Universal Library of Engineering Technology Research Article

ISSN: 3064-996X

Open Access | PP: 18-26

DOI: https://doi.org/10.70315/uloap.ulete.2022.003

Designing an Intelligent Cybersecurity Intrusion Identify Framework Using Advanced Machine Learning Models in Cloud Computing

Mukund Sai Vikram Tyagadurgam[1], Venkataswamy Naidu Gangineni[2], Sriram Pabbineedi[3], Mitra Penmetsa[4], Jayakeshav Reddy Bhumireddy[5], Rajiv Chalasani[6]

[1]University of Illinois at Springfield.
[2]University of Madras, Chennai.
[3]University of Central Missouri.
[4]University of Illinois at Springfield.
[5]University of Houston.
[6]Sacred Heart University.

Abstract

An effective method to attain improved security in identifying harmful activity in cloud computing during the past number of years has been A system for detecting intrusions (IDS). Efforts are being made on an intrusion detection system (IDS) to detect and classify network-level intrusions using machine learning (ML) techniques. Using a Bidirectional Long Short-Term Memory (Bi-LSTM) model, this paper proposes an intelligent cybersecurity intrusion detection system that can recognize complex attack patterns in network data. To apply a comprehensive data pre-processing pipeline over the CICIDS2017 dataset, it has performed numerical feature extraction, z-score normalization, and one-hot encoding for recognizing multi-class labels and SMOTE for the solution of class-imbalance problem. While its performance outpaced all others, the Bi-LSTM model improved at capturing both forward and backward time dependencies and obtained A 99% F1-score, a 98.51% accuracy rate, a 99% precision rate, and a 98% recall rate. Training and validation curves indicated strong generalization, and the normalized confusion matrix confirmed high classification accuracy across diverse intrusion types. A comparative analysis showed that the Bi-LSTM model outperformed traditional classifiers such as Naïve Bayes and Deep Multilayer Perceptron, establishing its effectiveness for advanced intrusion detection in intelligent cybersecurity systems. The study offers practical advice for choosing the best IDS models depending on certain network settings and security needs.

Keywords: *Cyber-Attacks, Cybersecurity, Intrusion Detection System, Machine Learning, Deep Learning, Bi-LSTM.*

INTRODUCTION

In recent years, cloud computing has become one of the most significant paradigm shifts in computing within the information technology sector. "Cloud computing" is a type of Internet-based computing that provides a shared pool of resources, such as processing power, memory, user apps, and networking bandwidth. End users may quickly and on-demand access these resources over the Internet with little infrastructure and maintenance costs [1]. Cloud computing services are becoming essential to people's daily life. Small businesses, major corporations, individuals, and government organizations all use cloud services for many of their operations. It has made it possible for customers to access businesses' services whenever they want and at the most affordable price via the Internet [2]. Cloud networks are susceptible to a variety of assaults in spite of their benefits. However, the hazards connected to cloud services have also escalated as their usage has accelerated. Consequently, measures have been taken to enhance cloud security, including the detection and categorization of cyberattacks and the monitoring of networks, which form the foundation of the cloud architecture. Consequently, one of the most important defenses for identifying assaults in the cloud computing network is an IDS [3][4][5].

However, a number of cybersecurity techniques have been employed to protect protecting the cloud environment against malicious internal and external attacks, including firewalls, antivirus software, data encryption algorithms, IDS, IPS, and others [6][7]. Therefore, to detect networks, an IDS is needed intrusions, monitor all packets, and determine if any incoming or the hack has affected outbound packets [8]. It can use one of two techniques to identify intrusions. Traditional intrusion detection methods based on signatures are used in the first. However, these systems have not proved adequate for vast network sizes, such cloud computing settings, and are limited in their ability to identify fresh assaults. Conversely, anomaly-based approaches based on learning techniques are used by intelligent intrusion detection systems [9]. The constant development of cyberattacks means that even while a number of conventional defense measures are employed to safeguard and identify assaults in cloud-based networks, threats and attacks continue to rise significantly. Because

ML and DL models can recognize intricate traffic patterns and accurately determine the assault type, they are able to identify new and unexpected threats [10][11].

This has led to a paradigm shift away from traditional defense mechanisms and towards more intelligent and adaptable ones, such as ML and its subdomain, DL, in an effort to improve security [12]. These days, ML has gained popularity in cybersecurity, and experts are working tirelessly to develop the most advanced cybersecurity techniques. However, because of A lot of machine learning-based detection methods are insufficiently effective for learning large-scale network traffic data because of the network's growing complexity and scope [13]. Because DL-based detection techniques can learn feature representations from raw data and perform better when dealing with intricate and large-scale network traffic, they have become more popular in recent years and can be used in a variety of attack scenarios. This project's objective is to use state-of-the-art cloud computing ML methods to build an intelligent cybersecurity IDS based on assault datasets

Motivation and Contribution of Study

The rapid expansion of digital infrastructure and increasing sophistication of cyberattacks, securing communication and data across cloud computing environments has become critical. Traditional security mechanisms like firewalls and VPNs are no longer sufficient to address advanced intrusion attempts. This study is motivated by the need to develop intelligent, adaptive, and high-performing IDS that leverage advanced ML techniques. The contribution consists in assessing and comparing both supervised and unsupervised learning models on modern datasets like CICIDS2017, using robust methods of a pre-processing nature, and suggesting a DL model based on Bi-LSTM. This approach makes detection more accurate, and enables better anomaly classification as well as overcoming the shortcomings of current systems when dealing with high-dimensional and unbalanced results.

