International Journal of Data and Network Science 9 (2025) 77–86

Contents lists available at GrowingScience

International Journal of Data and Network Science

homepage: www.GrowingScience.com/ijds

Adoption deep learning approach using realistic synthetic data for enhancing network intrusion detection in intelligent vehicle systems

Said A. Salloum[a,b*], Tarek Gaber[a,c], Mohammed Amin Almaiah[d], Rami Shehab[e*], Romel Al-Ali[f] and Theyazan H.H Aldahyani[g]

[a]*School of Science, Engineering, and Environment, University of Salford, United Kingdom*
[b]*Health Economic and Financing Group, University of Sharjah, Sharjah, United Arab Emirates*
[c]*Faculty of Computers and Informatics, Suez Canal University, Ismailia 41522, Egypt*
[d]*King Abdullah the II IT School, University of Jordan, Amman 11942*
[e]*Vice-presidency for Postgraduate Studies and Scientific Research, King Faisal University, Al-Ahsa 31982, Saudi Arabia*
[f]*Associate Professor, the National Research Center for Giftedness and Creativity, King Faisal University, Al-Ahsa 31982, Saudi Arabia*
[g]*Applied College in Abqaiq, King Faisal University, Al-Ahsa 31982, Saudi Arabia*

C H R O N I C L E

Article history:
Received: July 15, 2024
Received in revised format: August 20, 2024
Accepted: September 30 2024
Available online: October 1, 2024

Keywords:
Convolutional Neural Network (CNN)
Cybersecurity
Intelligent Vehicle Systems
Network Intrusion Detection Scapy
Network Traffic Analysis
Simulation
Threat Detection

A B S T R A C T

In the dynamic field of cybersecurity within intelligent vehicle systems, the sophistication of threats necessitates continual advancements in security technologies. Traditional Network Intrusion Detection Systems (NIDS) often fall short in detecting emerging and sophisticated intrusion methods, primarily due to their reliance on static datasets that fail to capture the nuanced dynamics and complexity of modern network intrusions. This study presents a sophisticated simulation for NIDS tailored to intelligent vehicle environments, utilizing the extensive capabilities of Scapy—a robust network manipulation tool—to generate a highly accurate dataset of network traffic reflective of real-world scenarios. We created a diverse dataset involving 100,000 network flows, covering a wide array of benign, malicious, and anomalous traffic patterns, to thoroughly evaluate the detection capabilities of our proposed system. This dataset was analyzed using a deep learning framework employing a Convolutional Neural Network (CNN), which demonstrated outstanding performance metrics: an accuracy of 99.08%, precision of 98.96%, recall of 99.11%, and an F1 score of 99.03%. These metrics showcase the system's enhanced capability to precisely classify various network flows, emphasizing the importance of realistic synthetic data in boosting the training and accuracy of NIDS in intelligent vehicles. The results of this research are significant, marking a step forward towards more flexible and preemptive security measures for intelligent vehicles, and effectively narrowing the gap between simulation-based testing and real-world network environments.

1. Introduction

In the rapidly evolving field of intelligent vehicle security, the escalation of network security threats presents substantial challenges, necessitating advanced protective measures (Biondi, 2010; Hahn et al., 2019). The emergence of sophisticated attack vectors, such as advanced persistent threats and zero-day exploits, underscores the critical role of Intrusion Detection Systems (IDS) in safeguarding vehicular network systems Nawaal et al. (2024). Hahn et al. (2019) depicts a collage representing the

* Corresponding author.
E-mail address Rtshehab@kfu.edu.sa (R. Shebab) salloum78@live.com (S. A. Salloum)

ISSN 2561-8156 (Online) - ISSN 2561-8148 (Print)

doi: 10.5267/j.ijdns.2024.10.001

advanced state of intelligent vehicle security systems, an area experiencing rapid evolution in the face of escalating network security threats. The visuals in the collage, featuring a dynamic display of vehicles on a road with glowing lines symbolizing active network connections, an individual interacting with a high-tech car interface, and a representation of network connectivity, highlight the necessity for sophisticated protective measures within vehicular systems. This scenario illustrates the critical importance of Intrusion Detection Systems (IDS) that must contend with sophisticated attack vectors, including advanced persistent threats and zero-day exploits. These systems are integral to the defense against cyber threats that target the complex networked ecosystems of modern vehicles, as seen in the central image where the user appears to be engaged with a security interface, potentially monitoring or managing the security status of the vehicle's network. As a primary defense mechanism, IDSs scrutinize vehicle network traffic to identify malicious activities and policy breaches in real time Schrötter et al. (2024). Despite their essential role, the efficacy of IDSs is continuously tested by the complex and evolving tactics employed by cyber adversaries (Kruegel et al., 2003; Giacinto et al., 2003).

