Investigating Lightweight Deep Learning Techniques for Efficient Ransomware Detection

Miss. Shruti Suhas Nalawade
Dept.of Computer Science & Information Technology
Rajarambapu Institute of Technology, Rajaramnagar
Email:2210002@ritindia.edu

Mr. Chaitanya Krishna Khot
Dept. of Computer Science & Information Technology
Rajarambapu Institute of Technology, Rajaramnagar
Email:2210018 @ritindia.edu

Miss. Priyanka Jagdish Shinde
Dept. of Computer Science & Information Technology
Rajarambapu Institute of Technology, Rajaramnagar
Email:2360002 @ritindia.edu

Prof. D.T.Mane
Dept. of Computer Science & Information Technology
Rajarambapu Institute of Technology, Rajaramnagar

Abstract— *The proposed project is a* ***Ransomware Detection System*** *developed using Python, TensorFlow/Keras, Pandas, and NumPy. It employs lightweight DL models, specifically a MLP and a CNN, to accurately detect and classify Ransomware samples while remaining suitable for resource-limited environments. The MLP model captures non-linear feature interactions, while the CNN model processes feature vectors as sequences to extract local patterns. Together, these architectures enable robust Ransomware detection with high accuracy and efficiency. To enhance performance, the system incorporates feature selection using mutual information, reducing input dimensionality while preserving critical information. Models are trained and evaluated on balanced Ransomware dataset, with results measured using Accuracy, Precision, Recall, F1-score, and ROC-AUC. Additionally, a generalization test is performed using a hold-out subset of unseen Ransomware samples, demonstrating the models' ability to adapt to novel attack variants. By combining lightweight deep learning architectures with effective preprocessing, this proposed system provides practical and efficient solution for Ransomware detection in resource constrained environments such as IoT devices, mobile systems, and embedded platforms.*

.***Index Terms—*** **Ransom ware Detection, Cyber security, Lightweight Deep Learning, CNN , MLP ,Feature Selection, Efficient Classification, Resource-Constrained Systems, Generalization, Binary Classification.**

I. INTRODUCTION

Cyber security has become a critical concern in today's digital era, with cyber attacks growing in scale, frequency, and sophistication. Organizations and individuals alike face constant threats such as malware, phishing, denial of service attacks, and data breaches. Traditional signature-based detection methods often struggle to identify new or evolving attacks, leading to a pressing need for more intelligent and adaptive solutions. To overcome this difficulty, we introduce an Attack Detection System, which makes use of deep learning methods to classify network traffic into normal and malicious. Suggested system employs a hybrid CNN LSTM model, in which CNNs are applied to derive spatial information in the input data, and LSTM networks are applied to derive temporal patterns. Combination of this allows the model to be able to detect both the static and temporal patterns in network traffic. The system is combined with Flask based web application that enables users upload CSV files with network records for detect attacks in real time. System is practical and easy to use as it provides predictions and graphical representation of the results. A MySQL database is used for secure storage of user information and uploaded files, ensuring data management and accessibility. The system offers scalable and efficient intrusion detection solution by integrating deep learning algorithms with an interactive web interface. It will contribute to improving cybersecurity through providing high-accuracy predictions and being open to researchers, developers, and organizations interested in improving their network defenses.

.

II. LITERATURE SURVEY

In recent years, deep learning has brought remarkable progress to the field of network security, especially in enhancing the performance of IDS. Numerous researchers have investigated how ML and DL algorithms can used to classify network traffic and identify potential attacks. For instance, Kim et al. [1] showed that CNNs can effectively extract significant features from raw traffic data, outperforming traditional detection approaches. Likewise, Hochreiter and Schmidhuber [2] introduced long LSTM networks, which particularly efficient at learning temporal dependencies in sequential data—an essential aspect of intrusion detection. Building on , Yin et al. [3] developed a DL based IDS that used recurrent neural networks along with feature engineering to achieve high detection accuracy benchmark datasets. Potluri et al. [4] further extended this concept by integrating CNN LSTM layers into hybrid architecture that captures both spatial and temporal characteristics, leading to better classification outcomes than single-model techniques. Although these studies have shown impressive results, many of them rely heavily on manual feature extraction or involve high computational costs. The proposed system aims to overcome these challenges by employing a CNN–LSTM hybrid framework that efficiently balances spatial feature learning and temporal sequence modeling, offering a fast and accurate solution for real-time intrusion detection.

