Sentinel Shield: Leveraging ConvLSTM and Elephant Herd Optimization for Advanced Network Intrusion Detection

Aparna Tiwari[1] and Dinesh Kumar[2]

[1]Research Scholar, Department of Computer Science and Engineering, GZSCCET, Maharaja, Ranjit Singh Punjab Technical University, Bhatinda, India
[2]Department of Computer Science and Engineering, GZSCCET, Maharaja Ranjit Singh, Punjab Technical University, Bhatinda, India

Abstract

Given the escalating intricacy of network environments and the rising level of sophistication in cyber threats, there is an urgent requirement for resilient and effective network intrusion detection systems (NIDS). This document presents an innovative NIDS approach that utilizes Convolutional Long Short-Term Memory (ConvLSTM) networks and Elephant Herd Optimization (EHO) to achieve precise and timely intrusion detection. Our proposed model combines the strengths of ConvLSTM, which can effectively capture spatiotemporal dependencies in network traffic data, and EHO, which allow the model to focus on relevant information while filtering out noise. To achieve this, we first preprocess network traffic data into sequential form and use ConvLSTM layers to learn both spatial and temporal features. Subsequently, we introduce Elephant Herd Optimization that dynamically assigns different weights to different parts of the input data, emphasizing the regions most likely to contain malicious activity. To evaluate the effectiveness of our approach, we conducted extensive experiments on publicly available network intrusion CICIDS2017 Dataset. The experimental results demonstrate the efficacy of the proposed approach (Accuracy = 99.98%), underscoring its potential to revolutionize modern network intrusion detection and proactively safeguard digital assets.

1. Introduction

The rapid expansion of interconnected systems and the ubiquitous presence of the Internet have ushered in unparalleled levels of connectivity and convenience. However, they have also ushered in a myriad of security challenges [1]. Constantly besieged by a variety of cyber threats, networked environments face an array of challenges, ranging from familiar intrusion techniques to emerging, highly sophisticated threats. Consequently, the importance of robust and proactive Network Intrusion Detection Systems (NIDS) has reached a paramount level [2]. Traditional NIDS approaches have typically relied on static rule-based methods or signature-based techniques to identify known attacks. While these methods can be effective to some extent, they often struggle to detect previously unseen or zero-day attacks, which are continuously evolving and adapting to evade detection. Furthermore, the substantial surge in network traffic data produced by contemporary networks poses a growing challenge for traditional NIDS systems to maintain synchronization and deliver prompt and precise threat detection. In light of these difficulties, both researchers and practitioners have embraced machine learning (ML) and deep learning (DL) methodologies to craft NIDS solutions that are more adaptable and resilient. Within the realm of machine learning paradigms, recurrent neural networks (RNNs) and convolutional neural networks (CNNs) have demonstrated potential in capturing the

temporal and spatial intricacies of network traffic data [3]. However, utilizing these techniques separately may not fully exploit the rich information embedded in network traffic [4]. This paper introduces a novel NIDS model that combines the power of Convolutional Long Short-Term Memory (ConvLSTM) networks with EHO to enhance network intrusion detection. ConvLSTM networks, which combine the capabilities of convolutional and recurrent neural networks, are well-suited for modeling the spatiotemporal dependencies in network traffic data. By integrating EHO into our model, we aim to dynamically focus on relevant network segments and detect anomalous patterns associated with intrusion attempts.

The primary goal of this research is to develop a highly accurate and adaptive NIDS that can effectively predict network intrusions by learning from historical traffic data. By leveraging ConvLSTM and EHO, our model seeks to overcome the limitations of traditional rule-based and signature-based approaches while achieving superior performance in terms of detection accuracy and false positive rates.

Subsequent sections will furnish a thorough examination of our envisaged network intrusion detection model, meticulously outlining its architectural components, data preprocessing techniques, experimental methodology, and outcomes. We hold the conviction that this pioneering approach bears the potential to markedly enhance the security stance of networked systems, particularly in an era defined by the continual evolution of cyber threats.

1.1 Motivation

The burgeoning complexity of network intrusions demands a paradigm shift in intrusion detection systems (IDS) towards more sophisticated approaches. Traditional methods often fall short in identifying subtle and evolving threats amidst the vast data streams of modern networks. This necessitates the exploration and integration of cutting-edge techniques to enhance the efficacy of intrusion detection. The motivation behind "Sentinel Shield" lies in addressing this critical need by harnessing the power of Convolutional Long Short-Term Memory (ConvLSTM) networks. These networks excel in capturing temporal dependencies and spatial patterns within network traffic data, providing a robust foundation for detecting intricate intrusion attempts. By leveraging ConvLSTM, Sentinel Shield aims to elevate the accuracy, agility, and resilience of IDS to combat emerging cyber threats effectively.

1.2 Contribution

The contribution of Sentinel Shield extends beyond the mere application of ConvLSTM networks. In addition to the innovative utilization of deep learning techniques, this paper introduces the Elephant Herd Optimization (EHO) algorithm as a novel means to enhance the performance of network intrusion detection systems. Inspired by the collaborative

and adaptive behavior of elephant herds in the wild, EHO offers a metaheuristic optimization approach uniquely suited to address the complex and dynamic nature of intrusion detection. Through the fusion of ConvLSTM and EHO, Sentinel Shield introduces a holistic framework that not only detects known intrusion patterns but also adapts swiftly to identify emerging threats in real-time. This synergistic integration represents a significant advancement in the field of network security, promising heightened levels of detection accuracy, efficiency, and scalability in safeguarding critical digital infrastructures against evolving cyber threats.

1.3 Organization of the paper

The remainder of the paper follows this structured organization: In Section 2, a review of related work is provided, offering insights into existing research and methodologies in the field. Section 3 delves into the problem definition and system model, proposed framework, detailing its components, methodologies, and the rationale behind their integration are discussed in section 4. Section 5 presents the results obtained from the application of the proposed approach, including any experimental findings or performance evaluations. Finally, Section 6 synthesizes the major conclusions drawn from the study, reflecting on the implications of the findings and suggesting avenues for future research and development.

2. Related Work

With the rapid advancement of Internet networks, there is an ever-increasing requirement for network resources. As a result, how network operators allocate and use these resources properly has become a focus of substantial research in the field of traffic prediction [5, 6]. In the present time of huge data requirement, one of the crucial technologies for network traffic prediction is the proper analysis and learning of the growth distribution patterns from vast historical network traffic data. The selection of suitable features from various network traffic data is critical for effective data processing [7]. This method entails finding certain features that correspond to the needs of various operators. To produce more precise forecasts of future network traffic, some elements that influence the forecasting approach must be considered. To have a more precise forecast of future network traffic, it is essential to leverage the characteristics of actual data. This involves extracting useful information from historical operator data and analysing their growth distribution. By doing so, we can build predictive models capable of anticipating forthcoming network traffic patterns. Network traffic data shows periodic changes, with major increases or decreases in data volume occurring at specific time periods [8]. Furthermore, data changes noticed an earlier time period can have an impact on subsequent data patterns [8]. As a result, any reliable

prediction model must account for these temporal oscillations as well as the impact of prior data points on future patterns. The blend of temporality and spatiality is important for an effective prediction method. The model should strike a delicate balance between these two characteristics, as they both hold valuable insights into the network's behaviour. By capturing the periodic changes over time and incorporating the spatial distribution, the forecasting algorithm can provide comprehensive and reliable predictions [9]. By incorporating temporality and spatiality into the forecasting method and leveraging historical operator data, we can design robust predictive models that adapt to the dynamic nature of network traffic, thus ensuring higher accuracy and effectiveness in forecasting [9].The historical flow data in numerous domains is enormous and constantly changing, making accurate prediction a challenging task for conventional techniques. However, deep learning has emerged as a promising solution to enhance prediction accuracy significantly. Numerous deep learning techniques have been proposed to upgrade network traffic prediction, among which the CNN is quite effective [10]. CNN leverages convolutional operations to learn local information effectively and has found to be very effective in various fields. By employing sliding filters in the convolution layer, CNNs can extract spatial features from input matrices. One notable advantage of CNNs is their weight sharing and sparse link characteristics, which demand fewer training operations compared to other networks with an equivalent number of neurons. As the research continues, many researchers have adopted hybrid models based on deep learning for network traffic forecasting [8-12]. These hybrid models have shown promising improvements in forecasting accuracy. By combining LSTM with RNN and focusing on the analysis of self-similarity coefficients, this new method significantly enhances forecasting accuracy [11]. However, network traffic prediction remains a challenging problem as network traffic exhibits specific temporal characteristics, such as spikes during holidays or specific time periods. These complex spatial and temporal relationships contribute to the ongoing challenge of accurately predicting network traffic [12]. Despite the advancements in prediction models, the dynamic nature of network traffic and the interplay between spatial and temporal factors necessitate further research and innovative approaches to tackle this challenging problem effectively.

This section explains how the current ML and DL algorithms are used to build reliable DDoS detection models. Fundamental methods for AI-based DDoS detection are also clarified. A dual strategy is used in the field of ML and DL, incorporating both supervised and unsupervised approaches. The prior labelling of data, ensuring that it is marked with the proper class labels, is a requirement for supervised algorithms. On the other hand, unsupervised algorithms work with unlabelled data and make use of their inherent structure and properties to find important patterns and insights.

