Epoch	200
Hidden Size	50
Layer	1
Batch Size	8
Learning Rate	0.001

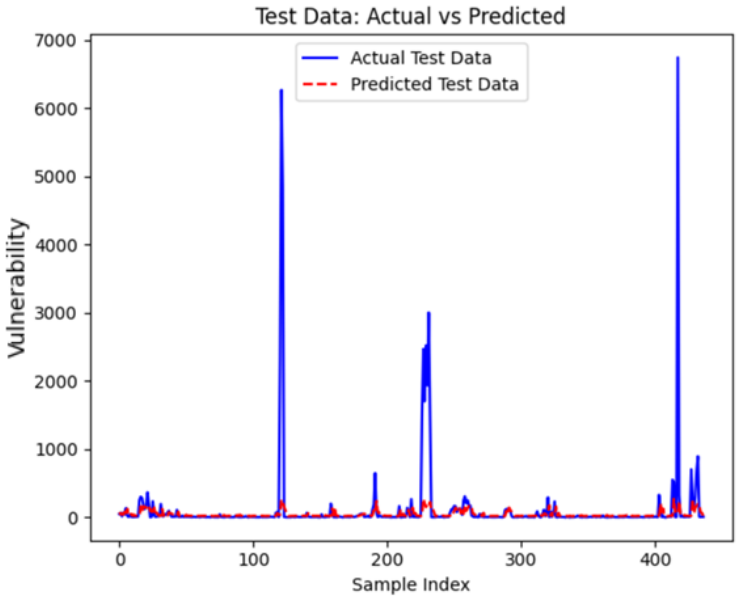

Figure 17 Comparison of Actual and Predicted Normalized Vulnerability Data

The evaluation results of the vulnerability model exhibit notable differences before and after data normalization. Prior to normalization, the model displayed RMSE scores indicating relatively high error rates, with the training set at a score of 174.1728 and the testing set at 554.7065. The R-squared scores were also suboptimal, standing at 0.5555 for the training set and 0.2358 for the testing set. After normalization, although there were improvements in some metrics, the model still struggled to achieve satisfactory results. The RMSE scores for both training and testing sets decreased but remained relatively high, while the R-squared scores experienced slight declines. Notably, changes were made to the model parameters during normalization. The number of epochs decreased from 250 to 200, the hidden size reduced from 75 to 50, the number of layers decreased from 2 to 1, and the batch size decreased from 16 to 8. Despite these adjustments, the model's performance improvements were marginal, underscoring the need for further refinement and optimization to enhance its effectiveness in the vulnerability model.

In pursuit of more optimal outcomes, a second experiment was conducted involving the standardization of both training and testing data using the StandardScaler function. This preprocessing step aims to transform the data to have a mean of 0 and a standard deviation of 1, facilitating better convergence and model performance. The evaluation results of this approach are detailed in Table 17, with the model parameters aligned with those specified in Table 18. While the initial normalization led to marginal improvements, standardization offers another avenue for refining the model's performance.

Table 17. Evaluation Results After Data Standardization

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	144.9539	607.277.056	Training	133.7222
			Testing	503.5521
R Squared			Training	0.5555
			Testing	0.2358
MAE			Training	32.7462
			Testing	110.3733

Table 18. List of Parameters Used in Training Vulnerability Model

Parameter	Value
Epoch	250
Hidden Size	75
Layer	2
Batch Size	16
Learning Rate	0.001

