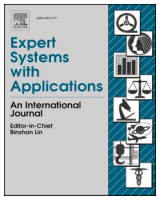
Expert Systems With Applications 245 (2024) 123027

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Dugat-LSTM: Deep learning based network intrusion detection system using chaotic optimization strategy

Ramkumar Devendiran, Anil V Turukmane *

School of Computer Science & Engineering, VIT-AP University, Amaravati, Vijayawada, Andhra Pradesh, 522237, India

ARTICLE INFO

Keywords:
Synthetic sampling approach
Principal component analysis
Honey badger optimization
Long short term memory
Gated recurrent unit

ABSTRACT

Network intrusion is a huge harmful activity to the privacy of the data sharing network. The activity will result in a cyber-attack, which causes damage to the system as well as the user's data. Unauthorized activities such as data tampering, illegal access to data and theft of credentials are increasing on the internet world every day. The detection of intrusion may be done by multiple methodologies; still, it is the biggest issue in the networks. Hence, an automated attack classification model is required to promote classification accuracy with fewer error possibilities based on the input parameters. To get relief from the insecurity of data, this paper presents an innovative model using deep networks. The proposed model is a deep learning based network intrusion detection system using a chaotic optimization strategy. The method is pre-processed using data cleansing and M-squared normalization. After pre-processing, the unbalanced datasets are balanced using the Extended Synthetic Sampling approach. After balancing, the features of the dataset are taken out using kernel-assisted principal component analysis. The optimal features are selected by the Chaotic Honey Badger optimization algorithm. After all required features have been extracted, the attacks are classified by the Gated Attention Dual Long Short Term Memory (Dugat-LSTM). The above process is performed using the TON-IOT and NSL-KDD datasets. The prototype is evaluated using the following metrics: accuracy, precision, recall, and F1 score. The accuracy value of the proposed model is 98.76% in the TON-IOT dataset and 99.65% in the NSL-KDD dataset. Thus, the accuracy and robustness of the model show that it outperforms other existing models.

1. Introduction

The interaction between hardware and software via the Internet is called the Internet of Things (IoT), which is a feasible technology (Koohang et al., 2022). It enables efficient and remote operations in hardware devices through software functions, reducing time consumption. However, certain issues can arise with this technology due to inadequate security measures and many malware attacks (Laghari et al., 2021). Cyber-attacks are becoming a challenging issue for IoT device security. Each security service has confidentiality, availability, and integrity, which are degraded by malicious software (Khraisat et al., 2019). Identifying the intrusion in the network is a complicated task to ensure the security of the data. There are mainly two types of cyber-attacks: passive and active attacks, where active attacks are easy to detect compared to passive attacks (Tummala & Inapakurthi, 2021; Naseri & Gharehchopogh, 2022). Fig. 1 describes some of the main types of intrusion attacks.

Privacy and security are important factors in data exchange and network processing. Various equipment, such as antivirus software, firewalls and intrusion detection systems (IDS), are used to improve security measures (Ahmad et al., 2021). Among these tools, IDS plays a crucial role in identifying intruders and monitoring abnormal behaviour in the network to protect users' data from threats (Ramkumar et al., 2022). In Mobile Ad-Hoc Networks (MANET), security is a big concern because finding intruders using IDS is complex (Ramkumar et al., 2019). IDS examines network traffic, looking for known and suspicious threats. To meet the needs of an efficient IDS, scholars have explored the potential of utilizing machine learning (ML) and deep learning (DL) techniques (Thakkar & Lohiya, 2023). ML and DL fall within the broad realm of artificial intelligence (AI) and determine to extract valuable insights from large data sets (Verma & Ranga, 2020). These methods have become extremely popular in network security over the past decade. Moreover, the development of high-performance graphics processing units (GPUs) is a major contributor to the development of deep networks.

Both ML and DL serve as effective means to extract valuable features

* Corresponding author.
E-mail addresses: ramkumar.d@vitap.ac.in (R. Devendiran), anil.turukmane@vitap.ac.in (A.V. Turukmane).

https://doi.org/10.1016/j.eswa.2023.123027
Received 4 August 2023; Received in revised form 8 December 2023; Accepted 22 December 2023
Available online 11 January 2024

and predict abnormal activities based on learned methods of attacks (Bharti et al., 2021). Deep learning intrusion detection systems (DL-IDS) can learn complicated characteristics from raw data without the need for feature engineering (Dushyant et al., 2022). IoT requires many features to protect connected applications. The applications are secured using deep learning models to monitor the process flow and data exchange (Kasongo & Sun, 2019; Abdullahi et al., 2022). IDS are classified into two types, namely network IDS and host IDS. Network-based intrusion detection systems provide network security, whereas host-based intrusion detection systems provide host security (Aldarwbi et al., 2022). Others are some categories of IDS commonly used in network security, such as signature-based, hybrid-based and anomaly-based systems (Choi et al., 2019).

Intrusion detection is based on past attacks in the signature-based IDS. Anomaly detection of cyber-attacks is based on the behaviour of signals or systems (Meira et al., 2020). Consequently, data processing activities are checked and investigated to protect the network from vulnerabilities (Kasongo & Sun, 2020; Lindemann et al., 2021). The detection is inefficient due to the zero-day attack and the lack of taxonomy, so many techniques are implemented to overcome these problems. This paper refers to an innovative deep learning model to fix the existing problems in IDS. They are maximizing the robustness and accuracy of an adaptive optimization based deep learning model for IDS. The data is pre-processed using pre-processing methods such as data cleaning and M-squared normalization.

The synthetic sampling approach to solving the imbalance is extended, and principal component analysis helps in feature extraction. The classification of these features is done using the gated attention dual long short term memory (LSTM), which helps to remember the attacks made in the early decades. LSTM is the dynamic modelling of a threat defence system (Lindemann et al., 2021; Laghrissi et al., 2021). TON-IOT and NSL-KDD datasets are taken for evaluation to show the enactment of the detection structure. The background, research gaps, motivation, contribution and organization are described in the following sections.

1.1. Background

An essential part of cybersecurity is intrusion detection systems (IDS), which are designed to detect and respond to hostile activity or unauthorized access to a computer network or system. IDS are primarily used to monitor system and network activities in real-time or near real-time to identify and remediate security incidents (He et al., 2023). Some host-based intrusion detection systems (HIDS) analyze events on a single device (host) (Sworna et al., 2023). Network-based intrusion detection systems (NIDS) track and identify unusual behavior or trends in network traffic. Previously, signature-based detection compares known malicious activity patterns (signatures) with given patterns of observed events (Díaz-Verdejo et al., 2022). Develops a standard for normal operation and, using anomaly-based detection, alerts the user to any changes from that standard that could indicate the presence of intruders.

Behavioral-Based detection seeks out unexpected patterns or aberrations in user and system behavior. IDS can be implemented directly on individual devices, within subnets, or throughout a network. When an IDS detects potential infractions or suspicious activities, it sends an alert. On the other hand, these warnings can initiate automated countermeasures, log the event, and notify administrators (Alem et al., 2023). One of the main challenges in IDS design is striking a balance between false positives, which mistakenly identify the typical activity as intrusions, and false negatives, which fail to detect true intrusions. IDS frequently function with firewalls, antivirus programs, and other security measures as part of larger security systems. IDS alerts when potential breaches or suspicious activity is detected. This allows automated countermeasures to be initiated, the incident registered, and administrators notified.

However, one of the main challenges in designing IDS is to ensure that false positives are identified as intruders and false negatives are not, as this can lead to false positives and false negatives. IDS is often used as part of a larger security system, such as a firewall, antivirus, and more. Recently, DL and ML based intrusion systems have been used for intrusion detection in networks. LSTM is a deep learning-based model that provides extremely suitable detection results due to its long-term dependencies. Likewise, many models are being introduced to detect

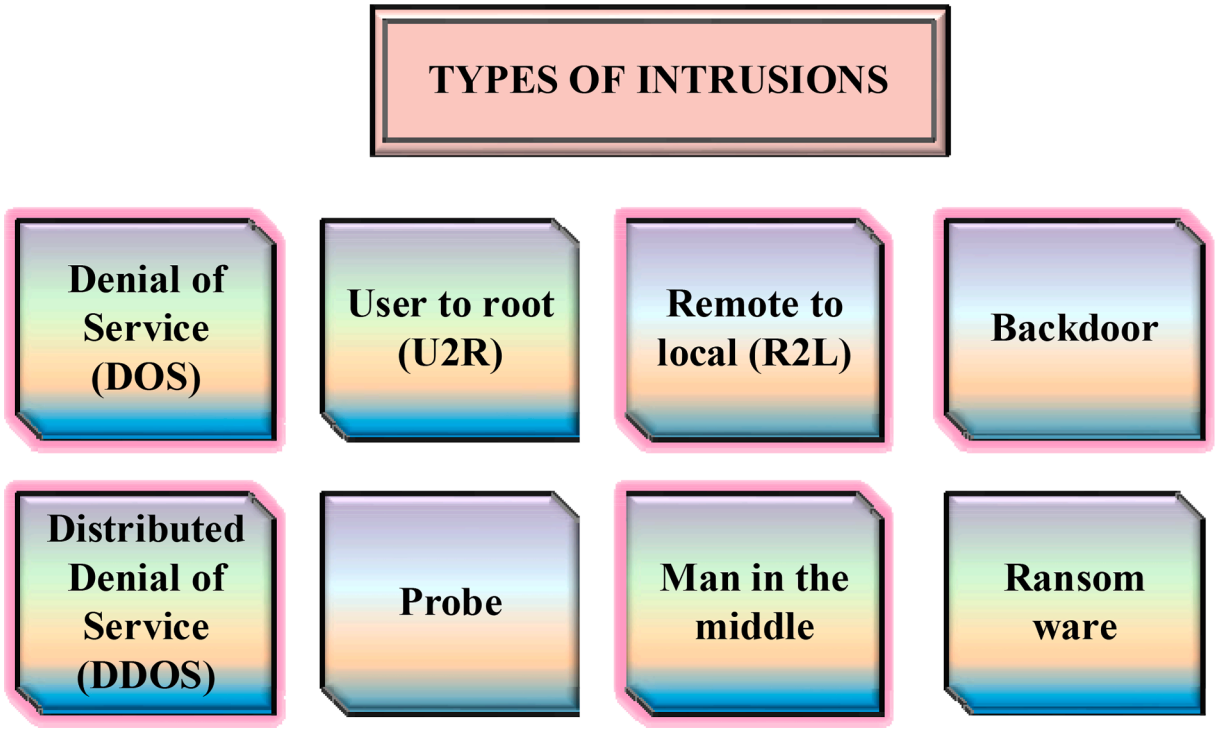


Fig. 1. Types of Intrusions.

intrusions (Saurabh et al., 2022). The research gaps will be given as follows to know the details about existing IDS approaches and challenges.

1.2. Research gaps

Rapidly evolving cyber threats pose a significant challenge to IDS. Research could focus on developing adaptive systems that quickly learn and respond to emerging threats without requiring frequent manual updates. Previously, detecting unknown or zero-day attacks remains a challenging problem (Guo, 2023). Research is needed to develop techniques to identify and mitigate attacks without known signatures or patterns. Improving anomaly detection methods to reduce false positives and improve the accuracy of detecting unusual behaviour. This involves developing more sophisticated models to better distinguish between normal and malicious activities (Xu et al., 2023). Research that could explore methods to improve the real-time detection and response capabilities of IDS, especially in large-scale (Javed et al., 2023) and high-speed networks where delays can be critical. While deep learning has shown promise in various domains, further research is needed to explore its potential for intrusion detection.

