International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 02 Issue: 10 October 2024
Page No: 2456 - 2462
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2024.0336

Ejection of Real-Time Malicious Intrusion and Attacks in IoT Empowered Infrastructure

S. K. Shameena Begum[1], Pulikonda Venkata Yoga Abhishek[2], Kadavakollu Lakshmi Chaitanya[3], Mandalapu Gopichand[4]
[1]Professor, Head of the department, Computer Science Engineering (CyberSecurity), Ramachandra College of Engineering, Eluru, Andhra Pradesh, India.
[2,3,4]UG, Computer Science Engineering, Ramachandra College of Engineering (CyberSecurity), Eluru, Andhra Pradesh, India.
Emails: *sameenamz@gmail.com[1], pulikondaabhishek03@gmail.com[2], chaitanya3419@gmail.com[3], gopichandchowdary108@gmail.com[4]*

Abstract
The project recognizes the pervasive threat of computer viruses, malware, and hostile attacks on computer networks, highlighting the critical role of intrusion detection as a proactive defense technology. The project introduces a novel approach based on deep learning to identify and mitigate cybersecurity vulnerabilities and breaches in IoT-driven cyber-physical systems, aiming for enhanced security measures. The project's objective is to elevate intrusion detection beyond the limitations of traditional systems by addressing issues like accuracy, detection effectiveness, and reducing false positives. This emphasizes the advancement and innovation in cybersecurity. To achieve the project's goals, the method employs a generative adversarial network, a cutting-edge deep learning technique. Additionally, it distinguishes itself by contrasting unsupervised and deep learning-based discriminative approaches, showcasing a comprehensive and effective approach to cybersecurity. In our project, we successfully implemented an ensemble method to boost predictive accuracy by integrating multiple individual models. Particularly noteworthy is the inclusion of a hybrid architecture, combining Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM), denoted as CNN+LSTM. This hybrid model achieved an impressive accuracy of 99% when applied to the KDD-Cup dataset, underscoring the efficacy of our ensemble technique for intrusion detection in IoT-based cybersecurity infrastructures.

Keywords: *Deep Learning, Intrusion Detection Systems (IDS), Cybersecurity, Anomaly Detection, Security Assaults.*

1. Introduction
Deep learning (DL) techniques employ several operators that prove advantageous for discrete mechanisms, particularly artificial neural networks (ANNs). There are three layers in total: hidden, output, and input [1-3]. But in DL, each layer functions nonlinearly, sending replies in response to the information supplied by the input layers. Lately, speech and audio recognition, image processing, signal processing, and graphic recognition have all been discovered using DL techniques. In fact, DL learning techniques are extensively applied in medicine to the study of illnesses and genomes. The architecture and operation of the DL methods demonstrate how to handle large data with an emphasis on forward and back propagation techniques by using sophisticated data organisation (such as text, pictures, and numerical hierarchies). The other query also concerns how devices alter values and hyperparameters with dimensions in order to calculate sample sizes that render various layers. There is a slight distinction in presentation and representation between training and testing when using successful methods. The family's customary teaching methods, which are structured and of high quality, were slightly altered, leading to the characteristics of outdated wisdom. Privacy and security considerations are crucial because of the reasons why DL approaches are assumed and used in many different contexts. Data movement—the transfer of data between encrypted forms in training,

testing, and interface modules—is the main problem with DL approaches. Furthermore, the DL used in all models for the training phase depends on a vast amount of user-sensitive, private, and confidential data, mostly training data [4].

2. Method

The methodology for implementing the Detecting real-time malicious intrusion and attacks in IoT-powered cybersecurity infrastructures typically involves a combination of methodologies Utilizing machine learning algorithms to detect deviations from normal behaviour in IoT devices and network traffic. Implementing pattern recognition techniques to identify known malicious patterns or signatures in network traffic or device behaviour [5-7]. security events from various IoT devices and network sources to identify potential security incidents. Integrating threat intelligence feeds to leveraging up-to-date information about emerging threats and attack vectors.

2.1. Proposed Work

The proposed system utilizes deep learning with a generative adversarial network to significantly enhance cybersecurity detection in IoT-enabled cyber-physical systems, achieving high accuracy, maintaining data privacy, and ensuring ease of deployment. The proposed system markedly enhances cybersecurity threat detection accuracy. It leverages deep learning, improving intrusion detection in complex settings. And also preserves critical data privacy and integrity for security. In our project, we successfully implemented an ensemble method to boost predictive accuracy by integrating multiple individual models. Particularly noteworthy is the inclusion of a hybrid architecture, combining Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM), denoted as CNN+LSTM. This hybrid model achieved an impressive accuracy of 99% when applied to the KDD-Cup dataset, underscoring the efficacy of our ensemble technique for intrusion detection in IoT-based cybersecurity infrastructures. The integration of a Flask-based user interface ensures practicality, offering a user-friendly testing environment, while secure authentication enhances the overall cybersecurity of the system. This amalgamation of advanced model architectures and user-friendly features positions our project as a robust and efficient solution for real-time intrusion detection in IoT-driven cybersecurity domains.