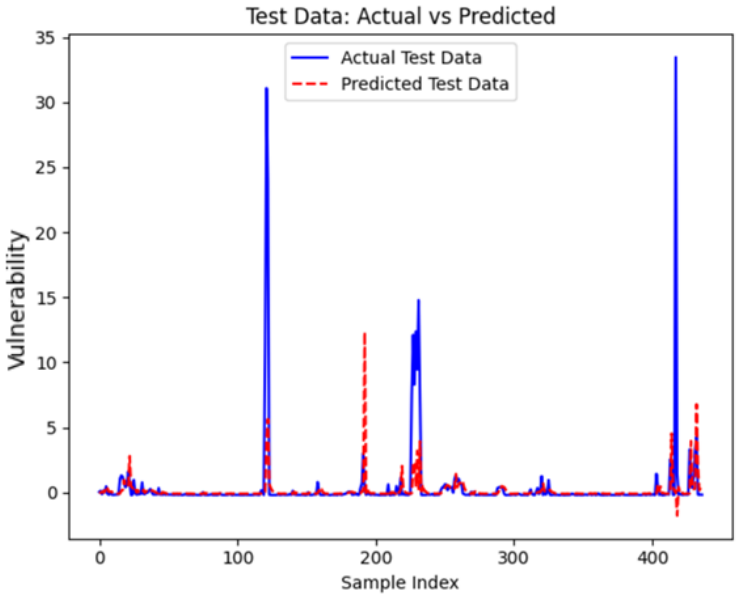

Figure 18. Comparison of Actual and Predicted Standardized Vulnerability Data

Despite the improvements observed after normalization and standardization, the vulnerability model's evaluation scores still fall short of expectations. Several factors may contribute to this persistent suboptimal performance. One plausible reason could be the complexity and variability of vulnerability attack patterns, which may not be adequately captured by the current model architecture and feature set. Additionally, the inherent noise and imbalances within the dataset could hinder the model's ability to generalize effectively to unseen data, leading to inflated error rates and diminished predictive accuracy. Furthermore, the selected model parameters, although optimized through experimentation, may not fully align with the underlying dynamics of vulnerability attacks, necessitating further fine-tuning and exploration. Lastly, external factors such as evolving attack tactics and data collection biases may introduce uncertainties that challenge the model's robustness and generalization capabilities.

In comparing our results with [9], several significant distinctions emerge in model performance and application context. Their study achieved notably lower MAE values, ranging from 0.001 to 1.45 for various network intrusion types, while our model demonstrated higher MAE values. This disparity can be attributed to the fundamental differences in our prediction tasks, where our study tackles the broader and more complex domain of vulnerability predictions. The variation in performance metrics also reflects the different scales and granularity of our target variables, as well as the inherent complexities in the underlying data distributions. These differences underscore the unique challenges presented by vulnerability prediction compared to specific intrusion detection tasks.

3.2 FB Prophet

In addition to the LSTM model evaluation, an alternative time series modeling approach using FB-Prophet was investigated to predict cyber attacks across three different attack types: spyware, flood, and vulnerability attacks. The FB-Prophet model was configured with carefully selected parameters to optimize its predictive performance.

The FB-Prophet model was meticulously configured through a manual parameter search approach to optimize prediction accuracy across different cyber attack types. The key parameters investigated included:

- **Changepoint Prior Scale**: Controls the model's flexibility in detecting trend changes. Tested values: 0.009, 0.007, 0.005, 0.001, 0.01, 0.05, 0.5, 1, and 5.
- **Seasonality Prior Scale**: Determines the model's sensitivity to seasonal patterns. Tested values: 100, 60, 30, 20, 10, 1, 0.1, 0.08, 0.05, and 0.01.
- **Interval Width**: Defines the confidence interval for predictions. Tested values: 100%, 95%, 90%, 85%, 80%, and 75%.

The manual parameter search aimed to identify the most optimal combination that would yield the best predictive performance for each attack type.

3.2.1 FB Prophet – Spyware

For spyware attack modeling, the FB-Prophet architecture was optimized using specific parameters. The Changepoint Prior Scale was set to 0.005, enabling the model to be more flexible in detecting sudden changes in data trends. The Seasonality Prior Scale was set to 0.1 to capture stronger seasonal patterns in spyware attack data. An Interval Width of 95% was established to provide a broader confidence interval that accommodates the high uncertainty typical in cybersecurity contexts. The evaluation results for the spyware FB-Prophet model are presented in Table 19.

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	0.6898	263.036.928	Training	211.3643
			Testing	216.6473
R Squared			Training	0.2744
			Testing	0.3751
MAE			Training	148.4477
			Testing	155.3015

The FB-Prophet model demonstrated unsatisfactory performance in predicting spyware attacks compared to the LSTM model. Notably, the model exhibited higher RMSE values and lower R-squared scores for both training and testing datasets, indicating higher prediction errors and a reduced ability to explain data variability.

3.2.2 FB Prophet – Flood

For flood attack modeling, the FB-Prophet architecture adopted a structured approach with carefully selected parameters. The Changepoint Prior Scale was set to 0.005 to sensitively detect significant trend changes, such as sudden increases or decreases in attack patterns. The Seasonality Prior Scale was set to 60 to capture seasonal patterns potentially influencing flood attacks. An Interval Width of 0.95 provided a sufficiently broad confidence interval to accommodate potential variations in predicted attack data. The evaluation results for the flood FB-Prophet model are presented in Table 20.

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	0.6368	262,516,736	Training	139.5942
			Testing	145.1554
R Squared			Training	0.5930
			Testing	0.6214
MAE			Training	109.3502
			Testing	112.1641

The FB-Prophet model for flood attacks revealed relatively lower performance compared to the LSTM model. The comparison showed higher RMSE values and lower R-squared scores for both training and testing datasets, indicating larger prediction errors and more limited model capabilities in explaining flood attack data variability.