A significant limitation in contemporary IDS methodologies is their dependence on datasets that fail to accurately reflect the complexity of real-world vehicular network traffic Okoli et al. (2024). Often, IDSs are developed using outdated or synthetic datasets that do not capture the variability and intricacy of actual network communications. This discrepancy leads to a 'reality gap,' where simulated environments do not accurately represent true network threats, thus compromising IDS reliability in practical scenarios Scarfone and Mell (2007). To tackle this challenge, our research develops a simulated IDS tailored for intelligent vehicles, utilizing the capabilities of Scapy, an open-source Python library known for its sophisticated packet manipulation features Buczak and Guven (2015). By employing Scapy, we aim to significantly enhance the realism of network flow data in our simulated environment, ensuring it closely mirrors the traffic observed in operational vehicle networks. Our innovative approach integrates Scapy within the simulation framework to create diverse network scenarios, encompassing a wide range of traffic patterns, protocols, and anomalous behaviors. This initiative is based on the hypothesis that more realistic training data improves the accuracy and generalizability of IDS models Openart AI. (2024). Additionally, it addresses the 'reality gap' by calibrating the IDS against a dataset that accurately simulates the current cyber threat landscape in intelligent vehicle networks. The structure of this paper is designed to sequentially unfold our research efforts. Section 2 reviews current literature to contextualize our study within the broader field of Intrusion Detection Systems. Section 3 outlines our methodological approach, highlighting the use of Scapy to generate a detailed dataset of network flows and explaining our experimental setup and evaluation metrics. This groundwork prepares for the rigorous assessment of the dataset through a Convolutional Neural Network (CNN), which is explored in Section 4. This section also discusses how our empirical findings relate to overall IDS effectiveness, examining the practical implications of our CNN's results. Section 5 discusses these implications in terms of their utility and relevance to IDS in intelligent vehicles. Finally, Section 6 summarizes our key conclusions and suggests directions for future research, emphasizing the importance of our contributions to the dynamic field of network security for intelligent vehicles.

2. Related Work

The realm of network intrusion detection primarily revolves around two methodologies: signature-based and anomaly-based systems Bansal and Bansal, N. (2015). Signature-based IDSs operate by comparing network traffic against a database of known threat signatures to identify intrusions Ring et al. (2019). This approach is highly effective in recognizing well-documented attacks but is inherently limited in detecting new, modified, or previously unseen threats Bacha et al. (2024). In contrast, anomaly-based IDSs leverage machine learning techniques Idrissi et al. (2023) and statistical models to establish what constitutes normal behavior within a network Phulre et al. (2024). Any deviation from this established baseline may indicate a potential intrusion. While capable of detecting novel attacks, anomaly-based systems often face challenges related to high rates of false positives (Bahlali et al., 2023; Lazarevic et al., 2005).

Simulated environments are crucial for assessing the effectiveness of IDS methodologies. These controlled environments allow for the systematic testing and training of IDS mechanisms under varied conditions Moustafa et al. (2021). Researchers have utilized testbeds, synthetic data generation, and network traffic simulation tools to create realistic network traffic scenarios Clausen et al. (2021). Nonetheless, the development of these simulations is frequently impeded by the scarcity of publicly available, labeled datasets for network intrusion, which restricts opportunities for thorough training and validation of IDS models Ciric et al. (2024).

Scapy, a flexible packet manipulation tool, is renowned for its ability to craft and decode packets across a multitude of network protocols, making it an invaluable resource in security research Xiao et al. (2019). It has been employed extensively for protocol creation, penetration testing, and the simulation of network attacks to evaluate security defenses Yu et al. (2024). However, the application of Scapy in academic research, particularly in the development and evaluation of IDS solutions, remains underexplored. This observation suggests a significant research opportunity.

There exists a conspicuous research gap concerning the utilization of Scapy to generate sophisticated and large-scale network traffic datasets that mimic the complexity of real-world intelligent vehicle networks. Most studies to date have not fully exploited

Scapy's capabilities in creating diverse network scenarios that could enhance IDS testing and validation processes. Our research aims to bridge this gap by harnessing Scapy not only for generating realistic traffic patterns but also integrating these capabilities within a machine learning framework using CNNs. This integration seeks to amplify detection accuracies and refine the predictive capabilities of IDS, specifically tailored for intelligent vehicle security systems (Tharwat, 2020).