Gao et al. [13] introduced an adaptive ensemble model for classifier selection within machine learning (ML) frameworks, but its performance was found to be limited on weaker attack classes. Sabeel et al. [14] incorporated Deep Neural Networks (DNNs) with Long Short-Term Memory (LSTM) layers, observing significant performance improvements, albeit limited to binary class classification. Asad et al. [15] proposed the Deep Detect model based on DNNs, outperforming other strategies, yet evaluation was conducted solely against Distributed Denial of Service (DDoS) assaults. Mural. et al. [16] achieved good accuracy in attack detection using DNNs but were restricted to assessing slow HTTP DoS attacks. Amaizu et al. [17] attained high accuracy rates with their DNN model, although its complexity may lead to longer detection times, impacting real-time usage. Hasan et al. [18] utilized a Deep Convolutional Neural Network (CNN) model, which showed better performance compared to three other ML methods, albeit with a dataset limited in size. Amma et al. [19] combined Fully Connected Neural Networks (FCNN) with Variable Convolutional Neural Networks (VCNN), outperforming basic classifiers, yet their reliance on outdated datasets and omitted trials may raise concerns. Chen et al. [20] employed CNNs, with their MCCNN showing promising results on constrained data, though no significant difference was observed between single and multi-class models. Haider et al. [21] utilized ensemble CNN techniques, which exhibited high accuracy but required longer training and testing periods. Wang et al. [22] combined entropy and deep learning techniques, demonstrating the effectiveness of CNNs, albeit with longer training requirements. Kim et al. [23] developed a CNN-based model effective in recognizing unique Denial of Service (DoS) attacks with similar features, yet detection using this model also incurred longer processing times. Hussain et al. [24] proposed a method to convert non-image network data for classification, achieving high accuracy in binary-class classification; however, the time required for data transformation was not considered. Li C et al. [25] utilized a deep learning approach for fairly accurate detection, but the model's performance required a significant amount of time. Shu et al. [26] introduced a hybrid-based Intrusion Detection System (IDS) and deep learning model based on LSTM, achieving high accuracy, albeit with a substantial time requirement for detection. Bhardwaj et al. [27] effectively addressed feature learning and overfitting issues with their DNN-based approach, yet their study was conducted offline, not utilizing recent datasets. Moh. et al. [28] combined LSTM with Bayes methods, maintaining stable performance with new data, but detection of attacks unsuitable for real-time applications may take longer. Finally, He et al. [29] proposed a strategy based on Dynamic Time-Lapse (DTL) for DDoS detection, achieving a 20.8% improvement in detecting the 8LANN network, albeit focusing solely on a single type of attack.

Patil and Pattewar introduce a Majority Voting and Feature Selection-based approach, emphasizing the importance of ensemble techniques and feature relevance in intrusion detection [30]. Venkateswaran and Prabaharan propose an efficient neuro deep learning intrusion detection system tailored for mobile adhoc networks, showcasing the efficacy

of deep learning in dynamic network environments [31]. Fatima et al. investigate the impact of feature reduction on machine learning-based intrusion detection systems, shedding light on the trade-offs between dimensionality reduction and detection accuracy [32]. In contrast, Zhang et al. focus on outlier detection from large distributed databases, presenting techniques applicable to anomaly-based intrusion detection systems [33]. Kabir et al. introduce a novel statistical technique for intrusion detection systems, offering insights into the statistical methods employed for anomaly detection [34]. Lastly, Alkanhel et al. [35] propose a network intrusion detection system integrating feature selection and hybrid metaheuristic optimization, highlighting the importance of optimization techniques in enhancing detection performance. Gul et al. [36] explore the realm of secure industrial IoT systems, employing RF fingerprinting amidst impaired channels with interference and noise.

3. Problem Definition and System Model

The problem addressed in this study lies at the intersection of cybersecurity and network infrastructure resilience, where the ever-evolving landscape of cyber threats poses a significant challenge to the integrity and security of digital systems. Traditional intrusion detection systems often struggle to keep pace with the sophistication and adaptability of modern attacks, particularly in environments characterized by dynamic network topologies, diverse communication protocols, and rapidly evolving attack vectors. The proposed system model seeks to address this challenge by introducing a novel approach that combines the robust temporal analysis capabilities of Convolutional Long Short-Term Memory (ConvLSTM) networks with the adaptive optimization mechanisms of Elephant Herd Optimization (EHO). This fusion of advanced techniques forms the backbone of the system model, which is designed to enhance the accuracy, agility, and resilience of intrusion detection in complex network environments. Within this system model, the architecture encompasses the hierarchical structure and components of the intrusion detection framework, including data preprocessing, feature extraction, anomaly detection, and decision-making modules. The algorithms and methodologies employed within this architecture leverage the capabilities of ConvLSTM networks to capture temporal dependencies and spatial patterns in network traffic data, enabling the system to discern subtle anomalies indicative of potential intrusions. Concurrently, the integration of EHO facilitates the optimization of system parameters and decision thresholds, enabling adaptive and efficient intrusion detection in the face of evolving threats and network conditions. Through this comprehensive system model, the study aims to contribute to the advancement of network security by providing a robust and adaptable framework for detecting and mitigating intrusions in real-time, thereby safeguarding critical digital infrastructures against a wide range of cyber threats.

4. Proposed Method

The methodology we propose utilizes a multi-step strategy designed to proficiently manage traffic classification and anomaly detection. This strategy encompasses multiple discrete phases, all of which collectively enhance the system's accuracy and efficiency. In this research, we introduce a hybrid deep architecture with the goal of enhancing network traffic prediction, as illustrated in Figure 1. The model presented in this study adopts a multi-module framework that consists of two Bi-LSTM modules and a Conv-LSTM module, collaborating effectively to capture complex spatiotemporal patterns inherent in traffic data. The holistic architecture is depicted in Figure 1, showcasing the interconnectedness of these modules. The Conv-LSTM module integrates CNNs and LSTM networks, amalgamating spatial and temporal information. The initial step involves the CNN extracting spatial features from the network traffic data, which are subsequently input into the LSTM to capture temporal dynamics. Concurrently, the Bi-LSTM modules contribute by capturing the recurrent patterns inherent in daily and weekly traffic flow variations. These modules facilitate the extraction of periodic features, crucial for understanding traffic behavior over extended time spans. In order to amalgamate the spatial-temporal aspects along with the periodic features, a feature fusion layer (FF layer) is incorporated into the architecture, resulting in the creation of an all-encompassing feature vector. This comprehensive feature vector then undergoes further processing through two fully-connected layers (FC layers) that serve as regression layers, proficiently carrying out the task of predictive forecasting. Furthermore, to enhance the model's performance, an innovative EHO has been seamlessly integrated into the Conv-LSTM module. This mechanism dynamically allocates varying degrees of importance to different flow sequences at various temporal points. By doing so, it allows the model to adapt and focus more on the most pertinent information during different stages of the prediction process, thereby improving its ability to capture and utilize critical patterns and dependencies within the data for accurate forecasting. In the subsequent subsections, each module is elucidated in detail, illuminating their individual functionalities and interactions within this holistic framework.

3.1 Conv-LSTM

The crucial Conv-LSTM module, which is the foundation for extracting the intricate spatial-temporal properties inherent in network traffic flow data, is the most important part of the given model. Figure 2 shows how a dynamic framework is created within this module by the combination of the CNN and the LSTM network.
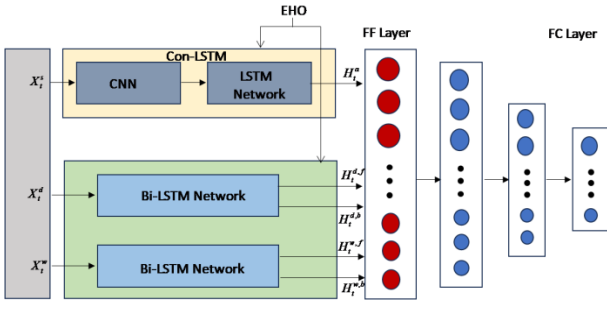
Figure 1. Block diagram of the future traffic prediction
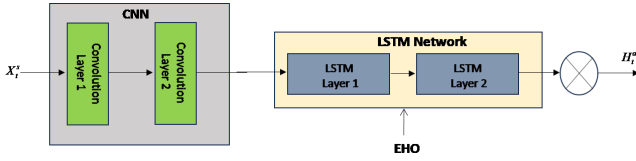

Figure 2. Block diagram of ConvLSTM with EHO

Within this framework, the ConvLSTM network is structured with two CNN layers followed by two LSTM layers, each contributing to the comprehensive analysis of network traffic data.

The first CNN layer functions as the initial feature extractor, convolving input data with learnable kernels to extract spatial features relevant to intrusion detection. These features are then passed through rectified linear unit (ReLU) activation functions to introduce non-linearity and enhance the network's representational capacity. Subsequently, max-pooling operations are applied to down-sample the feature maps, reducing computational complexity while retaining salient information.

The second CNN layer further refines the extracted features, leveraging additional convolutional filters to capture higher-level abstractions in the network traffic data. Similar to the first CNN layer, ReLU activation functions are employed to introduce non-linearities, facilitating the learning of complex patterns and correlations within the data.