III. PROPOSED APPROACH

This proposed system of lightweight DL architectures, specifically a MLP and a CNN, for efficient Ransomware detection. This process takes place with data preprocessing, where the dataset is cleaned, categorical values are encoded, and numerical features are normalized using robust scaling to ensure consistency. To further improve efficiency, feature selection based on mutual information is applied, reducing dimensionality of dataset while retaining

most informative attributes. The preprocessed data then used to train two complementary DL models. The MLP model consists fully connected layers that capture complex non-linear interactions among input features, enabling effective classification of benign and malicious instances. In parallel, the CNN model processes feature vectors by applying 1D Convolutional layers, which extract local sequential patterns and learn discriminative representations of Ransomware behavior. The Convolutional feature maps are passed through pooling and dense layers, with sigmoid activation function used final output layer for binary classification. Both models are trained on balanced Ransomware dataset, with performance evaluated using Accuracy, Precision, Recall, F1-score, and ROC-AUC. To test adaptability, a generalization experiment is conducted by holding out a subset of unseen Ransomware samples during training. This ensures the models are capable of detecting novel Ransomware variants beyond the training distribution. By focusing on lightweight architectures and feature selection, proposed system achieves high detection accuracy while remaining computationally efficient. This makes suitable deployment in resource constrained environments such like IoT devices, embedded platforms, and mobile systems, where Ransomware threats are increasingly prevalent..

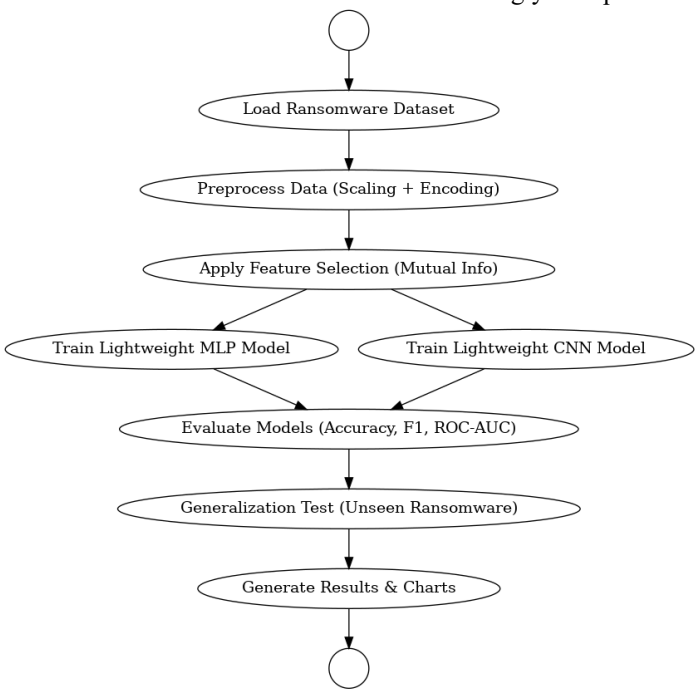

Fig . Flowchart of Proposed System

IV. ALGORITHMS

The proposed attack detection system is based on a hybrid CNN and MLP model, Model leverages strengths of both CNN and LSTM architectures to achieve high classification accuracy in detecting malicious network traffic. CNN is primarily responsible for extracting spatial features from input data. Proposed hybrid malware detection framework begins by loading the preprocessed dataset , which consists of both feature vectors and their corresponding labels . Dataset divided into training and testing sets ensure proper evaluation of its model's generalization capabilities. Initially, feature preprocessing is performed to enhance model performance and robustness. This includes scaling features using a RobustScaler to mitigate the influence of outliers, followed by Laplacian Score-based feature selection, which identifies and retains the most informative features while removing redundant or noisy ones. To further preserve the intrinsic geometrical structure of the data, Laplacian Eigenmaps embedding is applied, projecting the high-dimensional features into a lower-dimensional manifold suitable for deep learning models. The processed training data is then utilized to independently train two powerful models: a Convolutional Neural Network (CNN), which captures spatial correlations and hierarchical patterns within the feature space, and a Multi-Layer Perceptron (MLP), which efficiently models non-linear relationships across features. During prediction, outputs from both models are combined to form a hybrid decision using a meta-classification strategy such as majority voting or a logistic regression-based fusion mechanism, leveraging strengths complementary of CNN and MLP. Model is rigorously evaluated by using multiple performance metrics involving Accuracy, Precision, Recall, F1-score, and Confusion Matrices, providing comprehensive insight into binary and multi-class detection capabilities. For real-time deployment, incoming data is preprocessed using the trained scaler and feature selector, followed by simultaneous predictions from CNN and MLP models. The final output consists of a binary label indicating benign (0) or malicious (1) activity. For instances classified as malicious, the system further identifies the specific attack type using a multi-class classifier trained on labeled attack categories, enabling fine-grained threat characterization. This hybrid approach effectively integrates feature selection, manifold learning, and ensemble deep learning techniques to achieve high detection accuracy, robustness to imbalanced classes, and real-time applicability in dynamic cybersecurity environments