3. Methodology

The creation of the Simulated Network Intrusion Detection System (SNIDS) utilized the robust capabilities of Scapy, a powerful packet manipulation library, to produce an extensive dataset of synthetic network flows (Goodfellow et al., 2016). Our approach is grounded in the design of a controlled, yet intricate simulation environment that effectively replicates the diversity and unpredictability of real-world network traffic. This method allows for a more realistic representation of network conditions, enhancing the system's ability to identify and respond to varied security threats effectively.

3.1 Data Generation with Scapy

Scapy, a versatile Python-based packet crafting tool, was utilized to simulate a wide range of network behaviors Bansal and Bansal, N. (2015). This included benign activities, established attack patterns, and outlier traffic that could potentially signify emerging threats. Each network flow in our simulation was characterized by a set of distinct features as shown in Table 1.

Table 1
Simulation Features

Simulation Features	Description
src_ip	An anonymized integer representation of the source IP address.
src_port	The originating port number of the flow.
dest_ip	An anonymized integer corresponding to the destination IP address.
dest_port	The receiving port number.
protocol	A numerical value indicating the transport layer protocol (e.g., 6 for TCP, 17 for UDP).
bytes_in/out	The volume of bytes transferred in either direction.
num_pkts_in/out	The number of packets observed in the flow.
entropy	A statistical measure of randomness within the flow's payload.
total_entropy.	The aggregate entropy across all packets.
mean_ipt	The mean inter-packet arrival time.
time_start/end	Timestamps marking the flow's duration.
duration	The computed length of the flow in seconds.
label	The classification label assigned to the flow, distinguishing between benign, malicious, and outlier traffic.

The classification of each network flow was conducted using a heuristic algorithm, which employed predefined criteria designed to emulate standard intrusion detection mechanisms. This methodological approach allowed for systematic and precise categorization, aligning closely with the operational protocols typically observed in advanced intrusion detection systems.

3.2 Simulation Configuration

The architecture of our simulation environment was precisely engineered to create a substantial and balanced dataset comprising 100,000 individual network flows Xiao et al. (2019). This dataset is designed to capture a wide spectrum of classification labels, accurately reflecting the complex nature of real-world network traffic. Our goal was to develop a simulation that meets the quantitative demands for robustness and comprehensiveness while truly representing the qualitative nuances of network behavior across various contexts. To achieve a representative blend of traffic types, the simulation was structured to evenly distribute benign, malicious, and outlier flows. Benign flows mimic regular user activities and standard network communications, serving as a baseline for normal traffic. Malicious flows, on the other hand, represent patterns and payloads characteristic of established cyber threats, such as distributed denial-of-service (DDoS) attacks, malware propagation, and unauthorized data breaches. Additionally, outlier flows were incorporated to represent ambiguous activities that might indicate new or evolving threats, thereby providing a crucial resource for evaluating the responsiveness of intrusion detection algorithms. We utilized the Scapy tool to meticulously construct detailed packet-level interactions, granting precise control over the diversity of the flows. Through Scapy's capabilities to customize packet headers and payloads, we were able to simulate a range of network behaviors from standard protocol exchanges to sophisticated attack vectors. This level of detailed control over each flow's attributes—such as packet size, timing, sequence, and payload specifics—ensures an authentic replication of the complexity seen in active network environments.

3.3 Pseudocode Generation

In our research, we developed a simulation script to generate a comprehensive dataset of synthetic network traffic that mimics the complexities of real-world network behavior. This script utilizes prominent libraries, including Pandas for data handling, Numpy for numerical computations, and Random for generating random variables, and Scapy for detailed network packet manipulation.

Through a series of defined functions, we produce network flows each distinguished by attributes such as anonymized IP addresses, port numbers, protocols, and traffic metrics. Each flow is categorized as benign, malicious, or an outlier, with the classification determined by heuristic criteria concerning byte volume and entropy levels.
The core of the script is the simulate_network_flows function, which systematically generates 100,000 individual flows. These flows are then compiled into a Pandas DataFrame, offering a structured and accessible format for analyzing the simulated network traffic. This dataset is crucial for the validation of our intrusion detection system, providing a rich and varied environment to assess the system's capability to accurately detect and respond to potential threats.

