INNOVATIVE INTRUSION DETECTION FOR IOT: A RESOURCE-EFFICIENT NEURAL NETWORK APPROACH

Vijay Raj[1], Sree Malik Goud Sanem[2], Swetha Nadimishetty[2], S. Sai kiran[2]

[2]UG Scholar, [1,2]Department of Computer Science and Engineering

[1,2]Kommuri Pratap Reddy Institute of Technology, Ghatkesar, Hyderabad, Telangana.

ABSTRACT

In recent years, the increasing reliance on wireless networks has made them a prime target for cyberattacks. Intrusion detection systems (IDS) are critical for identifying and mitigating such threats. Traditional IDS methods have limitations, and leveraging advanced techniques like deep learning and synthetic data generation can enhance the effectiveness of intrusion detection. Traditional wireless network intrusion detection systems often rely on rule-based algorithms or signature-based detection methods. While effective to some extent, these methods may struggle with detecting novel or evolving attack patterns. The primary challenge is to design a wireless network intrusion detection system that can accurately identify and respond to various types of cyberattacks. This involves developing a model capable of learning complex patterns indicative of intrusions while adapting to changing attack strategies. Therefore, the need of wireless networks become more prevalent, the need for robust and accurate intrusion detection systems is paramount. Rapidly evolving cyber threats require adaptive and sophisticated approaches to safeguard network integrity and data confidentiality. The project addresses this need by combining advanced techniques for more effective intrusion detection. The project, "Innovating wireless network intrusion detection with adaptive synthetic sampling and enhanced convolutional neural network," aims to revolutionize wireless network security by combining two powerful techniques. The first is adaptive synthetic sampling, which generates additional training data to balance class distributions and improve model performance. The second is an enhanced convolutional neural network (CNN), which is adept at learning complex spatial patterns in network traffic data. By integrating these approaches, this research endeavours to develop a system capable of accurately detecting a wide range of intrusions in wireless networks. This advancement holds great promise for significantly enhancing the security posture of wireless networks, protecting critical assets and data from cyber threats.

Keywords: IOT, Intrusion, CNN, Cyberattacks, Security, Sampling technique, network, Deep Learning, network traffic

1. INTRODUCTION

Against the backdrop of escalating concerns surrounding cybersecurity and the imperative to fortify the integrity of sensitive data, this research endeavors to forge a system designed for the detection of unauthorized access and suspicious activities within computer networks. The significance of this undertaking lies in its pivotal role in ensuring the security and confidentiality of information traversing the digital landscape. The historical trajectory of this research traces its roots to the pressing need for effective wireless network intrusion detection. In the nascent stages of cybersecurity, traditional machine learning algorithms, such as Naive Bayes and Support Vector Machines (SVM), emerged as stalwart tools for detecting anomalies within network behavior. These algorithms, although effective to a certain extent, faced challenges in addressing the evolving landscape of cyber threats. The research, recognizing the limitations of traditional methodologies, takes a leap forward by integrating a deep learning approach into the intrusion detection framework.

Enter the Evolving Cloud-based Neural Network (ECNN), a cutting-edge innovation that harnesses the power of deep learning to discern intricate patterns and anomalies within network traffic. This integration of ECNN introduces a layer of adaptability and sophistication, allowing the system to learn and adapt to emerging threats over time. Moreover, the research incorporates the novel Adaptive Sampling and Synthesis (ASS) technique, enhancing the accuracy and efficiency of intrusion detection. ASS introduces a dynamic and responsive element to the detection process, ensuring that the system remains adept at identifying novel threats and evolving attack vectors. The user interface of the developed system is crafted with accessibility in mind, aiming to democratize intrusion detection capabilities. This ensures that users, even those without an in-depth understanding of machine learning or programming, can navigate the system seamlessly. The integration of traditional algorithms alongside deep learning methodologies is orchestrated to strike a balance between the interpretability of results and the adaptability required for addressing sophisticated cyber threats.