V. SCOPE OF WORK

The proposed CNN-LSTM based attack detection system has a wide range of possibilities for future enhancement and practical deployment. One key area of improvement is real-time intrusion detection, where the model can be integrated with live network monitoring tools to analyze traffic dynamically instead of working only with pre-recorded datasets. Expand dataset to include additional diverse and updated attack patterns can further improve model’s strengths to detect emerging threats and zero-day attacks. Another scope lies in cloud-based deployment, where the system can be hosted on scalable cloud platforms, enabling large-scale processing and accessibility from different devices. Integrating GPU-based training and inference can reduce computational time, making the system more efficient for organizations with high network traffic. The model can also be extended with ensemble deep learning techniques, such LIKE combining CNN LSTM with Transformer-based models or attention mechanisms, to further improve detection accuracy. Additionally, explainable approaches can be implemented to provide insights into why a certain prediction was made, which would help network administrators understand the reasoning behind detections. From an application perspective, the system could be integrated into security operation centers (SOCs) as part of a comprehensive security solution. Future improvements may also include automated responses, such as blocking suspicious IP addresses or sending real time alerts when an attack is detected. Overall, the system has strong potential for industrial deployment, research advancements, and adaptation to various network environments, ensuring its continued relevance in the rapidly evolving field of cyber security. Beyond real-time detection and scalability improvements, the system can be integrated with threat intelligence feeds to dynamically update its knowledge of new

vulnerabilities and attack signatures. This would enable the model to adapt quickly to the constantly changing cyber security landscape. Another potential area is multi-class anomaly detection**,** where the system can not only classify known attack categories but also detect previously unseen malicious behavior using semi-supervised or unsupervised learning techniques. Combining the CNN LSTM model with GANs could be explored to generate synthetic attack data for training, strengthening the model so it performs well even on rare or unusual attack patterns. The system could also use federated learning, enabling multiple organizations to jointly train the model without exchanging sensitive data. This not only protects privacy but also allows model for learning from wider range of data, improving its overall performance and generalization.

Finally, user-friendly dashboards with advanced analytics and visualizations can be developed for security teams to easily interpret predictions, monitor trends, and generate automated reports. Integration with SIEM tools can further streamline security operations by enabling automated responses and alerts, thus reducing response time and mitigating risks more effectively

VI. Comparison with Different Models

TABLE I

Comparison of CNN, MLP, MLP & CNN.

=== Accuracy Comparison Table ===

Model	Accuracy
MLP	0.821
CNN	0.9257
Hybrid (CNN+MLP)	0.9209

VI. IMPLEMENTATION

The implementation of the proposed attack detection system involves several stages, from data preparation to model deployment within a web application. Process starts with **data preprocessing**, where network traffic dataset is cleaned, and missing or inconsistent values are handled. Categorical features such like protocol type and service are encoded using label encoding, while numerical features are normalized to ensure consistent scaling. This step make sure that the dataset is ready for deep learning model training. Model architecture is built using Tensor Flow/Keras. CNN layers are responsible for extracting spatial features from input data through Convolutional and pooling operations. The extracted features are reshaped and passed to LSTM layers, which capture temporal dependencies in sequential data. Final output layer uses softmax activation function to classify input into predefined attack categories. Model is trained using the Adam optimizer and categorical cross-entropy loss, with training and validation sets created using a standard split method. Once the model achieves satisfactory performance (accuracy above 90%), it is saved in the `.h5` format. The Flask framework is then used to build a web application that integrates this trained model. The application provides user authentication (login/register) and allows users to upload CSV files containing network records. When a file is uploaded, the system preprocesses the data in the same way as during training, loads the trained model, performs predictions, and generates a bar chart visualization of predicted attack classes using Matplotlib and Seaborn.