ALGORITHM 1: Network Flow Simulation

```
Import necessary libraries: pandas, numpy, random, scapy

Define function generate_ip:
    Generate and return a random IP address string

Define function anonymize_ip with parameter ip:
    Return anonymized IP using hash function and modulus

Define function generate_flow:
    Create and return a dictionary representing a network flow with:
        Anonymized source and destination IP addresses
        Random source and destination port numbers
        Random protocol (TCP or UDP)
        Random byte and packet counts for both incoming and outgoing traffic
        Random entropy and total entropy values
        Random mean inter-packet arrival time
        Random start time and duration

Define function classify_flow with parameter flow:
    Classify flow as 'malicious', 'outlier', or 'benign' based on:
        Bytes transferred and entropy values

Define function simulate_network_flows with parameter n:
    Initialize an empty list to store flows
    Loop n times to generate and classify flows
    Add each flow to the list
    Return a DataFrame created from the list of flows

Execute simulate_network_flows function with 100000 flows
```

3.4 Validation Using CNN Model for Network Intrusion Detection

The evaluation of the simulated network flows was conducted through a Convolutional Neural Network (CNN) model, specifically designed to detect and categorize network intrusions. This phase is critical, confirming that the generated data adequately prepares machine learning models for training and ensures these models can effectively generalize to new, unseen data that mirrors actual real-world scenarios.

3.4.1 Model Architecture

The design of the CNN was specifically engineered to align with the unique attributes of network traffic data Yu et al. (2024). This model incorporates multiple convolutional layers, which are particularly adept at detecting patterns within the input features, relevant to one-dimensional time-series data typical of network Salam et al. (2021). Following the convolutional layers, pooling layers were employed to reduce the dimensionality of the data and mitigate the risk of overfitting Park and Kwak (2017). Additionally, to enhance the model's ability to generalize, dropout layers were strategically placed following the convolutional layers to thin out neural connections intermittently Murugan (2018). The architecture culminates in fully connected layers that lead to a softmax activation function, facilitating the classification of multiple classes Mehta et al. (2019).

3.4.2 Training Process

The CNN underwent training utilizing the backpropagation technique complemented by an Adam optimizer to enhance convergence efficiency Tharwat (2020). The training process was structured around mini-batches, facilitating incremental updates to the network's weights and enhancing its ability to generalize. To mitigate the risk of overfitting and to establish the most effective

moment to cease training, the model's accuracy and loss were consistently evaluated against a designated validation set throughout the training period.

3.4.3 Performance Metrics

To assess the performance of the Convolutional Neural Network (CNN) model, we utilized a comprehensive set of metrics, each providing distinct insights into the model's effectiveness:

- **Accuracy**: Measures the proportion of total correct predictions out of all predictions made, offering an overview of the model's overall effectiveness (Sokolova & Lapalme, 2009; Tavallaee et al., 2009).
- **Precision**: Indicates the ratio of true positive predictions to all positive predictions made by the model. This metric is particularly useful for understanding the model's ability to minimize false positives Powers (2020).
- **Recall**: The ratio of true positive predictions to all actual positive instances in the dataset, reflecting the model's sensitivity or its ability to detect all relevant cases van Rijsbergen (1979).
- **F1-Score**: The harmonic mean of precision and recall, this single metric helps balance the trade-offs between precision and recall, providing a more comprehensive view of model performance Fawcett (2006).
- **ROC Curve and AUC**: The Receiver Operating Characteristic (ROC) curve plots the true positive rate against the false positive rate at various threshold settings, while the Area Under the Curve (AUC) provides an aggregate measure of performance across all possible classification thresholds Fawcett (2006).

The CNN model's capability to accurately classify benign, malicious, and outlier network flows was rigorously evaluated using these metrics. The results were meticulously documented to affirm the efficacy of the simulated dataset in replicating real-world scenarios typically encountered in practical intrusion detection applications. This methodical evaluation helps validate the robustness and applicability of the CNN model within a simulated IDS environment, ensuring its preparedness for deployment in more dynamic and unpredictable real-world settings.

3.4.4 Model Validation

After the training phase, the Convolutional Neural Network (CNN) underwent a rigorous validation process using a test set that was not seen during training, derived from the generated dataset. This critical step was essential to verify the model's ability to generalize and maintain accuracy within a simulated operational environment, mirroring the conditions of real-world Intrusion Detection System (IDS) applications. This validation helps ensure that the CNN can reliably identify threats and anomalies, reflecting its practical efficacy and readiness for deployment in actual security settings.

4. Results

Upon running the simulation, the Simulated Network Intrusion Detection System (SNIDS) generated a comprehensive dataset that underwent thorough analysis. The findings were clearly illustrated through a series of detailed graphs, designed to reveal the intrinsic patterns in the traffic data and to demonstrate the system's classification abilities effectively. These visual representations played a crucial role in understanding the distribution and characteristics of network behaviors, as well as in assessing the precision of the system's detection mechanisms.