2. LITERATURE SURVEY

The development of attack recognition technology has gone through three stages: pattern matching algorithms, machine learning algorithms and deep learning algorithms. The pattern matching algorithm was first applied to intrusion detection tasks based on feature matching. In [1], Wu and Shen analysed the classical pattern matching algorithms, BM and AC, and proposed the corresponding improved algorithms BMHS and AC-BM; experiments illustrated that the enhanced algorithms greatly optimize the timeliness of IDS. Dagar et al. [2] applied RabinKarp and Knuth-MorrisPratt pattern matching algorithms to intrusion detection tasks and compared their execution efficiency. However, these pattern matching algorithms are difficult to adapt to today's network environment due to the diversity of network attacks. An attack recognition algorithm based on ML has been successfully applied to IDS and achieved excellent performance, which gradually replaced the traditional pattern matching algorithm. SVM is a typical supervised learning model in ML. Thaseen and Kumar [3] presented a novel attack recognition model based on chi-square feature selection and multi class support vector machine (SVM). The simulation illustrates that removing redundant features significantly improves the calculation accuracy and execution efficiency of the model. Ingre et al. [4] established a novel intrusion recognition system by combining the relevant feature screening algorithm with decision trees. Nancy et al. [5] designed a dynamic recursive feature selection algorithm. By extending the decision tree algorithm and combining it with convolutional neural networks. They proposed an intelligent fuzzy temporal decision tree algorithm. The new algorithm achieved a high detection rate of unknown attacks on the KDD cup dataset. An improved IDS based on a Bayesian network and feature selection algorithm was proposed in [6]. Although these ML detection algorithms achieved higher recognition accuracy in intrusion detection tasks, they not only need large-scale feature engineering but the model parameters are also difficult to adjust. However, the DL algorithm can autonomously abstract features from basic network traffic without complex feature engineering. Therefore, related research on intrusion detection is gradually focused on the DL method. An LSTM classifier with a gradient descent optimizer is used in IDS [7], which can effectively mine the association between features from the perspective of time. Su et al. [8] combined an attention mechanism and BLSTM (bidirectional long short-term memory) to propose a network anomaly detection model BAT, which extracts coarse-grained features by connecting forward LSTM and backward LSTM. The BAT model uses an attention mechanism to filter the network flow vectors generated by the BLSTM model to obtain the key characteristics of network traffic classification. Wei et al. [9] applied particle swarm optimization (PSO) to optimize the structure of DBN, and the improved DBN achieves significant anomaly detection ability.Gao et al. [10] designed an effective attack recognition method by combining association rules and improved deep neural networks (DNN), which uses the apriori algorithm to mine the association between discrete features and labels to

improve the recognition accuracy. Yin et al. [10] presented an effective attack recognition model by using feature enhancement and improved RNN; however, the feature enhancement method also increases the computational complexity of the model. CNN has been successfully applied to intrusion detection tasks because it can extract network traffic characteristics more effectively [4]. Lin et al. [5] designed a character-level CL-CNN model. The character-based encoding method makes the features more discretized, which contributes to improving the detection accuracy of IDS. Wu et al. [11] designed a CNN model with a simple structure and proved the necessity of converting the original data into a 2D format through experiments. In addition, the combination of this simple CNN and 2D data conversion greatly improves the detection efficiency of the model compared with the RNN model in [12]. Ding and Zhai [5] proposed a convolutional neural network model (MS-CNN) based on multistage features. Multistage features are obtained by connecting the outputs of all convolutional layers to the dense layer with the softmax classifier. By adding supplementary information (such as local information and detailed information lost by higherlevel convolutional layers), the expressive ability of the model significantly improves. Yang and Wang [13] extracted diverse features through a cross-layer aggregated CNN model, which greatly improved the expression ability of the model. Although the above attack recognition algorithms using CNN improve the detection accuracy, they ignore the interchannel information redundancy in the convolution layer. However, we cannot directly discard some channel information because we are not sure which channels are redundant. To reasonably eliminate the problem of interchannel information redundancy,

3. PROPOSED SYSTEM

This represents a comprehensive approach to intrusion detection in wireless networks using a combination of preprocessing techniques, various classification algorithms, and performance evaluation metrics

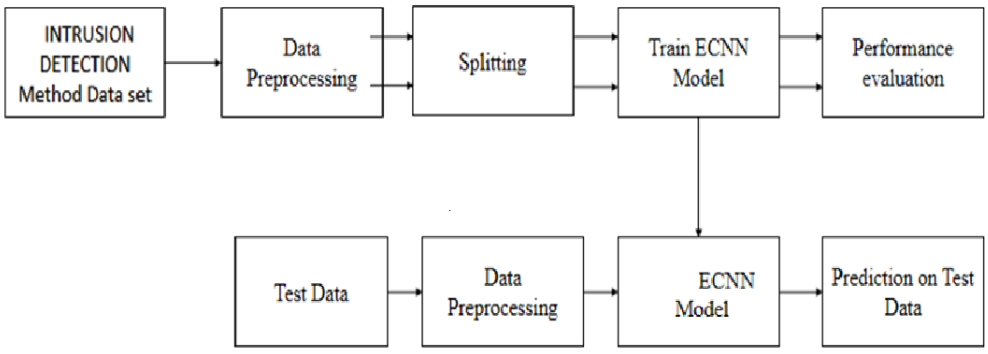

Fig.1: Arctectural block diagram of proposed diagram.

Dataset Upload and Exploration: The initial step involves uploading the NSL-KDD dataset, a common dataset for intrusion detection. The Tkinter GUI facilitates user-friendly file selection, and the dataset is displayed to provide a quick overview.