2.2. System Architecture

The system architecture for the project "Detection of Real-Time Malicious Intrusions and Attacks in IoT Empowered Cyber Security Infrastructures" follows a structured approach (Figure 1). It begins with dataset exploration to understand and identify key features, proceeds with data preprocessing to prepare the dataset for model training, and then splits the data into training and testing sets. The core of the architecture involves building machine learning models, including a hybrid CNN+LSTM model and a standalone CNN model to learn patterns and representations for intrusion detection. Model evaluation is performed using the testing set, assessing metrics like accuracy and precision, followed by a comprehensive analysis of model performance [8]. The integration of CNN+LSTM showcases a commitment to leveraging both spatial and temporal information for enhanced intrusion detection in real-time within IoT-driven cybersecurity environments.

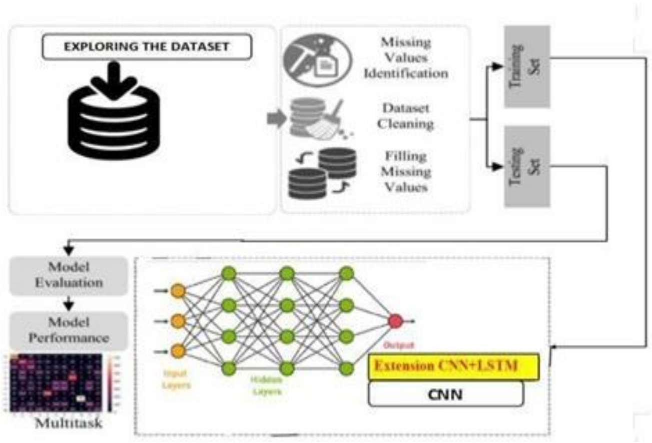
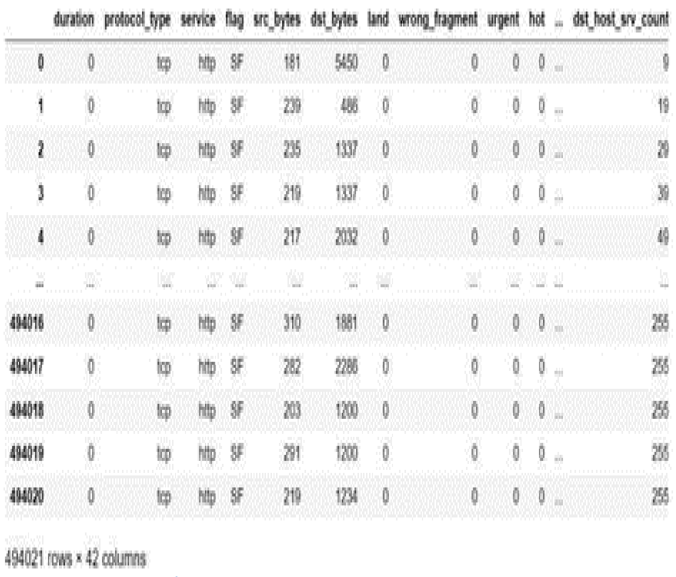


Figure 1 **Proposed Architecture**

2.3. Dataset Collection

Here, the project dives into the datasets that are crucial for training and evaluating the intrusion detection system [9]. Different datasets (KDDCUP99, NSL KDD, UNSW- NB15) are explored to understand their contents, features, and structure. This step helps in gaining insights into the data that is being worked with.



	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count
0	0	tcp	http	SF	181	5450	0	0	0	0	...	9
1	0	tcp	http	SF	239	486	0	0	0	0	...	19
2	0	tcp	http	SF	235	1337	0	0	0	0	...	29
3	0	tcp	http	SF	219	1337	0	0	0	0	...	39
4	0	tcp	http	SF	217	2032	0	0	0	0	...	49
...	...	...	...	...	...	...	...	...	...	...	...	...
494016	0	tcp	http	SF	310	1881	0	0	0	0	...	255
494017	0	tcp	http	SF	282	2286	0	0	0	0	...	255
494018	0	tcp	http	SF	203	1200	0	0	0	0	...	255
494019	0	tcp	http	SF	291	1200	0	0	0	0	...	255
494020	0	tcp	http	SF	219	1234	0	0	0	0	...	255