Following the CNN layers, the architecture incorporates two LSTM layers to capture temporal dependencies in the network traffic sequences. LSTM units within these layers are equipped with memory cells and gating mechanisms, enabling the network to selectively retain and update information over time. This allows the model to effectively model long-range dependencies and temporal dynamics inherent in network traffic data, thereby enhancing the system's ability to detect subtle and evolving intrusion patterns.

Moreover, the integration of EHO within the system architecture enables dynamic feature optimization and parameter tuning. EHO operates iteratively to adaptively adjust the weights and biases of the CNN and LSTM layers, optimizing the model's performance based on specified evaluation metrics such as detection accuracy or false alarm rate. By leveraging the collective intelligence and

exploration-exploitation capabilities of the EHO algorithm, the system can effectively navigate the high-dimensional search space of feature representations, thereby enhancing the discriminative power and robustness of the intrusion detection system.

The symbols and notations employed in this research are presented in detail within Table 1.

Table 1. Symbols and Notations

Symbols	Notations
T	Traffic flow
t	Time stamp
ω	weight
b	bias
U	Convolution layer output
H	Hidden state
σ	Activation function
f_t	Forget gate
i_t	Input gate
C_t	Cell state
o_t	Output gate
ζ	Elephant position
N	Total number of elephant
M	Number of dimension
r, β, ψ	Random variable
F	Fitness function
A	Classification Accuracy
δ	weight parameter
Υ_i	Selecetd number of features
L	Loss
Ω	Regularization terms

The Conv-LSTM model takes as input a matrix represented as T_t^s, which takes the spatial-temporal network traffic patterns. Equation (1) shows this matrix, which represents the historical network traffic flow of a Point under investigation.

At each time step t, the flow data T_t^s is subjected to a one-dimensional convolution process to extract spatial properties. Through this process, the model is able to gather localized perceptual data from the traffic flow by sliding a specific one-dimensional convolution kernel filter across the flow data.

$$T_t^s = \begin{bmatrix} T_{t-\tau}^s \\ T_{t-(\tau-1)}^s \\ \vdots \\ T_t^s \end{bmatrix} = \begin{bmatrix} r_{t-\tau}^1 & r_{t-(\tau-1)}^1 & \cdots & r_{t-\tau}^1 \\ r_{t-\tau}^2 & r_{t-(\tau-1)}^2 & \cdots & r_{t-\tau}^2 \\ \cdots & \cdots & \cdots & \cdots \\ r_{t-\tau}^k & r_{t-(\tau-1)}^k & \cdots & r_{t-\tau}^k \end{bmatrix} \quad (1)$$

The hour-wise traffic flow is also can be represented in the similar manner as

$$T_t^d = \begin{bmatrix} T_{t-\tau}^s \\ T_{t-(\tau-1)}^s \\ \vdots \\ T_t^s \end{bmatrix} = \begin{bmatrix} r_{t^d-\tau}^1 & r_{t^d-(\tau-1)}^1 & \cdots & r_{t^d-\tau}^1 & \cdots & r_{t^d+\tau}^1 \\ r_{t^d-\tau}^2 & r_{t^d-(\tau-1)}^2 & \cdots & r_{t^d-\tau}^2 & \cdots & r_{t^d+\tau}^2 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{t^d-\tau}^k & r_{t^d-(\tau-1)}^k & \cdots & r_{t^d-\tau}^k & \cdots & r_{t^d+\tau}^k \end{bmatrix} \quad (2)$$

The day-wise traffic flow is also can be represented in the similar manner as

$$T_t^w = \begin{bmatrix} T_{t-\tau}^s \\ T_{t-(\tau-1)}^s \\ \vdots \\ T_t^s \end{bmatrix} = \begin{bmatrix} r_{t^w-\tau}^1 & r_{t^w-(\tau-1)}^1 & \cdots & r_{t^w-\tau}^1 & \cdots & r_{t^w+\tau}^1 \\ r_{t^w-\tau}^2 & r_{t^w-(\tau-1)}^2 & \cdots & r_{t^w-\tau}^2 & \cdots & r_{t^w+\tau}^2 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{t^w-\tau}^k & r_{t^w-(\tau-1)}^k & \cdots & r_{t^w-\tau}^k & \cdots & r_{t^w+\tau}^k \end{bmatrix} \quad (3)$$

As shown in Figure 2, the EHO mechanism added to the Conv-LSTM architecture facilitates focusing on certain areas of interest within the spatial-temporal network traffic matrix. The output of the convolutional layer in terms of the input traffic (T_t^s) can be written as:

$$Y_t^s = \sigma\left(\omega_s * T_s^t + b_s\right) \quad (4)$$

In the given equation, ω_s represents the weights of the filter, bias is represented by b_s, σ is the activation function. Due to the fact that the spatial feature's dimensionality is not significant enough to warrant such pooling, we have decided not to include a pooling layer in the proposed model after the convolutional layer.. Let's define U_t^s as the resulting output after the second convolutional layer's operation. Once the spatial information has undergone processing through these two consecutive convolutional layers, the resulting output, U_t^s, is then seamlessly linked to LSTM network. This connection facilitates the integration of the refined spatial features into the LSTM's sequential processing mechanism.

LSTM networks are a type of recurrent neural network (RNN) architecture that excel in modeling sequential data by capturing long-term dependencies and mitigating the vanishing gradient problem encountered in traditional RNNs. Unlike standard RNNs, LSTM networks incorporate specialized memory cells and gating mechanisms, including input, forget, and output gates, which regulate the flow of information through the network over time (Figure 3). These gates enable LSTM units to selectively retain and update information, allowing the network to learn and remember patterns across extended temporal sequences. By effectively preserving relevant context and suppressing irrelevant information, LSTM networks are well-suited for tasks requiring the modeling of complex temporal dynamics, such as natural language processing, time series prediction, and, in the context of this study, the analysis of network traffic data for intrusion detection.

It is commonly known that traffic flow patterns show temporal relationships between neighbouring time periods.

LSTM networks provide a solution for learning extended temporal dependencies within sequential data to deal with this restriction. A memory block is housed in a cell that is part of an LSTM's architecture, along with the input gate, output gate, and forget gate (Figure 3). To ascertain the LSTM's current state, these gates work collaboratively. The input gate controls how much information from the current input is assimilated into the current cell state, while the forget gate controls how much information from the previous cell state is preserved in the present state. This architecture makes sure that significant elements of the present input are properly incorporated and that crucial information isn't lost. LSTM tries to improve our comprehension of the complex temporal dynamics hidden within the traffic flow patterns by utilizing the inherent capabilities.
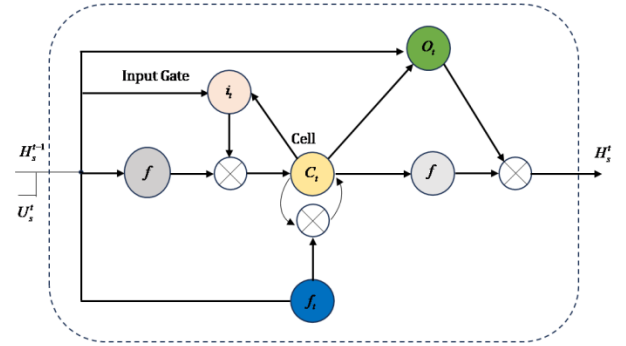

Figure 3. Block diagram of LSTM Network with various gates

Increasing the depth of the model by adding more layers is a common conventional approach to improving the performance of deep neural networks. With a specific focus on stacking numerous LSTM layers, we apply a similar strategy in the context of this study. With the help of this intentional stacking, the traffic flow data's underlying higher-level properties are gradually captured. This architectural decision promotes a hierarchical feature extraction process by enabling each LSTM layer to utilize the learnt representations from the prior layer's hidden state. Referring Figure 2, the computational process involves two LSTM layers. Initially, LSTM layer 1 is tasked with processing a sequential output derived from the preceding CNN module, $U_t^s = \left[U_{t-\tau}^s, \ldots U_{t-1}^s, U_t^s\right]$ encompassing the entire sequence from start to finish. Within this operation, LSTM layer 1 calculates and estimates a hidden state as $H_{1,t}^s = \left[H_{1,t-n}^s, \ldots H_{1,t-1}^s, H_{1,t}^s\right]$. Subsequently, the resulting sequence of hidden states, denoted as 'hidden state sequence,' $H_{1,t}^s$ serves as the input to LSTM layer 2. The primary objective of LSTM second layer (layer 2) is to estimate the hidden state $H_{2,t}^s$ specifically for a particular time stamp 't', which ultimately represents the final output

H_t^s. To precisely outline the computational procedures of each LSTM layer, we can refer to equations (5) through (9), which articulate the mathematical expressions governing these processes:

The expression of input gate (i_t) at time 't' is given by

$$i_t = \sigma\left(\Omega_{gi}T_t^s + \Omega_{hi}H_{t-1}^s + \Omega_{ci} \circ C_{t-1} + \beta_i\right) \quad (5)$$

where, I_t^s is the input to the LSTM layer. The expression for the forgot gate (f_t) is given by

$$f_t = \sigma\left(\Omega_{gf}T_t^s + \Omega_{hf}H_{t-1}^s + \Omega_{cf} \circ C_{t-1} + \beta_f\right) \quad (6)$$

The expression for the cell state (C_t) is given by

$$C_t = f_t \circ C_{t-1} + i_t \circ \tanh\left(\Omega_{gc}T_t^s + \Omega_{hc}H_{t-1}^s + \beta_c\right) \quad (7)$$

The expression for the out state (O_t) is given by

$$O_t = \sigma\left(\Omega_{go}T_t^s + \Omega_{ho}H_{t-1}^s + \Omega_{co} \circ C_t + \beta_o\right) \quad (8)$$

The output of the LSTM layer is given by

$$H_t^s = O_t \circ \tanh\left(C_t\right) \quad (9)$$

Further, U_t^s and H_t^s are inputs for layer 1 and layer 2 respectively and corresponding output for these two layers are represented by $H_{1,t}^s$ and $H_{2,t}^s$, respectively.