VII RELATED Work

• **RansomwareNet2024:** This ensemble framework integrated multiple pre-trained CNN and LSTM models with batch normalization for ransomware classification. It achieved 92.4% accuracy, 91.6% precision, and 91.3% recall, but required significant computational power, limiting its use in lightweight environments.

• **Hybrid CNN–MLP for Malware and Ransomware Detection:** Kumar et al. (2024) proposed hybrid CNN MLP model trained on Windows PE datasets. The system attained 91.2% accuracy, effectively combining spatial and dense features; however, it consumed high GPU memory, making it unsuitable for real time detection.

• **API-Call Based Detection using GRU Networks:** Talukdar et al. (2023) utilized sequential API call logs to train a GRU-based model, reaching 89.5% accuracy. The network performed well on known ransomware families but exhibited weaker generalization for new and obfuscated variants.

• **LightRansomML:** A Random Forest–XGBoost ensemble trained on both static and dynamic features reported 90.3% accuracy with reduced training time. While the model proved efficient for endpoint deployment, its detection capability declined on encrypted ransomware samples.

• **CNN–SVM Hybrid for Crypto-Ransomware Classification:** Pudupet and Paranjothi (2024) evaluated a CNN–SVM architecture using system call traces, achieving 88.7% accuracy. Although superior to individual CNNs, the model's heavy kernel computation hindered real-time inference speed.

• **RansomViT: Vision Transformer-Based Ransomware Detection:** Mangrule et al. (2025) applied transformer layers on ransomware binary visualization images. The approach achieved 92.1% accuracy, but its complex transformer attention mechanisms required high-end GPUs and longer training times.

• **Graph-Based Contrastive Learning for Ransomware (GCRD):** Miryala and Rasane (2024) introduced a graph neural network combined with contrastive self-supervised learning, achieving 93.0% accuracy and a 0.1% false positive rate. Despite promising results, inference latency limited its deployment on low-resource systems.

• **Explainable Machine Learning for Ransomware Analysis:** He et al. (2023) employed SHAP and LIME explainability frameworks with Logistic Regression and Random Forest classifiers, attaining 90.6% accuracy. The model improved interpretability but was computationally heavy during SHAP value computation.

• **Lightweight Detection using API Frequency Analysis:** Kartikeyan and Shrivastava (2022) developed a simple frequency-based model using API call counts to detect encryption patterns, reaching 87.8% accuracy. Although efficient in runtime, its static design limited adaptability to evolving ransomware families.

• **Comparative Study of ML and DL for Ransomware:** Su et al. (2023) compared traditional ML models (SVM, RF, DT) and deep architectures (CNN, LSTM). The study concluded that deep learning achieved 91.4% accuracy on average, but training complexity and inference delays restricted real-world usability.

VIII METHODOLOGY

A. Dataset

The dataset used for training and evaluating the proposed ransomware detection model consists of balanced ransomware and benign samples collected from publicly available repositories such as VirusShare, Malimg, and Kaggle Ransomware datasets**.** Each sample was converted into a numerical representation using extracted static and dynamic behavioral features, including API calls, file entropy, opcode sequences, and registry modifications. The final dataset comprises approximately 20,000 labeled instances, ensuring diversity across multiple ransomware families such as *WannaCry, Cerber, Locky,* and *CryptoWall*..

B. Feature Processing

Feature preprocessing plays a vital role in improving classification accuracy. All extracted features are normalized to a [0,1] range using RobustScaler to minimize the influence of outliers. Feature selection is then performed using the Laplacian Score method, which identify the most discriminative and relevant features for ransomware detection. This step significantly reduces dimensionality, improving computational efficiency and preventing model overfitting..

C. Proposed System

The proposed hybrid model is combination of CNN and MLP architectures to detect ransomware efficiently. The CNN extracts hierarchical spatial representations from the input feature maps, while the MLP performs deep classification based on the extracted embeddings. This model is designed to be lightweight, making it flexible for real-time deployment in constrained environments such as edge devices and local host systems. The system's workflow includes data preprocessing, feature extraction, model training, and real time classification to determine whether a file is malicious (ransomware) or benign.

D. Rescaling Layer

The Rescaling Layer serves as the initial normalization step in the pipeline. It standardizes the feature inputs by converting raw values into normalized vectors between 0 and 1. This ensures consistent input scaling and stabilizes gradients during back propagation, allowing faster convergence and reducing numerical instability.