494021 rows × 42 columns

Figure 2 KDDCUP Dataset

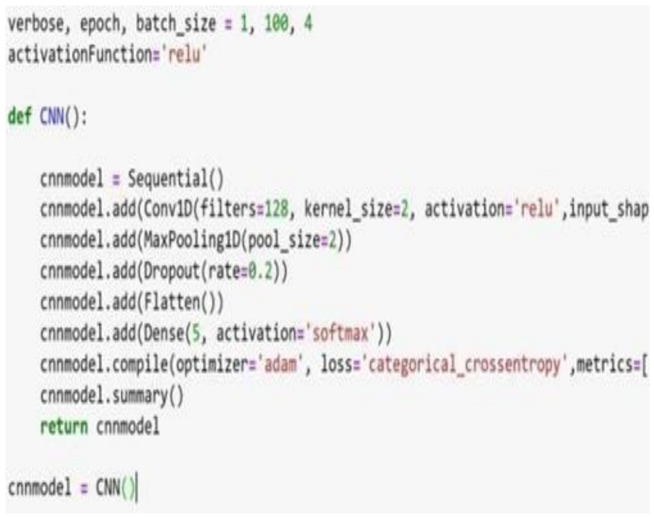
The most well-liked and frequently used datasets in academic research to assess various malicious actions and identify various assaults are KDDCup99, NSL-KDD, and UNSW-NB15. The NSL-KDD dataset is an expansion of the KDD99 dataset that addresses the inadequacies of the previous version [10-13]. Specifically, it aims to decrease redundant data from training and testing while also establishing a record count for training and testing sets (Figure 2). There are 42 features in the dataset, which are categorized into three groups: content features, traffic features, and content features. The KDD Cup 99 dataset is one of the popular datasets in IoT with cybersecurity. This dataset, which comes from the assessment program DARPA98 IDS, contains both labelled and unlabeled training and testing data. It corresponds to seven and two weeks. Perfect Storm (IXIA) and the UNSW Cyber Range Lab collaborated to build the UNSW-NB15 dataset, which is designed to simulate moderately aggressive behaviours and attacks. Process data is information that has been gathered, arranged, cleansed, checked, analyzed, and formatted into easily usable forms like documents or graphs. There are three ways to process data: mechanically, electronically, and manually. Enhancing the value of information and making decision-making easier are the goals. As a result, companies are able to enhance their operations and make critical decisions on time. This is mostly due to automated data processing technologies, such computer software development. It can assist in transforming vast volumes of data, including big data, into insightful understandings for decision-making and quality control.

2.4. Feature Selection

The process of identifying the most reliable, pertinent, and non-redundant features to employ in the creation of a model is known as feature selection. As the quantity and diversity of datasets increase, it is crucial to gradually reduce their size. Reducing modeling's computational cost and enhancing predictive model performance are the primary objectives of feature selection. The act of choosing the most crucial features to include in machine learning algorithms is known as feature selection, and it is one of the key elements of feature engineering. By removing redundant or unnecessary features and condensing the set of features to those that are most pertinent to the machine learning model, feature selection approaches are used to decrease the number of input variables. The primary advantages of selecting features proactively as opposed to relying on the machine learning model to determine their relative importance.

2.5. Algorithms

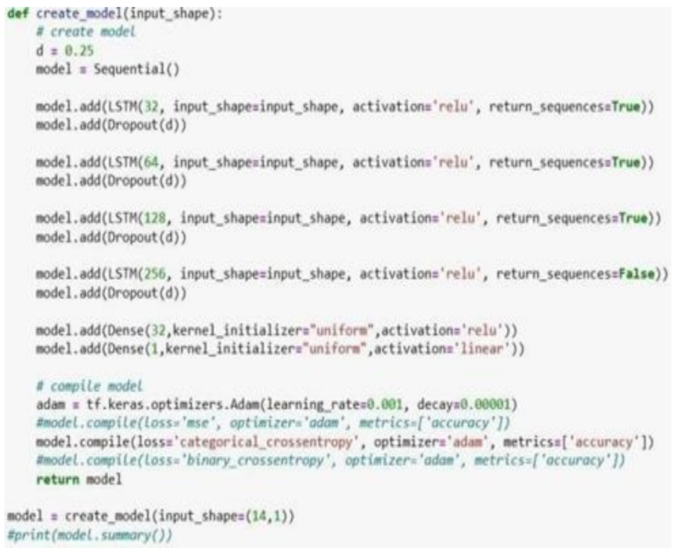
CNNs are specifically designed to handle data that resembles a grid, like photographs [14].To automatically and adaptively learn the spatial hierarchies of features from the input data, they employ convolutional layers (Figure 3). In tasks involving image and video identification, CNNs are frequently utilized.



```
verbose, epoch, batch_size = 1, 100, 4
activationFunction='relu'

def CNN():

    cnnmodel = Sequential()
    cnnmodel.add(Conv1D(filters=128, kernel_size=2, activation='relu',input_shap
    cnnmodel.add(MaxPooling1D(pool_size=2))
    cnnmodel.add(Dropout(rate=0.2))
    cnnmodel.add(Flatten())
    cnnmodel.add(Dense(5, activation='softmax'))
    cnnmodel.compile(optimizer='adam', loss='categorical_crossentropy',metrics=[
    cnnmodel.summary()
    return cnnmodel

cnnmodel = CNN()
```