3.2.3 FB Prophet – Vulnerability

In vulnerability attack modeling, the FB-Prophet parameters were configured to enhance predictive accuracy. The Changepoint Prior Scale was set to 0.5, influencing the model's flexibility in detecting sudden changes in vulnerability data. The Seasonality Prior Scale was set to 1, allowing the model to adjust to potential seasonal patterns affecting vulnerability trends. An Interval Width of 0.95 was maintained to provide a consistent confidence interval. The evaluation results for the vulnerability FB-Prophet model are presented in Table 21.

Measure	Processing Time	Consumed Memory	Sample	Score
RMSE	0.6099	258.109.440	Training	195.5025
			Testing	581.7067
R Squared			Training	0.0484
			Testing	-0.0267
MAE			Training	59.1409
			Testing	134.4789

The vulnerability attack predictions using FB-Prophet mirrored the performance of spyware and flood models, showing significantly lower performance compared to the LSTM approach. The evaluation across all three attack types demonstrated that FB-Prophet consistently produced higher RMSE values and lower R-squared scores for both training and testing datasets.

4. LIMITATIONS AND FUTURE WORK

Despite promising results in cyber attack prediction using LSTM networks, several key limitations exist in the current implementation. The model faces significant computational constraints, particularly in processing large-scale attack data in real-time environments, with resource requirements increasing substantially when handling multiple attack types simultaneously. Implementation challenges extend to scalability issues in enterprise-level deployments, including difficulties in efficiently handling high-volume network traffic and complex integration requirements with existing Security Information and Event Management (SIEM) systems. Additionally, the model's effectiveness heavily depends on historical attack data quality, potentially limiting its ability to detect and predict novel attack patterns.

Future work should focus on several critical areas for improvement. Model enhancements could involve investigating hybrid approaches combining LSTM with other deep learning architectures and developing adaptive learning mechanisms for real-time updates. Architectural improvements should address scalability through distributed processing architectures and efficient data partitioning strategies. Integration capabilities could be enhanced by developing standardized APIs and plug-and-play modules for popular SIEM platforms. Advanced analytics research, including causality analysis and explainable AI techniques, would improve the model's interpretability and effectiveness.

The implementation of these improvements can be structured across different time horizons, from short-term goals focusing on basic model improvements and initial scalability solutions (6-12 months) to medium-term objectives involving enhanced prediction capabilities (1-2 years) and long-term goals addressing advanced analytics and enterprise-ready deployment solutions (2+ years). While the current study demonstrates the potential of LSTM networks in cyber attack prediction, these proposed improvements and research directions provide a structured framework for advancing both the theoretical foundation and practical application of the system in real-world cybersecurity applications.

5. CONCLUSION

This research, we developed and evaluated Long Short-Term Memory (LSTM) regression models to predict three types of cyber attacks: flood, spyware, and vulnerability. Our models aimed to forecast the number of attacks that would occur in the next hour. Various training parameters were optimized, including hidden size, layer, batch size, and learning rate, to enhance the model's learning and predictive accuracy. For each type

of attack, we conducted initial evaluations and explored different data preprocessing techniques, such as normalization using MinMaxScaler and standardization using StandardScaler, to improve model performance.

The experiments demonstrate that preprocessing techniques such as normalization and standardization can positively impact model performance by reducing prediction errors and enhancing accuracy. However, the extent of improvement varies across different types of cyber attacks. The flood attack model benefited the most from these techniques, while the vulnerability attack model showed limited improvement, underscoring the need for more sophisticated modeling approaches for complex attack patterns.

In conclusion, while the LSTM models developed in this research exhibit potential in predicting cyber attacks, further optimization and exploration of advanced techniques are required to achieve more robust and accurate predictions, particularly for complex and variable attack types like vulnerabilities. Continued efforts in fine-tuning model parameters, enhancing data preprocessing methods, and exploring alternative model architectures will be essential for advancing predictive capabilities in cybersecurity threat modeling.

Future work could focus on integrating additional contextual information, such as network traffic patterns and user behavior analytics, to enhance model performance. Additionally, exploring hybrid models that combine LSTM with other machine learning techniques, such as convolutional neural networks (CNN) or ensemble methods, may provide better prediction accuracy. Moreover, real-time implementation and testing of these models in live network environments could validate their practical applicability and effectiveness. Finally, expanding the scope of research to include adaptive learning mechanisms that update model parameters in response to evolving cyber threat landscapes could significantly improve the resilience and accuracy of cyber attack predictions.

ACKNOWLEDGMENTS

The authors thank to Bina Nusantara University for supporting this research.

REFERENCES

[1] T. Stevens, "Global cybersecurity: New directions in theory and methods," *Politics and Governance*, vol. 6, no. 2. Cogitatio Press, pp. 1–4, 2018. doi: 10.17645/pag.v6i2.1569.