3.2 Elephant Herding Optimization

Elephants are extremely social animals that display a sophisticated social hierarchy dominated by females and their young. A matriarch is the leader of a group of many elephants in an elephant clan, directing and supervising their actions. The gregariousness of female elephants is demonstrated by their preference to live with their family. Male elephants, on the other hand, usually move somewhere else and become increasingly self-sufficient over time until they finally break away from their families [30]. The goal of the Elephant Herding Optimization (EHO) optimization technique is to mimic the synchronized and cooperative movements of elephants in their natural herding activity. The EHO approach is based on a number of assumptions that are consistent with the traits and actions of elephants that have been seen in their natural environments. Elephants' general population distribution is depicted in Figure 4, which offers a graphic depiction of the group dynamics of these amazing animals. The below given assumptions are taken into account in EHO.
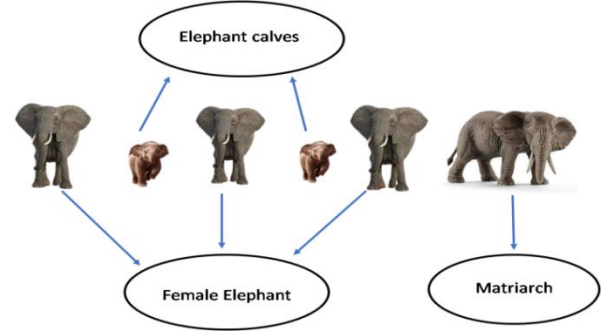

Figure 4. Elephant population type

1. The elephant population comprises distinct clans, each with a specific number of elephants.
2. Every generation witnesses a set number of male elephants breaking away from their family group to establish independent residency apart from the main elephant population.
3. Within each clan, the leadership role is assumed by a matriarch who guides the elephants.

3.2.1. Clan-updating Operator

In accordance with the inherent behaviors of elephants, a matriarch assumes a leadership role within each clan, guiding the collective actions of the elephants. The mathematical expression for calculating the position of elephant [j] within clan [γ_i] is denoted by Equation (10).

$$\zeta_{new,\gamma_{i,j}} = \zeta_{new,\gamma_j} + \beta \times \left(\zeta_{best,\gamma_i} - \zeta_{\gamma_{i,j}}\right) \times r \quad (10)$$

$\zeta_{new,\gamma_{i,j}}$ and $\zeta_{\gamma_{i,j}}$ represent the most recent and previous positions of elephant [j] in clan [γ_i], respectively. [ζ_{best,γ_i}] signifies the matriarch in clan [γ_i], representing the optimal elephant within the group. The parameter scale factor is indicated by $\beta \in [0,1]$ and $r \in [0,1]$. The determination of the best elephant in each clan is achieved through the application of Equation (11).

$$\zeta_{new,\gamma_{i,j}} = \psi\zeta_{center,\gamma_i} \quad (11)$$

Here, the variable $\psi \in [0,1]$ denotes impact of the center individual (ζ_{center,γ_i}) on new individual ($\zeta_{new,\gamma_{i,j}}$) on clan γ_i. In the mth dimension can be evaluated as

$$\zeta_{center,\gamma_i,m} = \frac{1}{n_{\gamma_i}} \times \sum_{j=1}^{n_{\gamma_i}} \zeta_{\gamma_{i,j,m}} \quad 1 \le m \le M \quad (12)$$

n_{γ_i} defines the number of elephants in clan γ_i.

Elephants with the lowest fitness values within their clans are ignored during the optimization process, and their locations are randomly regenerated throughout the search space. Known as the "separating operator," this procedure simulates how male elephants naturally depart from their groups.

3.2.2. Separating Operator

The separation operator can be used to include the occurrence of male elephants leaving their family groups into an optimization framework. In the context of optimization, the separating operator serves as a mechanism inspired by the natural behavior of male elephants as they gradually distance themselves from their family units. This operator encapsulates the essence of separation, mirroring the transition observed in elephant social structures [37]. Its implementation, guided by the individual with the least favourable fitness, reflects the dynamic adaptation of optimization algorithms to real-world scenarios, drawing parallels between the natural world and computational problem-solving strategies. Equation (13) serves as the mathematical representation of this separating operator, capturing the essence of the departure process in the optimization context.

$$\zeta_{worst,\gamma_{i,j}} = \zeta_{min} + \beta \times (\zeta_{max} - \zeta_{min} + 1) \times r \qquad (13)$$

Where ζ_{max} and ζ_{min} represents the upper and lower bound of the individual respectively. $\zeta_{worst,\gamma_{i,j}}$ defines the worst individual in clan γ_i.

The EHO algorithm faces challenges when the current optimal individual is drawn towards a local extremum, lacking an effective mechanism to break free from constraints and risking premature convergence. This paper suggests an improvement by incorporating Levy flight behavior, utilizing it to simulate the elephant's position update. This modification (IEHO) aims to leverage Levy flight for broader search ranges, preventing individuals from getting trapped in local optima and enhancing the algorithm's ability to avoid premature convergence.

$$\zeta_{worst,\gamma_{i,j}} = \zeta_{min} + \frac{r_1(1,m) \times \sigma}{|r_2(1,m)|^{1/\tau}} + \beta \times (\zeta_{max} - \zeta_{min} + 1) \times r \qquad (14)$$

Feature selection constitutes a crucial step in the data processing pipeline for intrusion detection. Its essence lies in choosing subsets from the original dataset in a way that maximizes the classification effectiveness or, equivalently, yields the highest fitness function value. Formally, for a given set of data samples $\{f_1, f_2, ..., f_L\}$ where L is the size of the feature set, any feature subset can be denoted by a binary vector $\{s_1, s_2, ..., s_L\}$ with s_i indicating whether the ith feature is selected (s_i=1) or not (s_i=0).

The overarching objective of feature selection is to achieve superior classification outcomes with the minimum number of features. This involves tackling two fundamental challenges: devising effective search strategies and evaluation functions for the generation and assessment of feature subsets. The fitness function, synonymous with the evaluation function, plays a pivotal role in determining the quality of each individual in the population. In the context of intrusion detection, the fitness function predominantly revolves around the number of selected features and the accuracy of classification. Therefore, the fitness function for feature selection can be defined as Equation (15):

$$F_i = (1 - \delta)A_i + \delta \Upsilon_i \qquad (15)$$

Here, F_i signifies the fitness value of the feasible solution, A_i denotes the classification accuracy (eqn. 19) , Υ_i represents the selected number of features, and δ is a weight parameter typically set to 0.01. This function encapsulates the dual aspects of classification accuracy and the economy of selected features in the evaluation of the solution's efficacy in the context of intrusion detection.

The IEHO (Improved Elephant Herding Optimization) algorithm enhances the convergence speed and global optimization capability of the EHO algorithm, thereby elevating the overall classification performance. However, given that feature selection involves a binary combination optimization problem, and the IEHO algorithm is designed for continuous problems, a direct application for feature selection is not feasible. To address this, this work considers an iintrusion detection feature selection method by combining binary coding with the IEHO algorithm, termed the binary IEHO algorithm.

In the iterative process of the binary IEHO algorithm for feature selection, the elephant group functions as search agents exploring the solution space, with each elephant individual representing a potential solution. The binary IEHO algorithm employs a binary coding format, where each elephant's position corresponds to a feasible solution. In this binary coding, each dimension of the elephant is represented as a binary value; taking either 0 or 1 depending on feature is not selected or selected in the feasible solution. The binary vector is constructed using the sigmoid function, which is intended to be

$$Sign(\zeta_{new,\gamma_{i,j}}) = \frac{1}{1 + \exp^{-(\zeta_{new,\gamma_{i,j}})}} \qquad (16)$$

The binarization is done using

$$\zeta_{new,\gamma_{i,j}} = \begin{cases} 1 & \text{if } Sign(\zeta_{new,\gamma_{i,j}}) > rand(1) \\ 0 & \text{otherwise} \end{cases} \qquad (17)$$

Determining the number of dimensions that this equation will update raises an important question. In the foundational EHO algorithm, when all dimensions are simultaneously altered, every elephant within the clan essentially assumes the role of the matriarch. Conversely, updating one dimension at a time leads to delayed convergence, presenting a noteworthy challenge for the conventional EHO algorithm [38].

To address these concerns and strike a balance between diversity and convergence speed across updated dimensions, a parameter known as the dimension load (DL) has been introduced. This parameter plays a crucial role in determining the number of dimensions that undergo updates. The DL parameter serves as a pivotal mechanism to optimize the algorithm by influencing the trade-off between exploring diverse solution spaces and achieving efficient convergence. The careful adjustment of this parameter allows for a more nuanced and adaptive approach, mitigating the issues associated with simultaneous or sequential updates of dimensions in the algorithm.