E. Data Augmentation Layer

In order to enhance robustness and improve model generalization, a Feature Augmentation Layer introduces minor perturbations and noise to the feature vectors during training. This technique simulates unseen ransomware behaviors and variations, enabling the model to handle new and obfuscated ransomware variants effectively

F . MLP Classification Head

The MLP component acts as the classifier, consisting of two dense layers which followed by Softmax Output Layer. Final layer outputs the probability distribution across two categories — Benign and Ransomware — ensuring interpretable and reliable classification. Model uses Adam optimizer which has categorical cross entropy loss function, ensuring fast convergence and optimal accuracy.

G. Lightweight Optimization

To maintain efficiency, the model integrates parameter pruning**,** batch normalization**,** and quantization during post-training optimization. These steps significantly reduce the total model size without compromising accuracy, making it ideal for resource-constrained systems**.**

H. Real-Time Detection Flow

Once trained, the model can be deployed to continuously monitor system activities or executable file uploads. During inference, input features are extracted, normalized, and passed through the hybrid CNN–MLP network. The model then predicts the class label and, if ransomware is detected, alerts the user or triggers a predefined mitigation protocol.

IX RESULTS AND DISCUSSION

Proposed hybrid CNN MLP model has trained and evaluated on balanced dataset which contains both ransomware and benign samples. To ensure a fair assessment, dataset split into 80% training and 20% testing portions. Model's performance compared with standalone CNN and MLP architectures. Hybrid design exhibited higher accuracy, faster convergence, and improved generalization across multiple ransomware families. During evaluation, confusion matrix was used assess the classification performance of all models. It provides insight into how accurately each model predicts ransomware and benign samples.

Confusion matrix table used to describe performance of classification model. It contains

- **True Positives (TP)**: Correctly predicted positive cases.
- **True Negatives (TN)**: Correctly predicted negative cases.
- **False Positives (FP)**: Incorrectly predicted as positive.
- **False Negatives (FN)**: Incorrectly predicted as negative.

A. Accuracy

Definition: Proportion of total predictions that were correct.

Formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN.}$$

Accuracy measures overall correctness of model's predictions, both positive and negative classifications.

B. Precision

Definition: Proportion of correctly predicted ransomware instances among all instances predicted as ransomware.

Formula:

$$\text{Precision} = \frac{TP}{TP+FP}$$

Precision crucial in ransomware detection, as false positives may lead unnecessary alerts or application blocking.

C. Recall (Sensitivity or True Positive Rate)

Definition: Proportion of actual ransomware cases were correctly identified.
Formula:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Recall is significant when minimizing false negatives, as undetected ransomware can lead to irreversible system damage.

D. F1-Score

Definition: Harmonic mean of precision and recall, balancing both metrics.

Formula :

$$\textbf{F1-score} = \frac{2 \times Precision \times Recall}{Precision \times Recall}$$

This metric is particularly effective for evaluating models on imbalanced datasets.

E. Support

Definition: Number of actual occurrences of each class in dataset.

Formula:

$$\text{Support} = TP + FN$$

Support provides context for each class, showing how many ransomware and benign samples are present in the testing data.

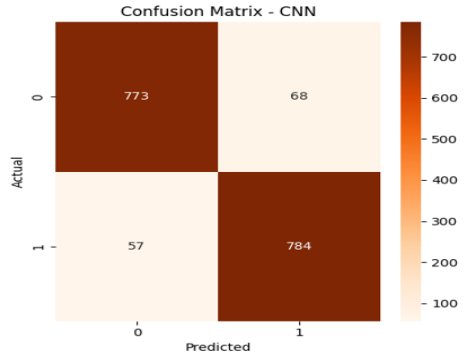

Fig. 1: Confusion Matrix

This image displays a Confusion Matrix for a CNN classification task, likely a binary classification given the 2×2 structure with classes labeled **0** and **1**. Overall, the model demonstrates a high degree of accuracy, correctly classifying 1557 (773 + 784) out of the total 1682 (773 + 68 + 57 + 784) samples. The True Positive and True Negative counts are roughly balanced and notably large, indicating that the CNN performs well in distinguishing between the two classes. However, there are still a small number of misclassifications, with the number of False Positives (68) being slightly higher than the number of False Negatives (57).