This involves fixing problems with resource efficiency, adversarial attacks, and interpretability. The increasing integration of IoT devices and industrial control systems into networks introduces new challenges for intrusion detection. Research can focus on developing specialized IDS for these environments, considering their unique characteristics and constraints. Improved privacy for individuals and organizations is possible through investigation techniques that enable effective intrusion detection. In addition to growing privacy concerns, this is more crucial than ever. For intrusion detection systems (IDS), the goal is to design resilient and dynamic designs that can adjust to new infrastructure, attack methods, and network configurations. Gaining confidence in the system and enhancing security analysts' understanding can be achieved by making IDS outputs more explainable (Pande & Khamparia, 2023). Transparent and interpretable models are crucial for effective collaboration between automated systems and human analysts. Methods were also created to integrate threat intelligence from various sources into IDS effectively. This involves addressing data heterogeneity, reliability, and real-time updating issues. Research in this area can help in developing countermeasures that make IDS more robust against evasion attempts. Intrusion detection solutions need to be resource-efficient, particularly for resource-constrained environments or edge computing scenarios. Researchers and practitioners in the field are likely to focus on these and other emerging challenges to enhance the capabilities of IDS in the face of evolving cyber threats.

1.3. Motivation

The development of IoT network technology simultaneously creates a loophole for insecure data transmissions due to insecure network connections. This leads to an urgent need to protect the computer networks from unauthorized access, malicious activities and potential security threats. An intrusion into the network can pose a risk to the security of user data and also lead to incorrect functionality. In addition, the increasing sophistication of cyberattacks poses a significant threat to the confidentiality, integrity and availability of data in computer networks. Various approaches have been developed to solve the intrusion problem, although they have failed to detect it accurately. These methods also have the disadvantage of gaining computational cost, lower performance detecting various attacks, etc. These challenges motivated research efforts to detect and prevent unauthorized access, data breaches, and other malicious activities by implementing a deep learning-based approach. These approaches aim to create a robust and flexible model to achieve adequate earlier detection of network intrusions.

1.4. Contribution

The major involvement of the paper about the model is described as follows,

- To present efficient pre-processing using data cleaning and M-square normalization to transform the original data into the useful data needed
- To solve an imbalance problem, the Extended Synthetic Sampling (ExSyn) approach was developed.
- To present Kernel assisted Principal Component Analysis (KerPCA) for well-defined feature extraction.
- To provide a Chaotic Honey Badger optimization algorithm (CHbO) for the selection of optimal features.
- To implement a Gated Attention dual long short term memory (Dugat-LSTM) to classify the various types of attacks.

In this paper, following the abstract and introduction, section 2 refers to the existing models of IDS and its drawbacks. Section 3 explains the proposed model, and section 4 describes the results of the model evaluation. Finally, section 5 describes the conclusion and future works of the proposed models.

2. Related works

This section covers the survey of some existing works that are deployed for detecting intrusion in the networks.

Ravi et al. (2022) constructed an intrusion system with recurrent deep learning-based feature fusion that ensemble meta-classifiers. The model gathers the characteristics from the hidden layers of recurrent models and uses a kernel-based principal component analysis (KPCA) feature selection method to get the best features. Finally, an ensemble meta-classifier was used to classify data after the best features of several recurrent models were combined. The suggested approach outperformed other existing models as well as current methodologies, according to experimental analysis and findings on many benchmark network intrusion datasets. Furthermore, the proposed method showed the highest accuracy and 97 % in network threat categorization using the SDN-IoT dataset.

Caville et al. (2022) introduced an Anomal-E, a Graph neural networks (GNN)-based intrusion and anomaly detection method that used a graph's topological structure and edge properties in a self-supervised way. This strategy was the first to use network flows within an edge-leveraging, self-supervised GNN for network intrusion detection. Two contemporary benchmarks NIDS datasets' experimental findings show a noteworthy improvement above raw features and other baseline algorithms when employing Anomal-E. This further asserts the ability of Anomal-E to identify intrusions on actual network traffic.

Halbouni et al. (2022) utilized the spatial feature extraction capabilities of the CNN and the temporal feature extraction capabilities of the LSTM Network, and a hybrid intrusion detection system model was produced. In the model, dropout layers and batch normalization are included to improve its efficiency. The model was trained using the binary and multi-class classification as the basis. CIC-IDS 2017, UNSW-NB15, and WSN-DS are the three datasets. The suggested model obtained an accuracy of 94.53 %.

Kasongo (2023) suggested a learning technique for the IDS using the recurrent neural network (RNN). This model used different types of RNNs, most notably LSTM, Gated Recurrent Unit (GRU) and Simple RNN. The NSL-KDD and UNSW-NB15 datasets were considered to examine the performance. XGBoost-LSTM achieved a TAC of 88.13 %, a VAC of 99.49 % and a training time of 225.46 s in the binary classification using NSL-KDD. For the multi-class classification scheme, XGBoost-LSTM achieved an 86.93 % TAC versus NSL-KDD, and XGBoost-GRU achieved a 78.40 % TAC versus the UNSW-NB15 dataset.

Hnamte and Hussain (2023) developed a model to form a network-

based detection. The developed model combined a convolutional neural network (CNN) and bidirectional LSTM with more than 10 hidden layers. Pre-processing was done using what are known as one-hot encoding and data normalization techniques, and an auto-encoder was used to reduce dimensionality. The model was trained on live traffic datasets such as CICIDS2018 and Edge_IIoT. The effectiveness of the model is examined using categorization in multiple classes, and a precision rate of 100 % and 99.64 % was achieved when trained and examined on the datasets. The disadvantage of the developed model was its complexity due to the longer training time.

Abd Elaziz et al. (2023) created a model for IDS in a cloud and IoT environment. The created model used swarm intelligence algorithms stacked with deep networks to build a productive IDS. Firstly, deep networks were employed to determine ideal properties from the IoT IDS datasets. A productive technique of feature selection based on a Swarm optimizer known as the Capuchin Search Algorithm (CapSA) was then proposed. The effectiveness was assessed using four IoT cloud datasets, specifically NSL-KDD, BoT-IoT, KDD99 and CIC2017.

Muhammad et al. (2023) constructed a Security Information and Event Management (SIEM) system on the basis of real-time analytics. IDS uses ML integrated with SIEM, and the model requires a consolidated framework with multiple methods and services. For live analysis, detection and monitoring of cyber-attacks, the built system used open source components. The combination of Elastic (ELK) Stack, Slips and Zeek IDS was used to build the system. The system was tested using a Denial of Service (DoS) test with packets at a speed of 344.1/s. The performance tests show that elastic search is the component that consumes the most CPU and RAM, accounting for 78 % of CPU usage and 2300 MB of RAM usage.

Altunay and Albayrak (2023) advanced an IoT and cloud-based IDS model in the IoT ecosystem. Three models had been created to identify IIoT network intrusions using the CNN and LSTM hybrid deep learning architecture. The prototype was developed using the UNSW-NB15 datasets that achieved an accuracy of 93.21 % for binary classification and 92.9 % for multi-class classification in the UNSW-NB15 dataset. On the other hand, the model offers a high level of complexity due to the use of hybrid networks.

The overall performance of existing models and their drawbacks are shown in Table 1.

3. Proposed methodology

Network security is an extremely important feature when exchanging data over the Internet. Cyber-attacks are becoming increasingly common in the IoT ecosystem due to reduced security measures. These existing models have many disadvantages, such as synthetic sample imbalance, lower detection accuracy, zero-day attacks, lack of taxonomy, etc. This article introduces an innovative network IDS model on the basis of a deep learning approach. The proposed model provides automated attack classification with high accuracy and fewer errors. Gated attention is performed using dual LSTM for intruder detection, so the model is referred to as Dugat-LSTM. The systematic representation is shown in Fig. 2.

3.1. Pre-processing of data

Pre-processing is the first phase of the proposed model to improve the robustness by removing the variants and noises. This will done by data cleaning and M-square normalization. Those techniques are described below.

3.1.1. Data cleaning

Data cleaning (Toupas et al., 2019) is a process that removes inaccurate data on the network to improve network traffic quality. Data cleaning measures remove duplicates and correct errors, contributing to better data integrity. The process also identifies those pieces of data that are irrelevant to the given network. Given the high false alarm rate of IDS, data sanitization is a necessary process to detect intruders. The cleaning process takes place in three steps: data quality, measurement and improvement. The steps of data cleaning are described below,

Stage 1- Data Quality

The datasets TON-IOT and NSL-KDD are utilized as input here, and the data is supplied in the data analysis step. The analysis determines the quality dimension of the given data and outputs a list of quality dimensions.

Stage 2- Measurement

The output of the first stage is used as an input in the measurement stage, which performs a quality measurement based on quality metric definitions. As a result, the quality of the data value and the problems are determined.

Stage 3- Improvement

By comparing the requirement quality with the data quality score, the quality of the input data is improved by trying different methods. This process has a key feature of improving the overall quality of the dataset by identifying and rectifying errors. High-quality data is required for accurate insights and making educated decisions. Clean data helps model analyses produce more accurate and trustworthy results. Incorrect or inconsistent data might lead to incorrect conclusions and faulty detection.

3.1.2. M-square normalization

After the specified data is moved out of the data cleansing phase, data normalization is performed using M-squared normalization. Normalization provides robust data, and any data size can be normal-

Table 1
Survey of existing models.

Author name and reference	Technique used	Dataset used	Performance	Drawbacks
Ravi et al. (2022)	RNN	SDN-IoT	Achieve accuracy up to 97.00 %	Did not categorize the attacks
Caville et al. (2022)	GNN	NIDS datasets	Achieve accuracy up to 98.63 %	Low efficiency in temporal feature extraction
Halbouni et al. (2022)	CNN-LSTM	CIC-IDS 2017, UNSW-NB15, and WSN-DS	Achieve accuracy up to 94.53 %	Inefficient in imbalanced datasets
Kasongo (2023)	XGBoost-LSTM	NSL-KDD and the UNSW-NB15	Test accuracy of 88.13 %, a validation accuracy of 99.49 %	Less performance in minority classes
Hnamte and Hussain (2023)	DCNNBiLSTM	CICIDS2018 and Edge_IIoT datasets	99.64 % precision rate	There is no categorization of incoming traffic
Abd Elaziz et al. (2023)	CNN-CapSA	NSL-KDD, BoT-IoT, KDD99, and CIC2017	Acquire optimal characteristics extraction	computational costs are high
Muhammad et al. (2023)	ELK stack SIEM	Denial of Service (DoS) test	DoS test with 344.1/sec	The accuracy of detecting different cyber-attacks is less
Altunay and Albayrak (2023)	CNN + LSTM	UNSW-NB15 dataset	Accuracy of 93.21 % and 92.95 for binary and multi-classification, respectively.	The model has a memorization problem.