- A Bi-LSTM model is used in the study's intelligent intrusion detection technique to improve detection accuracy by detecting in the network data, there are both forward and backward temporal linkages.
- A robust preprocessing pipeline was used, which included methods for class imbalance, data normalization, one-hot encoding, SMOTE, and efficient representation of multiclass features.
- The suggested model has been thoroughly tested using the widely used CICIDS2017 data set for computer security benchmarking; this set sheds light on a variety of threats and benign traffic.
- The Bi-LSTM model achieved better results than the baseline models (NB and DMLP) regarding F1-score, recall, accuracy, and precision. This suggests that it could be useful in cyber-systems in the real world.

Justification and Novelty of paper

This study's originality and rationale stem from the use of a Bi-LSTM model for intrusion detection, It makes an effort to solve the difficulties that come with recognising complex attack patterns in cybersecurity. Drawing on Bi-LSTM's ability to record both forward and backward temporal dependencies, the study enhances the model's capability to uncover intricate attack behaviors that emerge over time, a significant shortcoming of the conventional ML model. To address the class imbalance, the model is being fine-tuned using more advanced pre-processing techniques such as SMOTE, in conjunction with z-score normalization and one-hot encoding. This combination of cutting-edge DL methods with careful data preparation offers a novel intrusion detection methodology that differs from current methods and is very relevant to practical cybersecurity scenarios.

Structure of paper

The framework of the article is as follows: Studies that have examined cloud intrusion detection systems are covered in Section II. Section III describes the proposed method. Section IV presents the results and comparative analysis, while Section V presents the findings along with recommendations and insights for further research.

LITERATURE REVIEW

There have been a lot of proposals for anomaly detection methods to build successful NIDS in the last 30 years. These methods try to improve the speed of network packet traffic and have strong prediction accuracy to identify assaults. There is a wide range of techniques here, from old-school ML to more modern DL-based methods, and everything in between. The comparative analysis of background study based on their Methodology, Key Findings, Performance, Limitations, and Future work are provided in Table I.

Portela et al. (2019) the modernization of IDS has been made possible by research into intelligent solutions that identify abnormalities on a range of communications and computer systems while protecting the privacy of data. The effectiveness of many intrusion detection techniques, including supervised ones (KNN and SVM) and unsupervised ones (Isolation Forest and K-Means), is evaluated in this study using the UNSW-NB12 data set. The findings showed that the SVM gaussian fine-supervised method has a 92% accuracy rate in differentiating between normal and abnormal data[14].

Srivastava et al. (2019) have detected unusual patterns in the recently presented dataset using unique feature reduction-based ML methods. A rapidly expanding field is intrusion detection. In the past, network traffic intrusion detection relied on supervised learning approaches. The attained level of accuracy is 86.15% [15].

Gao et al. (2019) proposes a framework for IDS-based adaptable group education. They create many DT and alter the training data proportion to produce the Multitree approach. To increase the total impact of detection, it uses a number of fundamental classifiers, including decision trees, random forests, kNNs, and DNNs, and constructs an ensemble adaptive voting system. they validate their approach using

NSL-KDD Test+; the adaptive voting algorithm obtains an accuracy of 85.2%, while the Multitree technique achieves an accuracy of 84.2% [16].

Kurniawan et al. (2019) research shows that The proposed approach is capable of detecting five types of intrusions: dos, r2l, u2r, normal, and probe. The evaluation results show that the recommended approach outperforms the others in terms of accuracy (97.02%), detection rate (97%), and false alarm rate (0.16%) than the intrusion detection techniques, but it does not perform well in terms of processing time [17].

Wani et al. (2019) the company's own cloud environment was breached An IDS and an attack tool called Tor Hammer were used to generate a fresh dataset. This work uses a variety of ML methods, NB, RF, and SVM all achieved 98.0% overall accuracy, while SVM achieved 97.6% in classification [18].

Pradeepthi and Kannan (2018) suggest a new method that detects botnet traffic using neuro-fuzzy classification algorithms. A number of open-source botnet simulation tools were used to target an application that was deployed to the Eucalyptus cloud in order to create the dataset required for the research. The system's accuracy rate was 94.78% using 56 characteristics and 15,000 occurrences[19].

Salman et al. (2017) examine methods that combine AD with anomaly classification, as is becoming increasingly frequent in recent studies. Several attack types were detected and categorized using learning models that they developed and evaluated using a widely used publicly available dataset. It has specifically employed two supervised ML methods, namely RF and LR. demonstrate that similarities across assaults might lead to less accurate classification, even in the case of 100% detection. their results demonstrate a detection accuracy of over 99% and a classification accuracy of 93.6% [20].

Despite significant progress in ML-based IDS several research gaps remain unaddressed. Many existing approaches achieve high accuracy but often struggle with processing efficiency, limiting their applicability in real-time environments. The accurate categorization of similar attack types continues to be a challenge, affecting the reliability of multi-class classification. Moreover, most models are evaluated on specific datasets, raising concerns about their generalizability to diverse or real-world network conditions. Issues such as handling imbalanced data, ensuring scalability in cloud infrastructures, and the lack of advanced models that paradigms emphasize the necessity for IDS that are both more powerful and more adaptable

Table I. Literature summary on Intrusion Detection using Machine Learning Approaches in cloud