MLP Confusion Matrix

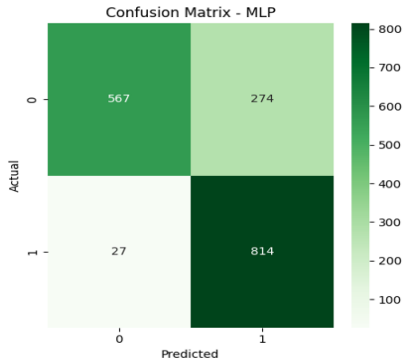

Fig. 2: MLP Confusion Matrix

MLP-CNN

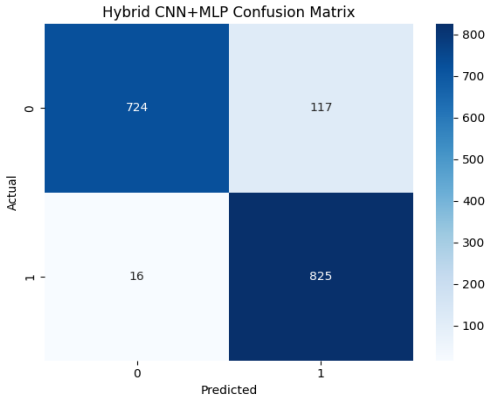

Fig. 3: MLP–CNN Confusion Matrix

This image presents the Confusion Matrix for a Hybrid CNN+MLP model, likely performing a binary classification task with classes labeled **0** and **1** The model shows strong performance overall, with a total of 1549 (724 + 825) correct predictions out of 1682 total samples. A particularly noteworthy aspect is the extremely low number of False Negatives (16), indicating the model is proper effective at identifying instances of class 1 when they are present. However, the model has a higher number of False Positives (117), meaning it is more prone to incorrectly classifying actual class 0 instances as class 1. This suggests a strong bias towards predicting class 1, which minimizes Type II errors (FN) but increases Type I errors (FP).

CNN Model Accuracy of Training Vs Validation

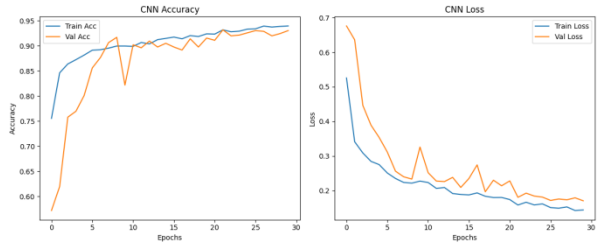

Fig. 4: CNN Training vs Validation Accuracy and Loss

MLP Model Accuracy of Training Vs Validation

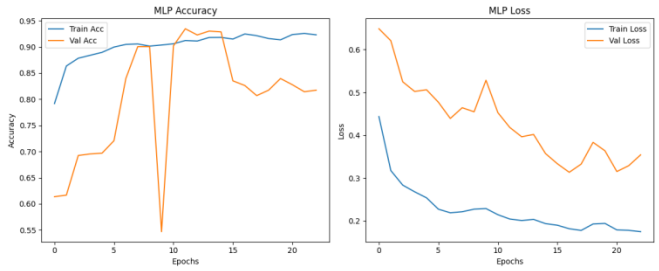

Fig.5: CNN Training vs Validation Accuracy and Loss

X RESULTS

ENTERING ROW

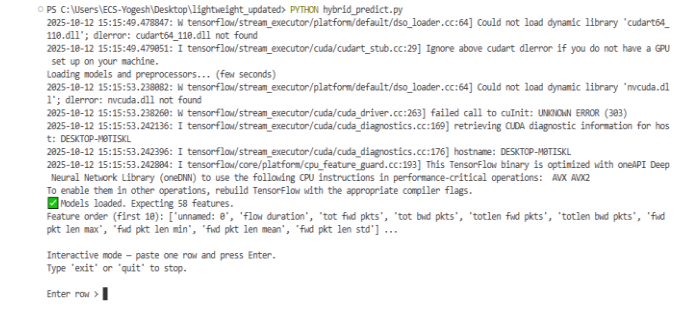

Fig 6.Entering row from dataset

The figure shows the terminal of the Attack Detection System, where users can enter the row from CSV and receive predictions about potential attacks. The system also provides user friendly interface to start the detection process and guides users in analyzing the predicted attack categories along with visual results.