Preprocessing: After uploading the dataset, preprocessing steps are applied to ensure it is ready for model training. The process involves handling missing values, label encoding for categorical features, and normalization. The LabelEncoder from scikit-learn is utilized to convert categorical variables into numerical format, making them suitable for machine learning algorithms.

Data Augmentation using SMOTE: The code implements Synthetic Minority Over-sampling Technique (SMOTE) for data augmentation. This technique helps balance the class distribution by

oversampling the minority classes. A bar graph visualizes the distribution of different attack types before and after augmentation.

Train-Test Split: The dataset is split into training and testing sets, with 80% of the records used for training and 20% for testing. This step ensures the model's performance is evaluated on unseen data.

Training a Specialized CNN Model (ECNN): The code defines and trains a specialized Convolutional Neural Network (CNN) model, referred to as ECNN (Enhanced CNN), using the Keras library. The model architecture includes convolutional layers, max-pooling layers, and dense layers. The training process involves saving the best model weights to a file for later use.

Applying Existing Classifiers: Two existing classifiers, Naive Bayes and Support Vector Machine (SVM), are implemented for intrusion detection. Each classifier is trained on the preprocessed data, and their performances are evaluated using accuracy, precision, recall, and F1-score metrics.

Performance Evaluation and Comparison: The code calculates and displays performance metrics such as accuracy, precision, recall, and F1-score for both the proposed ECNN model and the existing classifiers. Confusion matrices are visualized to provide insights into the classification results.

Prediction on Test Data: Finally, the code allows users to upload a new dataset for prediction using the trained ECNN model. The selected dataset undergoes the same preprocessing steps, and the model predicts the attack types for each record. The results are displayed in the Tkinter GUI

ECNN

Convolutional Neural Networks (CNNs) are a type of deep neural network designed for processing structured grids of data, such as images or spatial data. Unlike traditional neural networks, CNNs leverage spatial hierarchies of features and local receptive fields to capture patterns efficiently. They are widely used in computer vision tasks, including image classification, object detection, and segmentation.

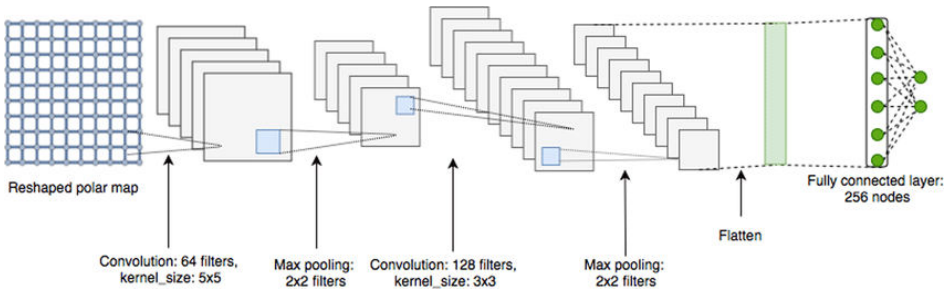

Fig. 2: Architectural of CNN model.

Convolutional Layers: At the core of a CNN are convolutional layers. Each layer consists of a set of learnable filters (or kernels) that slide over the input data, performing element-wise multiplication with local regions and producing feature maps. This process allows the network to learn hierarchical representations of patterns in the data. Mathematically, for an input I and a filter K, the output feature map O is computed as:

$$O(i,j)=\sum m \sum n I(i+m,j+n)\cdot K(m,n)$$

where i,j represents the spatial location in the output feature map.

Pooling Layers: Pooling layers follow convolutional layers and serve to downsample the spatial dimensions of the feature maps while retaining important features. Max pooling, for instance, selects the maximum value from each region of the feature map defined by a pooling window, thus reducing the spatial size and providing translation invariance.

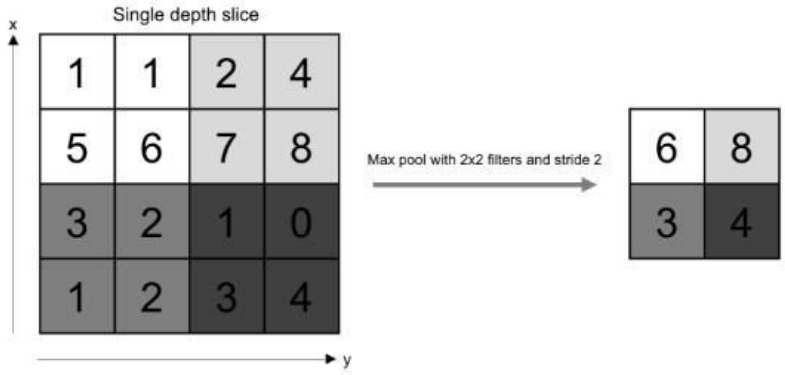

Fig. 3: Maxpooling Layers.