[2] T. Vimy *et al.*, "ANCAMAN SERANGAN SIBER PADA KEAMANAN NASIONAL INDONESIA," *J. Kewarganegaraan*, vol. 6, no. 1, 2022.

[3] L. Daria, Z. Dmitry, and Y. Anastasiia, "Predicting cyber attacks on industrial systems using the Kalman filter," *Proc. 3rd World Conf. Smart Trends Syst. Secur. Sustain. WorldS4 2019*, pp. 317–321, 2019, doi: 10.1109/WorldS4.2019.8904038.

[4] J. F. Torres, D. Hadjout, A. Sebaa, F. Martínez-Álvarez, and A. Troncoso, "Deep Learning for Time Series Forecasting: A Survey," *Big Data*, vol. 9, no. 1. Mary Ann Liebert Inc., pp. 3–21, Feb. 01, 2021. doi: 10.1089/big.2020.0159.

[5] Burkov, "The hundred pages machine learining book," p. 433, 2019.

[6] Y. Bengio, I. Goodfellow, and A. Courville, "Deep Learning," 2015.

[7] M. H. Alsharif, M. K. Younes, and J. Kim, "Time series ARIMA model for prediction of daily and monthly average global solar radiation: The case study of Seoul, South Korea," *Symmetry (Basel).*, vol. 11, no. 2, Feb. 2019, doi: 10.3390/sym11020240.

[8] S. Siami-Namini, N. Tavakoli, and A. Siami Namin, "A Comparison of ARIMA and LSTM in Forecasting Time Series," in *Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018*, Institute of Electrical and Electronics Engineers Inc., Jul. 2018, pp. 1394–1401. doi: 10.1109/ICMLA.2018.00227.

[9] G. Werner, S. Yang, and K. McConky, "Time series forecasting of cyber attack intensity," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Apr. 2017. doi: 10.1145/3064814.3064831.

[10] W. G. Mueller, A. Memory, and K. Bartrem, "Forecasting Network Intrusions from Security Logs Using LSTMs," in *Communications in Computer and Information Science*, Springer Science and Business Media Deutschland GmbH, 2020, pp. 122–137. doi: 10.1007/978-3-030-59621-7_7.

[11] S. H. Amin Mahmood, S. Mustafa Ali Abbasi, A. Abbasi, and F. Zaffar, "Phishcasting: Deep Learning for Time Series Forecasting of Phishing Attacks," *Proc. - 2020 IEEE Int. Conf. Intell. Secur. Informatics, ISI 2020*, vol. d, 2020, doi: 10.1109/ISI49825.2020.9280509.

[12] Y. Zhang and Z. Ji, "Anomaly Detection and Trend Prediction in Intelligent Operations Based on Prophet and S-ESD," *Acad. J. Comput. Inf. Sci.*, vol. 5, no. 4, pp. 46–51, 2022, doi: 10.25236/ajcis.2022.050408.

[13] A. R. Al-Ghuwairi, Y. Sharrab, D. Al-Fraihat, M. AlElaimat, A. Alsarhan, and A. Algarni, "Intrusion detection in cloud computing based on time series anomalies utilizing machine learning," *J. Cloud Comput.*, vol. 12, no. 1, 2023, doi: 10.1186/s13677-023-00491-x.

[14] C. Chen, J. Twycross, and J. M. Garibaldi, "A new accuracy measure based on bounded relative error for time series forecasting," *PLoS One*, vol. 12, no. 3, pp. 1–23, 2017, doi: 10.1371/journal.pone.0174202.

BIOGRAPHY OF AUTHORS

Lukman Hakim is currently pursuing a Master of Computer Science at Bina Nusantara University. He has a strong interest in cybersecurity, machine learning, and artificial intelligence. With a professional background as a backend developer, Lukman brings a practical and technical perspective to his research endeavors. He is passionate about leveraging advanced technologies to address complex cybersecurity challenges and is dedicated to advancing the field through innovative research. Lukman can be reached at lukman.hakim002@binus.ac.id for any academic or professional inquiries related to his areas of expertise.

Lili Ayu Wulandhari is a computer engineering lecturer at Bina Nusantara University. She specializes in machine learning, data science, and text mining. She successfully obtained her master's and Ph.D. degrees in computer science from the Malaysian University of Technology. Lili possesses a decade of experience as an academic as well as expertise as a professional data scientist. She has successfully applied AI and machine learning techniques in several domains and data types, including text, image, and financial data. She is actively involved as both the chairman and a member of the campus's internal and national grants. In the past five years, the research has resulted in over 10 articles that have been distributed in respected national and international journals.