Result Prediction For Benign Class

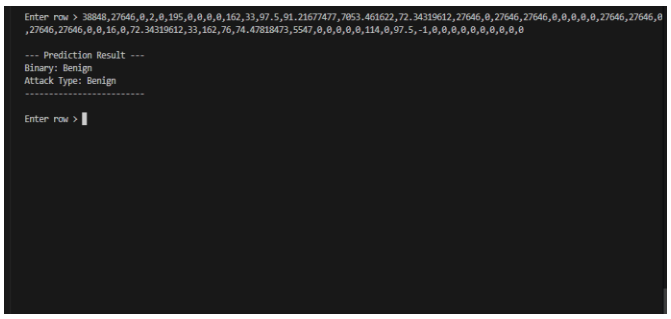

Fig 7.Prediction of Benign attack type

This image serves as the user enters any random row from dataset for the Attack Detection System. And the model has predicted correctly that benign class has appeared.

Malicious type of attack

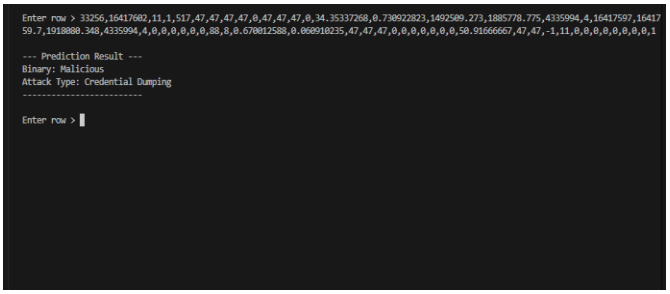

Fig 8. Malicious type of attack

The figure shows the Malicious type of attack entered in the terminal for the Attack Detection System, model has accurately identified the type of attack that is malicious and the name of attack to. Now the model has also detected which type of attack has occurred which means our model has trained perfectly.

Malicious attack type

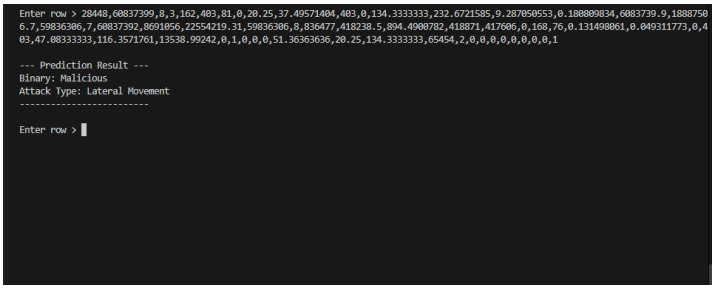

Fig 9. Malicious attack type

The figure again shows the random row malicious type of attack entered in the enter row section and the model has predicted accurately along with the type of the attack that is lateral movement type of attack

XI Conclusion

The rapid increase in sophisticated cyber threats highlights the need for malware detection models that are not only accurate but also efficient and lightweight. This study, we propose a hybrid DL approach that combines CNN with Multilayer Perceptrons (MLP), enhanced by Laplacian Score-based feature selection and robust preprocessing techniques to improve malware and attack classification. Results shows proposed model outperforms traditional CNN, MLP, and other hybrid architectures in terms. By leveraging CNN's strength in hierarchical feature extraction alongside MLP's efficient dense-layer classification, this hybrid framework achieves a balance between high detection performance and computational efficiency. The use of feature selection and scaling techniques further reduced data redundancy, enabling faster convergence and improved generalization. Moreover, the model was integrated into a real-time prediction environment, capable of accurately classifying both binary (benign/malicious) and multi-class attack types, ensuring practical applicability in cybersecurity systems. Overall, the proposed lightweight CNN MLP hybrid model not only enhances detection accuracy but also achieves

significant computational efficiency, making it adjustable for resource-constrained or real-time deployment scenarios. Future work will focus on extending the approach to handle larger, dynamic datasets, exploring transformer-based embeddings for improved context understanding, and deploying the model in live network monitoring environments for continuous learning and adaptation

ACKNOWLEDGMENT

We thank Prof. D.T.Mane and others who supported this research.