Activation Functions: Activation functions like ReLU (Rectified Linear Unit) are applied after each convolutional and pooling layer to introduce non-linearity, allowing the network to learn complex relationships in the data.

Architectural Components

Fully Connected Layers: Following multiple convolutional and pooling layers, fully connected layers aggregate features learned by previous layers to make final predictions. These layers connect every neuron from one layer to every neuron in the next layer, enabling high-level reasoning.

Dropout: To prevent overfitting, dropout layers randomly deactivate a fraction of neurons during training, forcing the network to learn redundant representations and improving generalization.

Loss Functions: CNNs are typically trained using gradient-based optimization methods such as stochastic gradient descent (SGD). Common loss functions include softmax cross-entropy for classification tasks and mean squared error for regression.

Back propagation: The backpropagation algorithm computes gradients of the loss function with respect to the network parameters, enabling efficient updates of weights through gradient descent.

4. RESULTS AND DISCUSSION

Figure 4 depicts a visual representation of the GUI application developed for wireless network intrusion detection. It showcases the various components and features of the application, allowing users to interact with the ASS and ECNN models for intrusion detection. Figure 5 presents a bar chart (count plot) illustrating the distribution of different attack types in the NSL-KDD dataset. Each bar represents a specific type of network attack, and the height of the bar indicates the frequency or count of that particular attack type in the dataset.

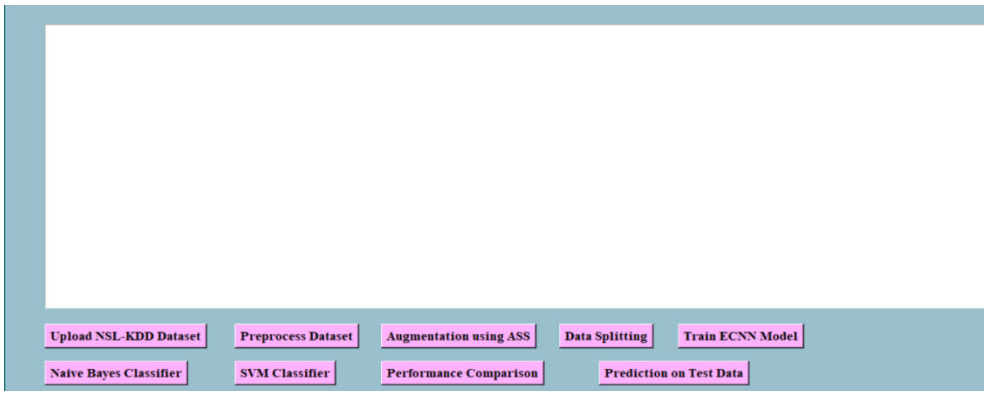

Figure 4: Illustration of UI application for wireless network intrusion detection using ASS and ECNN.

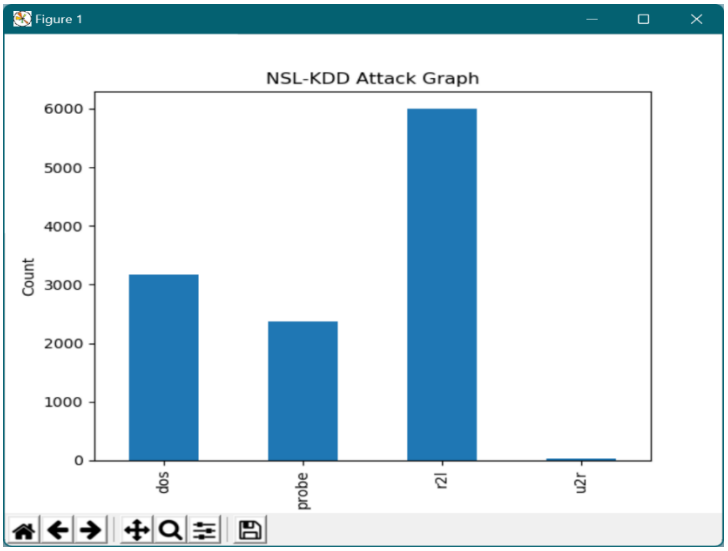

Figure 5: Count plot of attack types in NSL-KDD dataset.

Figure 6 shows a screenshot of the user interface after successfully loading the NSL-KDD dataset. It displays the loaded dataset with detailed information, such as the number of rows (11578) and columns (42). Users can interact with this loaded dataset through the application's interface.