Author	Methodology	Dataset	Key Findings	Limitation	Future Work
Portela, Almenares Mendoza and Benavides (2019)	Supervised (KNN, SVM) and Unsupervised (Isolation Forest, K-Means) ML models	UNSW-NB15	SVM (Gaussian Fine) achieved 92% accuracy in distinguishing normal and abnormal traffic	Evaluated only a limited set of algorithms and datasets	Explore deep neural networks and test on real-time traffic
Srivastava, Agarwal and Kaur (2019)	Feature reduction combined with ML classification models	NSL-KDD	Achieved 86.15% accuracy with reduced feature space, improving detection efficiency	Only tested supervised models	Incorporate deep learning and evaluate on more complex datasets
Gao et al. (2019)	MultiTree and Adaptive Voting using Decision Tree, RF, kNN, and DNN is an example of adaptive ensemble learning.	NSL-KDD (Test+)	Accuracy of MultiTree: 84.2%; Adaptive Voting: 85.2%	Moderate accuracy; lacks real-time evaluation	Apply ensemble method to real-world datasets and online detection systems
Kurniawan et al. (2019)	Supervised classification for intrusion types in IoT systems	IoT-specific dataset with five attack types (Normal, Probe, DoS, R2L, U2R)	Accuracy: 97.02%, Detection Rate: 97%, FAR: 0.16%	High processing time limits real-time deployment	Optimize time efficiency and adapt to scalable IoT environments
Wani et al. (2019)	ML-based IDS using SVM, Naive Bayes, Random Forest on cloud dataset	Custom dataset from OwnCloud (Tor Hammer attack simulation)	Accuracy: SVM: 99.7%, RF: 97.6%, NB: 98.0%	Dataset not generalizable; limited to specific attack tool	Validate results on standard datasets and in diverse cloud setups
Pradeepthi and Kannan (2018)	Neuro-Fuzzy classifier for botnet detection	Custom botnet traffic dataset (Eucalyptus cloud)	Accuracy: 94.78% with 15,000 instances and 56 features	Dataset and method are not generalized	Apply to broader cloud environments and integrate adaptive learning
Salman et al. (2017)	Supervised ML for detection and categorization (Linear Regression, Random Forest)	Public network dataset (likely NSL-KDD or similar)	Detection: >99%, Categorization: 93.6%	High detection, but reduced categorization accuracy due to similar attack patterns	Improve multi-class differentiation and test across more datasets

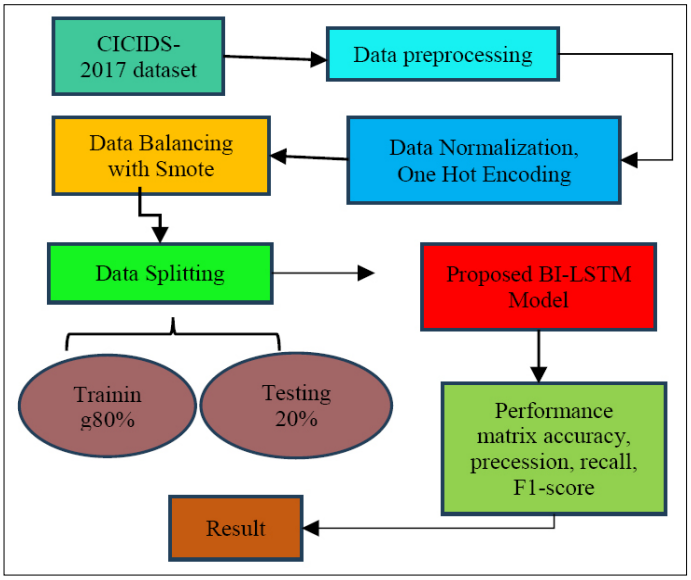

Fig 1. Flowchart for Cybersecurity Intrusion Detection System

METHODOLOGY

This research presents a smart cybersecurity IDS that uses a Bi-LSTM model and extensive data pre-treatment methods to improve model accuracy and reliability. The pre-processing pipeline includes numerical feature extraction, z-score normalization to mitigate gradient dispersion during training, and one-hot encoding applied specifically to label values for efficient multi-class classification. In order to increase model sensitivity and address class imbalance, the SMOTE creates synthetic instances in minority classes. The dataset is split into Subsets for testing (20%) and training (80%) are used to guarantee precise model assessment. The Bi-LSTM architecture, capable of capturing contextual dependencies in is used to enhance sequence learning and the identification of intricate assault patterns that go both forward and backward in time. The model's performance is evaluated using common measures such as accuracy, precision, recall, and F1-score, which offer a comprehensive assessment of the model's ability to categorise various types of intrusions. Figure 1 Proposed flowchart for Intrusion Detection in Cybersecurity.

These flowchart steps are discussed below:

Data Collection

To teach ML to spot suspicious activity, a dataset is very necessary. The utilized CICIDS-2017 dataset records network data in real time and includes both hostile and benign activity. The main goal of creating this dataset was to collect background traffic in real time. By adopting the B-profile approach, harmless background traffic was collected. harmless traffic from the HTTP, HTTPS, FTP, SSH, and email accounts of 25 different users. Five days' worth of network traffic was gathered and dumped, with one day having normal traffic and the other days having attacks inserted into it. Heartbleed, Brute Force FTP, Web Attack, Infiltration, and Botnet, Brute Force SSH, 'DoS,' and 'DDoS' were some of the attacks that were presented. Their investigation of the CICIDS2017 dataset included a thorough EDA.