Figure 3 CNN

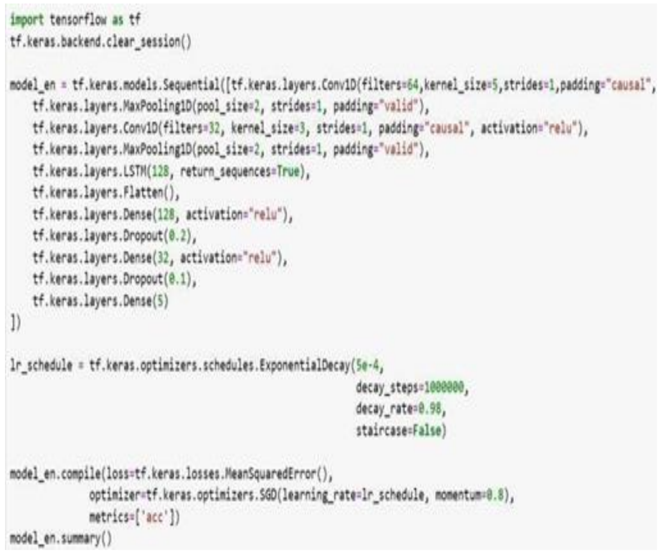
RNNs are designed to work with sequential data by maintaining an internal state or memory [15]. They process inputs in a way that information cycles through a loop, allowing the network to consider previous context (Figure 4). This makes them suitable for tasks involving sequences or time-series data.



```python
def create_model(input_shape):
    # create model
    d = 0.25
    model = Sequential()

    model.add(LSTM(32, input_shape=input_shape, activation='relu', return_sequences=True))
    model.add(Dropout(d))

    model.add(LSTM(64, input_shape=input_shape, activation='relu', return_sequences=True))
    model.add(Dropout(d))

    model.add(LSTM(128, input_shape=input_shape, activation='relu', return_sequences=True))
    model.add(Dropout(d))

    model.add(LSTM(256, input_shape=input_shape, activation='relu', return_sequences=False))
    model.add(Dropout(d))

    model.add(Dense(32,kernel_initializer="uniform",activation='relu'))
    model.add(Dense(1,kernel_initializer="uniform",activation='linear'))

    # compile model
    adam = tf.keras.optimizers.Adam(learning_rate=0.001, decay=0.00001)
    #model.compile(loss='mse', optimizer='adam', metrics=['accuracy'])
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    #model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

model = create_model(input_shape=(14,1))
#print(model.summary())
```

Figure 4 **RNN**

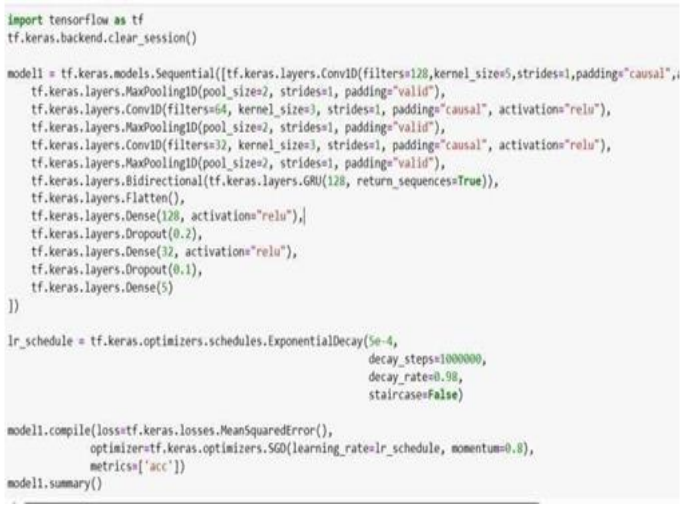
Combining CNN and LSTM leverages: CNN's ability to capture spatial features from data (e.g., images) and LSTM's capability to understand and retain temporal dependencies (Figure 5). This hybrid approach is effective for tasks involving both spatial and sequential data.



```python
import tensorflow as tf
tf.keras.backend.clear_session()

model_en = tf.keras.models.Sequential([tf.keras.layers.Conv1D(filters=64,kernel_size=5,strides=1,padding="causal",
    tf.keras.layers.MaxPooling1D(pool_size=2, strides=1, padding="valid"),
    tf.keras.layers.Conv1D(filters=32, kernel_size=3, strides=1, padding="causal", activation="relu"),
    tf.keras.layers.MaxPooling1D(pool_size=2, strides=1, padding="valid"),
    tf.keras.layers.LSTM(128, return_sequences=True),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation="relu"),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(32, activation="relu"),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(5)
])

lr_schedule = tf.keras.optimizers.schedules.ExponentialDecay(5e-4,
                                                             decay_steps=1000000,
                                                             decay_rate=0.98,
                                                             staircase=False)

model_en.compile(loss=tf.keras.losses.MeanSquaredError(),
              optimizer=tf.keras.optimizers.SGD(learning_rate=lr_schedule, momentum=0.8),
              metrics=['acc'])
model_en.summary()
```