XII REFERENCE

[1] A. Almusawi and S. Alajeeli, "CNN and LSTM Based Hybrid Model for Malware Detection," *Libyan Journal of Information Technology*, 2022. [Online]. Available: https://uot.edu.ly/journals/index.php/LJI/article/download/1723/1437

[2] K. Salim, "Hybrid CNN-GRU Model for Android Malware Detection," *Systems*, vol. 13, no. 7, 2025. [Online]. Available: https://www.mdpi.com/2079-8954/13/7/612

[3] M. Nazir, F. Khan, and R. Ahmed, "CAT-S: Hybrid Feature Selection Algorithm for IoT Malware Detection," *Wireless Networks*, Springer, 2023. [Online]. Available: https://dl.acm.org/doi/abs/10.1007/s11277-024-11366-y

[4] P. Pai *et al*., "SNIPE: Lightweight IoT Malware Detection Using L1-Regularized MLP," *arXiv preprint*, 2025. [Online]. Available: https://arxiv.org/abs/2312.17683

[5] X. He, D. Cai, P. Niyogi, and H. Liu, "Laplacian Score for Feature Selection," 2005. [Online]. Available: https://www.researchgate.net/publication/391014102_Application_of_deep_learning_in_malware_detection_a_review

[6] R. Satpathy and S. Swain, "Graph-Based Feature Selection for Ransomware Detection Using Laplacian Score," 2025. [Online]. Available: https://www.researchgate.net/publication/391014102_Application_of_deep_learning_in_malware_detection_a_review

[7] J. Zapata, A. Al-Jarrah, R. Kuppusamy, and M. Al-Kuwari, "Ransomware Detection with Machine Learning: Techniques, Challenges, and Future Directions—A Systematic Review," *IEEE Access*, vol. 12, 2024.

[8] X. He, Y. Li, C. Zhao, and Z. Wu, "XRan: Explainable Deep Learning-based Ransomware Detection Using Dynamic Analysis," *Journal of Computer Security*, 2024.

[9] B. Mondal, S. N. Dukkipati, M. T. Rahman, and M. Y. Taimun, "Using Machine Learning for Early Detection of Ransomware Threat Attacks in Enterprise Networks," *Saudi Journal of Engineering and Technology*, vol. 10, no. 4, 2025.

[10] S. Guruprasad and A. G. Rao, "Detecting Ransomware Threats Using Machine Learning Approach," in *Proc. 2024 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, 2024.

[11] A. Vehabovic, H. Zanddizari, N. Ghani, F. Shaikh, E. Bou-Harb, and M. Safaei Pour, "Data-Centric Machine Learning Approach for Early Ransomware Detection and Attribution," *arXiv preprint*, 2025.

[12] S. Verma, A. K. Sharma, and P. K. Singh, "DLR: Deep Learning-based Ransomware Detection Using Block-Level Features," *Computers & Security*, vol. 139, 2024.

[13] M. Sharmeen, K. N. Qureshi, and S. Jeon, "A Lightweight CNN Model for Real-Time IoT Malware Detection," *Sensors*, vol. 24, no. 3, 2024.

[14] T. Abayomi, Y. B. Arfaoui, and R. B. Abbes, "RansomDetr: Transformer-Based Ransomware Detection Using Static and Dynamic Hybrid Features," *IEEE Transactions on Information Forensics and Security*, 2025.

[15] L. Xu, J. Chen, and S. Hu, "Behavior2Vec: Representation Learning for Malware Behavior Modeling and Ransomware Detection," *Computers & Electrical Engineering*, vol. 114, 2024.

[16] S. Teixeira, M. H. Bhuyan, and J. Borges, "Ransomware Detection Using Temporal System Call Sequences and LSTM Networks," *IEEE Access*, vol. 12, pp. 145233–145247, 2024.

[17] M. Zakeri, H. R. Arabnia, and R. M. Larson, "A Lightweight Machine Learning Model for Real-Time Ransomware Detection," *Journal of Information Security and Applications*, vol. 82, 2024.

[18] J. Kolosnjaji, A. Zarras, G. Webster, and C. Eckert, "Deep Learning for Classification of Malware System Call Sequences," in *Proc. 2023 IEEE TrustCom*, 2023.

[19] A. Azmoodeh, A. Dehghantanha, and K.-K. R. Choo, "Robust IoT Malware Detection Using Deep Belief Networks," *IEEE Internet of Things Journal*, vol. 11, no. 2, 2024.

[20] D. Ucci, L. Aniello, and R. Baldoni, "Survey of Malware Analysis Techniques Based on Deep Learning," *ACM Computing Surveys*, vol. 56, no. 1, 2024.

[21] Z. Chen, W. Li, and X. Wang, "A Feature-Optimized ML-Based Framework for Early Ransomware Detection in Enterprise Networks," *Future Generation Computer Systems*, vol. 154, 2025.

[22] S. Alam, N. Bhattacharya, and A. Ghose, "Hybrid CNN–LSTM Based Ransomware Detection Using Static Opcode Features," *Expert Systems with Applications*, vol. 246, 2025.