For example, Figure 5 shows the updating procedure for each elephant, assuming that DL is set to 0.4 and the total

number of dimensions (M) is 10. In this particular case, the number of dimensions that need to be updated is four, as determined by multiplying DL by M. Crucially, these dimensions are chosen at random, guaranteeing their uniqueness from one another. By carefully utilizing the dimension load parameter, the updating process is given a controlled variability that provides a sophisticated resolution to the convergence and similarity problems that the fundamental EHO algorithm presents.

Figure 5. Elephant update process in Binary EHO

Similar to the solution search equation (Eq. 10) in the basic Elephant Herding Optimization method algorithm, Eq. (9) ensures that the positions of the matriarch remain unchanged when dealing with binary values. However, it becomes evident that Eq. (17) is unsuitable for computing the central position of the clan to update the matriarch, given that Eq. (11) is not applicable in the context of binary values. Consequently, a mutation operator is introduced to facilitate the matriarch's update and enhance population diversity. A careful approach is employed to prevent excessive disruption to the matriarch, who represents the best elephant in each clan. Specifically, only one dimension, chosen randomly, undergoes mutation. Fig. 6 illustrates a representative mutation operation. This strategic mutation process aims to strike a balance between introducing variability within the population and preserving the stability brought by the matriarch, thereby contributing to the algorithm's robustness. The algorithm allows for adaptation to the binary character of the search space by introducing the mutation operator. This ensures that the matriarch's position is updated in a way that maintains population variety while striking a balance between exploration and exploitation.

Figure 6. Mutation process in Binary EHO

Bi-Directional LSTM

The bi-directional LSTM model's architectural arrangement consists of two layers, each of which contains unique unidirectional LSTM components stacked in both an ascending and descending order (Figure 7). This split setup creates a forward pass LSTM and a companion backward pass LSTM, which together provide a thorough grasp of the input data's temporal structure. As previously explained, a key component of successfully handling the historically cyclical nature of traffic flow is the integration of numerous LSTM layers. Specifically, a dual-layer composition with one pair for the forward pass and another for the backward pass emerges within each bi-directional LSTM network.
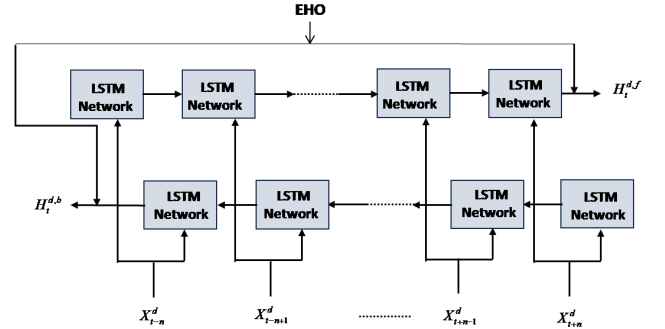

Figure 7. Bi-LSTM Network

The traffic flow data from the previous day's near past and immediate future, as well as the parallel temporal span from the prior week, are included in the input dataset used for forecasts at a given time instance. Equations (2) and (3) serve as respective representations of this data structure. The concatenated data streams are subsequently input into the bi-directional LSTM architecture. The generated hidden states from these twin passes are then processed, harmonizing them to produce an overall output. Combining the information obtained from the forward and backward LSTM sequences gives the model a richer contextual perspective, which inherently improves its predictive effectiveness. A visual representation of the comprehensive framework encompassing the suggested bi-directional LSTM module inside our overall model is provided in Figure 5. The LSTM module's input data streams are represented by the elements marked by T_t^d and T_t^ω, respectively. According to the input data streams T_t^d and T_t^ω, respectively, the outputs of the forward pass LSTM are represented by $H_t^{d.f}$ and $H_t^{\omega.f}$, while the results of the backward pass LSTM are represented by $H_t^{d.b}$ and $H_t^{\omega.b}$.

3.4 Feature Fusion

The spatial-temporal characteristics H_t^a, the daily periodicity features $H_t^{d.f}$, and $H_t^{d.b}$, as well as the weekly periodicity features $H_t^{\omega.f}$ and $H_t^{\omega.b}$ are obtained, as shown in Fig.1, Then, to perform forecasting, two regression layers input a feature vector that has all of these features concatenated together.

The loss function is modelled as

$$L = \left(\frac{1}{\tau} \sum_{t=1}^{\tau} (PT - IT)^2 + \lambda_1 \sum_{t=1}^{\tau} |\Omega_i| + \lambda_2 \sum_{t=1}^{\tau} \sqrt{\Omega_i^2} \right) \quad (18)$$

The first term is MSE of input traffic (IT) and predicted traffic (PT), while second and third terms are regularization terms to avoid overfitting and specific feature dominance respectively.

4. Results

In this section, we present the results of our comprehensive network traffic prediction model, which leverages advanced machine learning and data analysis techniques. These results not only provide insights into the model's accuracy and effectiveness but also offer a glimpse into the potential for more proactive and adaptive network management.

4.1 CICIDS2017 Dataset

The CICIDS2017 dataset, short for "Canadian Institute for Cybersecurity Intrusion Detection Evaluation Dataset 2017," is a collection of network traffic data specifically designed for evaluating and benchmarking IDS and intrusion prevention systems (IPS) [39]. The dataset is intended to represent real-world network traffic scenarios, including both benign and malicious activities. It includes various types of attacks, anomalies, and normal network traffic to provide a comprehensive set of data for testing the effectiveness of intrusion detection and prevention mechanisms.

Over the course of the week, a range of cyber security events and incidents were documented, as displayed in Table 2. On Monday, there were 529,918 events classified as "Benign," indicating harmless or non-malicious activities. Tuesday witnessed 445,909 events related to "Brute Force" attacks, where unauthorized access attempts were made through repeated username and password combinations. On Wednesday, 692,703 events were associated with "Heartbleed" and "DoS" (Denial of Service) attacks, which can potentially lead to data leaks and service disruption. Thursday had 170,366 events categorized under "Web," which could encompass various web-related activities. Additionally, on Thursday, 288,602 events were logged as "Infiltration," signifying attempts or incidents of unauthorized access. On Friday, 190,133 events were linked to "Botnet" activities, often involving networks of compromised devices for malicious purposes. Friday also recorded 286,467 "Port Scan" events, indicating reconnaissance activities, and 225,745 "DDoS" events, involving distributed denial of service attacks aimed at making services unavailable. These events are typically monitored and analyzed to enhance cyber security measures and respond to potential threats.

Table 2. Day wise traffic distribution for CICIDS2017 dataset

Day	Attack Type	Number of records
Monday	Benign	529918
Tuesday	Brute Force	445909
Wednesday	Heartbleed/DoS	692703
Thursday	Web	170366
Thursday	Infiltration	288602
Friday	Botnet	190133
Friday	Port Scan	286467
Friday	DDoS	225745

In the dataset, each piece of data is associated with one of sixteen distinct labels. Among these labels, one of them is labelled as "Benign," signifying normal network activities devoid of any malicious intent. This benign record is constructed by mimicking genuine user behaviour and encompasses various protocols such as Mail services, SSH, FTP, HTTP, and HTTPS.
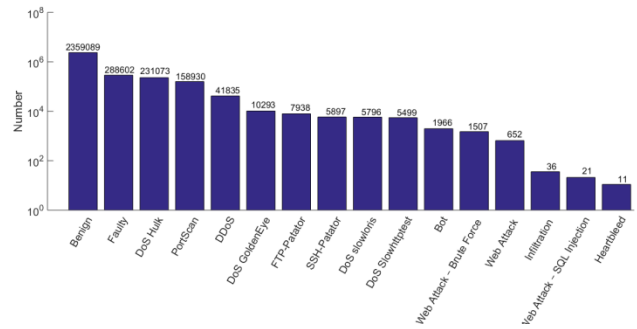

Figure 8. Traffic type with labels and number

Essentially, it replicates the patterns of genuine user data, ensuring that it does not pose any threat or harm to the network. The remaining fourteen labels in the dataset are dedicated to representing different types of network attacks. For a comprehensive list of the specific labels and their corresponding numbers, please refer to Figure 8.
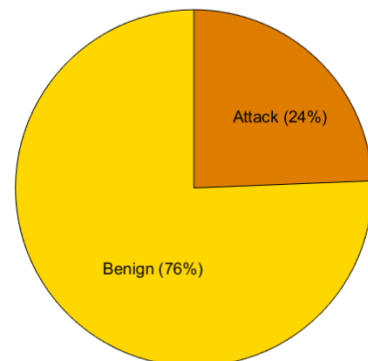

Figure 9. Traffic distribution chart

Out of the entire network traffic, a significant portion, precisely 76 percent, is categorized as "benign" traffic (Figure 9). This benign traffic represents the normal and non-harmful activities occurring on the network. It includes

activities like Mail services, SSH, FTP, HTTP, and HTTPS, which simulate genuine user interactions and contribute to the smooth functioning of the network. This substantial majority of benign traffic is essential for organizations to operate efficiently and without interruptions. In contrast, the remaining 24 percent of the network traffic is classified as "attack" traffic. This portion consists of various types of malicious activities, which could potentially harm the network's integrity, compromise data security, or disrupt services. These attacks encompass a wide range of techniques and strategies employed by cybercriminals, including brute force attacks, DoS attacks, web vulnerabilities, infiltration attempts, botnet activities, port scans, and DDoS attacks, among others. The feature set presented comprises 78 distinct attributes or characteristics used in network traffic analysis. These features offer comprehensive insights into the behaviour of network flows. Notably, the "Label" feature serves as a crucial identifier, categorizing network flows as either benign or indicative of an attack. Other features provide information on the duration of flows, packet inter-arrival times, packet lengths, and various statistics about packet data. Collectively, these attributes enable network analysts and machine learning algorithms to scrutinize and classify network activities effectively.