Figure 5 **CNN + LSTM**

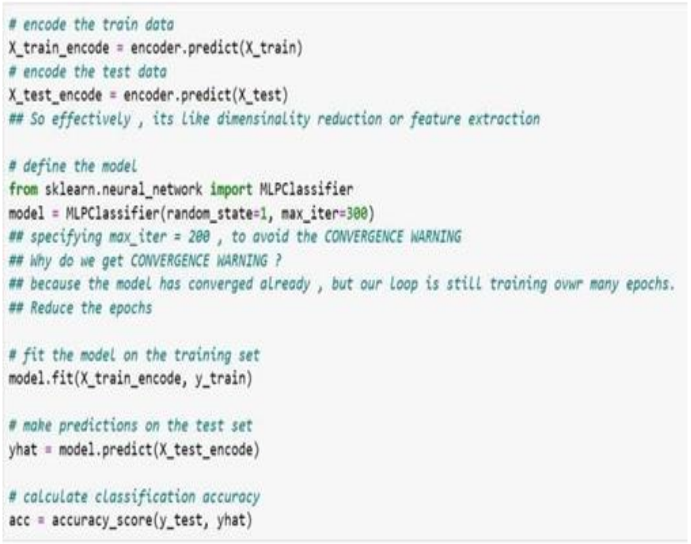
RBM is a generative stochastic artificial neural network used for unsupervised learning. Combining CNN with BiGRU suggests using a mix of convolutional layers for feature extraction (CNN) and bidirectional gated recurrent layers (BiGRU) to capture sequential patterns, potentially for complex pattern recognition tasks (Figure 6).



```python
import tensorflow as tf
tf.keras.backend.clear_session()

model1 = tf.keras.models.Sequential([tf.keras.layers.Conv1D(filters=128,kernel_size=5,strides=1,padding="causal",
    tf.keras.layers.MaxPooling1D(pool_size=2, strides=1, padding="valid"),
    tf.keras.layers.Conv1D(filters=64, kernel_size=3, strides=1, padding="causal", activation="relu"),
    tf.keras.layers.MaxPooling1D(pool_size=2, strides=1, padding="valid"),
    tf.keras.layers.Conv1D(filters=32, kernel_size=3, strides=1, padding="causal", activation="relu"),
    tf.keras.layers.MaxPooling1D(pool_size=2, strides=1, padding="valid"),
    tf.keras.layers.Bidirectional(tf.keras.layers.GRU(128, return_sequences=True)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation="relu"),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(32, activation="relu"),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(5)
])

lr_schedule = tf.keras.optimizers.schedules.ExponentialDecay(5e-4,
                                                             decay_steps=1000000,
                                                             decay_rate=0.98,
                                                             staircase=False)

model1.compile(loss=tf.keras.losses.MeanSquaredError(),
              optimizer=tf.keras.optimizers.SGD(learning_rate=lr_schedule, momentum=0.8),
              metrics=['acc'])
model1.summary()
```

Figure 6 **RBM**

DNNs consist of multiple layers of interconnected nodes, and when organized in a multi-layer perceptron architecture, they are great at learning intricate patterns and features from the data. They are widely used in various machine learning tasks for classification and regression (Figure 7).



```python
# encode the train data
X_train_encode = encoder.predict(X_train)
# encode the test data
X_test_encode = encoder.predict(X_test)
## So effectively , its like dimensinality reduction or feature extraction

# define the model
from sklearn.neural_network import MLPClassifier
model = MLPClassifier(random_state=1, max_iter=300)
## specifying max_iter = 200 , to avoid the CONVERGENCE WARNING
## Why do we get CONVERGENCE WARNING ?
## because the model has converged already , but our loop is still training ovwr many epochs.
## Reduce the epochs

# fit the model on the training set
model.fit(X_train_encode, y_train)

# make predictions on the test set
yhat = model.predict(X_test_encode)

# calculate classification accuracy
acc = accuracy_score(y_test, yhat)
```