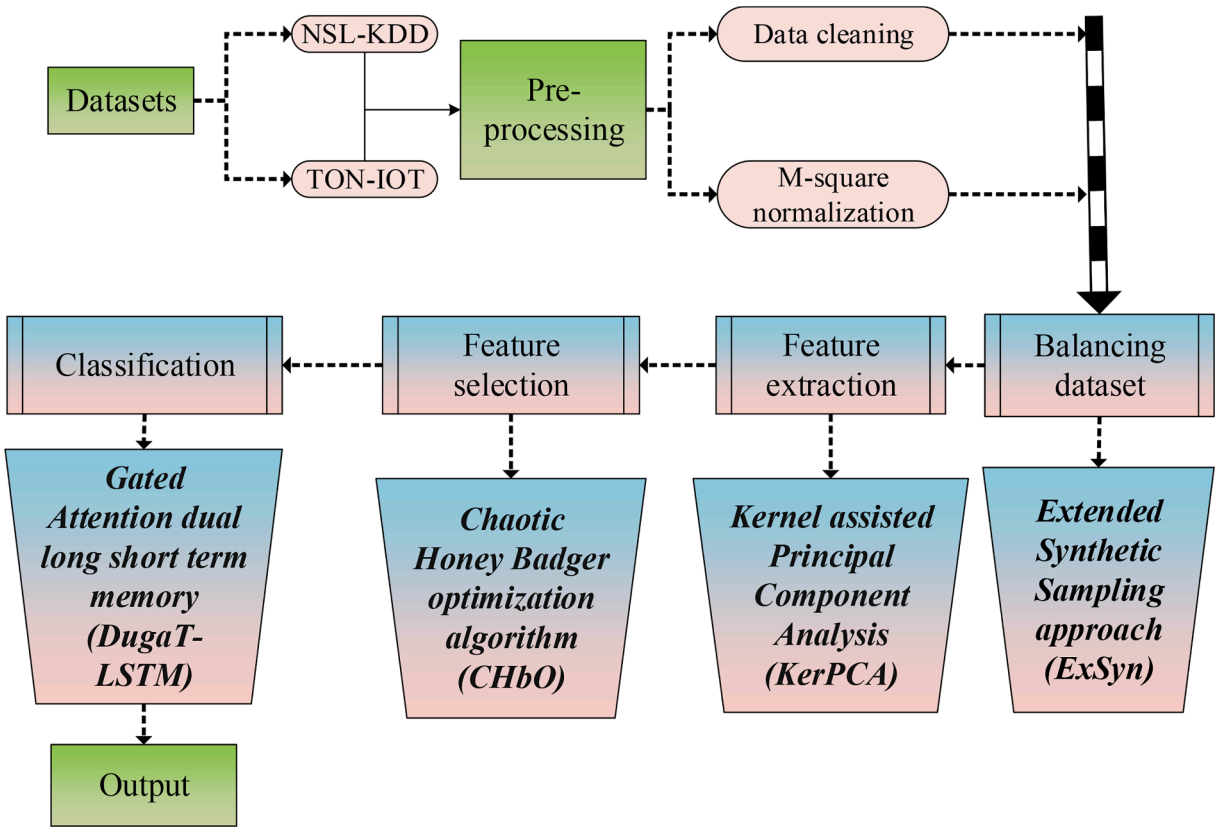


Fig. 2. Overview of the proposed model.

ized. First, a linear regression model which have S^2 value in each set that represents in the form of a given equation,

$$Zs = \frac{\widehat{\mu}_{me} - y_j}{\widehat{\sigma}_{mad}} \tag{1}$$

Where Zs is a Z score,$\widehat{\mu}_{me}$ is a Median estimator and $\widehat{\sigma}_{mad}$ is a Median Absolute Deviation, and any size of data is mentioned as an integer type. The normalized data ensures that the input conforms to a median of 0 to a normal distribution and a mean absolute deviation of 1. Eliminates multiple collinearity while generating a multi-variant linear regression model while improving the capability of the model. The advantage of M-squared normalization is that identical center lines are obtained. This normalization scales features to a common range, making the model less sensitive to differences in the scales of individual features. Normalizing features to a common scale can facilitate the interpretation of model coefficients. The principal component analysis works well when features are within a specific range. Outliers or extreme values in features can have a disproportionate impact on the performance of the model. Min-max normalization can help mitigate this influence by constraining the feature values to a specific range, reducing the impact of outliers. This advancement makes the process efficient in feature extraction and makes the model robust enough to detect a range of attacks. The result of the normalization is that the centrality value for two data sets of different sizes has almost the same value. In addition, the data distribution for both datasets decreases, and the minimum value for both datasets has almost no difference (Singh & Kundu, 2022). The dataset with normalized data is then balanced in the next sampling phase.

3.2. Extended synthetic sampling approach

The size of the classes in the data set can be unbalanced in some cases. These classes are balanced using a kernel-based approach (Liu & Hsieh, 2019). There is a significant learning bias after using this method. An advanced synthetic sampling approach (ExSyn) is implemented to solve the problem. The sampling approach is based on the SMOTE approach. An example of the ExSyn approach using the algorithm is explained further.

The dataset D with samples s $\{a_x, b_x\}$, $x = 1....s$ where a_x an instance of the dimensional feature space is indicated as A and $b_x \in B = \{1, -1\}$ is a label with a_x. The majority and minority class is indicated as max and min, therefore, max⩽min and max + min = s. To define the degree of class imbalance

$$E = \frac{\max}{\min} \tag{2}$$

where $E \in (0, 1)$ is a degree, considered the preset threshold of degree in the class imbalance ratio. A count of synthetic data is generated for the minority classes

$$g = (\min - \max) \times \beta \tag{3}$$

here $\beta \in [0, 1]$ represents the balancing level,$\beta = 1$ is a fully balanced dataset where $a_x \in$ min*orityclass*. To find the K nearest neighbour ratio R is needed to be found using the below equation,

$$R_x = \frac{\Delta_x}{Kn}, \ x = 1, ..., s \tag{4}$$

where Δ_x is the samples in K nearest neighbours of a_x which belongs to the majority class; therefore $R_x \in [0, 1]$, the ratio is normalized according to the following equation

$$\widehat{R}_x = \frac{R_x}{\sum_{x=1}^{s} R_x} \tag{5}$$

Here R_x is a density distribution, then count the synthetic data samples to find the minority a_x

$$S_x = \widehat{R}_x \times g \tag{6}$$

where S_x is a total range of synthetic data samples that create a minority class defined in Eq. (2). The loop is continued from 1 to S_x then one minority data is chosen randomly Y_{xi} from the K nearest neighbour for data a_x. The equation for creating the synthetic data is given as,

$$g_x = a_x + (a_{ix} - a_x) \times \lambda \tag{7}$$

Where $(a_{ix} - a_x)$ is the vector of difference in the dimensional area and λ is a randomly assigned number $\lambda \in [0, 1]$ after creating the synthetic data, the loop will end and then move to feature extraction.

3.3. Feature extraction using KerPCA

The pre-processed and adjusted data set then enters the feature extraction phase. The characteristics of each data are extracted using the following extraction techniques.

3.3.1. Kernel assisted principal component analysis

Kernel assisted principal component analysis (KerPCA) is an extended form of principal component analysis (Uddin et al., 2021). In the KerPCA, the process of feature extraction is done in two phases that are described as follows

Phase 1: Training in offline

The trained data matrix is evaluated in the first analysis phase, and the α will indicate the co-efficient matrix (Eigenvector). The vector corresponds to eigenvalues arranged in descending order. A kernel matrix decomposition is provided to obtain the Eigenvalue. The covariance matrix CO_m is initially found out by the following equation,

$$CO_m = \frac{1}{x}\sum_{j=1}^{x} \Phi(y_j)\Phi(y_j)^d \tag{8}$$

Here, the Eigen vector e and Eigenvalues λ are satisfied by the following equation. In the equation, i represent the i^{th} dimension $i = [1, 2, ..., n]$ in the feature space.

$$CO_m e_i = \lambda_i e_i \tag{9}$$

Using above equation, the following equation will be obtained,

$$\frac{1}{x}\left[\sum_{j=1}^{x} \Phi(y_j)\Phi(y_j)^d\right] e_i = \frac{1}{x}\sum_{j=1}^{x} (\Phi(y_j).e_i)\Phi(y_j) = \lambda_i e_i \tag{10}$$

The linear combination is given as eigenvectors with m map data values.

$$e_i = \frac{1}{x\lambda}\sum_{j=1}^{x} (\Phi(y_j).e_i)\Phi(y_j) = \sum_{k=1}^{x} \alpha_{i.k}.\Phi(y_k) \tag{11}$$

At this point, e_i is inset to get the further equations,

$$\frac{1}{x}\sum_{j=1}^{x} \Phi(y_j)\Phi(y_k)^d \sum_{k=1}^{x} \alpha_{i.k}\Phi(y_k) = \lambda_i \sum_{k=1}^{x} \alpha_{i.k}\Phi(y_k) \tag{12}$$

By the definition of the kernel function, $\kappa(y_j, y_k) = \Phi(y_j)^d\Phi(y_k)$ and $\Phi(y_l)^d$ is multiplied in both sides of equation (12), such that the equation changed as follows,

$$\frac{1}{x}\sum_{j=1}^{x} \kappa(y_l, y_j) \sum_{k=1}^{x} \alpha_{i.k}\kappa(y_j, y_k) = \lambda_i \sum_{k=1}^{x} \alpha_{i.k}\kappa(y_l, y_k) \tag{13}$$

Where $l = 1, 2, ..., n$ then the Eq. (13) is converted into a matrix

$$\kappa^2\alpha = \lambda x \kappa \alpha \tag{14}$$

where κ belongs to the matrix of the real number $\Re^{m\times n}$, the mean of mapped data is shifted from m the sensors, $\Phi(y_j)$ to 0, and Eq. (16) gives the centre of kernel matric with the definition of Eq. (15).

$$\widetilde{\Phi}(y_{j\,or\,k}) = \widetilde{\Phi}(y_{j\,or\,k}) - \frac{1}{m}\sum_{j=1}^{x} \Phi(y_{j\,or\,k}) \tag{15}$$

$$\widetilde{\kappa}(y_j, y_k) = \widetilde{\Phi}(y_j)^d\widetilde{\Phi}(y_k) \tag{16}$$

At this time $\widetilde{\kappa}$ is the center of the kernel matrix; thus, equation (16) is rewritten in the form of a matrix,

$$\widetilde{\kappa} = \kappa - 1_x\kappa - k1_x + 1_x\kappa 1_x \tag{17}$$

Where 1_x is a $m \times n$ matrix every element of the matrix is equal to $\frac{1}{x}$. In Eq. (14), $\widetilde{\kappa}$ is replaced to κ, the Eigen problem is resolved by the following equation,

$$\widetilde{\kappa}\alpha = \lambda x \alpha \tag{18}$$

The above equation gives the Eigenvectors $\alpha_1, \alpha_2, \alpha_{x-1}\alpha_x$ and the corresponding Eigenvalues are $\lambda_1 \geqslant \lambda_2 \geqslant ... \geqslant \lambda_{x-1} \geqslant \lambda_x$. The normalization of the Eigenvector is done by the following,

$$e_i^d e_i = \sum_{j=1}^{x}\sum_{k=1}^{x} \alpha_{i.j}\alpha_{i..k}\Phi(y_j)^d\Phi(y_k) = \lambda_i\alpha_i^d\alpha_i = 1 \tag{19}$$

where $i = 1, 2, ..., n$ and $k = 1, 2, ..., x$, then, the score of the matrix represents the principal component space will calculated using

$$P_{k,i} = \sum_{j=1}^{x} \alpha_{i.j}\widetilde{\kappa}(y_j, y_k) \tag{20}$$

In the above equation P is the score matrix that helps calculate Hotelling's T^2-statistic and Q-statistic. The next step is the monitoring process to check the normal operation.