4.1 Traffic Volume Analysis

The data visualization of total incoming bytes, segmented by protocol and label, was graphically represented and indicated that both TCP (protocol 6) and UDP (protocol 17) exhibited similar volumes of benign and malicious traffic. Notably, outliers contributed substantially to the traffic volumes in both protocols (refer to Fig. 1). This analysis highlights the significant role of these protocols in network traffic and underscores the importance of monitoring outlier behavior, which could provide insights into unusual or potentially harmful network activities.

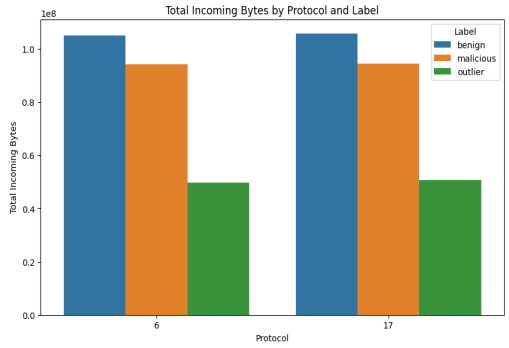

Fig. 1. Traffic Volume by Protocol

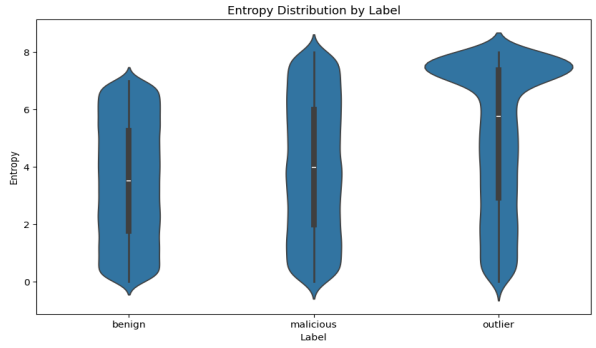

Fig. 2. Entropy Distribution by Label

4.2 Entropy Analysis

The analysis of entropy distribution by label revealed that flows identified as 'malicious' exhibited a significantly higher average entropy than those labeled as 'benign'. Moreover, flows categorized as 'outliers' displayed the greatest variability in entropy, indicating these could be key targets for further enhancements in anomaly detection efforts (refer to Figure 2). This pattern suggests a nuanced understanding of how different types of network flows manifest in terms of entropy, pointing to potential areas for improving the precision of anomaly detection algorithms.

4.3 Total Entropy Distribution

A consistent pattern was also evident in the overall entropy distribution, further affirming the importance of entropy as a critical distinguishing characteristic in flow analysis (refer to Fig. 3). This observation underscores the utility of entropy in differentiating between various types of network flows, thereby reinforcing its role as a valuable metric in the analysis of network traffic.

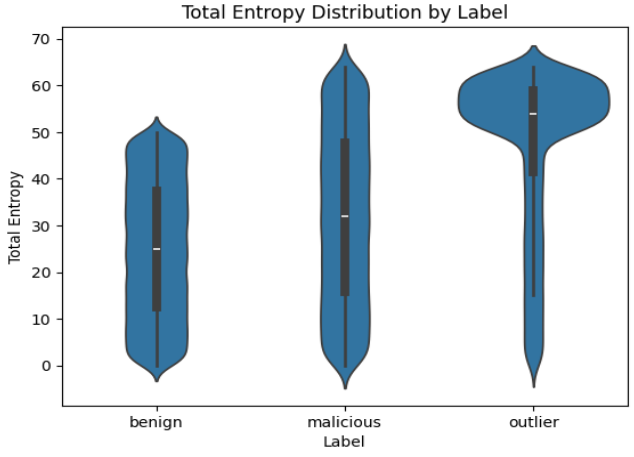

Fig. 3. Total Entropy Distribution by Label.

