Journal of Science, Computing and Engineering Research (JSCER)
Volume-7, Issue-11, November 2024.
DOI: *https://doi.org/10.46379/jscer.2023.071102*

Network Intrusion Detection System Using Hybrid Deep Learning

[1]R. KARTHICK, [2]J. MOHAMMED ZAYED RAHAMAN, [3]S. ABISHEIK, [4]A. NAMBI RAJA, [5]L. ARUN PRASATH

[1-5] Computer Science and Engineering, KLN College of Engineering, Madurai, India,

Article Information

Received : 10 Nov 2024
Revised : 13 Nov 2024
Accepted : 15 Nov 2024
Published : 17 Nov 2024

Corresponding Author:
Arun prasath

Abstract— This Network Intrusion Detection System (NIDS) are essential for safeguarding computer networks, analyzing traffic to detect malicious activities. With increasing network traffic and sophisticated cyber-attacks, traditional rule- and signature-based IDS struggle with scalability and adaptability. This research examines the use of hybrid deep learning techniques—like Convolutional Neural Networks (CNNs) combined with Bidirectional Long Short-Term Memory (BiLSTM)—to enhance NIDS capabilities in detecting both known and unknown attacks. Recent advancements demonstrate that deep learning-based NIDS improve detection accuracy and reduce false positives, offering a robust and scalable solution for network security.

Keywords: ***anomaly detection, convolutional neural network, cyber attacks, deep learning, network security***

Citation: R. Karthick, J. Mohammed Zayed Rahman, S. Abisheik, A. Nambi Raja, L. Arun Prasath, "Network Intrusion Detection System Using Hybrid Deep Learning", Journal of Science, Computing and Engineering Research, 7(11), November 2024.

I. INTRODUCTION

Intrusion Detection is a way of monitoring the events happening within a network or on a local system to detect any signs of abnormal or malicious breaching the security or standard policies. Intrusion Detection Systems (IDSs) are broadly classified into host-based and network-based. The former monitors an individual computer system (e.g., operating system files and logs) looking for malicious activities, whereas the latter examines network traffic to recognize any malicious and anomalous activities that can be part of an attack. Hybrid deep learning combines multiple deep learning models or methodologies to enhance performance, robustness, and adaptability in various applications. This can involve the combination of different neural network architectures, such as convolutional neural networks (CNNs) for image data and recurrent neural networks (RNNs) for sequential data, allowing for a more comprehensive analysis of multimodal data.

II. RELATED WORKS

Programming education is undergoing a revolution thanks to automated assessment systems and data mining techniques, which provide predictive analytics, individualized learning experiences, and thorough insights into student performance. But conventional approaches frequently lack hands-on coding experience, necessitating a more efficient strategy. We suggest OptiCode, a cutting-edge approach created to close this gap by improving accessibility and efficacy in programming skill acquisition. [1] Current method for Network Intrusion Detection System: The hybrid model overcomes the drawbacks of conventional signature-based systems by producing synthetic data that imitates malicious and legitimate traffic, improving the ability to identify new threats. [2] Detection Systems with Deep Learning: It examines several architectures, including Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs), emphasizing their better detection rates and real-time processing capabilities in comparison to conventional techniques.

III. PROPOSED SYSTEM

By incorporating cutting-edge AI techniques for individualized learning and code improvement, OptiCode is a creative system that improves programming education. After users choose a programming language and subject, Jina Embedding converts their queries into numerical vectors. Large Language Models (LLMs) and Retrieval Augmented Generation (RAG) are used by the system to produce precise code snippets that enable interactive optimizations and corrections. By utilizing real-time data, this all-inclusive workflow creates an immersive learning environment and raises the bar for programming education.