Phase 2: Monitoring

The data may be in abnormal conditions at this stage of the model, the normalized test data in a dimensional space indicates $y_{td} \in \Re^n$, that the kernel vector is represented by $k_{td} \in \Re^{l\times m}$. The training data is formulated as follows.

$$[k_{td}]_j = \kappa_{td}(y_j, y_{td}) \tag{21}$$

where $y_j \in \Re^n, j = [1, 2, ..., m], td = [1, 2, ..., k]$ and k is the count of test data values. The centre test kernel vector is calculated using,

$$\widetilde{k}_{td} = k_{td} - 1_{td}\kappa - k_{td}1_x + 1_{td}\kappa 1_x \tag{22}$$

In the above equation $1_{td} = \frac{1}{x[1,1,...,1]} \in \Re^{l\times x}$ 1, which is for retaining the score matrix for the test data values.

$$P_{td,i} = \sum_{j=1}^{x} \alpha_{i,j}\widetilde{k}_{td}(y_j, y_{td}) \tag{23}$$

The mapped data is represented by the score matrix with the count of some principal components (PC). It must be decided to prove that there is no lack of data on each data. Because the goal of KerPCA is to deduce the dimensionality of data and extract features. The cumulative explained variance (CEV) selects the correct variety of main additives. Thus, CEV describes the variant set of the selected main characteristics with the total variant, which refers to all main components.

$$CEV(\%) = \frac{\sum_{j=1}^{i}\lambda_j}{\sum_{j=1}^{m}\lambda_j} \times 100 \tag{24}$$

The PC is increased until the CEV gives the right function. The number of main components is chosen so the CEV exceeds 90 %. Tracking T^2 – and Q-statistics is a method to notice abnormalities when the T^2 or Q values exceed the control limits. The variation in the data can be calculated as follows,

$$T_i^2 = [p_{i,1}, ..., p_{k,i}]\Omega^{-1}[p_{i,1}, ..., p_{k,i}]^d \tag{25}$$

p is a score matrix, Ω is a diagonal matrix with Eigenvalues to the selected feature of the principal components. i is extracted principal components where $i = 1, 2, ..., n$ and n is a number of samples. The features are moved to the feature selection step after they have been picked using the KerPCA approach.

3.3.2. Feature selection using CHbO

The optimal intrusion detection function is selected using the Chaotic Honey Badger (CHbO) optimization algorithm. Some existing optimization approaches are used for feature selection, such as ant optimization (Kunhare et al., 2020), Pigeon optimization (Alazzam et al., 2020), and Whale optimization (Vijayanand & Devaraj, 2020). This approach possesses flaws like low convergence speed, easy localization and inducing sensitivity. CHBO aims to balance exploration (searching for new solutions) and exploitation (refining promising solutions). This balance can help the algorithm to navigate the solution space efficiently and avoid premature convergence to suboptimal solutions, so this algorithm is used to select optimal features. In addition, this approach is definitely used to optimize the hyperparameter and minimize the loss, which shows the independence of this approach in various processes (El-Sehiemy et al., 2022). The Honey Badger Algorithm (HBA) (Khan et al., 2022) is an algorithm that copycats the activities of the honey badger. There are two phases mentioned using the following arithmetic model. This algorithm takes a total of 5 steps to find the optimal feature. The steps are explained below, starting with population candidate solutions C.

$$C = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1d} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nd} \end{bmatrix} \tag{26}$$

Step 1: First, the number of balanced entries of records and their position is calculated using

$$a_j = l_j + s_1 \times (u_j - l_j) \tag{27}$$

where s_1 is chosen randomly as 0 or 1, where a_j mention the j^{th} feature point for estimating the solution of the candidate in whole dataset D, while u_j *and* l_j are the upper and lower bounds, respectively, in the given pre-processed dataset.

Step 2: Finding the intensity (IN) is the second step of optimization to analyze the absorption strength of the interruption. This algorithm is also used in this step to study the attack patterns, where the algorithm is very adaptable. Also, calculate the distance between good and bad features using the Inverse-square law.

$$IN = s_2 \times \frac{ST}{4\pi D_j^2} \tag{28}$$

$$ST = (a_j - a_{j+1})^2 \tag{29}$$

$$D_j = a_{tar} - a_j \tag{30}$$

where ST is a strength and distance is given by D_j and s_2 is a second random number.

Step 3: Density factor updating where indicated as α that controls the variation in temporal randomization to assure an easy changeover from exploration to exploitation. The choice of parameter increases the possibility of finding the optimal feature. Update the most optimal feature with the decreasing factor and use iterations that reduce randomization.

$$\alpha = CON \times ep\left(\frac{-i}{i_{\max}}\right) \tag{31}$$

here CON is a constant $\geqslant 1 (default = 2)$ and i is a number of iterations in maximum.

Step 4: Local optimum: The below steps are used to escape from the region of local optima with the help of a flag Fg. The flag helps change the direction in which optimal function occurs most often.

Step 5: Position of the feature: The process will update position sequentially as a_{new} this will be categorized into two steps: digging and the honey phase.

Digging Phase: The process of selection is done in the form of a Cardioid shape, so the movement is calculated using the following equation.

$$a_{new} = a_{tar} + Fg \times \beta \times IN \times a_{tar} \times s_3 \times \alpha \times D_j \times |\cos(2\pi s_4) \times [1 - \cos(2\pi s_5)]| \tag{32}$$

where a_{new} is the position of the feature, which is an optimized feature, $\beta \geqslant 1 (default = 6)$ is a capability to find the optimal feature. s_3, s_4 and s_5 are different random numbers.

$$Fg = \begin{cases} 1 & if\ s_6 \leqslant 0.5 \\ -1 & else \end{cases} \tag{33}$$

Features depend on the intensity IN of the target a_{tar} , the disturbance b may occur is easily identified, and features are selectively replaced.

Honey phase: In this phase, the honey badger chases the honey guide bird to get to the hive. The equation is,

$$a_{new} = a_{tar} + Fg \times s_7 \times \alpha \times D_j \tag{34}$$

Where s_7 a random real number and the above equation are used to refer to the new position of the honey badger in the honey phase, the CHbO algorithm is given as a pseudocode in Table 2.

The Chaotic Honey Badger optimization improves feature selection by increasing random feature selection. The population initialization plays a crucial role in the selection process. It enables an efficient approach to selecting the random number of the iteration to reduce the complexity of selecting the random number. When the number is selected by the following chaotic equation, the confusion caused by choosing the number is solved, and the convergence speed is also increased. Different initializations can lead to different convergence speeds. A well-chosen initialization may allow the algorithm to converge faster by starting closer to the optimal solution, while a poor initialization may result in slower convergence. As the convergence speed increases, the effectiveness of feature selection also increases. Thus, the random number s_1 is taken as $tent_n$, such that the Eq. (27) is replaced by

Table 2
Pseudocode of CHbO algorithm.

Start
Load the parameters $i_{\max}, m, \beta, CON$.
Start the population with a random position
Take input from the Chaotic HBO format from equation (35)
Estimate the fitness of each feature position a_j using objective function and set $O_j, j \in [1, 2, ..., m]$
Save the optimal feature position a_{tar} and assume fitness to O_{tar}
while $i \leqslant i_{\max}$ **do**
  Update the decreasing factor α using equation (31)
  for $k = 0$ to m **do**
    Estimate the intensity IN_j using equation (28), (29), (30)
    if $s < 0.5$ **then**
      update the position a_{new} using equation (32)
    else
      update the position a_{new} using equation (34)
    end if
    Calculate new features and assign to O_{new}
    if $O_{new} \leqslant O_j$ **then**
      Set $a_j = a_{new}$ and $O_j = O_{new}$
    end if
    if $O_{new} \leqslant O_{tar}$ **then**
      Set $a_{tar} = a_{new}$ and $O_{tar} = O_{new}$
    end if
  end for
end while optimal features are obtained
Return a_{tar}
Stop

the following equation,

$$a_j = l_j + tent_n \times (u_j - l_j) \tag{35}$$

$$tent_n = \begin{cases} \dfrac{tent}{z} \\ \dfrac{(1-tent)}{1-z} \end{cases} tent \in [0, z] \tag{36}$$

Here $tent_n$ is a random number and $z \in (0,1)$, which is a real number limit that can be taken as a random.

Feature selection optimization is done with chaotic Honey Badger optimization due to the versatility of getting a selective feature to detect attacks. The main strength of the approach is its parameter independence. Only a few parameters are required for processing and also ensure convergence speed. The main problem with this approach is the ease of falling into the local optimum. To fix this error, chaotic functions were added to the approach. Due to its flexibility compared to other approaches, this approach works easily even with large data sets. To maintain the large dataset, this algorithm can be combined with other approaches to make selection possible and reduce the computational cost. HBO can be hybridized with other optimization techniques or algorithms, combining their strengths. Hybrid approaches can take advantage of multiple algorithms for improved performance. Furthermore, feature selection is based on the number of features extracted from the extraction phase, which reduces the number of features to be classified. After the feature extraction and selection process, the type of attack is classified using the proposed Dugat-LSTM model.

3.4. Dugat-LSTM model

Cyber-attacks are harmful to the privacy of user data, which is frequently done on networks. Therefore, the identification of this intrusion is difficult. For such purposes, Gated attention dual long short term memory (Dugat-LSTM) is implemented as a suggested novel method to detect the intrusion in a feasible manner. Furthermore, interpretability is a crucial aspect of the deep learning model. Here are some efforts researchers often make to interpret and explain the decisions of deep learning models. Dual-gate LSTMs provide a form of interpretability by highlighting specific parts of the input sequence that contribute more to model recognition. Attention weights can indicate the relevance of different elements in the sequence. Analyzing the importance of input features can provide insights into the aspects of the data that strongly influence model decisions. Techniques such as feature importance scores or color gradients can be used for this purpose. Examining the activations of different layers in the neural network can provide insight into how the model transforms and processes information at each stage. This can help understand hierarchical representations. In this proposed model, Softmax is used as an activation function to detect network intrusions. Here, Long Term short-term memory (LSTM) (Ayetiran, 2022) and Gated recurrent unit (GRU) (Gao et al., 2021) are neural network structural designs utilized in DL. Neural networks encounter those issues of disappearing gradients. To address the vanishing gradient problem, LSTM is combined with the GRU model. GRU has multiple factors different from the LSTM model and processes data in different ways. To normalize the flow of data across the network, the LSTM employs input, output, and forget gates, whereas the GRU employs a reset and update gate to regulate the flow of GRU networks. The systematic diagram is shown in Fig. 3.

In the given hybrid model, two LSTM layers are combined with GRU units with an attention mechanism. The gates of both networks perform different functions, with an activation function at the end of the shift. The input gate determines the amount of data used to be fed, the forgetting gate determines the removal of data from the memory cell, and the output gate determines how much data is output and defines the rest of the network. Softmax is an activation function that gives results based on the probability of various outcomes. After the LSTM layers, the reduction in complexity is achieved with a 10 % dropout. The GRU layers combine the entity and hidden layers to create an update gate that is used to replace the functions that provide intrusion detection benefits.

Consider the selected feature as input to the layers of the hybrid model, which will represent as $I = [i_1, i_2, i_3, \ldots i_n] \in R$ here R is a real number. The detection of different attacks is retained as output, which is indicated as $O = [o_1, o_2, o_3, \ldots, o_m] \in R$ where n and m is the size of input and output data. The function for mapping the layers and gates is given by $O = MG(R)$, and using the function, the hidden gate and the reset gate are calculated.