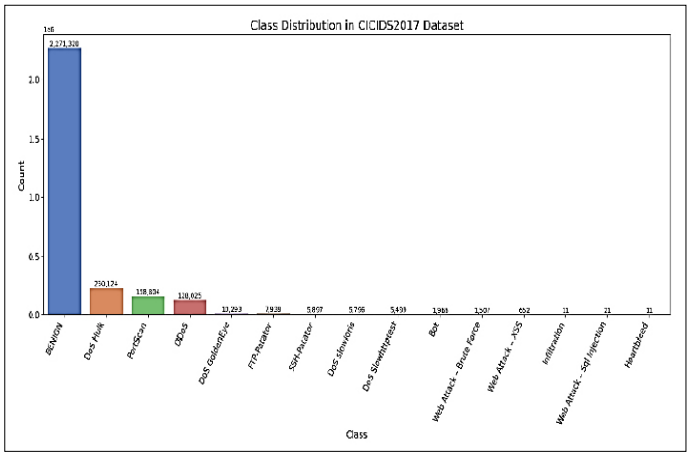

Fig 2. Class Distribution in the CICIDS2017 Dataset

To visualize the distribution of classes in the 'Label' column, it used Seaborn, a Python-based program. Figure 2's visualization highlights the dataset's imbalance: the 'Benign' class, which ranks second in CICIDS2017, has about 2.2 million records, far more than the 'DoS Hulk' class, which has around 230,000 entries.

Data Preprocessing

The initial step of data preparation addressed the issue with the dataset's missing values. Due to the magnitude of the CICIDS2017 dataset, removing these missing values should not compromise the dataset's overall dependability and quality. Using the data is null().sum() method as a verification tool, it made sure that the dataset was completely free of missing values after cleanup. Data normalization, numerical division, one-hot encoding, and data balancing were employed in this procedure.

Data Normalization

The numerically processed data has a lot of variances in its individual characteristics, normalizing this data helps prevent gradient dispersion when the backpropagation technique is used. The z-score normalization approach is used to change all of the CICIDS2017 data.by Equation (1)

$$m'i = \frac{m_i - \overline{m}}{x} \quad (1)$$

It denotes the value of the data sample before and after normalization as $m'i$ and m_i respectively. At the same time, $\overline{m}$ is used to depict the average data value of the feature prior normalization.

One-Hot Encoding:

In ML, One such method that makes use of categorical variables is one-hot encoding. It includes manifesting every category as a vector of binary. The procedure creates a binary vector for every distinct category, where all other items are set to zero and only the matching element for the observed category is set to one. Consequently, the dataset's category variables are represented by a matrix of binary vectors.

Data Balancing with SMOTE

An efficient technique for oversampling that is often used in medical applications to address the issue of class-imbalanced data is the SMOTE method. By employing Euclidean distance and expanding the minority class's quantity of data samples as it achieves this by using the measurements to pseudo-randomly create synthetic points from their closest neighbors. Since these new instances are created using the original features, they are intended to look like the original data. It is important to keep in mind, though, that SMOTE might not be the best option for high-dimensional data since it might introduce extra noise. The SMOTE approach is applied in this work to create a fresh training dataset.

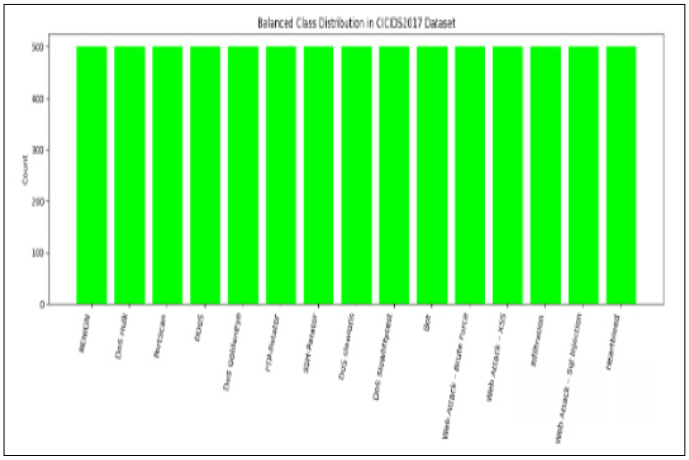

Fig 3. After Data Balanced Graph In CICIDS2017 Dataset

The bar chart, of balanced class distribution in CICIDS2017 dataset visually represents the class distribution after the application of the SMOTE, shows in Figure 3. Every bar represents a distinct assault category within the CICIDS2017 dataset, and the height of each bar indicates the count of instances belonging to that category. Following the balancing procedure using SMOTE, the chart demonstrates a uniform distribution across all attack types, with each class now containing approximately 500 instances. It is essential to use this balanced representation while training ML models, as it mitigates the bias that can arise from imbalanced datasets, ensuring that the model gives equal importance to each class during the learning process and improves its overall generalization performance across different attack types.

Data Splitting

It divided the data set into three parts, each with a specific function. 20% came from the test set, while 80% came from the training set.

Proposed Bi-LSTM Model

In a Belts model, which consists of the LSTM is a kind of RNN and comes in two varieties: forward and backward. To address the gradient vanishing issue in conventional RNN, a gating unit is incorporated into the LSTM. This improves the RNN's ability to recognise and utilise the dependencies seen in long-distance data and increases the LSTM's ability to capture long-term relationships. The four main parts of the new structure that each LSTM cell unit adopts are the input gate i_t,, output gate o_t, forget gate f_t, and storage unit c_t, Figure 4 depicts the internal architecture of a single Bi-LSTM module cell.

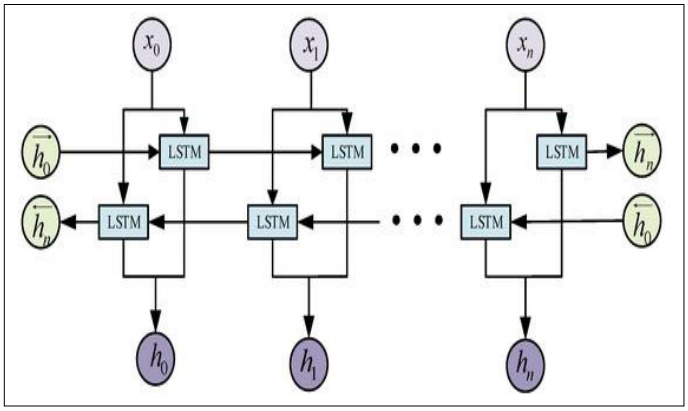

Fig 4. Bilstm Network Model.