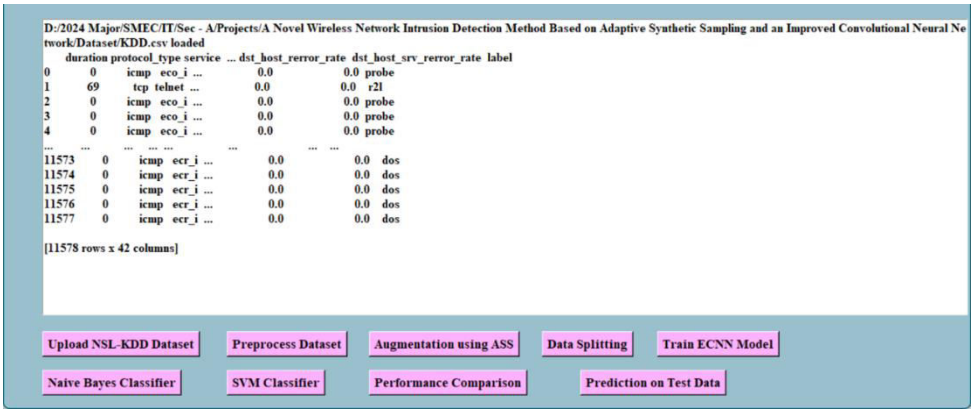

Figure 6: Illustrating the UI application after loading the NSL-KDD dataset with 11578 rows and 42 columns.

Figure 7 showcases the dataset after undergoing preprocessing steps. Preprocessing typically includes tasks like encoding categorical variables, handling missing values, normalizing features, and shuffling the data. The processed dataset is essential for training machine learning models. Figure 8 displays the dataset after the application of the ASS algorithm for data augmentation. ASS is used to balance the class distribution in the dataset by generating synthetic samples of minority class instances. This step helps in addressing class imbalance issues.

Figure 9 represents the user interface after splitting the dataset into training and testing sets. In this case, 80% of the data (19177 records) is used for training, and 20% (4795 records) is reserved for testing the machine learning models. This step ensures that the models are trained on a subset of the data and evaluated on an independent subset to assess their generalization performance.

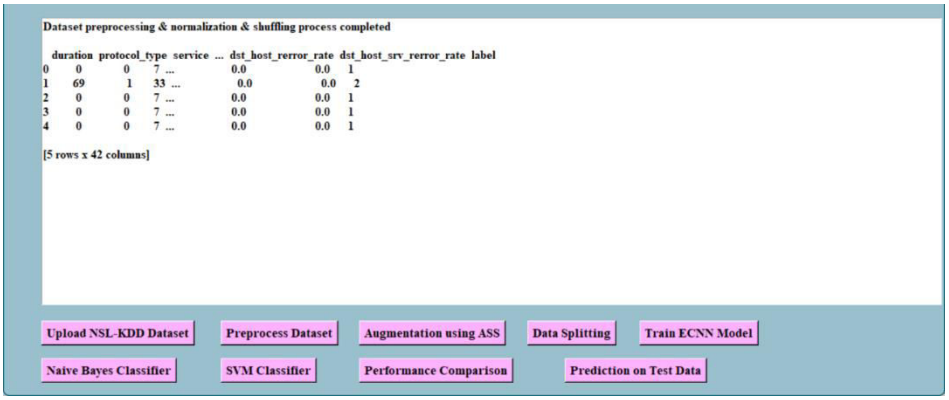

Figure 7: Dataset after preprocessing, normalization and shuffling process.

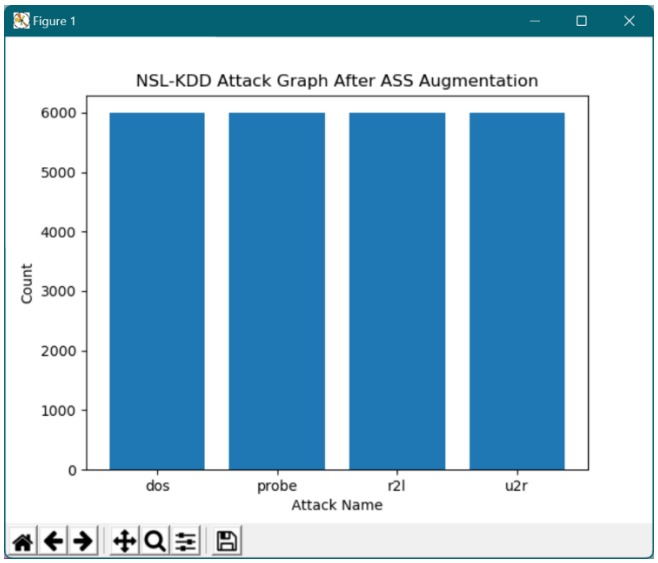

Figure 8: Dataset after applying augmentation using ASS algorithm.