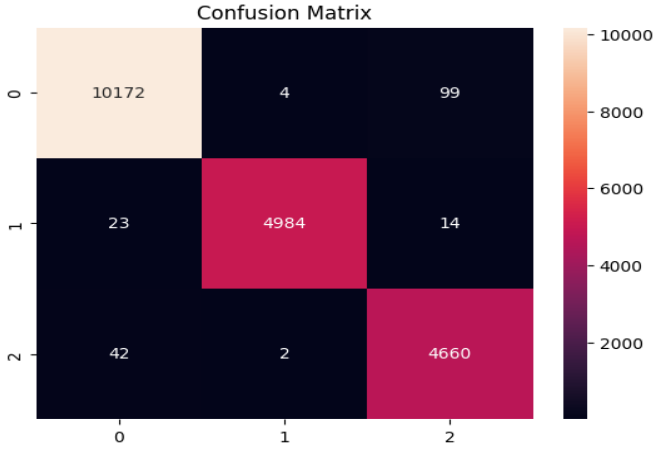

Fig. 4. Confusion Matrix visualizing the classification results of the CNN model

4.4 Performance of the CNN Model

Testing the generated network flows with the Convolutional Neural Network (CNN) model resulted in exceptionally high performance metrics (see Fig. 4), highlighting the effectiveness of the CNN model and the quality of the simulated dataset in identifying network intrusions. The CNN model was evaluated using a test set distinct from the training data to ensure an unbiased assessment. It achieved an accuracy of 99.08%, showcasing an excellent rate of correct classifications. The precision of the model, which measures its ability to correctly identify instances of network attacks, was approximately 98.96%. This high precision indicates a minimal occurrence of false positives, affirming that most flows identified as attacks were indeed malicious. The model's recall rate was 99.11%, demonstrating its sensitivity and ability to detect nearly all actual malicious flows present in the network. The F1 Score, calculated as the harmonic mean of precision and recall, stood at 99.03%. This score reflects a strong balance between precision and recall, verifying the model's capability to maintain both a low false positive rate and a high true positive rate.

Further supporting the CNN model's discriminatory prowess are the confusion matrix and the Receiver Operating Characteristic (ROC) curve. The confusion matrix offers a detailed view of the model's performance across different categories, showing few

misclassifications. The ROC curve, which illustrates the trade-off between the true positive and false positive rates, achieved an area under the curve (AUC) nearly perfect at 1.00 for each class, indicating exceptional performance (see Fig. 5).

The clarity of the confusion matrix and the ROC curves underscores the high detection capabilities of the CNN model, with the matrix showing a high accuracy of correct classifications across benign, malicious, and outlier categories. The ROC curves, with an AUC of 1.00 for each class, confirm the model's excellent ability to classify correctly at various decision thresholds.

The remarkable performance metrics of the CNN model, including accuracy, precision, recall, and F1 Score, along with corroborative insights from the confusion matrix and ROC curves, indicate that the combination of our simulation framework and Scapy's ability to generate network traffic creates a data environment ideal for training efficient intrusion detection models. The findings confirm that our approach to generating synthetic data produces a dataset that is both complex and realistic, making it an excellent stand-in for real-world data when evaluating and training intrusion detection systems.

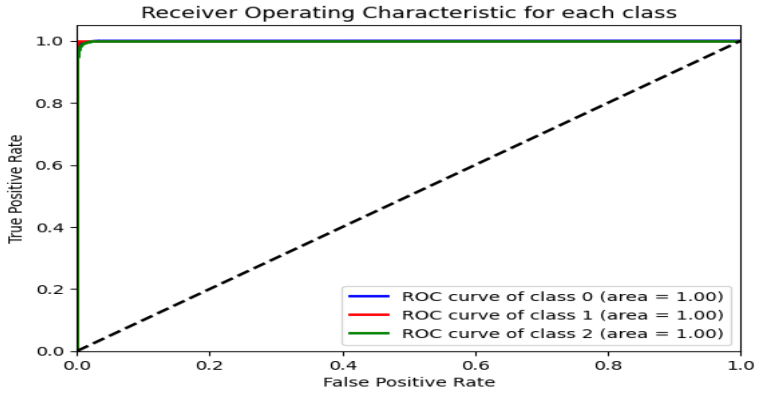

Fig. 5. Receiver Operating Characteristic (ROC) curves for each class.

5. Discussion

The simulated dataset created using Scapy closely mirrored real network traffic, establishing a robust foundation for testing detection algorithms within intelligent vehicle systems. The Convolutional Neural Network (CNN) model, with its impressive results—99.08% accuracy, 98.96% precision, 99.11% recall, and an F1 score of 99.03%—demonstrated its effectiveness in differentiating between benign and malicious network activities.

The success of the CNN model can largely be attributed to the diverse and rich dataset generated by Scapy. Scapy's capability to finely tune packet attributes enabled the simulation of various network scenarios, from routine user activities to complex cyber-attacks. This detailed replication of network behavior is crucial for training Intrusion Detection Systems (IDS) to be both adaptive and comprehensive in their threat detection capabilities.