The LSTM update formula is as follows:

Forget Gate Mechanism

The forget gate f_t is a memory cell that has been reset. x_t is the input at time t, $W_{(f)}$ is a weight matrix, and σ is the Sigmoid activation function. The short-term memory is represented by h_{t-1}, the hidden state, and U is the hidden layer output's weight matrix. It is the offset vector, b_f, it formulates Equation (2):

$$f_t = \sigma(W_{(f)}x_t + u_{(f)}h_{t-1} + b_{(f)}) \quad (2)$$

Input Gate Mechanism:

The input gates that regulate the memory cell's input are denoted by $W_{(f)}$ is a weight matrix, x_t is the input at time t, and σ is the Sigmoid activation function. The hidden state, denoted by h_{t-1}, is the short-term memory, and U is the weight matrix of the hidden layer output. This is the offset vector, $b_{(i)}$, it formulates Equation (3):

$$i_t = \sigma(W_{(f)}x_t + u_f\ h_{t-1} + b_{(f)}) \quad (3)$$

Current Unit Status

The proposed memory cell is $\tilde{c}$. A reset memory cell is the forget gate f_t. The long-term memory is represented by c_{t-1}, which is the cell state at $t-1$, it formulates Equation (4):

$$\tilde{c} = f_1 \odot c_{t-1} \quad (4)$$

Update Unit Status

The long-term memory is represented by the cell state, c_t is, whereas the candidate memory cell is denoted by $\tilde{c}$. It stands for the input gates that regulate the memory cell's input. x_t is the input at time t, and $W_{(c)}$ is a weight matrix. The short-term memory is represented by h_{t-1}, the hidden state, whereas U represents the hidden layer output's weight matrix. The offset vector is $b_{(c)}$, it formulates Equation (5):

$$c_t = \tilde{c} + i_t \odot tanh\,(W_{(c)}x_t + u_c\ h_{t-1} + b_{(c)}) \quad (5)$$

Output Gate Mechanism

The output gate the output gates that regulate the memory cell's output are denoted by o_t . $W_{(o)}$ is a weight matrix, x_t is Sigmoid activation function σ is the input at time t. The hidden state, represented by h_{t-1} , is the short-term memory, and U is the weight matrix of the hidden layer output. The offset vector is $b_{(o)}$, it formulates Equation (6).

$$o_t = \sigma(W_{(o)}x_t + u_o\, h_{t-1} + b_{(o)}) \quad (6)$$

The Current State of the Hidden Layer

h_t is the state that is concealed. Tanh is the sigmoid activation function. The cell state, or C_t, stands for long-term memory. 0_t stands for the output gates that control the output of the memory cell in Equation (7) is formulated:

$$h_t = 0_t * \ tanh(C_t) \quad (7)$$

The model underwent 60 rounds of training with a batch size of 512 and a seed value of 0, and three heads of the multi-head attention mechanism. The learning rate of the model is 0.0003, and it is optimized using the Adam method. Adam may adaptively modify the momentum and learning rate in response to gradient information, preventing an early descent into a local minimum. In the output component of this model, a Sigmoid activation function is used and adds two dense layers after BiLSTM. In the meanwhile, a dropout layer with parameters of 0.8 and 0.3 is placed between the two dense layers and after the embedding layer. Using the CICIDS2017 dataset, the model takes 33 minutes to train in total.

Performance Matrix

In this work employ the model using the measures of F1-Score, Accuracy, Precision, and Recall, with FN representing the amount of incorrectly predicted samples, The numbers of properly predicted negative samples, successfully predicted positive samples, and favorably anticipated samples that were mis predicted are denoted by TP, TN, and FP, respectively.

Accuracy: Reflects, as in Equation (8), the ratio of correctly categorized occurrences to all classifications:

$$Accuracy = \frac{\mathrm{TN} + \mathrm{TP}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \quad (8)$$

Precision: shows, as in Equation (9) the ratio of all expected positive cases to the number of correctly predicted positive cases.

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP+FP}} \quad (9)$$

Recall: The fraction of correctly predicted positive occurrences relative to all favourable circumstances, as in Equation (10).

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \quad (10)$$

F1 Score: The mean of the accuracy and recall weighted values, as indicated in Equation (11).

$$\mathrm{F1} = \frac{2*(\mathrm{precision}*\mathrm{recall})}{\mathrm{precision}+\mathrm{recall}} \quad (11)$$

These performance matrices are used to do comparative analysis and evaluate the model's intrusion detection performance in order to enhance cybersecurity on the CICIDS2017 dataset.

RESULTS AND DISCUSSION

As an illustration, the outcomes of the proposed Bi-LSTM-based intelligent cybersecurity IDS are shown, together with the comparative analysis compared to baseline models. The cloud-based system was based on a 64-bit operating system and was supported by high power resources and powered by Python 3 working with TensorFlow and Scikit-Learn libraries. Performance of Bi-LSTM model was measured using performance measures on the CICIDS2017 dataset.

Table II. Findings of Bi-LSTM Models for IDS key Performance matrix

Performance Metric	Bi-LSTM
Accuracy	98.51
Precision	99
Recall	98
F1-score	99

As seen in Table II, the Bi-LSTM model performed exceptionally well, reaching 99% F1-score, 98.51% accuracy, 99% precision, and 98% recall. The methodology is highly helpful in real-time cybersecurity applications in cloud computing settings since these metrics show that it can accurately identify intrusions with few false alarms and false termites.