Advantages of the Proposed System:

- The hybrid architecture can detect complex attack patterns that may not be apparent through either spatial or temporal analysis alone.
- The hybrid model is designed to detect anomalies and attacks in real-time, allowing for immediate action to be taken

A. SOFTWARE REQUIREMENTS

1. Python

Python is like a versatile toolbox for programmers, offering a wide array of tools and gadgets to tackle virtually any coding task with ease. Python is a friendly guide through the dense forest of programming. Its syntax, or grammar, is designed to be easily understood, making it accessible even to those venturing into the coding wilderness for the first time. With Python, you don't need to 20 decipher complex hieroglyphics; instead, you're greeted with familiar words and phrases, making your journey smoother and more enjoyable. This programming language operates like a skilled interpreter, executing your commands line by line in real-time. Whether you need to manipulate data, build a website, or train a machine learning model, Python has a module ready to assist you. These modules act as trusty companions, offering shortcuts and solutions to common challenges, saving you time and effort along your coding expedition. Despite its simplicity, Python is a powerful language, capable of handling complex tasks with grace and precision. Like a skilled craftsman, Python allows you to create intricate structures and designs, whether you're building a simple script or a sophisticated application. Python boasts a vibrant and welcoming community of developers. Here, you'll find support, guidance, and camaraderie, as you embark on your coding journey together

2. Pandas

Pandas is an open-source data manipulation and analysis library for Python that provides data structures and functions needed to work with structured data seamlessly. It introduces two primary data structures: Series, which is a one-dimensional labelled array capable of holding any data type, and Data Frame, a two-dimensional labelled data structure similar to a spreadsheet or SQL table. Pandas makes it easy to clean, transform, and analyse data, allowing users to perform operations like filtering, grouping, merging, and reshaping datasets with simple syntax. One of the key strengths of Pandas is its ability to handle missing data and time series data, making it a powerful tool for data scientists and analysts dealing with real-world datasets. The library supports various file formats for data input and output, including CSV, Excel, and SQL databases, facilitating smooth data integration. Additionally, Pandas leverages the performance of NumPy, making it efficient for large datasets and complex computations.

3. Tensorflow or PyTorch

TensorFlow and PyTorch are two of the most widely used deep learning frameworks that enable developers to build, train, and deploy complex machine learning models efficiently. TensorFlow, developed by Google, provides a robust and scalable platform for large-scale machine learning applications, offering features such as automatic differentiation and a flexible architecture that supports both CPU and GPU computations. Its comprehensive ecosystem includes tools for model deployment, such as TensorFlow Serving and TensorFlow Lite, which facilitate the integration of machine learning models into production environments. On the other hand, PyTorch, developed by Facebook, is known for its dynamic computation graph, which allows for more intuitive model building and debugging. This feature makes PyTorch particularly appealing for researchers and developers who prioritize flexibility and speed during the experimentation phase. Both frameworks provide extensive libraries and tools for handling data, defining neural networks, and optimizing model training, making them ideal for tasks such as image recognition, natural language processing, and, importantly, network intrusion detection systems (NIDS) using hybrid deep learning. By leveraging either TensorFlow or PyTorch, developers can harness powerful algorithms and techniques to improve the accuracy and efficiency of NIDS, ultimately enhancing cybersecurity measures.

4. Google collab

Google Collab, or Google Collaboratory, is a cloud-based platform that enables users to write, execute, and share Python code directly from their web browser. It is especially favored in the data science and machine learning communities for its accessibility and powerful features. Users can create Jupyter notebooks, which allow for a mix of code, visualizations, and rich text, making it easy to document workflows and findings. One of the standout features of Google Colab is its free access to powerful computing resources, including GPUs and TPUs, which are crucial for training complex machine learning models. This capability democratizes access to high-performance computing, allowing individuals and teams to work on data-intensive projects without needing expensive hardware. Integration with Google Drive streamlines file management, enabling users to easily save, share, and collaborate on notebooks in real time. Additionally, Google Collab supports popular libraries such as TensorFlow, PyTorch, and scikit-learn, making it a versatile tool for various machine learning and data analysis tasks. With its user-friendly interface and collaborative features, Google Collab serves as an invaluable resource for learners and professionals alike, fostering experimentation and innovation in the field of data science. You can run different versions of your model in

parallel by opening new colab sessions or tabs. This is helpful for hyperparameter tuning and testing different configurations of your CNN and BiLSTM. The platform is suitable for both beginners and experienced developers due to its intuitive interface.

IV. IMPLEMENTATION

The implementation of a Network Intrusion Detection System (NIDS) using CNN and BiLSTM involves several key stages, each designed to build a robust model capable of identifying malicious network activities. Below is an in-depth overview of the steps involved:

Data Preprocessing

The first step is choosing a comprehensive dataset that reflects realistic network traffic, such as UNSW-NB15 or CICIDS2017. These datasets contain labeled instances of normal and attack traffic, which are essential for training and testing the NIDS.