Figure 7 **DNN with MLP**

3. Results

Precision: Precision (Figure 8) evaluates the fraction of correctly classified instances or samples Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

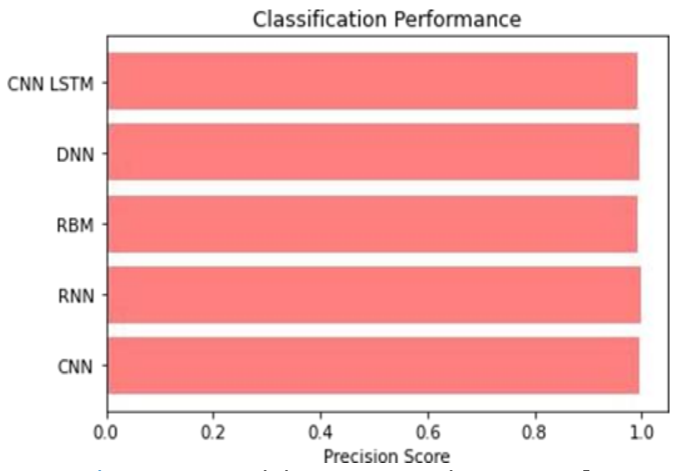


Figure 8 **Precision Comparison Graph**

Recall: In machine learning, recall is a metric that assesses a model's capacity to locate all pertinent instances of a given class. It is the ratio of correctly predicted positive observations to the total actual positives, providing insights into a model's completeness in capturing occurrences of a certain class (Figure 9).

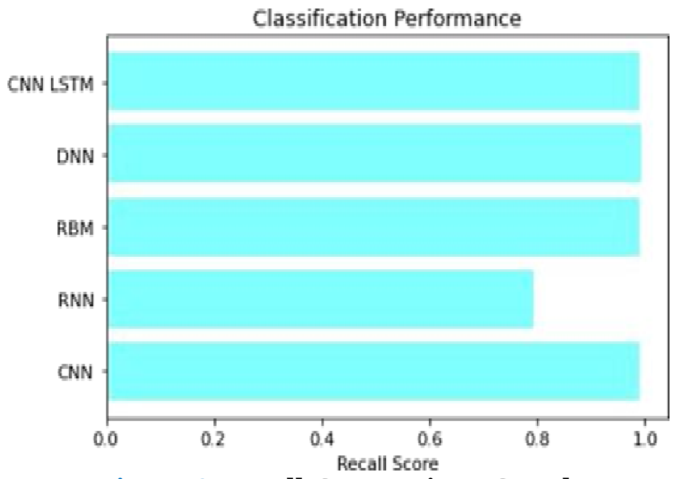


Figure 9 **Recall Comparison Graph**

Accuracy: The percentage of accurate predictions made in a classification task is known as accuracy, and it indicates how accurate a model's predictions are overall (Figure 10).

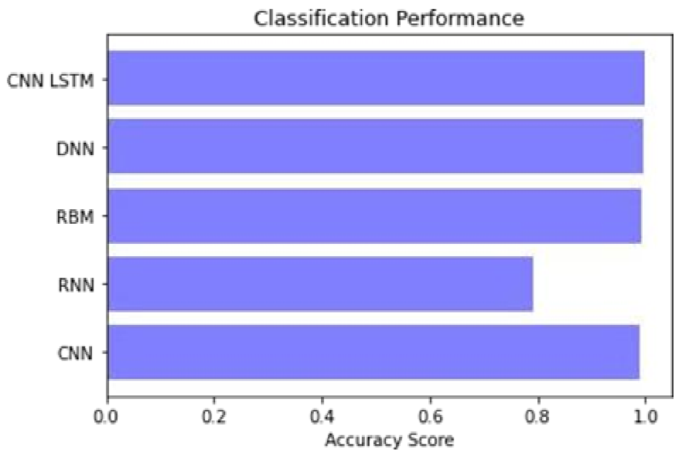


Figure 10 **Accuracy Graph**

F1 Score: The F1 Score is appropriate for imbalanced datasets because it provides a balanced metric that takes into account both false positives and false negatives. It is calculated as the harmonic mean of precision and recall. Refer Figures 11 to 15.