They play a pivotal role in the development of intrusion detection systems, traffic anomaly detection, and network security assessment by providing the necessary data for identifying and responding to potential threats and vulnerabilities in network traffic. Our proposed model leverages these features to effectively distinguish between benign and malicious network traffic. By analyzing and interpreting the information contained within these features, and selecting optimal features our model can make informed decisions regarding potential threats, thereby bolstering the overall cyber security defences.

4.2 Results

A confusion matrix is a tool used in machine learning and statistics to evaluate the performance of a classification model.

Predicted Classes

Figure 10. Confusion Matrix

It is especially useful for assessing the accuracy of a model's predictions when dealing with binary classification problems, such as distinguishing between "benign" and

"attack" labels. The confusion matrix typically consists of four values: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). The details are shown in Figure 10. The accuracy is defined as

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{19}$$

The precession is defined as

$$P = \frac{TP}{TP + FP} \tag{20}$$

The recall is defined as

$$R = \frac{TP}{TP + FN} \tag{21}$$

The F1-score is defined as

$$F1 = \frac{2PR}{P + R} \tag{22}$$

Table 3 provides a comprehensive overview of the simulation parameters utilized in the study, detailing the various factors and settings essential for conducting the experiments and analyzing the results.
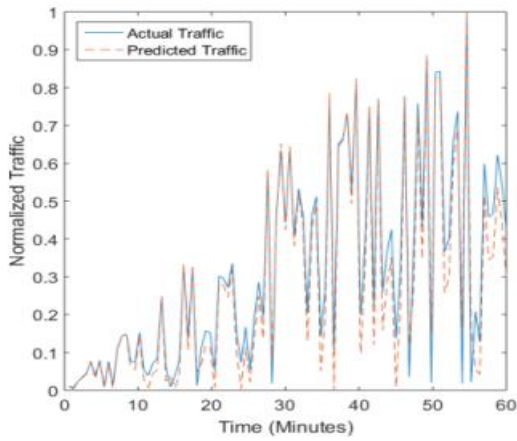
Table 3. Simulation Details

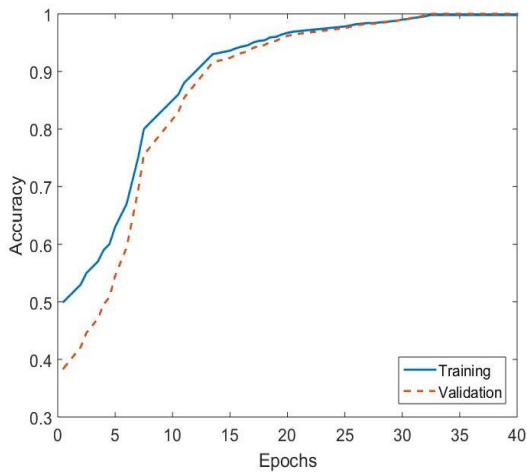| Parameters	Value
ConvLSTM1 Filter	128
Kernel Size	(1,3)
ConvLSTM 2 Filter	64
Kernel Size	(1,3)
LSTM1	200
LSTM 2	200
Activation	ReLU
Drop out	0.2
Regularization L2	0.001
Elephant Population Size	32
Number of clans	4
β	0.9
ψ	0.1
Number of iterations	100

Figure 11 presents a detailed analysis of network traffic patterns over time, measured in hours, and the performance of a predictive model. In part (a), it illustrates the comparison between the actual network traffic data and the predictions generated by proposed forecasting model. This visual representation allows analysts to assess how closely the model's predictions align with the observed real-world traffic patterns. It can be clearly visualized that the proposed model has good prediction of traffic. Part (b) provides insight into the model's accuracy over time, showing how well it captures the actual network traffic. In the initial epochs the accuracy is very low but as number of epochs increases the accuracy increases with maximum attainable accuracy of 99.981%. The provided classification model exhibits outstanding performance metrics. It achieves a

precision of 99.98%, signifying an extremely low rate of false positives. Additionally, it attains a perfect recall of 100%, implying that it correctly identifies all positive instances without any false negatives. Moreover, the F1-Score, which balances precision and recall, also reaches 99.98%, indicating a well-rounded and highly accurate model.
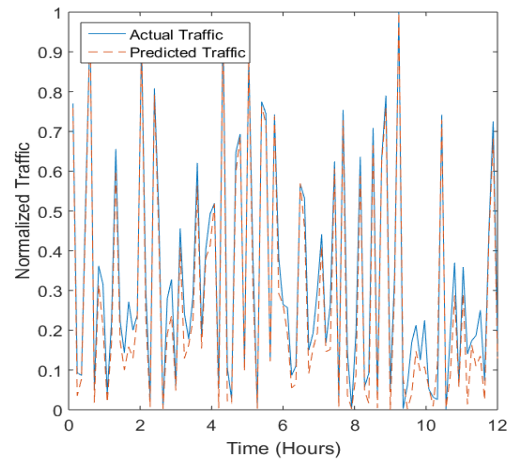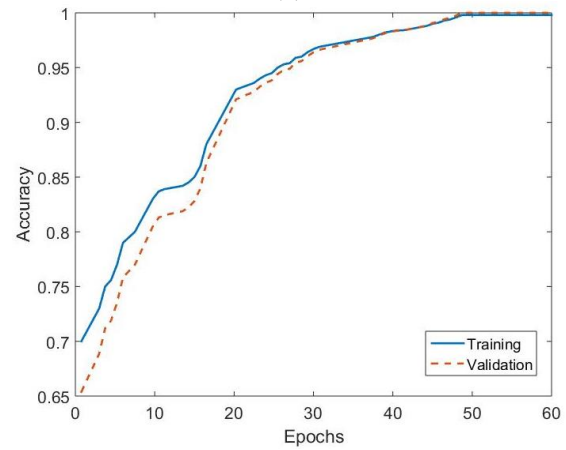

(a)

(b)

Figure 11. (a) Actual and Predicted Traffic (Minutes) (b) Accuracy (Minutes)

Figure 12 offers an extensive examination of network traffic trends over hourly intervals, presenting an evaluation of a predictive model's performance. In section (a), it visualizes the comparison between actual network traffic data and the predictions generated by the proposed forecasting model. This graphical representation serves as a means for analysts to gauge the alignment between the model's predictions and the real-world traffic patterns. Notably, it's evident that the proposed model exhibits a commendable ability to predict traffic accurately. In section (b), the figure sheds light on the model's accuracy across time, revealing how effectively it captures actual network traffic patterns. The accuracy initially starts at a lower level but steadily increases as the number of epochs progresses, eventually reaching a remarkable peak accuracy of 99.979%. The classification model performs exceptionally well, with precision, recall,
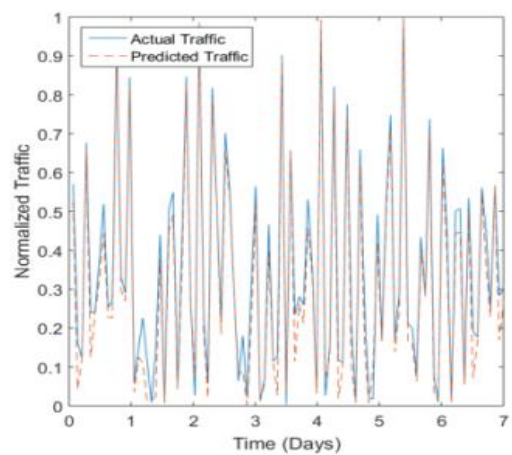
accuracy, and F1-Score all at an impressive 99.99%, indicating minimal errors and strong predictive capabilities.

(a)

(b)

Figure 12. (a) Actual and Predicted Traffic (Hours) (b) Accuracy (Hours)
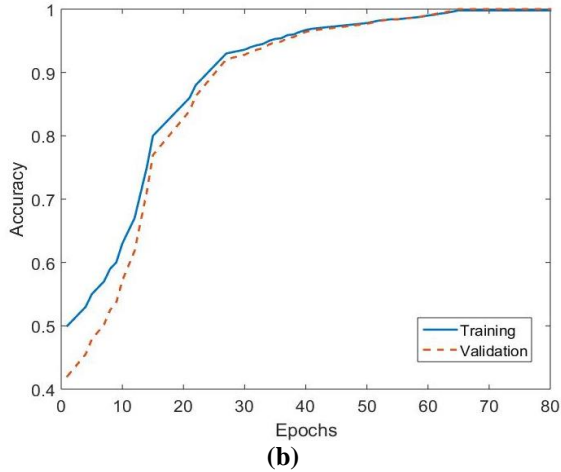

(a)

(b)

Figure 13. (a) Actual and Predicted Traffic (Days) (b) Accuracy (Days)

Figure 13 provides an extensive analysis of network traffic patterns over daily intervals, offering an assessment of the predictive model's performance. In section (a), it visualizes the comparison between actual network traffic data and the predictions generated by the proposed forecasting model. This graphical representation enables analysts to evaluate the alignment between the model's predictions and real-world traffic patterns. Importantly, it is evident that the proposed model demonstrates a noteworthy ability to accurately predict traffic.