Scapy's flexibility in packet manipulation played a pivotal role in adding realism to the synthetic dataset, allowing security professionals to replicate a wide spectrum of network traffic scenarios accurately. This variety is essential for the IDS to encounter and learn from a broad spectrum of potential threats, enhancing its detection capabilities and ability to generalize across different scenarios.

The CNN model's robust performance metrics validate the synthetic dataset's suitability for training purposes. The model's high precision and recall are indicative of its capability to minimize both false positives and false negatives, maintaining the integrity and reliability of the network environment.

A key achievement of this study was ensuring that the synthesized data reflects the complexity of actual network behaviors, addressing a common shortfall in many existing datasets. This achievement suggests that the model, when trained on this synthetic dataset, is likely to perform well in real-world applications, having been exposed to a comprehensive spectrum of potential network situations.

These results have significant implications for the field of network security, particularly in intelligent vehicle systems. A simulated IDS, equipped with high-fidelity data, is better positioned to detect and mitigate real-world intrusions effectively. The performance of the CNN model on the synthesized data supports the potential of such systems to be highly effective in actual environments.

However, the study is not without its limitations. Future research could benefit from integrating additional network flow characteristics, simulating more sophisticated attack vectors, and exploring other deep learning models to further enhance the capabilities of IDS. Moreover, comparing these results with datasets derived from actual network traffic would provide deeper insights into the model's practical applicability and effectiveness in real-world scenarios.

6. Conclusions and Future Work

In conclusion, the development of the Simulated Network Intrusion Detection System (SNIDS) for intelligent vehicle security represents a notable advancement in the field of cybersecurity. Utilizing the sophisticated packet crafting and manipulation capabilities of the Scapy library, we generated a synthetic dataset that effectively mimics the complexities of real-world vehicular network traffic. The deployment of a Convolutional Neural Network (CNN) for this dataset's validation demonstrated excellent performance, evidenced by high metrics of accuracy (99.08%), precision (98.96%), recall (99.11%), and F1 Score (99.03%). The model's robust predictive ability was further confirmed by an impressive Receiver Operating Characteristic (ROC) curve. The use of Scapy enabled precise control over the attributes of network traffic, allowing us to simulate a diverse array of behaviors, from routine operations to advanced cyber-attacks. This versatility facilitated a thorough evaluation of the CNN-based IDS within various scenarios, effectively narrowing the gap between theoretical research and practical application in operational systems. Nevertheless, this study has its limitations. The diversity of the synthetic dataset, while comprehensive, does not capture all conceivable network behaviors and attack vectors recognized in cybersecurity. Moreover, the heuristic classification of network flows could be enhanced to better reflect the dynamic nature of vehicular communications and the constantly evolving landscape of cyber threats. Future research will aim to expand the simulation environment to incorporate more intricate and covert attack scenarios, such as Advanced Persistent Threats (APTs) and insider threats, which are notably challenging to detect. Integrating this simulation system with real-world traffic could further improve the realism and applicability of the dataset. There also lies potential in exploring various neural network architectures and machine learning strategies to optimize detection performance. The overarching goal is to develop a simulation system that serves dual purposes: as a rigorous testing environment for IDS solutions and as an educational tool for network administrators and security professionals. By continuously enhancing the simulated dataset and refining the classification methodologies, this system will evolve in sync with the new threats it aims to mitigate, ensuring its ongoing relevance and efficacy in the protection of intelligent vehicle networks. This research makes a significant contribution to the cybersecurity domain by providing a robust simulated environment for IDS development and demonstrating the practical benefits of synthetic datasets in enhancing IDS performance. These promising results set a solid foundation for future advancements in network security, paving the way for more effective and resilient defenses against an array of cyber threats.

Acknowledgment

This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia (Grant No. KFU241959)

References

Bacha, S., Aljuhani, A., Ben Abdellafou, K., Taouali, O., Liouane, N., & Alazab, M. (2024). Anomaly-based intrusion detection system in IoT using kernel extreme learning machine. *Journal of Ambient Intelligence and Humanized Computing, 15*(1), 231–242.

Bahlali, A. R., & Bachir, A. (2023). Machine learning anomaly-based network intrusion detection: Experimental evaluation. In *International Conference on Advanced Information Networking and Applications* (pp. 392–403).

Bansal, S., & Bansal, N. (2015). Scapy—a Python tool for security testing. *Journal of Computer Science & Systems Biology, 8*(3), 140.

Biondi, P. (2010). Scapy documentation (!). *469*, 155–203.

Buczak, A. L., & Guven, E. (2015). A survey of data mining and machine learning methods for cybersecurity intrusion detection. *IEEE Communications Surveys & Tutorials, 18*(2), 1153–1176.