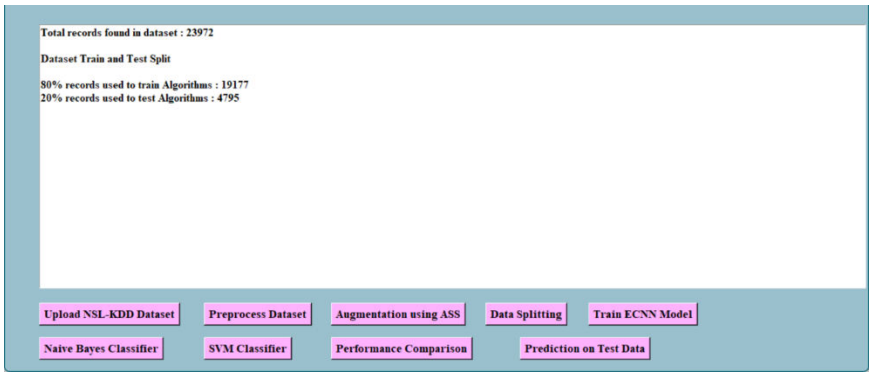

Figure 9: UI after applying data splitting into train and test with 80%, and 20% of 23972 records.

Figure 10 presents confusion matrices for the results obtained using different classifiers: Naïve Bayes, SVM, and the proposed ASS with ECNN model. Confusion matrices provide detailed information about the model's performance, showing the true positive, true negative, false positive, and false negative predictions for each class.

Table 1 presents a comparison of various performance metrics (such as accuracy, precision, recall, and F1-score) obtained from different models. These metrics provide insights into the effectiveness of each model in detecting network intrusions.

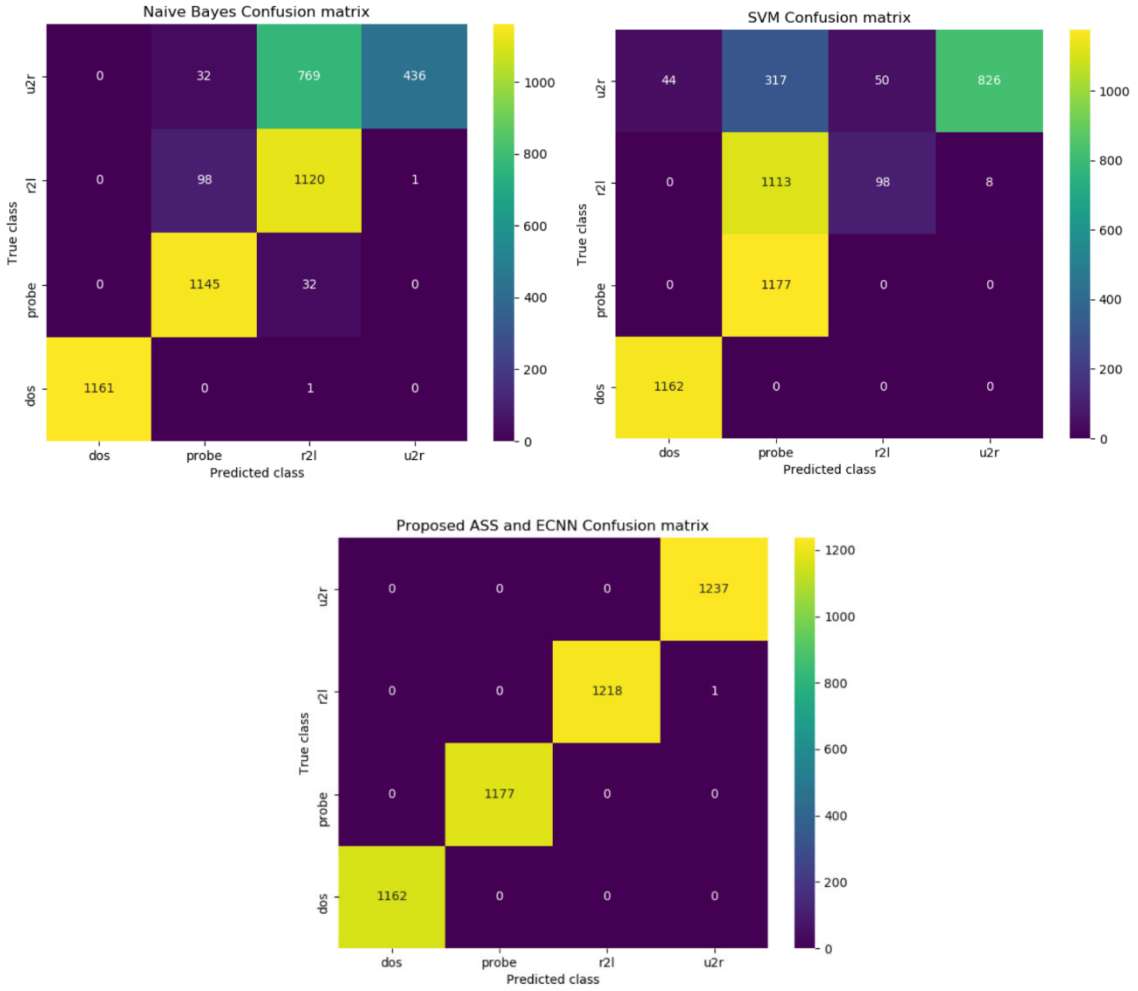

Figure 10: Confusion matrix obtained using existing naïve bayes, SVM classifiers and proposed ASS with ECNN model.