The raw dataset may include irrelevant or redundant features and missing values. Data cleaning ensures these issues are addressed by removing or imputing missing data and discarding unnecessary features.

Feature extraction involves selecting relevant attributes that enhance the model's ability to learn from the data. This step may include domain-specific knowledge to focus on features that indicate network behavior.

To maintain uniformity, features are normalized or standardized. This step helps in scaling numerical data to a consistent range, improving the convergence rate during training.

The preprocessed data is divided into training, validation, and testing sets, ensuring that the model has separate data for learning, tuning, and evaluation.

CNN Feature Extraction

The initial layer of the model employs a 1D CNN to identify local dependencies and extract key features from the input data. This helps in capturing patterns that indicate potential intrusions.

The convolutional layer is configured with suitable kernel sizes and a number of filters to optimize feature detection. ReLU (Rectified Linear Unit) activation functions are applied to introduce non-linearity, enhancing the model's capacity to learn complex patterns.

Max-pooling layers follow the convolutional layer to reduce the feature map's dimensionality, ensuring computational efficiency and highlighting the most significant features.

Bidirectional Long Short-Term Memory (BiLSTM) Layer:

The extracted features from the CNN are fed into a BiLSTM layer. This type of recurrent neural network processes data in both forward and backward directions, enabling the model to learn long-term dependencies and context within the sequential data.

BiLSTMs are particularly useful in analyzing time-series data, such as network traffic, where understanding the order of events is crucial for distinguishing between normal and malicious activity.

Dropout layers are added after the BiLSTM to prevent overfitting by randomly dropping a portion of neurons during each training iteration. This promotes the model's ability to generalize to new data.

Model Compilation and Training

The model is compiled using an optimizer like Adam or RMSprop, known for adaptive learning rate properties that improve convergence. The loss function typically used for classification tasks is categorical cross-entropy if the problem is multi-class or binary cross-entropy for binary classification.

The training phase involves fitting the model to the training data, with hyperparameters such as the learning rate, batch size, and number of epochs being tuned for optimal performance. Techniques like early stopping and learning rate schedulers can be used to halt training when the validation performance plateaus, preventing overfitting.

During training, validation data is used to monitor the model's progress and fine-tune the architecture if necessary. This step ensures that the model learns effectively and generalizes well to unseen data.

Model Evaluation

Once trained, the model is evaluated on the test set to gauge its performance. Metrics such as accuracy, precision, recall, and F1-score are calculated to assess how well the NIDS can identify and classify intrusions.

Accuracy measures the overall correct classifications made by the model.

Precision indicates the proportion of true positive detections out of all positive detections, measuring the model’s reliability in identifying attacks.

Recall (Sensitivity) assesses the model's ability to identify all actual positives (attack instances).

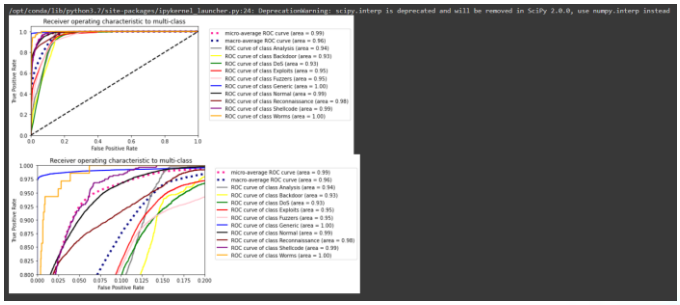
F1-score provides a balance between precision and recall, giving a single metric that accounts for both false positives and false negatives.

Optimization and Fine-tuning

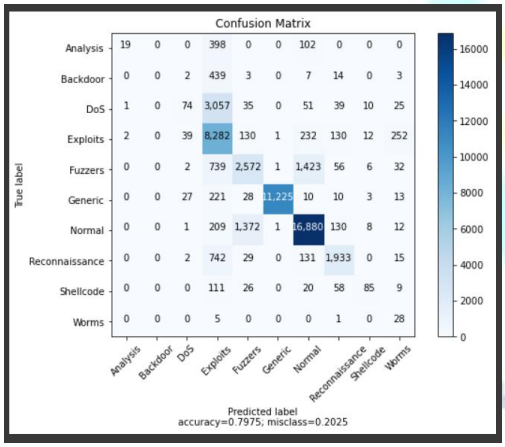
Techniques like grid search or random search can be employed to find the best combination of hyperparameters (e.g., number of CNN filters, LSTM units, learning rate).