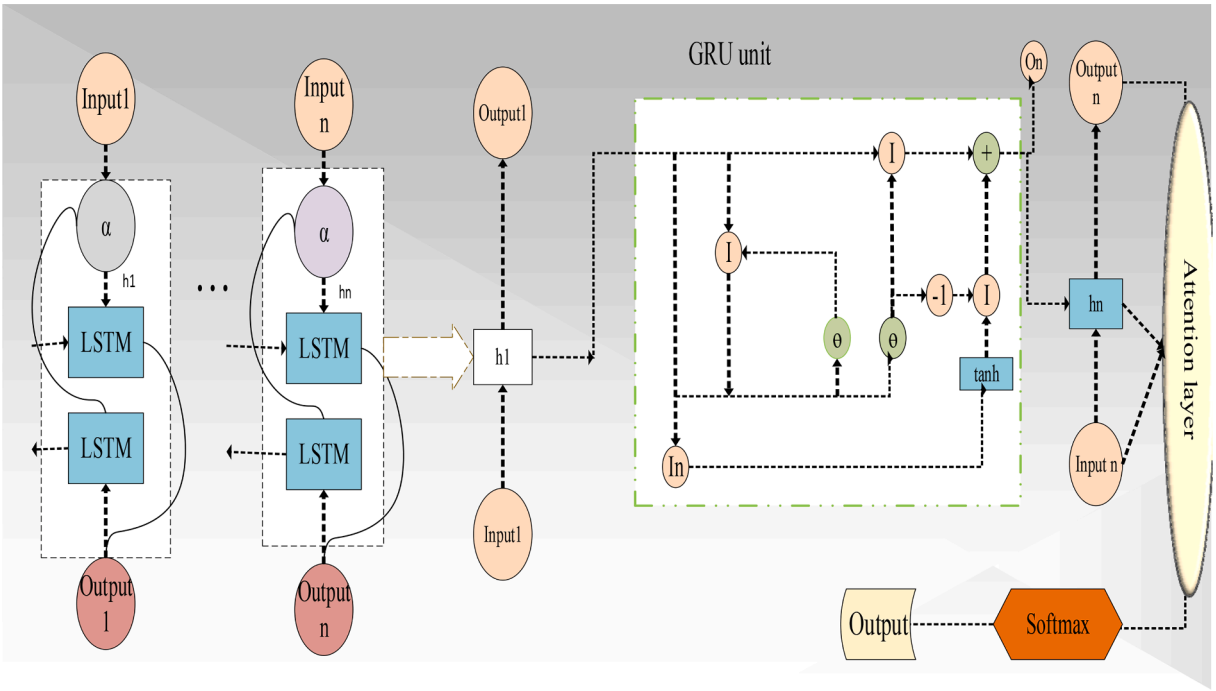


Fig. 3. Architecture of Dugat-LSTM.

$$rs = \sigma(wm_i[H_{n-1}, i_n]) \tag{37}$$

In the above equation, *rs* is a reset gate, *wm* is a weight matrix and H is a hidden gate. The softmax function is used to maintain a limit on given values. The currently stored feature neglects the other features when estimating the fault types. The fault is identified using the following equation,

$$CF_n = \tan H(wm_H[rs \times H_{n-1}, i_n]) \tag{38}$$

The functions are updated in the update gate using the following equation,

$$U = \sigma(wm_H[H_{n-1}, i_1]) \tag{39}$$

The final detection of the type of cyber-attack is done in the hidden state and normalized by the activation function to provide an output.

$$H_n = (1 - U_n)^* H_{n-1} + U_n \times CF_n \tag{40}$$

The above classification process is performed with attention mechanisms that provide a highly optimal solution in detecting the type of attack on the network. Here, the degree of attention is given below,

$$\begin{aligned} &G = S^n \tan H(TH_n + V o^{'}_{o-1} + bias) \\ &o \in N,\ 1 \leqslant o \leqslant N,\ n \in N, 1 \leqslant n \leqslant m \end{aligned} \tag{41}$$

where *S*, *T*& *V* are the real numbers and represent the matrix of learning weights. Normalization is done using the following equation

$$D_{io} = \frac{\exp(sof_{io})}{\sum_{o-1}^{n} \exp(sof_{io})} \tag{42}$$

D_{io} is a network unit of the GRU that keeps the type of attack as output. The novel method delivers accurate results due to its attention mechanism and deep classification using GRU. The LSTM layer is helpful in overcoming the vanishing gradient problem that can occur when processing large amounts of data simultaneously. This model improves intrusion detection performance by improving processing speed, efficient memory usage, and adaptability. Additionally, the dual gating mechanisms in the model enable improved control over the flow of information within the memory cell. This allows the model to selectively remember or forget information based on its relevance to the task, resulting in improved long-term memory retention. The dual gating mechanism can make the model less sensitive to small changes in the input data, resulting in improved robustness. This advancement enables large amounts of data to be processed with low latency. The model is appraised by comparing the performance metrics, precision, F-measure, specificity, recall and accuracy, etc., with other models.

4. Results and discussion

The proposed model uses a combined network, where an LSTM is often used in classification techniques that process disappearance gradients. The models are known for their capacity to practice huge amounts of data. Here, Attention GRU is combined with the LSTM network to provide a detailed classification of storage capacity and identify the type of attack. In the end, to evaluate the performance of the IDS, a comparison is made with other existing systems in terms of accuracy, precision, F1 score, recall and specificity, etc.

4.1. Dataset description

The input comes from two sets of data, namely NSL KDD and TON-IOT, which are used to classify different cyber-attacks. These two datasets are used to evaluate the proposed model due to their versatility. The TON-IOT data set is platform independent. The datasets were collected from a realistic and large-scale network developed at UNSW Canberra Cyber's IoT Lab. Various hacking techniques such as DoS, DDoS and ransomware are detected. Likewise, researchers use NSL-KDD to study the generalization capabilities of intrusion detection models. For practical applicability, it is crucial to understand how well models trained on this dataset can adapt to different network environments.

4.1.1. Details of NSL KDD dataset

The dataset comprises 41 characteristics categorized into basic, content-based, and time-based attributes. These features are further divided into three types: Nominal (3 features), Binary (6 features), and Numeric (32 features). The training set contains 22 attacks, and the test set contains 16 attacks. Attacks are divided into four categories: DoS, probe attacks (PA), remote-to-local (R2L) attacks, and user-to-root (U2R) attacks.

4.1.2. Details of TON-IOT dataset

The second dataset used for detection is the TON-IOT dataset, which is a next-generation dataset for evaluating the effectiveness of cyber security measures. These datasets consist of various source data obtained from telemetry datasets from IoT and IIoT sensors. The TON-IoT dataset includes both normal and compromised data, such as DDoS, ransomware, DoS, password cracking and data injection attacks. These datasets were collected using a realistic representation of a medium-sized network at the University of New South Wales (UNSW) IoT and Cyber Range Labs in Canberra. Using both datasets below, graphs are created with metrics to ensure the efficiency of the proposed model.

4.2. Performance analysis

The model's efficiency and robustness are tested with different attributes noted as accuracy, precision, recall, specificity, F-measure, and attack detection rate. These attributes measure the capabilities of the Dugat-LSTM model. A detailed description of metrics can be found below,

4.2.1. Accuracy

The accuracy indicates the correctness and quality of the model as a percentage, which is the ratio of certain correctness to the total number of cases. Accuracy is calculated using the following equation,

$$Accuracy = \frac{T.\ negative + T.positive}{T.\ negative + F.\ positive + F.\ positive + T.positive} \tag{43}$$

4.2.2. Precision

Precision defines the classification strength of the given model. The equation of precision is calculated using the below equation,

$$precision = \frac{T.positive}{T.positive\ + F.positive} \tag{44}$$

4.2.3. Specificity

The efficiency of the model is estimated based on specificity. The equation of specificity is given below,

$$Specificity = \frac{T.positive}{T.negative + F.positive} \tag{45}$$

4.2.4. F-measure

The F-measure gives the amount of correctness in classification. The equation of the F-measure is given below

$$F - measure = 2 \times \frac{precision \times \mathrm{Re}call}{precision + \mathrm{Re}call} \tag{46}$$

4.2.5. Recall

The recall measure gives the analysis of a measure that indicates proficient classification. The equation of the recall is given below,

$$\mathrm{Re}call = \frac{T.positive}{T.positive\ + F.negative} \tag{47}$$

4.2.6. Attack detection rate

The detection rate concisely mentions the rate and exactness of the detection process. The equation of recall metrics will be relatively the same for the attack detection rate.

4.3. Evaluation using the NSL KDD dataset

Many evaluations and tests are carried out using the attributes mentioned above. The results of the evaluation based on the NSL KDD data sets are listed in Table 3 (Bhuvaneshwari et al., 2022).

The table above shows the values obtained in the performance metrics and clarifies the uniqueness of the proposed model, which provides greater accuracy in classifying attacks. Among other models, Dugat-LSTM performs better in the detection process, with a high attack detection rate of 96.89 %. The performance plot is easily observed in graphical format.

In Fig. 4, the proposed model gives the related details with other intrusion detection models, namely Dragonfly Optimizer (DFO), Feasible Selection of Mutual Information Features (FMIFS), Forming Limit Curves at Fracture (FLCFS), SMOTE Ensemble Neural Network (SENN), and Improved Dragonfly Optimizer (IG-DFO). The CHbO feature selection strategy will excite the most relevant features. This strategy enables the model to produce reliable findings. The graphs present the result with an accuracy of 99.65 %, which is relatively higher than other neural network models.

In Figs. 5 and 6, the given model is matched to other intrusion detection models, namely DFO, FMIFS, FLCFS, SENN and IG-DFO. The graphs represent the result of 97.01 % sensitivity and 97.23 % specificity, which is relatively higher than the other network model. Fig. 6 implies the specificity performance comparison of proposed and existing models. The CHBo Approach is primarily concerned with changing the position of the honey badger, which provides an effective function for selecting characteristics. The majority of the features that will be set contribute to a better knowledge of normal and aberrant network behavior, boosting the sensitivity of IDS.

Fig. 7 shows the attack detection rate of the proposed structure compared to other models. The detection rate is achieved at 96.98 %, which is higher compared to the newer detection models. The graph was created using the NSL-KDD datasets, and the score may differ when using supplementary datasets. The hybrid model and attention function produced accurate results.

Fig. 8 shows the accuracy rate curve of the proposed model, which demonstrates that the given prototype is very accurate in detecting network intrusions. The model is also evaluated against another data set to show that the model is extremely effective at detection. The other models have lower accuracy due to their lower ability to detect attacks. Here, the false positive rate achieved by the proposed model using the TON-IoT dataset is shown in Fig. 9.

The graph shows that the proposed model has a lower false positive rate than existing models. The existing models considered for comparison are Random Forest, Decision Tree, Support Vector Machine, Nave Bayes, BN, Ada Boost and XG Boost. The value achieved with the proposed model is 0.003, while the other approaches have a higher false positive rate.

Table 3
Comparison of the proposed model with another model.