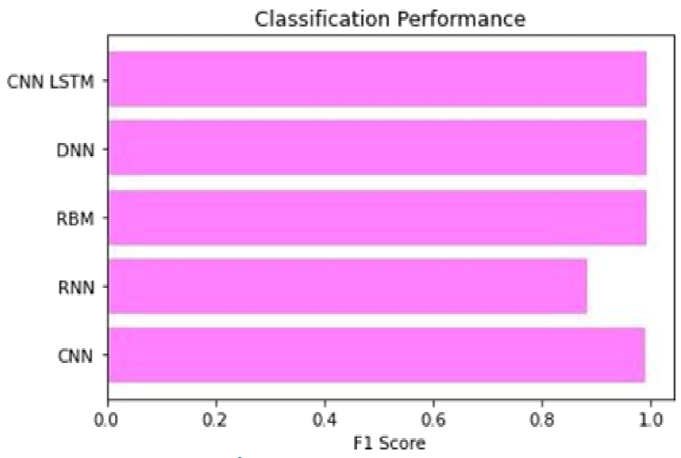
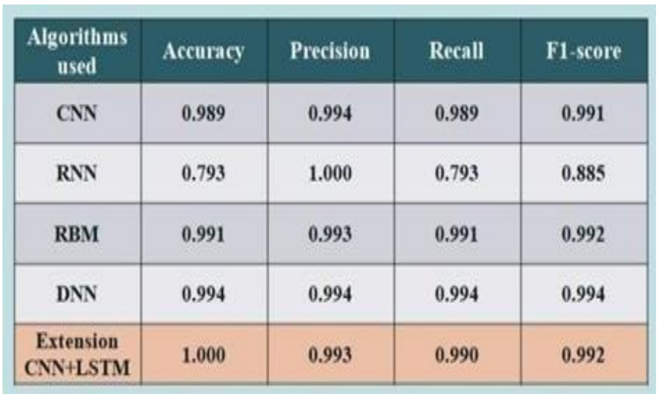


Figure 11 **F1Score**



Algorithms used	Accuracy	Precision	Recall	F1-score
CNN	0.989	0.994	0.989	0.991
RNN	0.793	1.000	0.793	0.885
RBM	0.991	0.993	0.991	0.992
DNN	0.994	0.994	0.994	0.994
Extension CNN+LSTM	1.000	0.993	0.990	0.992

Figure 12 **Performance Evaluation**

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 02 Issue: 10 October 2024
Page No: 2456 - 2462
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2024.0336

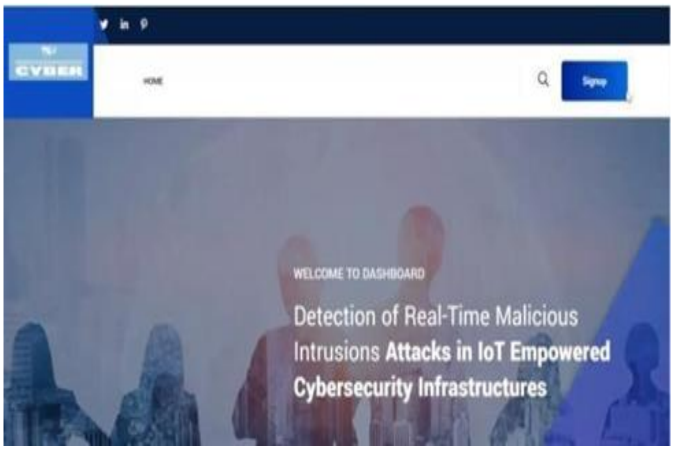

CYBER
HOME
Signup
WELCOME TO DASHBOARD
Detection of Real-Time Malicious Intrusions Attacks in IoT Empowered Cybersecurity Infrastructures

Figure 13 Home Page

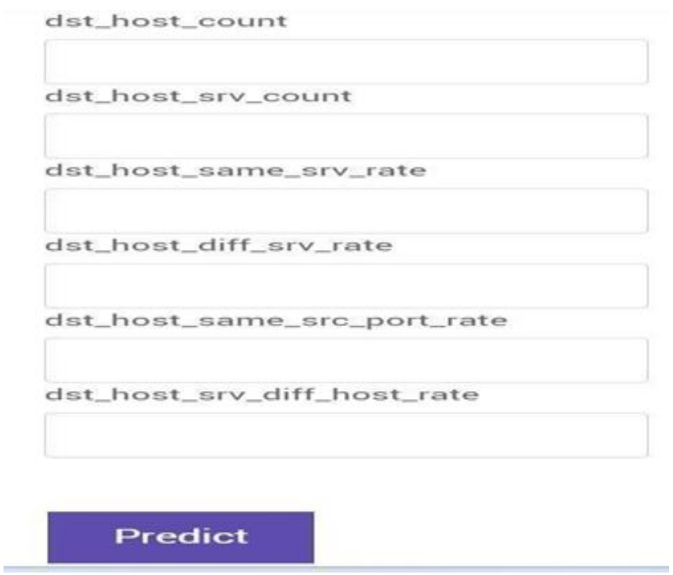
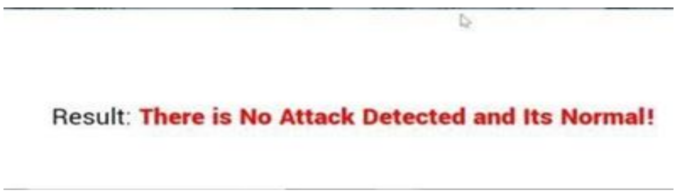


Figure 14 User Input



Result: There is No Attack Detected and Its Normal!