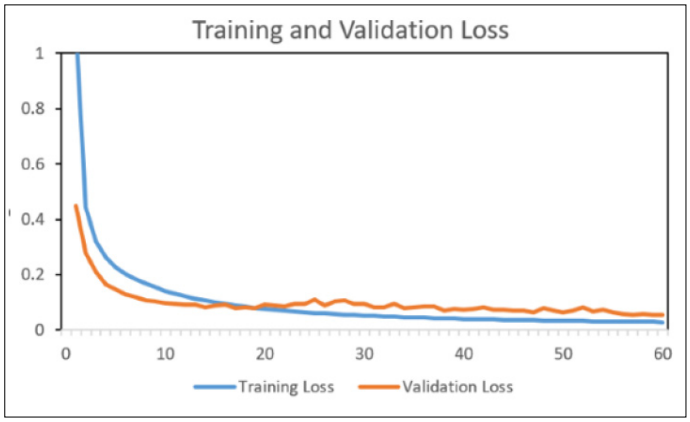

Fig 5. Loss Curve of BI-LSTM Model.

The variation in training and validation loss under training the model upon the CICIDS2017 dataset for 60 epochs is depicted in Figure 5. In the beginning, when its learning rate and prediction error drastically drop, training and validation losses also drop rather dramatically. A good learning process without overfitting from the model is shown by the losses gradually reducing and stabilising over the course of the epochs. Good generalisation throughout the epochs, the training and validation loss curves' near-alignment indicates the model's performance on unknown data.

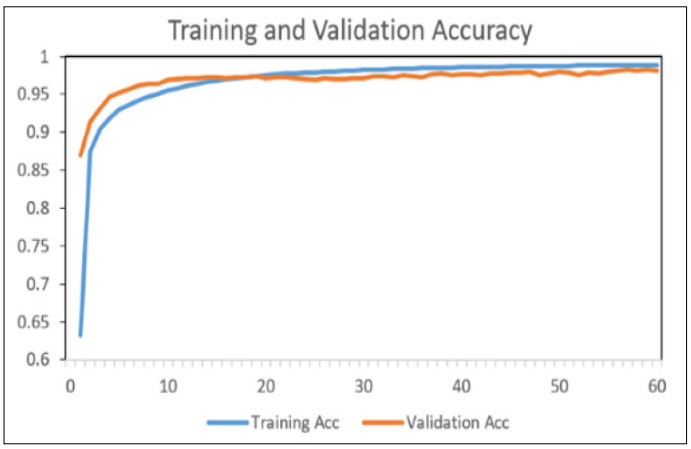

Fig 6. Accuracy Curve of BI-LSTM Model.

The model's utilizing the CICIDS2017 dataset to examine trends in Training and validation accuracy across 60 epochs is shown in Figure 6. During the early epochs, both curves demonstrate a sharp rise in accuracy, indicating effective learning. Additionally, the validation accuracy rises and stabilizes around 97%, while the training accuracy keeps improving progressively, approaching 100%., indicating a good capacity for generalization. The close alignment of the two curves indicates that throughout the training phase, the model exhibits little overfitting and consistently performs well on both visible and invisible data.

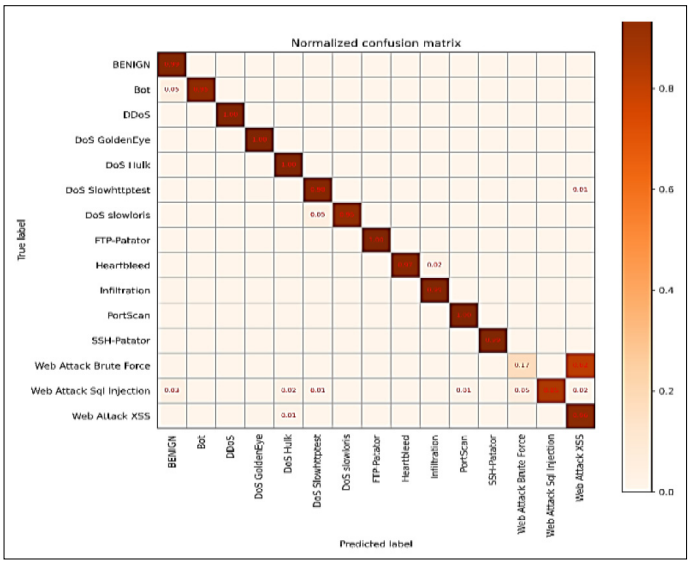

Fig 7. Normalized Confusion Matrix of CICIDS2017 Dataset.

Figure 7 displays the model's normalised confusion matrix for the CICIDS2017 dataset, which also shows the classification accuracy for benign traffic and different forms of attacks. The diagonal dominance shows that the majority of the samples are properly classified by the model, with high accuracy observed for classes like BENIGN, DDoS, DoS Goldeneye, and Ports can. However, minor misclassifications occur in Web Attack XSS, sometimes mistaken for Web Attack Brute Force and SQL Injection, and Heartbleed are among the classes, which shows some confusion with Infiltration. Overall, the matrix reflects strong classification capability with only a few confusions in closely related attack categories.