Figure 11 displays a graphical representation of the performance evaluation results for existing models (such as Naïve Bayes and SVM) compared with the proposed ASS and ECNN model. The graph could show trends or comparisons of metrics across different models, providing a visual summary of the evaluation results. 12 Figure shows the user interface after performing predictions on test data using the trained models. It displays the results of the prediction, indicating the predicted labels for the test instances and potentially providing additional information on the confidence or probability scores associated with the predictions.

Table 1: Performance comparison of obtained quality metrics for wireless network intrusion detection system.

Algorithm Name	Accuracy	Precision	Recall	FSCORE
Proposed ASS and ECNN	99.97914494264859	99.97980613893375	99.97949138638228	99.9796404753319
Naive Bayes	80.54223149113659	86.96193032274499	81.08007955135716	79.18894225539381
SVM	68.05005213764338	76.68905959581373	68.70345771578145	63.61434711909168

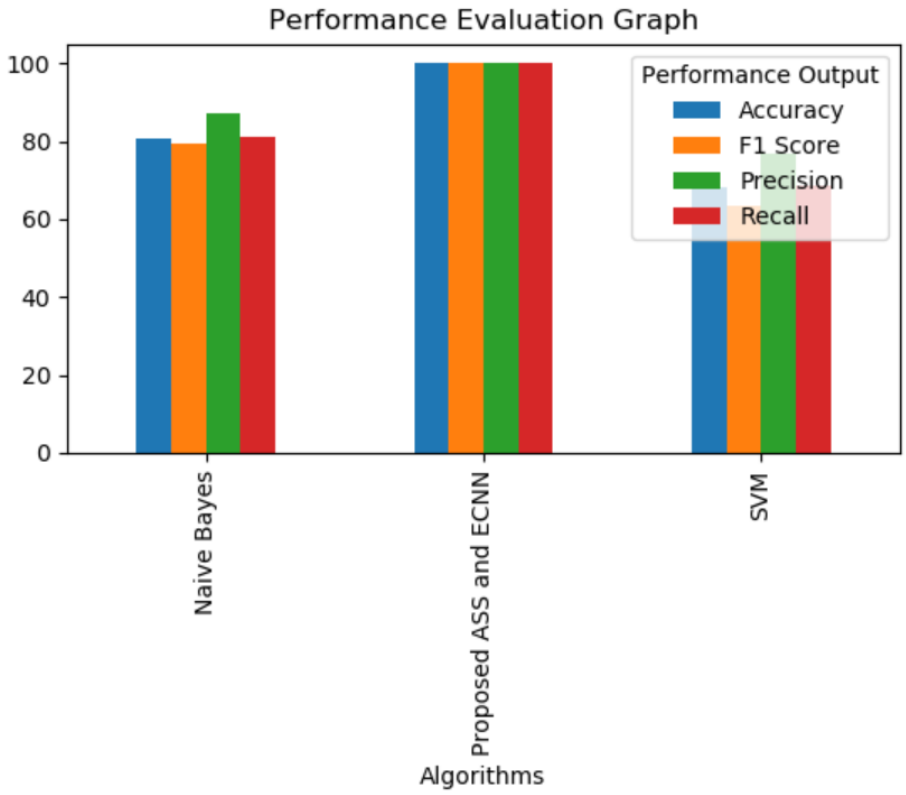

Figure 11: Performance evaluation graph of existing and proposed models.

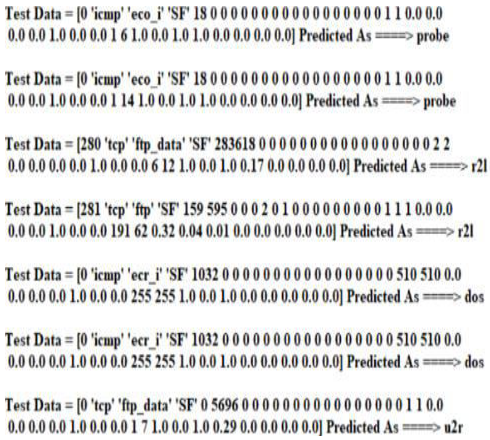

Figure 12: Illustration of UI application after performing prediction on test data.