Metrics	DFO	FMIFS	FLCFS	SENN	IG-DFO	Dugat-LSTM
Accuracy	95.97	91.49	94.81	94.02	98.72	**99.65**
Sensitivity	88.4	83.62	87.37	88.05	94.88	**97.01**
Specificity	84.64	87.03	89.76	85.32	95.22	**97.23**
Attack detection rate	90.78	84.98	87.71	88.05	94.54	**96.98**

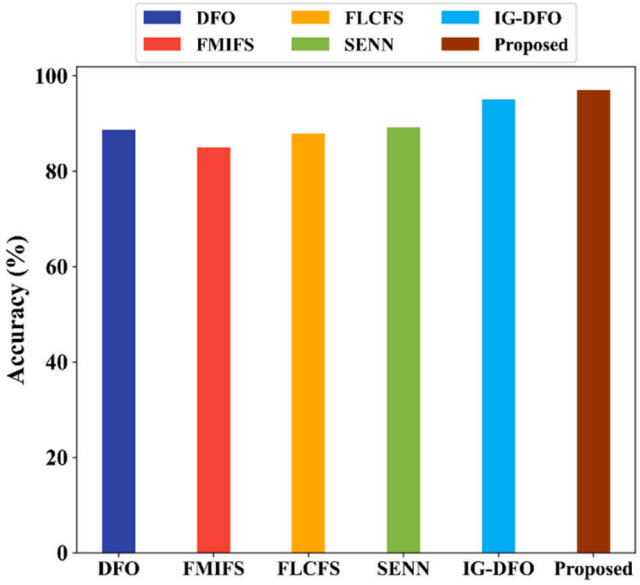


Fig. 4. Accuracy comparison with other models.

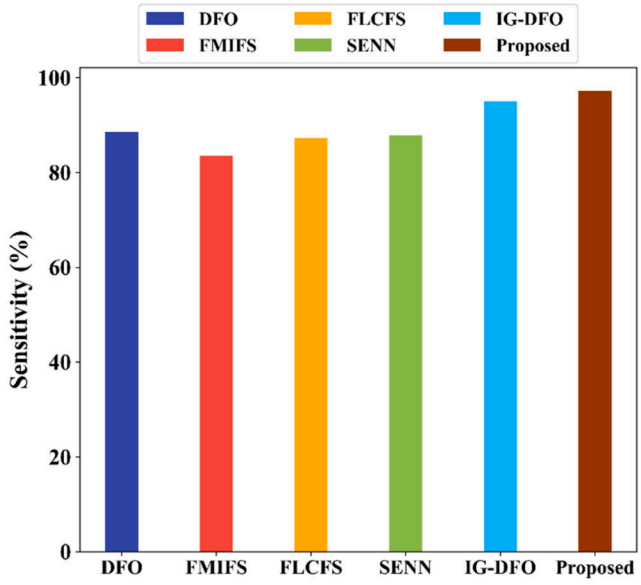


Fig. 5. Sensitivity comparison with existing models.

4.4. Evaluation using the TON-IOT dataset

The results of the evaluation using the TON-IOT data sets are listed in Table 4 (Pampapathi et al., 2022).

Table 4 shows the values that demonstrate the uniqueness of the proposed model, offering greater accuracy in the classification of attacks. Among other models, Dugat-LSTM scores better in the detection. The plot of performance is easily observed in graphical representation.

In Fig. 10, the proposed design is compared with other intrusion detection designs, namely Deep Learning Neural Networks (DLNN), Forward Deep Learning Neural Networks (FDLNN), and Artificial Neural Networks (ANN). The graphs present the result with an accuracy of 98.76 %, which is relatively higher than another neural network model.

In Fig. 11, a comparison of the recall metric shows that the proposed model provides the result of 97.87 %. The recall score ensures that the proposed model provides a well-defined model for finding the attacks. The recall represents the ability to find selected characteristics or data as

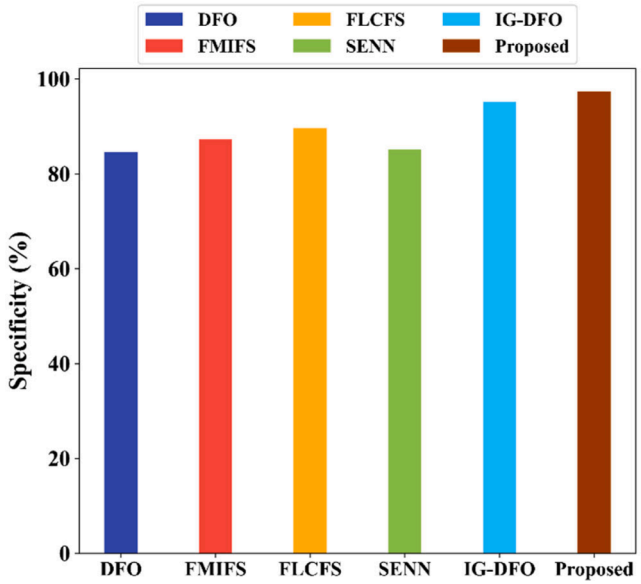


Fig. 6. Specificity comparison with other models.

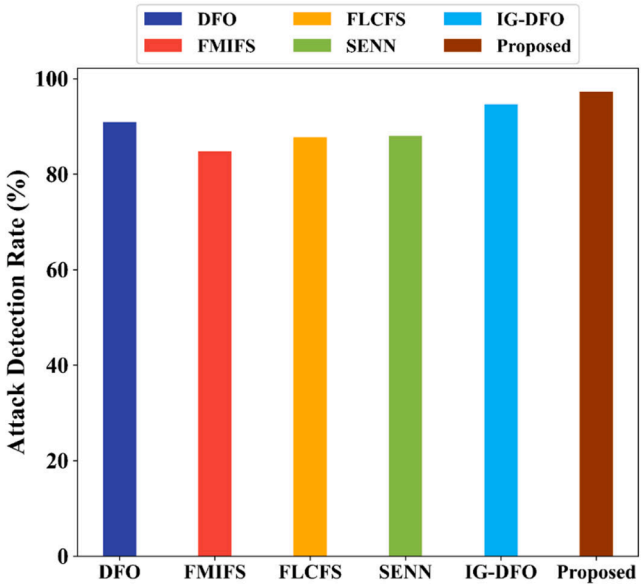


Fig. 7. ADR comparison of the proposed model with other models.

output as needed.

In Fig. 12, a comparison of the F-measure is shown that the proposed model provides the result of 97.56 %. The score satisfies that the proposed model provides a well-defined model for finding the attacks. The disturbance that may occur in the detection will automatically redirect due to the defined neural network model.

In Fig. 13, comparing the accuracy, it is shown that the proposed model delivers a better result of 98.76 %. The existing models scored less accuracy when compared to the proposed intrusion detection model. Fig. 14 depicts an AUC-ROC graph for the proposed model for further analysis.

The recall analysis shows the quality of attack detection in both datasets with an AUC of 0.993. The AUC-ROC measurement is calculated using the true positive rate and the false positive rate. In addition, the constant course of the drawn line demonstrates the completeness of the intrusion detection. The approaches require more time for training data, and the better phase for the model train is done in research. The analysis

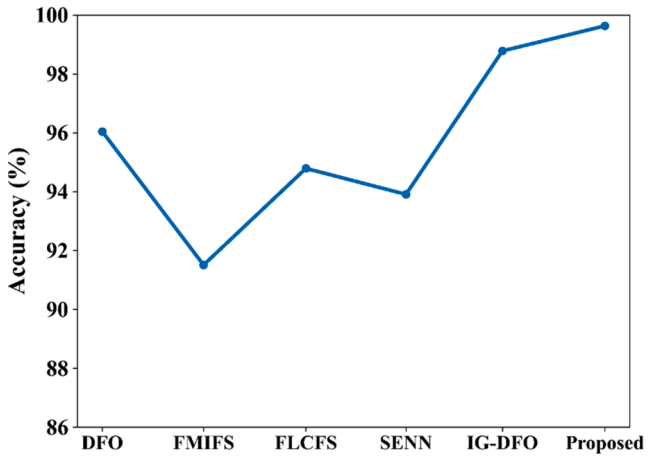


Fig. 8. Accuracy rate attained by the proposed model in NSK KDD.

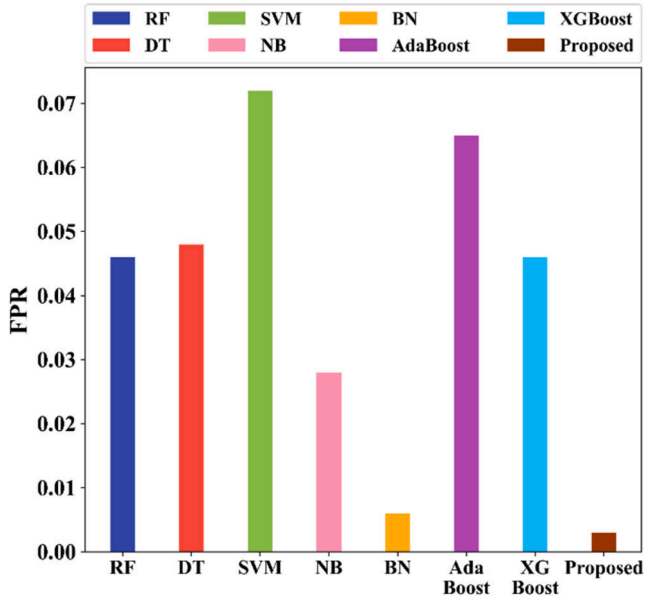


Fig. 9. FPR analysis of proposed model vs. existing approaches.

Table 4
Comparison of the proposed model with another model.

Metrics	FDLNN	DLNN	ANN	Dugat-LSTM
Accuracy	94.75	90.03	94.81	**98.76**
Recall	95.23	90.03	84.67	**97.87**
F-measure	94.98	90.0	84.9	**97.23**
Precision	94.75	89.99	85.14	**96.98**

of training time with other models is shown in Table 5.

The computational complexity of deep learning models is a crucial consideration, and providing information about the computational resources required for training and deployment is valuable in assessing practical feasibility. Here are some key aspects related to computational complexity that are often mentioned when describing a proposed deep learning model. Inference speed is the time it takes the trained model to process a single input and produce a prediction. Faster inference is desirable for real-time applications. This metric is often influenced by model architecture, optimization techniques, and hardware specifications. In addition, the inference speed when using the proposed model is high compared to the existing model. The model also requires less

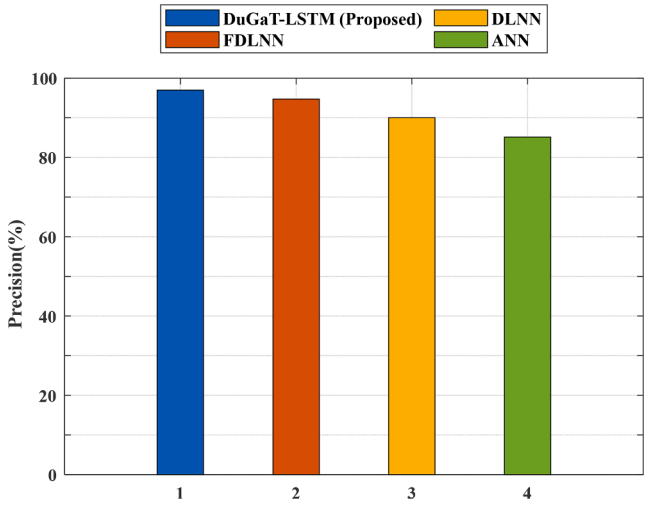


Fig. 10. Precision comparison with other models.