Comparative Analysis

The comparative study of IDS using the CICIDS2017 dataset is presented in this section. The same dataset and evaluation matrix used in Table III serve as the basis for the model comparisons. A comparison between the performance of the proposed Bi-LSTM model and baseline intrusion detection methods from earlier research. In contrast to the DMLP model, which recorded 91% accuracy, and the NB model, which recorded 86.72%, Bi LSTM obtains the highest number of 98.51% accuracy. This comparison demonstrates the Bi-LSTM model's superior capacity to precisely identify intrusions thereby confirming its suitability for application in intelligent cybersecurity systems.

Table III. Comparison between existing and propose model performance for intrusion detection

Moled	Accuracy
Bi-LSTM Performance	98.51
Naïve Bayes[21]	86.7180
DMLP[22]	91

The proposed Bi-LSTM-based IDS offers several key advantages. Its bidirectional architecture achieves the temporal dependencies from both the forward and the backward directions, better allowing for advanced and dynamically changing attack patterns' detection. Improved preprocessing procedures, including z-score normalization, one-hot encoding, and SMOTE, maximize model sensitivity, balance data, and guarantee stable training. As opposed to the conventional models, the Bi-LSTM has a significantly better It is appropriate for intricate real-time cybersecurity uses in dynamic network environments because to its accuracy and generalizability.

CONCLUSION AND FUTURE SCOPE

In the increasing complexity of cyberthreats and the growing traffic strain on networks, this study explores the use of DL, IDS in gigantic network systems. In the presented Bi-LSTM-based IIDS shows better performance for detecting and categorizing different kinds of cyber threats based on the CICIDS2017 dataset. The model obtains a 99% F1-score, 98.51% accuracy, 99% precision, and 98% recall. much more effective than more conventional models like NB and DMLP. These results check the efficiency and reliability of the Bi-LSTM model for its real-world application to the intelligent cybersecurity systems as a prospective solution for detecting intrusions in complicated real-time network-based systems. Despite its effectiveness, the suggested Bi-LSTM model has certain drawbacks. It may have difficulty identifying the rare or new attack type because of the fundamental nature of the supervised mechanism and the use of historical records. Also, even though SMOTE can mitigate class imbalance, it can induce noise, particularly in high-dimensional data and may compromise precision. In future investigations,

the use of hybrid models combining Bi-LSTM and attention mechanisms or transformer-based architecture may improve detection performance. Real-time deployment and testing on live traffic, along with transfer learning to adjust to fresh assault trends, are also promising directions for advancing the system's practicality and robustness.

REFERENCES

1. M. Zekri, S. El Kafhali, N. Aboutabit, and Y. Saadi, "DDoS Attack Detection Using Machine Learning Techniques in Cloud Computing Environments," in *2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech)*, IEEE, Oct. 2017, pp. 1–7. doi: 10.1109/CloudTech.2017.8284731.
2. M. Jouini and L. B. A. Rabai, "A Security Framework for Secure Cloud Computing Environments," *Int. J. Cloud Appl. Comput.*, 2016, doi: 10.4018/ijcac.2016070103.
3. O. Achbarou, M. A. El Kiram, O. Bourkoukou, and S. Elbouanani, "A New Distributed Intrusion Detection System Based on Multi-Agent System for Cloud Environment," *Int. J. Commun. Networks Inf. Secur.*, 2018, doi: 10.17762/ijcnis.v10i3.3546.
4. M. Hafsa and F. Jemili, "Comparative Study Between Big Data Analysis Techniques in Intrusion Detection," *Big Data Cogn. Comput.*, vol. 3, no. 1, p. 1, Dec. 2018, doi: 10.3390/bdcc3010001.
5. S. S. S. Neeli, "Serverless Databases : A Cost-Effective and Scalable Solution," *IJIRMPS*, vol. 7, no. 6, 2019.
6. N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Netw. Appl.*, 2019, doi: 10.1007/s12083-017-0630-0.
7. S. Singamsetty, "Fuzzy-Optimized Lightweight Cyber-Attack Detection For Secure Edge-Based IoT Networks," *J. Crit. Rev.*, vol. 6, no. 7, 2019, doi: 10.53555/jcr.v6:i7.13156.
8. S. Shamshirband, N. B. Anuar, M. L. M. Kiah, and A. Patel, "An Appraisal and Design of a Multi-Agent System Based Cooperative Wireless Intrusion Detection Computational Intelligence Technique," *Eng. Appl. Artif. Intell.*, vol. 26, no. 9, pp. 2105–2127, Oct. 2013, doi: 10.1016/j.engappai.2013.04.010.
9. J. Lee and K. Park, "AE-CGAN Model based High Performance Network Intrusion Detection System," *Appl. Sci.*, vol. 9, no. 20, p. 4221, Oct. 2019, doi: 10.3390/app9204221.
10. J. Gu, L. Wang, H. Wang, and S. Wang, "A Novel Approach to Intrusion Detection Using SVM Ensemble with Feature Augmentation," *Comput. Secur.*, vol. 86, pp. 53–62, Sep. 2019, doi: 10.1016/j.cose.2019.05.022.
11. A. Gogineni, "Novel Scheduling Algorithms For Efficient Deployment Of Mapreduce Applications In Heterogeneous Computing," *Int. Res. J. Eng. Technol.*, vol. 4, no. 11, 2017.
12. F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, "TSDL: A Two-Stage Deep Learning Model for Efficient Network Intrusion Detection," *IEEE Access*, 2019, doi: 10.1109/ACCESS.2019.2899721.
13. S. H. Kok, A. Abdullah, N. Z. Jhanjhi, and M. Supramaniam, "A review of intrusion detection system using machine learning approach," *International Journal of Engineering Research and Technology*. 2019.
14. F. G. Portela, F. A. Mendoza, and L. C. Benavides, "Evaluation of the performance of supervised and unsupervised Machine learning techniques for intrusion detection," in *2019 IEEE International Conference on Applied Science and Advanced Technology (ICASAT)*, IEEE, Nov. 2019, pp. 1–8. doi: 10.1109/iCASAT48251.2019.9069538.
15. A. Srivastava, A. Agarwal, and G. Kaur, "Novel Machine Learning Technique for Intrusion Detection in Recent Network-based Attacks," in *2019 4th International Conference on Information Systems and Computer Networks, ISCON 2019*, 2019. doi: 10.1109/ISCON47742.2019.9036172.
16. X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An Adaptive Ensemble Machine Learning Model for Intrusion Detection," *IEEE Access*, 2019, doi: 10.1109/ACCESS.2019.2923640.
17. A. A. Kurniawan, H. A. Santoso, M. A. Soeleman, and A. Z. Fanani, "Intrusion Detection System as Audit in IoT Infrastructure using Ensemble Learning and SMOTE Method," in *Proceeding - 2019 5th International Conference on Science in Information Technology: Embracing Industry 4.0: Towards Innovation in Cyber Physical System, ICSITech 2019*, 2019. doi: 10.1109/ICSITech46713.2019.8987524.
18. A. R. Wani, Q. P. Rana, U. Saxena, and N. Pandey, "Analysis and Detection of DDoS Attacks on Cloud Computing Environment using Machine Learning Techniques," in *Proceedings - 2019 Amity International Conference on Artificial Intelligence, AICAI 2019*, 2019. doi: 10.1109/AICAI.2019.8701238.
19. K. V. Pradeepthi and A. Kannan, "Detection of Botnet traffic by using Neuro-fuzzy based Intrusion Detection," in *2018 10th International Conference on Advanced Computing, ICoAC 2018*, 2018. doi: 10.1109/ICoAC44903.2018.8939109.
20. T. Salman, D. Bhamare, A. Erbad, R. Jain, and M. Samaka, "Machine Learning for Anomaly Detection and Categorization in Multi-Cloud Environments," in *Proceedings - 4th IEEE International Conference on Cyber*