5. CONCLUSION

This research successfully implemented a wireless network IDS by integrating the ASS technique with the ECNN model. This hybrid approach proved to be highly effective in addressing the class imbalance challenge commonly encountered in intrusion detection datasets. The system demonstrated superior performance compared to traditional classifiers like Naive Bayes and SVM. The ASS with ECNN model outperformed its counterparts, showcasing higher accuracy, which is vital for precise identification of network anomalies. Moreover, the model exhibited superior precision, indicating a reduced rate of false positives, and higher recall, indicating fewer false negatives. The F1-score, representing a balanced trade-off between precision and recall, highlighted the system's efficiency in handling both types of classification errors. These findings emphasize the model's effectiveness in accurate intrusion detection. In addition, these enhanced metrics underscore the system’s robustness in

accurately distinguishing between normal and intrusive network activities. In conclusion, the wireless network IDS is poised to evolve, adapting to emerging threats, and maintaining its effectiveness in safeguarding network infrastructures. The ongoing research and implementation efforts will contribute significantly to the field of network security, ensuring robust and proactive intrusion detection capabilities.

REFERENCES

[1] P.-F. Wu and H.-J. Shen, ‘‘The research and amelioration of patternmatching algorithm in intrusion detection system,’’ in Proc. IEEE 14th Int. Conf. High Perform. Comput. Commun. IEEE 9th Int. Conf. Embedded Softw. Syst., Liverpool, U.K., Jun. 2012, pp. 1712–1715, doi: 10. 1109/HPCC.2012.256.

[2] V. Dagar, V. Prakash, and T. Bhatia, ‘‘Analysis of pattern matching algorithms in network intrusion detection systems,’’ in Proc. Int. Conf. Adv. Comput., 2016, pp. 1–5.

[3] I. S. Thaseen and C. A. Kumar, ‘‘Intrusion detection model using fusion of chi-square feature selection and multi class SVM,’’ J. King Saud Univ.- Comput. Inf. Sci., vol. 29, no. 4, pp. 462–472, Oct. 2017.

[4] B. Ingre, A. Yadav, and A. K. Soni, ‘‘Decision tree based intrusion detection system for NSL-KDD dataset,’’ in Proc. Int. Conf. Inf. Commun. Technol. Intell. Syst., Ahmedabad, India, 2017, pp. 207–218, doi: 10.1007/978- 3-319-63645-0_23.

[5] P. Nancy, S. Muthurajkumar, S. Ganapathy, S. V. N. S. Kumar, M. Selvi, and K. Arputharaj, ‘‘Intrusion detection using dynamic feature selection and fuzzy temporal decision tree classification for wireless sensor networks,’’ IET Commun., vol. 14, no. 5, pp. 888–895, Mar. 2020, doi: 10. 1049/iet-com.2019.0172.

[6] M. A. Jabbar, R. Aluvalu, and S. S. Satyanarayana Reddy, ‘‘Intrusion detection system using Bayesian network and feature subset selection,’’ in Proc. IEEE Int. Conf. Comput. Intell. Comput. Res. (ICCIC), Coimbatore, India, Dec. 2017, pp. 1–5, doi: 10.1109/ICCIC.2017.8524381.

[7] T.-T.-H. Le, J. Kim, and H. Kim, ‘‘An effective intrusion detection classifier using long short-term memory with gradient descent optimization,’’ in Proc. Int. Conf. Platform Technol. Service (PlatCon), Busan, South Korea, Feb. 2017, pp. 1–6, doi: 10.1109/PlatCon.2017.7883684.

[8] T. Su, H. Sun, J. Zhu, S. Wang, and Y. Li, ‘‘BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset,’’ IEEE Access, vol. 8, pp. 29575–29585, 2020, doi: 10.1109/ACCESS.2020. 2972627.

[9] P. Wei, Y. Li, Z. Zhang, T. Hu, Z. Li, and D. Liu, ‘‘An optimization method for intrusion detection classification model based on deep belief network,’’ IEEE Access, vol. 7, pp. 87593–87605, 2019, doi: 10. 1109/ACCESS.2019.2925828.

[10] M. Gao, L. Ma, H. Liu, Z. Zhang, Z. Ning, and J. Xu, ‘‘Malicious network traffic detection based on deep neural networks and association analysis,’’ Sensors, vol. 20, no. 5, p. 1452, Mar. 2020.

[11] R. Vinayakumar, K. P. Soman, and P. Poornachandran, ‘‘Applying convolutional neural network for network intrusion detection,’’ in Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI), Udupi, India, Sep. 2017, pp. 1222–1228, doi: 10.1109/ICACCI.2017.8126009.

[12] Y. Ding and Y. Zhai, ‘‘Intrusion detection system for NSL-KDD dataset using convolutional neural networks,’’ in Proc. 2nd Int. Conf. Comput. Sci. Artif. Intell. (CSAI), 2018, pp. 81–85.

[13] H. Yang and F. Wang, ‘‘Wireless network intrusion detection based on improved convolutional neural network,’’ IEEE Access, vol. 7, pp. 64366–64374, 2019, doi: 10.1109/ACCESS.2019.2917299.