Figure 15 Predict Result for Given Input

Conclusion

The project lays a substantial emphasis on the usefulness of applying deep learning techniques such as Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN) and Deep Neural Networks (DNN) for early detection of cyber-attacks and identification of malware. By leveraging the capabilities of deep learning, the project showcases its potential to significantly enhance cybersecurity measures, providing a proactive approach to identifying and mitigating threats. Among the various models employed, the extension CNN + LSTM ensemble model stands out by achieving an impressive 99% accuracy. This remarkable result underscores the robustness and adaptability of the ensemble model in real-time scenarios. The combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks prove to be a powerful solution for achieving high accuracy in cyber-attack detection, showcasing the effectiveness of ensemble techniques.

References

[1]. Y. LeCun, Y. Bengio, and G. Hinton, ‘‘Deep learning,’’ Nature, vol. 521, no.7553, pp. 436–444, 2015.
[2]. A. Krizhevsky, I. Sutskever, and G. E. Hinton, ‘‘ImageNet classification with deep convolutional neural networks,’’ Commun. ACM, vol. 60, no. 2, pp. 84–90, Jun. 2017.
[3]. M. K. Islam, M. S. Ali, M. M. Ali, M. F. Haque, A. A. Das, M. M. Hossain, D. S. Duranta, and M. A. Rahman, ‘‘Melanoma skin lesions classification using deep convolutional neural network with transfer learning,’’ in Proc. 1st Int. Conf. Arif. Intell. Data Analytics (CAIDA), Apr. 2021.
[4]. A. Ahmim, M. Derdour, and M. A. Ferrag, ‘‘An intrusion detection system based on combining probability predictions of a tree of classifiers,’’ Int. J. Commun. Syst., vol. 31, no. 9, p. e3547, Jun. 2018.
[5]. A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour, and H. Janicke, ‘‘A novel hierarchical intrusion detection system based on decision tree and rules-based models,’’ in Proc. 15th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS), May 2019, pp. 228–233.
[6]. Z. Dewa and L. A. Maglaras, ‘‘Data mining and intrusion detection systems. J. Adv. Comput. Sci. Appl., vol. 7, no. 1, pp. 1–10, 2016.
[7]. B. Stewart, L. Rosa, L. A. Maglaras, T. J. Cruz, M. A. Ferrag, P. Simoes, and H. Janicke, ‘‘A novel intrusion detection

mechanism for SCADA systems which automatically adapts to network topology changes,'' EAI Endorsed Trans. Ind. Netw. Intell. Syst., vol. 4, no. 10, p. e4, 2017.

[8]. M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, ''Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study,'' J. Inf. Secur. Appl., vol. 50, Feb. 2020, Art. no. 102419.

[9]. Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, ''A bidirectional LSTM deep learning approach for intrusion detection,'' Expert Syst. Appl., vol. 185, Dec. 2021, Art. no. 115524.

[10]. A. A. Salih, S. Y. Ameen, S. R. Zeebaree, M. A. Sadeeq, S. F. Kak, N. Omar, I. M. Ibrahim, H. M. Yasin, Z. N. Rashid, and Z. S. Ageed, ''Deep learning approaches for intrusion detection,'' Asian J. Res. Comput. Sci., vol. 9, no. 4, pp. 50–64, 2021.

[11]. J. Azevedo and F. Portela, ''Convolutional neural network—A practical case study,'' in Proc. Int. Conf. Inf. Technol. Appl. Singapore: Springer, 2022, pp. 307–318.

[12]. K. He, X. Zhang, S. Ren, and J. Sun, ''Deep residual learning for image recognition,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognin. (CVPR), Jun. 2016, pp. 770–778.

[13]. J. Kosinski, J. Clune, Y. Bengio, and H. Lipson, ''How transferable are features in deep neural networks?'' in Proc. Adv. Neural Inf. Process. Syst., vol. 27, 2014, pp. 1–9.

[14]. G. Awed, C. G. Snoek, A. F. Smeaton, and G. Quénot, ''Trecvid semantic indexing of video: A 6-year retrospective,'' ITE Trans. Media Technol. Appl., vol. 4, no. 3, pp. 187–208, 2016.

[15]. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, ''Rethinking the inception architecture for computer vision,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2818–2826