Lastly, in section (b), the figure illuminates the model's accuracy over time, revealing its effectiveness in capturing actual network traffic patterns. The accuracy begins at a lower level initially but steadily improves as the number of epochs progresses, ultimately reaching an impressive peak accuracy of 99.989%. The performance metrics provided for the classification model are outstanding in every aspect. With a precision score of 0.9999, the model demonstrates an impressive ability to correctly identify positive instances, with an incredibly low rate of false positives. The recall score of 0.9999 indicates that it effectively captures nearly all of the actual positive instances, leaving very few false negatives. The high accuracy score of 0.99989 signifies that the model's overall predictions are almost flawless, correctly classifying the vast majority of instances in the dataset. Furthermore, the F1-Score, which harmonizes precision and recall, also stands at an impressive 0.9999, showing a well-balanced performance.

Figure 14 depicts a visual representation of the comparison of accuracy among different models or methodologies using bar graphs. Each bar in the graph corresponds to a specific model or approach, and the height of the bar indicates the level of accuracy achieved by that particular model. The accuracy values associated with the bars are reported as percentages, with precise measurements of 99.981%, 99.979%, and 99.989%, for minutes, hours and days respectively. These percentages signify the proportion of correctly classified instances relative to the total number of instances evaluated by each model.
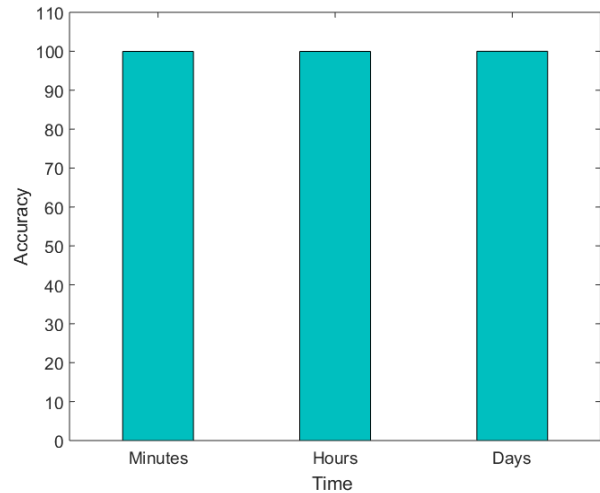

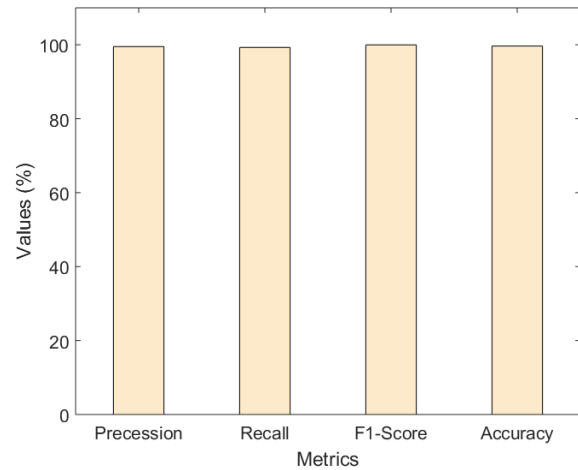**Figure 14.** Accuracy comparison

Figure 15. Metric values (Average)

Figure 15 presents the average values of key metrics, including precision, recall, F1-score, and accuracy, depicted as percentages. These metrics provide insights into the performance of the evaluated models or methodologies in terms of their ability to correctly classify instances within a given dataset. The reported average values for precision, recall, F1-score, and accuracy are 99.52%, 99.30%, 99.99%, and 99.66%, respectively.

The Table 4 summarizes the results of various research studies focused on intrusion detection systems (IDS) and their corresponding accuracies. Gao et al. [13] employed a machine learning approach and achieved an accuracy of 85.2%. The precession, recall and F1-score is also low. While their adaptive ensemble model showed effectiveness, it had limitations in detecting weaker attack classes. In contrast, Sabeel et al. [14] obtained an impressive Precision of 94.9% and an excellent F1-Score of 97.4%, leading to a high Accuracy of 98.72%. However, their model was limited to binary class classification.

Asad et al. [15] proposed a DNN-based model with a high accuracy of 98% but evaluated it solely against DDoS

13

attacks. Mural. et al. [16] used a DNN approach and achieved a strong accuracy of 99.61%, though their method was specific to detecting slow HTTP DoS attacks.

Table 4. Performance comparison with the state-of-the-art methods

Reference	Precession	Recall	F1-Score	Accuracy
Gao et al. [13]	86.5%	85.2%	84.9%	85.2%
Sabeel et al. [14]	94.9%	-	97.4%	98.72%,
Asad et al. [15]	-	-	99.0%	98%
Muraleedharan. et al. [16]	99%	-	-	99.61%.
Amaizu et al. [17]	99.52%	99.30%	99.99%	99.66%.
Hasan et al. [18]	99%	99%	99%	99%
Amma et al. [19]	-	-	-	99.3%
Chen et al. [20]	-	-	-	98.87%
Haider et al. [21]	99.57%	99.64%	99.61%	99.45%
Wang et al. [22]	98.99%	98.96%	98.97%	98.98%
Kim et al. [23]	-	-	-	99%,
Hussain et al. [24]	87%	86%	86%	87.06%
Li C et al. [25]	-	-	-	98%
Shu et al. [26]	-	-	-	99.19%
Bhardwaj et al. [27]	99.22%	97.12%	98.57%	98.43%
Moh. et al. [28]	98.42%	97.6%	98.05%	98.15%
He et al. [29]	-	-	-	87.8%
Proposed	99.98%	1	99.99%	99.98%

Amaizu et al. [17] employed a DNN and obtained an exceptional performance with a high Precision of 99.52%, a Recall of 99.30%, and an outstanding F1-Score of 99.99%. However, the complexity of their suggested model might result in longer detection times, which could impact real-time use. Hasan et al. [18] utilized a Deep CNN model, reaching 99% accuracy, outperforming three other machine learning methods. Nevertheless, their dataset had a limited number of cases. Amma et al. [19] combined Fully Connected Neural Networks (FCNN) and Variational Convolutional Neural Networks (VCNN), obtaining an accuracy of 99.3%. They outperformed basic classifiers and even more sophisticated systems but relied on an outdated dataset. Chen et al. [20] used a CNN model, with MCCNN performing well on constrained data and achieving an accuracy of 98.87%. However, they found no significant difference between multi-class and single-class classification models.

Haider et al. [21] introduced a Deep CNN ensemble technique that outperformed existing approaches, with an accuracy of 99.45% with high precession, recall and F1 score. Nevertheless, their method required longer training and testing periods. Wang et al. [22] employed an Entropy and Deep Learning technique, with their CNN model outperforming alternatives across multiple metrics but requiring a longer detection time.Kim et al. [23] utilized a CNN-based model that effectively recognized unique DoS attacks with similar features, achieving 99% accuracy, albeit at the cost of longer detection times. Hussain et al. [24] achieved a moderate accuracy of 87.06% by converting non-image network data into a suitable format for deep learning, although they did not consider the time required for data preparation. Li C et al. [25] applied a DL method and achieved 98% accuracy in DDoS assault detection. However, their model demanded a significant amount of time for detection. Shu et al. [26] combined hybrid-based IDS with LSTM, reaching an accuracy of 99.19%. Nonetheless, their method also required a substantial amount of time for detection.

Bhardwaj et al. [27] effectively addressed feature learning and overfitting issues using a DNN, resulting in an accuracy of 98.43% with reasonably good precession, recall and F1-score. However, their study was conducted offline and did not utilize recent datasets. Moh. et al. [28] combined LSTM with Bayes, achieving stable performance with new data and positive outcomes. Still, detecting attacks unsuitable for real-time applications may take longer with their LSTM–Bayes approach. Finally, He et al. [29] employed a strategy based on DTL for DDoS detection, achieving a 20.8% improvement in detecting the 8LANN network. However, they considered only single type of attack. The proposed system, represented as "Proposed," attained the highest accuracy of 99.98% among the mentioned studies, showcasing its potential as an effective intrusion detection system.

5. Conclusions

In conclusion, the research presented in this paper represents a significant step forward in the realm of network intrusion detection. The integration of ConvLSTM and EHO mechanisms has yielded a powerful model capable of accurately predicting network traffic and identifying potential security breaches. The findings not only contribute to advancing the field of intrusion detection but also have practical implications for enhancing network security in real-world scenarios. In nutshell, the findings from this study affirm the predictive model's exceptional performance in network traffic classification, with average values for precision, recall, F1-score, and accuracy are 99.52%, 99.30%, 99.99%, and 99.66%, respectively. This level of precision is particularly noteworthy in the context of network security, where the ability to accurately distinguish between benign and attack traffic is of paramount importance. The results underscore the model's reliability and its potential for application in real-world network security and traffic analysis tasks, where precise detection of security threats is essential for safeguarding network

integrity. In the future works, other meta-heuristic algorithm can be considered.

References

[1] Abdel-Rahman, Mohamed. "Advanced Cybersecurity Measures in IT Service Operations and Their Crucial Role in Safeguarding Enterprise Data in a Connected World." *Eigenpub Review of Science and Technology* 7, no. 1 (2023): 138-158.