Combining the CNN-BiLSTM model with other detection approaches or using techniques like stacking can enhance overall performance.

V. RESULTS



This is the home page of our system where users are presented with options to choose programming languages and text box to ask their question, fill their own logic in the generated code, code analysis for their own logic code and some sample questions.



VI. CONCLUSION

The Network Intrusion Detection System (NIDS) using hybrid deep learning, which integrates Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM), offers an advanced and efficient approach to real-time intrusion detection. This hybrid model combines the strengths of CNN, which excels in spatial feature extraction, and BiLSTM, which is adept at identifying temporal patterns, to enhance the detection of both known and unknown network threats. By leveraging these deep learning models, the NIDS can accurately identify anomalies in complex network traffic, distinguishing between benign activities and malicious intrusions such as Denial of Service (DoS), Distributed Denial of Service (DDoS), and port scanning attacks.In conclusion, this hybrid deep learning-based NIDS demonstrates a highly scalable, adaptable, and reliable solution for modern network security challenges. Through extensive unit, functional, and integration testing, it has proven to be effective in accurately detecting a wide range of intrusions while maintaining real-time performance, making it a crucial tool for safeguarding network infrastructure in today's ever-evolving cyber threat landscape.

VII. FUTURE WORKS

There are several potential enhancements that can be made to further improve the performance and functionality of the Network Intrusion Detection System (NIDS) using hybrid deep learning. This could help the system adapt to different network configurations and types of traffic. Moreover, introducing unsupervised learning techniques alongside the current supervised CNN-BiLSTM architecture could enhance the detection of unknown or novel attacks without relying solely on labeled data. Another future enhancement could involve using feature selection techniques such as Recursive Feature Elimination (RFE), which would optimize the input features used by the system, improving accuracy and reducing computational complexity.

REFERENCES

[1]. Al-QatfM. I. A., M. Lasheng, M. O. Al-Habib, and K. Al-Sabahi, "Deep learning approach combining sparse autoencoder with SVM for network intrusion detection," IEEE Access, vol. 6, pp. 52843-52856, Sep. 2018.

[2]. AltwaijryN, TuraikiAl , "A convolutional neural network for improved anomaly-based network intrusion detection," Big Data, vol. 9, no. 3, pp. 233-252, Jun. 2021

[3]. Alam.M, Javaid. A, Niyaz. Q &Sun. W, "A deep learning approach for network intrusion detection system," Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), pp. 21-26, Dec. 2015.

[4]. Bailey. D.H,Zhang. Y, Xie. N, Wang. W &Li. X , "Deep learning based network intrusion detection system with feature selection method," IEEE Access, vol. 7, pp. 18560-18575, Feb. 2019.

[5]. Chehri. A,Quy. V.K, Quy. N.M, Han. N.D, and Ban. N.T, "Innovative trends in the 6G Era: A comprehensive survey of architecture, applications, technologies, and challenges," IEEE Access, vol. 11, pp. 39824-39844, 2023.

[6]. Lashkari. A.HSharafaldin. M, and Ghorbani. A.A, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), pp. 108-116, Jan. 2018.

[7]. Lasheng. M, Al-Qatf. M.I.A, Al-Habib. M.O, and Al-Sabahi. K, "Deep learning approach combining sparse autoencoder with SVM for network intrusion detection," IEEE Access, vol. 6, pp. 52843-52856, Sep. 2018.

[8]. Pandey. M et al., "The transformational role of GPU computing and deep learning in drug discovery." Nat. Mach. Intell., vol. 4, no. 3. pp. 211-221, 2022.

[9]. Yin. Z, and Zhu. Z, "A hybrid model using deep auto-encoder and one-class SVM for anomaly detection," Knowledge-Based Systems, vol. 195, no. 1, 105648, Apr. 2020.