Security and Cloud Computing, CSCloud 2017 and 3rd IEEE International Conference of Scalable and Smart Cloud, SSC 2017, 2017. doi: 10.1109/CSCloud.2017.15.

21. S. S. Panwar, P. S. Negi, and Y. P. Raiwani, "Implementation of Machine Learning Algorithms on CICIDS-2017 Dataset for Intrusion Detection using WEKA," *Int. J. Recent Technol. Eng.*, vol. 8, no. 3, Sep. 2019, doi: 10.35940/ijrte.C4587.098319.
22. S. Ustebay, Z. Turgut, and M. A. Aydin, "Intrusion Detection System with Recursive Feature Elimination by Using Random Forest and Deep Learning Classifier," in *International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism, IBIGDELFT 2018 - Proceedings*, 2019. doi: 10.1109/IBIGDELFT.2018.8625318.
23. Chinta, P. C. R., & Karaka, L. M. (2020). Agentic AI and Reinforcement Learning: Towards More Autonomous and Adaptive AI Systems.
24. Kalla, D., Kuraku, D. S., & Samaah, F. (2021). Enhancing cyber security by predicting malwares using supervised machine learning models. *International Journal of Computing and Artificial Intelligence*, *2*(2), 55-62.
25. Katari, A., & Kalla, D. (2021). Cost Optimization in Cloud-Based Financial Data Lakes: Techniques and Case Studies. *ESP Journal of Engineering & Technology Advancements (ESP-JETA)*, *1*(1), 150-157.
26. Kalla, D., Smith, N., Samaah, F., & Polimetla, K. (2021). Facial Emotion and Sentiment Detection Using Convolutional Neural Network. *Indian Journal of Artificial Intelligence Research (INDJAIR)*, *1*(1), 1-13.
27. Routhu, K., Bodepudi, V., Jha, K. M., & Chinta, P. C. R. (2020). A Deep Learning Architectures for Enhancing Cyber Security Protocols in Big Data Integrated ERP Systems. *Available at SSRN 5102662*.
28. Kuraku, S., & Kalla, D. (2020). Emotet malware—a banking credentials stealer. *Iosr J. Comput. Eng*, *22*, 31-41.
29. Kalla, D., & Samiuddin, V. (2020). Chatbot for medical treatment using NLTK Lib. *IOSR J. Comput. Eng*, *22*, 12.
30. Routhu, K., & Jha, K. M. (2021). Leveraging Big Data Analytics and Machine Learning Techniques for Sentiment Analysis of Amazon Product Reviews in Business Insights. *Available at SSRN 5106490*.
31. Chinta, P. C. R., & Katnapally, N. (2021). Neural Network-Based Risk Assessment for Cybersecurity in Big Data-Oriented ERP Infrastructures. *Neural Network-Based Risk Assessment for Cybersecurity in Big Data-Oriented ERP Infrastructures*.
32. Karaka, L. M. (2021). Optimising Product Enhancements Strategic Approaches to Managing Complexity. *Available at SSRN 5147875*.
33. Boppana, S. B., Moore, C. S., Bodepudi, V., Jha, K. M., Maka, S. R., & Sadaram, G. AI And ML Applications In Big Data Analytics: Transforming ERP Security Models For Modern Enterprises.

Citation: Mukund Sai Vikram Tyagadurgam, Venkataswamy Naidu Gangineni, et al., "Designing an Intelligent Cybersecurity Intrusion Identify Framework Using Advanced Machine Learning Models in Cloud Computing", Universal Library of Engineering Technology, 2022; 18-26. DOI: https://doi.org/10.70315/uloap.ulete.2022.003.