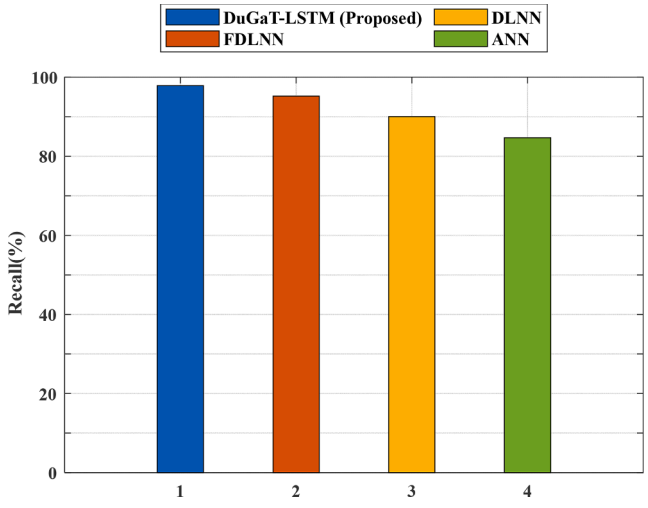


Fig. 11. Recall comparison with other models.

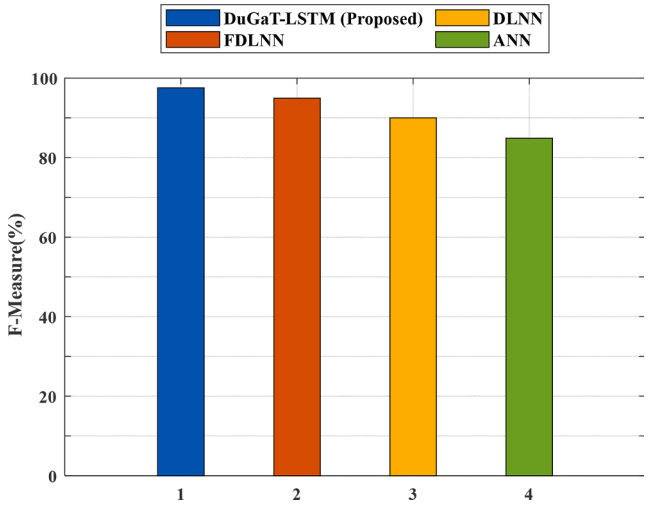


Fig. 12. F-Measure comparison with other models.

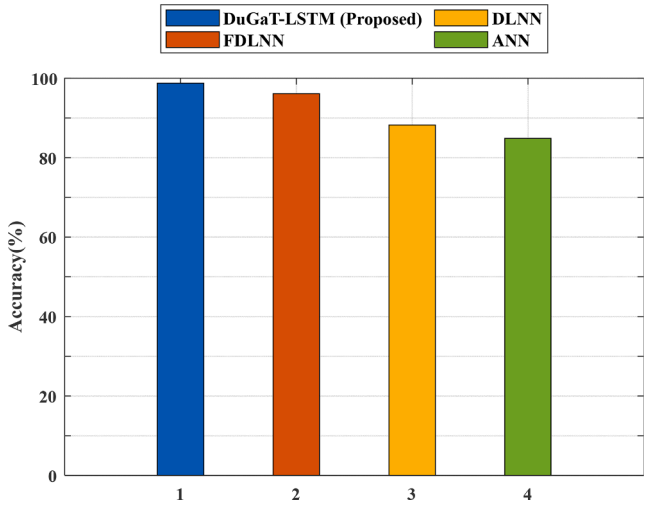


Fig. 13. Accuracy comparison with other models.

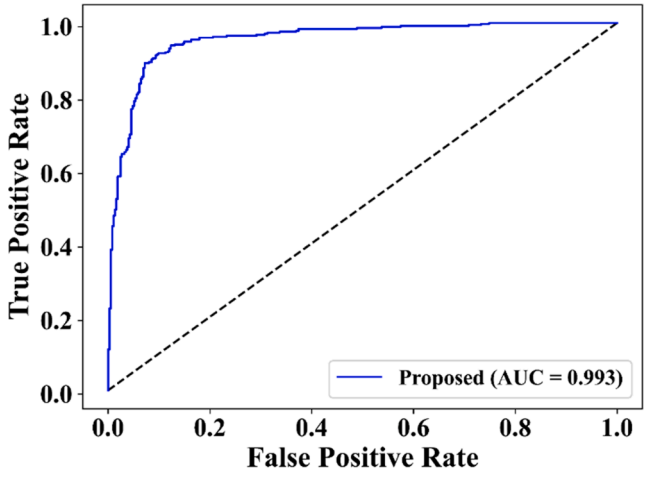


Fig. 14. AUC-ROC graph analysis of the proposed model.

Table 5
Analysis of training time by the proposed model.

Models	Training time (s)
AlexNet	12900.93
LeNet-5	713.02
CNN	2929.62
BiLSTM	6949.43
CNN-BiLSTM	4367.99
Proposed	700.12

memory for the recognition process. Table 5 shows the training time analysis of the proposed model with AlexNet, LeNet-5, CNN, BiLSTM, and CNN-BiLSTM (Jiang et al., 2020). Compared to other models, the proposed model requires less time to train the data. The cross-validation of the proposed model is shown in Table 6.

In the cross-validation analysis with the XGBoost-DNN model (Devan & Khare, 2020), the proposed model shows its efficiency by achieving higher accuracy in each convolution. Due to the independence, the model is flexible for both datasets as well as other datasets, such as the UNSW-NB15 dataset and the CICIDS2017 dataset, which have an accuracy of more than 90 %. This demonstrates the generalizability and robustness of the proposed model across different network attack scenarios. The proposed model is less computationally intensive as compared to the state-of-the-art models due to the adaptation of the

Table 6
10-fold cross validation of the proposed model.

Models	1 Fold	2-fold	3-fold	4-fold	5-fold	6-fold	7-fold	8-fold	9-fold	10-fold
XGBoost-DNN	97.39	97.19	96.99	97.19	96.79	97.6	96.19	96.19	97.39	96.39
Proposed	**98.67**	**98.65**	**98.59**	**98.51**	**97.89**	**97.76**	**97.55**	**97.43**	**96.97**	**96.77**

Table 7
Analysis of accuracy with other intrusion system.

Models	Accuracy (%)
CNN + LSTM	93.21
RNN	97.00
DNN-RNN	94.00
Proposed	**99.65**

approaches. However, the proposed model is compared with other intrusion systems, as shown in Table 7.

Balancing the dataset with ExSyn makes detection faster and increases the scalability of the model. This scalability metric of the proposed model is compared with other models in Table 8.

The evaluation assumes that the proposed model provides a well-defined attack detection model with high scalability. The model also requires less memory for attack detection, as shown in Table 9.

Table 9 demonstrates that the model has less memory than existing models. Furthermore, the proposed neural network model automatically reroutes any interference that may occur during detection. As a result, the inference speed is quick, and the model performs differently on different data sets. This type of action has no effect on the model's performance. This validation result is presented in section 4.5 to ensure a clear evaluation observance.

4.5. Discussion

Various assessments of the proposed Dugat-LSTM have shown that the model outperforms anomaly in intrusion detection. As part of a deep learning-based network intrusion detection system, the proposed model applies a chaotic optimization strategy. This method aids in obtaining the most relevant results in intrusion detection. In the method's pre-processing, M-squared normalization and data purification are used. This procedure is utilized to reduce noise from the network in order to offer proper results through its normalizing approach. Following pre-processing, the ExSyn method is employed to balance the unbalanced datasets. The dataset is balanced to resolve any biasing issues that may develop during feature extraction. After the dataset has been balanced, KerPCA is used to extract its properties, allowing for the separability of nonlinear data through the use of kernels. The Chaotic Honey Badger optimization approach then selects the best characteristics. After extracting all essential features, the attacks are classified using the (Dugat-LSTM). The suggested model is validated using the TON-IOT and NSL-KDD datasets by these improved models.

The prototype is evaluated using the following metrics: F1 score, accuracy, precision, and recall. The suggested model's accuracy values

Table 8
Scalability analysis of the proposed model.

Techniques	Scalability (MB)
RNN	135
GNN	315
CNN-LSTM	153
XGBoost-LSTM	345
DCNNBiLSTM	134
CNN-CapSA	256
ELK stack SIEM	344
CNN + LSTM	256
Proposed	**128**

Table 9
Memory Usage Analysis.

Techniques	Memory usage (MB)
RNN	137
GNN	360
CNN-LSTM	189
XGBoost-LSTM	900
DCNNBiLSTM	788
CNN-CapSA	300
ELK stack SIEM	344
CNN + LSTM	256
Proposed	**124**

in the TON-IOT dataset and the NSL-KDD dataset are 98.76 % and 99.65 %, respectively. The CHbO technique for feature selection will stimulate the most relevant features, allowing the model to produce correct results. The CHBo Approach is primarily concerned with changing the position of the honey badger, which provides an effective function for picking characteristics. The majority of the features that will be set will contribute to a better knowledge of normal and abnormal network behavior, boosting the sensitivity and specificity of IDS. Due to the parameter independence, the model's memory utilization is as low as 124 MB. Because of the reduced memory utilization, scalability increases as well, as the suggested has 128 MB. To confirm the model's trustworthiness, the 10 fold cross validation was also assessed, and the model obtained 96.77 % accuracy in 10 fold validation by improving extraction relevant characteristics. The AUC-ROC evaluation shows that using dual and gated attention in each layer yields accurate findings. The suggested model's recall score is high because it achieves the most accurate positive incidents by studying every feature with dual attention in the classification model. Similarly, the model achieves a high F1-score in both datasets by using the same dual attention in all layers of the classification model. The model provides excellent validation results due to the multiple advancements in techniques.

5. Conclusion

This paper presents an innovative model that is characterized by the use of a robust methodology in detecting network intrusions. Existing works have failed to improve the robustness and flexibility of network intrusion detection. The pre-processing phase included data cleaning and M-squared normalization, effectively eliminating unnecessary input data. This approach advances improved data visualization and improves optimization convergence. The ExSyn approach was used to balance the training and testing datasets to improve the effectiveness of subsequent feature extraction. The feature extraction enabled by KerPCA was performed seamlessly on the balanced datasets, demonstrating the effectiveness of the method in identifying relevant features. This shows the benefits of effectiveness in reducing the dimensionality of data with nonlinear structures. In particular, the CHbO approach played a crucial role in carefully selecting the best features and optimizing the overall functionality. This simultaneously improves the algorithm's adaptability to different problem domains and its ability to handle different types of constraints. The proposed Dugat-LSTM model showcased remarkable performance in classifying cyber-attacks, achieving an impressive accuracy of 98.76 % in the TON-IOT dataset and 99.65 % in the NSL-KDD dataset. Beyond accuracy, the model exhibited outstanding metrics, including a recall of 97.87 %, precision of 96.98 %, specificity of 97.23 %, sensitivity of 97.01 %, F-measure of 97.56 %, and

an attack detection rate of 96.98 % in both datasets. These results highlight the superiority of the proposed prototype over existing approaches and demonstrate its effectiveness in achieving high accuracy and reliability in the challenging task of network intrusion detection.

Because the model outperforms the TON-IOT and NSL-KDD datasets, it represents an important contribution to the field. Furthermore, the model is adaptable to both datasets as well as other datasets with higher precision, such as the UNSW-NB15 dataset and the CICIDS2017 dataset. This proves the proposed model's generalizability and robustness across various network attack situations. In order to increase the attack detection rate, future work will concentrate on refining the feature selection method. The technique may impede finding newly discovered attacks in many processes, which may be expanded in future research. Furthermore, by examining models that enable real-time analysis in dynamic environments with greater volumes of network data, the model's scalability is increased. A new model's potential for enhanced performance in detecting new attacks with practical applications is examined. The model will be examined with more datasets to demonstrate considerably greater generalizability.