[2] Adhikari, Naresh, and Mahalingam Ramkumar. "IoT and Blockchain Integration: Applications, Opportunities, and Challenges." *Network* 3, no. 1 (2023): 115-141.

[3] Khan, Abid, Awais Ahmad, Mansoor Ahmed, Jadran Sessa, and Marco Anisetti. "Authorization schemes for internet of things: requirements, weaknesses, future challenges and trends." *Complex & Intelligent Systems* 8, no. 5 (2022): 3919-3941.

[4] Pan, Gary, Poh Sun SEOW, Calvin Chan, and Chu Yeong LIM. "Analytics and cybersecurity: The shape of things to come." (2015): 1.

[5] Vinayakumar, R., K. P. Soman, and Prabaharan Poornachandran. "Applying deep learning approaches for network traffic prediction." In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 2353-2358. IEEE, 2017.

[6] Ramakrishnan, Nipun, and Tarun Soni. "Network traffic prediction using recurrent neural networks." In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 187-193. IEEE, 2018.

[7] Mahdavinejad, Mohammad Saeid, Mohammadreza Rezvan, MohammadaminBarekatain, Peyman Adibi, Payam Barnaghi, and Amit P. Sheth. "Machine learning for Internet of Things data analysis: A survey." *Digital Communications and Networks* 4, no. 3 (2018): 161-175.

[8] Zhao, Ling, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. "T-gcn: A temporal graph convolutional network for traffic prediction." *IEEE transactions on intelligent transportation systems* 21, no. 9 (2019): 3848-3858.

[9] Yang, Shuguan, Wei Ma, Xidong Pi, and Sean Qian. "A deep learning approach to real-time parking occupancy prediction in transportation networks incorporating multiple spatio-temporal data sources." *Transportation Research Part C: Emerging Technologies* 107 (2019): 248-265.

[10] Ranjan, Navin, Sovit Bhandari, Hong Ping Zhao, Hoon Kim, and Pervez Khan. "City-wide traffic congestion prediction based on CNN, LSTM and transpose CNN." *IEEE Access* 8 (2020): 81606-81620.

[11] Zhu, Hailong, Yawen Xie, Wei He, Chao Sun, Kaili Zhu, Guohui Zhou, and Ning Ma. "A novel traffic flow forecasting method based on RNN-GCN and BRB." *Journal of Advanced Transportation* 2020 (2020): 1-11.

[12] Yu, Bing, Haoteng Yin, and Zhanxing Zhu. "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting." *arXiv preprint arXiv:1709.04875* (2017).

[13] Gao, Xianwei, Chun Shan, Changzhen Hu, Zequn Niu, and Zhen Liu. "An adaptive ensemble machine learning model for intrusion detection." *Ieee Access* 7 (2019): 82512-82521.

[14] Sabeel, Ulya, Shahram Shah Heydari, Harsh Mohanka, Yasmine Bendhaou, Khalid Elgazzar, and Khalil El-Khatib. "Evaluation of deep learning in detecting unknown network attacks." In *2019 International Conference on Smart Applications, Communications and Networking (SmartNets)*, pp. 1-6. IEEE, 2019.

[15] Asad, Muhammad, Muhammad Asim, Talha Javed, Mirza O. Beg, Hasan Mujtaba, and Sohail Abbas. "Deepdetect: detection of distributed denial of service attacks using deep learning." *The Computer Journal* 63, no. 7 (2020): 983-994.

[16] Muraleedharan, N., and B. Janet. "A deep learning based HTTP slow DoS classification approach using flow data." *ICT Express* 7, no. 2 (2021): 210-214.

[17] Amaizu, Gabriel Chukwunonso, Cosmas Ifeanyi Nwakanma, Sanjay Bhardwaj, J. M. Lee, and Dong-Seong Kim. "Composite and efficient DDoS attack detection framework for B5G networks." *Computer Networks* 188 (2021): 107871

[18] Hasan, Md Zahid, KM Zubair Hasan, and Abdus Sattar. "Burst header packet flood detection in optical burst switching network using deep learning model." *Procedia computer science* 143 (2018): 970-977.

[19] Amma, Narayanavadivoo Gopinathan Bhuvaneswari, and Selvakumar Subramanian. "Vcdeepfl: Vector convolutional deep feature learning approach for identification of known and unknown denial of service attacks." In *TENCON 2018-2018 IEEE Region 10 Conference*, pp. 0640-0645. IEEE, 2018.

[20] Chen, Jinyin, Yi-tao Yang, Ke-ke Hu, Hai-bin Zheng, and Zhen Wang. "DAD-MCNN: DDoS attack detection via multi-channel CNN." In *Proceedings of the 2019 11th International Conference on Machine Learning and Computing*, pp. 484-488. 2019.

[21] Haider, Shahzeb, Adnan Akhunzada, Iqra Mustafa, Tanil Bharat Patel, Amanda Fernandez, Kim-Kwang Raymond Choo, and Javed Iqbal. "A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks." *Ieee Access* 8 (2020): 53972-53983.

[22] Wang, Lu, and Ying Liu. "A DDoS attack detection method based on information entropy and deep learning in SDN." In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1, pp. 1084-1088. IEEE, 2020.

[23] Kim, Jiyeon, Jiwon Kim, Hyunjung Kim, Minsun Shim, and Eunjung Choi. "CNN-based network intrusion detection against denial-of-service attacks." *Electronics* 9, no. 6 (2020): 916.

[24] Hussain, Faisal, Syed Ghazanfar Abbas, Muhammad Husnain, Ubaid U. Fayyaz, Farrukh Shahzad, and Ghalib A. Shah. "IoT DoS and DDoS attack detection using ResNet." In *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pp. 1-6. IEEE, 2020.

[25] Li, Chuanhuang, Yan Wu, Xiaoyong Yuan, Zhengjun Sun, Weiming Wang, Xiaolin Li, and Liang Gong. "Detection and defense of DDoS attack–based on deep learning in OpenFlow-based SDN." *International Journal of Communication Systems* 31, no. 5 (2018): e3497.

[26] Shurman, Mohammad M., Rami M. Khrais, and Abdulrahman A. Yateem. "DoS and DDoS attack detection using deep learning and IDS." *Int. Arab J. Inf. Technol.* 17, no. 4A (2020): 655-661.

[27] Bhardwaj, Aanshi, Veenu Mangat, and Renu Vig. "Hyperband tuned deep neural network with well posed

stacked sparse autoencoder for detection of DDoS attacks in cloud." *IEEE Access* 8 (2020): 181916-181929.

[28] Mohammadnia, Hamzeh, and Slimane Ben Slimane. "IoT-NETZ: Practical spoofing attack mitigation approach in SDWN network." In *2020 Seventh International Conference on Software Defined Systems (SDS)*, pp. 5-13. IEEE, 2020.

[29] He, Jiawei, Yejin Tan, Wangshu Guo, and Ming Xian. "A small sample DDoS attack detection method based on deep transfer learning." In *2020 International Conference on Computer Communication and Network Security (CCNS)*, pp. 47-50. IEEE, 2020.

[30] Patil, Dharmaraj R., and Tareek M. Pattewar. "Majority voting and feature selection based network intrusion detection system." *EAI Endorsed Transactions on Scalable Information Systems* 9, no. 6 (2022).

[31] Venkateswaran, N., and S. P Prabaharan. "An efficient neuro deep learning intrusion detection system for mobile adhoc networks." *EAI Endorsed Transactions on Scalable Information Systems* 9, no. 6 (2022).

[32] Fatima, Masooma, Osama Rehman, and Ibrahim MH Rahman. "Impact of features reduction on machine learning based intrusion detection systems." *EAI Endorsed Transactions on Scalable Information Systems* 9, no. 6 (2022): e9-e9.

[33] Zhang, Ji, Xiaohui Tao, and Hua Wang. "Outlier detection from large distributed databases." *World Wide Web* 17 (2014): 539-568.

[34] Kabir, Enamul, Jiankun Hu, Hua Wang, and Guangping Zhuo. "A novel statistical technique for intrusion detection systems." *Future Generation Computer Systems* 79 (2018): 303-318.

[35] Alkanhel, Reem, El-Sayed M. El-kenawy, Abdelaziz A. Abdelhamid, Abdelhameed Ibrahim, Manal Abdullah Alohali, Mostafa Abotaleb, and Doaa Sami Khafaga. "Network Intrusion Detection Based on Feature Selection and Hybrid Metaheuristic Optimization." *Computers, Materials & Continua* 74, no. 2 (2023).

[36] Gul, Omer Melih, Michel Kulhandjian, Burak Kantarci, Azzedine Touazi, Cliff Ellement, and Claude D'amours. "Secure industrial iot systems via rf fingerprinting under impaired channels with interference and noise." *IEEE Access* 11 (2023): 26289-26307.

[37] Li, Juan, Hong Lei, Amir H. Alavi, and Gai-Ge Wang. "Elephant herding optimization: variants, hybrids, and applications." *Mathematics* 8, no. 9 (2020): 1415.

[38] Ismaeel, Alaa AK, Islam A. Elshaarawy, Essam H. Houssein, Fatma Helmy Ismail, and Aboul Ella Hassanien. "Enhanced elephant herding optimization for global optimization." *IEEE Access* 7 (2019): 34738-34752.