Ciric, V., Milosevic, M., Sokolovic, D., & Milentijevic, I. (2024). Modular deep learning-based network intrusion detection architecture for real-world cyber-attack simulation. *Simulation Modelling Practice and Theory, 102916*.

Clausen, H., Grov, G., & Aspinall, D. (2021). Cbam: A contextual model for network anomaly detection. *Computers, 10*(6), 79.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*(8), 861–874.

Giacinto, G., Roli, F., & Didaci, L. (2003). Fusion of multiple classifiers for intrusion detection in computer networks. *Pattern Recognition Letters, 24*(12), 1795–1803.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning*. MIT Press.

Hahn, D., Munir, A., & Behzadan, V. (2019). Security and privacy issues in intelligent transportation systems: Classification and challenges. *IEEE Intelligent Transportation Systems Magazine, 13*(1), 181–196.

Idrissi, M. J., et al. (2023). Fed-anids: Federated learning for anomaly-based network intrusion detection systems. *Expert Systems with Applications, 234*, 121000.

Kruegel, C., & Toth, T. (2003). Using decision trees to improve signature-based intrusion detection. In *International Workshop on Recent Advances in Intrusion Detection* (pp. 173–191).

Lazarevic, A., Kumar, V., & Srivastava, J. (2005). Intrusion detection: A survey. In *Managing Cyber Threats: Issues, Approaches, and Challenges* (pp. 19–78).

Mehta, S., Paunwala, C., & Vaidya, B. (2019). CNN based traffic sign classification using Adam optimizer. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)* (pp. 1293–1298).

Moustafa, N., Keshk, M., Choo, K.-K. R., Lynar, T., Camtepe, S., & Whitty, M. (2021). DAD: A distributed anomaly detection system using ensemble one-class statistical learning in edge networks. *Future Generation Computer Systems, 118*, 240–251.

Murugan, P. (2018). Implementation of deep convolutional neural network in multi-class categorical image classification. *arXiv Preprint arXiv1801.01397*.

Nawaal, B., Haider, U., Khan, I. U., & Fayaz, M. (2024). Signature-based intrusion detection system for IoT. In *Cyber Security for Next-Generation Computing Technologies* (pp. 141–158). CRC Press.

Network Flows. (2024). Kaggle. https://www.kaggle.com/datasets/saidasalloum/network-flows.

Okoli, U. I., Obi, O. C., Adewusi, A. O., & Abrahams, T. O. (2024). Machine learning in cybersecurity: A review of threat detection and defense mechanisms. *World Journal of Advanced Research and Reviews*.

Openart AI. (2024). https://openart.ai/home.

Park, S., & Kwak, N. (2017). Analysis on the dropout effect in convolutional neural networks. In *Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part II 13* (pp. 189–204).

Phulre, A. K., Jain, S., & Jain, G. (2024). Evaluating security enhancement through machine learning approaches for anomaly-based intrusion detection systems. In *2024 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)* (pp. 1–5).

Powers, D. M. W. (2020). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv Preprint arXiv2010.16061*.

Ring, M., Wunderlich, S., Scheuring, D., Landes, D., & Hotho, A. (2019). A survey of network-based intrusion detection data sets. *Computers & Security, 86*, 147–167.

Salam, M. A., Azar, A. T., Elgendy, M. S., & Fouad, K. M. (2021). The effect of different dimensionality reduction techniques on machine learning overfitting problem. *International Journal of Advanced Computer Science and Applications, 12*(4), 641–655.

Scarfone, K., & Mell, P. (2007). Guide to intrusion detection and prevention systems (IDPS). *NIST Special Publication*, *800*(94).

Schrötter, M., Niemann, A., & Schnor, B. (2024). A comparison of neural-network-based intrusion detection against signature-based detection in IoT networks. *Information, 15*(3), 164.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management, 45*(4), 427–437.

Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy* (pp. 305–316).

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications* (pp. 1–6).

Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics, 17*(1), 168–192.

van Rijsbergen, C. J. (1979). *Information retrieval*. Butterworth-Heinemann.

Xiao, Y., Xing, C., Zhang, T., & Zhao, Z. (2019). An intrusion detection model based on feature reduction and convolutional neural networks. *IEEE Access, 7*, 42210–42219.

Yu, H., Wang, Z., Xie, Y., & Wang, G. (2024). A multi-granularity hierarchical network for long-and short-term forecasting on multivariate time series data. *Applied Soft Computing, 111537*.