6. Compliance with ethical standards

Funding: No funding is provided for the preparation of the manuscript.

Ethical approval: This article does not contain any studies with human participants or animals performed by authors.

Consent to participate: All the authors involved have agreed to participate in this submitted article.

Consent to publish: All the authors involved in this manuscript give full consent for publication of this submitted article.

CRediT authorship contribution statement

Ramkumar Devendiran: Conceptualization, Methodology, Software, Writing – original draft, Visualization. **Anil V Turukmane:** Formal analysis, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Abd Elaziz, M., Al-qaness, M. A., Dahou, A., Ibrahim, R. A., & Abd El-Latif, A. A. (2023). Intrusion detection approach for cloud and IoT environments using deep learning and Capuchin Search Algorithm. *Advances in Engineering Software, 176*, Article 103402.

Abdullahi, M., Baashar, Y., Alhussian, H., Alwadain, A., Aziz, N., Capretz, L. F., & Abdulkadir, S. J. (2022). Detecting cybersecurity attacks in Internet of things using artificial intelligence methods: A systematic literature review. *Electronics, 11*(2), 198.

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies, 32*(1), e4150.

Alazzam, H., Sharieh, A., & Sabri, K. E. (2020). A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer. *Expert systems with applications, 148*, Article 113249.

Aldarwbi, M. Y., Lashkari, A. H., & Ghorbani, A. A. (2022). The sound of intrusion: A novel network intrusion detection system. *Computers and Electrical Engineering, 104*, Article 108455.

Alem, S., Espes, D., Nana, L., Martin, E., & De Lamotte, F. (2023). A novel bi-anomaly-based intrusion detection system approach for industry 4.0. *Future Generation Computer Systems*.

Altunay, H. C., & Albayrak, Z. (2023). A hybrid CNN+ LSTMbased intrusion detection system for industrial IoT networks. *Engineering Science and Technology, an International Journal, 38*, Article 101322.

Ayetiran, E. F. (2022). Attention-based aspect sentiment classification using enhanced learning through CNN-BiLSTM networks. *Knowledge-Based Systems, 252*, Article 109409.

Bharti, R., Khamparia, A., Shabaz, M., Dhiman, G., Pande, S., & Singh, P. (2021). Prediction of heart disease using a combination of machine learning and deep learning. *Computational intelligence and neuroscience, 2021*.

Bhuvaneshwari, K. S., Venkatachalam, K., Hubálovský, S., Trojovský, P., & Prabu, P. (2022). Improved dragonfly optimizer for intrusion detection using deep clustering CNN-PSO classifier. *Computers, Materials & Continua, 70*(3).

Caville, E., Lo, W. W., Layeghy, S., & Portmann, M. (2022). Anomal-E: A self-supervised network intrusion detection system based on graph neural networks. *Knowledge-Based Systems, 258*, Article 110030.

Choi, H., Kim, M., Lee, G., & Kim, W. (2019). Unsupervised learning approach for network intrusion detection system using autoencoders. *The Journal of Supercomputing, 75*, 5597–5621.

Devan, P., & Khare, N. (2020). An efficient XGBoost–DNN-based classification model for network intrusion detection system. *Neural Computing and Applications, 32*, 12499–12514.

Díaz-Verdejo, J., Muñoz-Calle, J., Alonso, A. E., Alonso, R. E., & Madinabeitia, G. (2022). On the detection capabilities of signature-based intrusion detection systems in the context of web attacks. *Applied Sciences, 12*(2), 852.

Dushyant, K., Muskan, G., Annu, Gupta, A., & Pramanik, S. (2022). Utilizing machine learning and deep learning in cybesecurity: An innovative approach. *Cyber Security and Digital Forensics*, 271–293.

El-Sehiemy, R., Shaheen, A., Ginidi, A., & Elhosseini, M. (2022). A honey badger optimization for minimizing the pollutant environmental emissions-based economic dispatch model integrating combined heat and power units. *Energies, 15*(20), 7603.

Gao, Y., Wang, R., & Zhou, E. (2021). Stock prediction based on optimized LSTM and GRU models. *Scientific Programming, 2021*, 1–8.

Guo, Y. (2023). A review of Machine Learning-based zero-day attack detection: Challenges and future directions. *Computer Communications, 198*, 175–185.

Halbouni, A., Gunawan, T. S., Habaebi, M. H., Halbouni, M., Kartiwi, M., & Ahmad, R. (2022). CNN-LSTM: Hybrid deep neural network for network intrusion detection system. *IEEE Access, 10*, 99837–99849.

He, K., Kim, D. D., & Asghar, M. R. (2023). Adversarial machine learning for network intrusion detection systems: a comprehensive survey. *IEEE*.

Hnamte, V., & Hussain, J. (2023). DCNNBiLSTM: An efficient hybrid deep learning-based intrusion detection system. *Telematics and Informatics Reports, 10*, Article 100053.

Javed, Y., Khayat, M. A., Elghariani, A. A., & Ghafoor, A. (2023). PRISM: A hierarchical intrusion detection architecture for large-scale cyber networks. *IEEE Transactions on Dependable and Secure Computing*.

Jiang, K., Wang, W., Wang, A., & Wu, H. (2020). Network intrusion detection combined hybrid sampling with deep hierarchical network. *IEEE access, 8*, 32464–32476.

Kasongo, S. M. (2023). A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework. *Computer Communications, 199*, 113–125.

Kasongo, S. M., & Sun, Y. (2019). A deep learning method with filter based feature engineering for wireless intrusion detection system. *IEEE access, 7*, 38597–38607.

Kasongo, S. M., & Sun, Y. (2020). Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. *Journal of Big Data, 7*, 1–20.

Khan, M. H., Ulasyar, A., Khattak, A., Zad, H. S., Alsharef, M., Alahmadi, A. A., & Ullah, N. (2022). Optimal sizing and allocation of distributed generation in the radial power distribution system using honey badger algorithm. *Energies, 15*(16), 5891.

Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity, 2*(1), 1–22.

Koohang, A., Sargent, C. S., Nord, J. H., & Paliszkiewicz, J. (2022). Internet of Things (IoT): From awareness to continued use. *International Journal of Information Management, 62*, Article 102442.

Kunhare, N., Tiwari, R., & Dhar, J. (2020). Particle swarm optimization and feature selection for intrusion detection system. *Sādhanā, 45*, 1–14.

Laghari, A. A., Wu, K., Laghari, R. A., Ali, M., & Khan, A. A. (2021). A review and state of art of Internet of Things (IoT). *Archives of Computational Methods in Engineering*, 1–19.

Laghrissi, F., Douzi, S., Douzi, K., & Hssina, B. (2021). Intrusion detection systems using long short-term memory (LSTM). *Journal of Big Data, 8*(1), 65.

Lindemann, B., Maschler, B., Sahlab, N., & Weyrich, M. (2021). A survey on anomaly detection for technical systems using LSTM networks. *Computers in Industry, 131*, Article 103498.

Lindemann, B., Müller, T., Vietz, H., Jazdi, N., & Weyrich, M. (2021). A survey on long short-term memory networks for time series prediction. *Procedia CIRP, 99*, 650–655.

Liu, C. L., & Hsieh, P. Y. (2019). Model-based synthetic sampling for imbalanced data. *IEEE Transactions on Knowledge and Data Engineering, 32*(8), 1543–1556.

Meira, J., Andrade, R., Praça, I., Carneiro, J., Bolón-Canedo, V., Alonso-Betanzos, A., & Marreiros, G. (2020). Performance evaluation of unsupervised techniques in cyber-attack anomaly detection. *Journal of Ambient Intelligence and Humanized Computing, 11*, 4477–4489.

Muhammad, A. R., Sukarno, P., & Wardana, A. A. (2023). Integrated security information and event management (SIEM) with intrusion detection system (IDS) for live analysis based on machine learning. *Procedia Computer Science, 217*, 1406–1415.

Naseri, T. S., & Gharehchopogh, F. S. (2022). A feature selection based on the farmland fertility algorithm for improved intrusion detection systems. *Journal of Network and Systems Management, 30*(3), 40.

Pampapathi, B. M., Guptha, N., & Hema, M. S. (2022). Towards an effective deep learning-based intrusion detection system in the Internet of things. *Telematics and Informatics Reports, 7*, Article 100009.

Pande, S., & Khamparia, A. (2023). Explainable deep neural network based analysis on intrusion detection systems. *Computer Science, 24*(1).

Ramkumar, D., Annadurai, C., & Nelson, I. (2022). Iris-based continuous authentication in mobile ad hoc network. *Concurrency and Computation: Practice and Experience, 34*(8), Article e5542.

Ramkumar, D., Annadurai, C., & Nirmaladevi, K. (2019). Continuous authentication consoles in mobile ad hoc network (MANET). *Cluster Computing, 22*, 7777–7786.

Ravi, V., Chaganti, R., & Alazab, M. (2022). Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Computers and Electrical Engineering, 102*, Article 108156.

Saurabh, K., Sood, S., Kumar, P. A., Singh, U., Vyas, R., Vyas, O. P., & Khondoker, R. (2022). Lbdmids: LSTM based deep learning model for intrusion detection systems for IOT networks. In *2022 IEEE World AI IoT Congress (AIIoT)* (pp. 753–759).

Singh, G., & Kundu, S. (2022, December). Outlier and Trend Detection Using Approximate Median and Median Absolute Deviation. In *2022 5th International Conference on Computational Intelligence and Networks (CINE)* (pp. 01-06). IEEE.

Sworna, Z. T., Mousavi, Z., & Babar, M. A. (2023). NLP methods in host-based intrusion detection Systems: A systematic review and future directions. *Journal of Network and Computer Applications*, Article 103761.

Thakkar, A., & Lohiya, R. (2023). Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. *Information Fusion, 90*, 353–363.

Toupas, P., Chamou, D., Giannoutakis, K. M., Drosou, A., & Tzovaras, D. (2019, December). An intrusion detection system for multi-class classification based on deep neural networks. In *2019 18th IEEE International conference on machine learning and applications (ICMLA)* (pp. 1253-1258). IEEE.

Tummala, A. S., & Inapakurthi, R. K. (2021). A two-stage Kalman filter for cyber-attack detection in automatic generation control system. *Journal of Modern Power Systems and Clean Energy, 10*(1), 50–59.

Uddin, M. P., Mamun, M. A., Afjal, M. I., & Hossain, M. A. (2021). Information-theoretic feature selection with segmentation-based folded principal component analysis (PCA) for hyperspectral image classification. *International Journal of Remote Sensing, 42*(1), 286–321.

Verma, A., & Ranga, V. (2020). Machine learning based intrusion detection systems for IoT applications. *Wireless Personal Communications, 111*, 2287–2310.

Vijayanand, R., & Devaraj, D. (2020). A novel feature selection method using whale optimization algorithm and genetic operators for intrusion detection system in wireless mesh network. *IEEE Access, 8*, 56847–56854.

Xu, H., Sun, Z., Cao, Y., & Bilal, H. (2023). A data-driven approach for intrusion and anomaly detection using automated machine learning for the Internet of Things. *Soft Computing, 27*(19), 14469–14481.