Machine Learning based DDoS Detection for Connected Automotive Vehicles

Prerona Ghosh
Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada
p2ghosh@uwaterloo.ca

***Abstract*—Connected vehicles are vulnerable to various cyber threats, including Distributed Denial of Service (DDoS) attacks, which can disrupt the vehicle's network connectivity and compromise its functionality. To address this, a research project can be undertaken to develop an advanced DDoS detection system using machine learning techniques. This project aims to enhance the security and reliability of connected vehicles by effectively identifying and mitigating DDoS attacks. The paper focuses on the specific problem of detecting DDoS attacks targeting connected vehicles using machine learning algorithms and how correct feature selection can help improve the accuracy of the detection. DDoS attacks involve overwhelming a network or system with a flood of traffic, rendering it inaccessible. Such attacks can impact the connectivity and communication capabilities of connected vehicles, jeopardizing passenger safety and vehicle operations. Therefore, developing an efficient DDoS detection system becomes critical in ensuring the robustness and reliability of connected vehicle networks.**

***Index Terms*—Automotive Networks; Machine Learning; Classification; DDoS; Feature Engineering; Connected Vehicles; Internet of Things; Security.**

I. Introduction

There has been a growing interest in vehicular network architectures, protocols, and applications. Vehicular networks serve as ad-hoc networks that play a vital role in various transportation applications, including road safety, traffic management, speed control, in-vehicle infotainment services, and support for driverless cars. The traditional vehicular network architecture comprises of on-board unit (OBU), edge controllers, road-side units, centralized device control and trusted authority and communication takes place between one single localized vehicle network communicating with an edge network, which in turn passes information to a backbone network via a wired or wireless link. The different types of communication involve vehicle-to-vehicle (V2V), vehicle-to-RSU (V2R), infrastructure-to-infrastructure (I2I), and others. However, most of these do not provide much security to the data being transfer. Earlier, mechanisms such as password protected authentication, key-based entry authentication and biometric protections methods have been installed in vehicles, but they fail to validate if the data is real or spoofed.

Vehicular networks are susceptible to attacks as the traditional and commonly used authentication techniques such as password protection, biometric authentication, key-based entry do not detect if the data was real or spoofed. They just check real-time input data against the data stored in its memory. These can also be difficult to be implemented on low-powered vehicle security systems. Hence, there has been significant growth in leveraging machine learning algorithms (ML) for faster and highly accurate attack predictions in vehicle security. ML has shown great promise when used on wireless networks and a wide range of ML agorithms have been already applied on different wireless applications. My underlying understanding from existing literature is that vehicular networks have a different format compared to most other forms of data on networks, such as latency and bandwidth parameters, states (vehicles or IoT devices tend to have fixed states obtained from the devices), because of which their network activity is more predictable

Drawing from these observations, I have constructed a machine learning pipeline that undertakes the tasks of data collection, feature extraction, and data classification on the network data. The objective is to discern whether the data corresponds to an attack or not. Due to the lack of available dataset specific to vehicular networks, I have built an experimental low-scale IoT network mimicking devices installed within a car. I have setup a local router connected with various sensors, and an Android phone connected to the network as a Bluetooth-enabled user. Thus, the features in this study are tailored to exploit IoT-specific network behaviors, while also taking advantage of network flow characteristics like packet length, inter-packet intervals, and protocol. I have implemented various machine learning algorithms such as K-nearest neighbours, Support vector machine, Decision tree using Gini impurity scores, Random Forest using Gini impurity scores, along with building a deep neural network and compared the results from all for attack detection.

It was found that Random Forest, K-Nearest Neighbours and neural networks effectively identified attacks compared to other classifiers with an accuracy higher than 95%. However, deep learning classifiers may be expected to perform better with additional data from the real-world deployments. The pipeline has been created for middleboxes such as routers, firewall devices or network switches to identify anamolous

network behaviour, and not end systems. This framework is flow-based, stateless and not specific to any particular network protocol.

II. Background and Related Work

In this section, a brief introduction into network anomaly detection and middlebox properties have been discussed.

A. Vehicular Networks Communication

Vehicular network communication takes place through a combination of technologies and protocols designed to facilitate efficient and reliable data exchange between vehicles and infrastructure components. The communication in vehicular networks is primarily aimed at improving road safety, traffic management, and providing various services to drivers and passengers. Some emerging and important types of communication that takes place is low-level transferring data between nodes in V2V, V2P, V2N and V2S (vehicle-to-sensor).

V2V communication enables direct communication between nearby vehicles. It allows vehicles to exchange information, such as position, speed, acceleration, and other relevant data. This communication is crucial for enabling cooperative safety applications, such as collision avoidance, cooperative adaptive cruise control, and emergency warning systems. V2V communication is typically facilitated using short-range wireless technologies like Dedicated Short Range Communications (DSRC) or Cellular Vehicle-to-Everything (C-V2X).

V2I communication involves communication between vehicles and roadside infrastructure, such as Roadside Units (RSUs) or traffic signals. RSUs are equipped with wireless communication capabilities, enabling them to interact with vehicles in their vicinity. V2I communication provides additional information to vehicles, such as traffic conditions, traffic light status, and road infrastructure updates. This information aids in traffic management, traffic signal optimization, and other intelligent transportation system applications.

Other types of communication involves V2N (Vehicle-to-Network) where vehicles communicate with broader networks such as cloud and data centres. Vehicle-to-Cloud (V2C) communication in particular refers to communication with cloud services, accessing data from cloud-based applications and aims to benefit from their computing capabilities to perform complex tasks, such as traffic analysis.

B. Security Attack Types

Among the above mentioned vehicular communication variants, the Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication can be highly susceptible to spoofing attacks. In V2V communication, vehicles exchange information directly with nearby vehicles and due to this openness, an attacker could impersonate a legitimate vehicle by transmitting false information, such as incorrect position, speed, or acceleration data. This can affect the communication among vehicles and affect safety-critical systems such as collision avoidance systems or cooperative adaptive cruise control.

In V2I communication, vehicles interact with Roadside Units (RSUs) or traffic signals. Spoofing attacks may involve an attacker impersonating an RSU or other infrastructure components and may falsely transmit incorrect traffic information, change traffic signal timings, provide misleading data to vehicles which can result in traffic disruptions or potential safety hazards. These types of attacks are also called Man-in-the-Middle (MitM) Attacks.

Hardware based attacks may involve GPS spoofing, wherein global position data may be tampered with. Routing attacks, wherein the routing tables within the routers are altered by miscommunicating the node's presence and location. Sensor based attacks are when a vehicle manipulates it own sensor readings and uploads it to the network. Sensors can also be jammed in order to interfere with data transmission, thereby creating a false environment around the sensor and the vehicle.

One of the most widely implemented but difficult to handle attacks are DoS (Denial of Service) attacks, where an attacker can overwhelm the target system's resources, such as bandwidth, processing power, or memory, making it incapable of responding to legitimate requests or services. At present, SDN is gaining a lot of attention in the field of transporation systems. However, a centralized control provides a singular and easy point of access to launch distributed DoS attacks (DDoS) by sending multiple requests to a single SDN controller. These attacks can be ICMP flood, UDP fragmentation attack, SYN flood attack, DNS amplification attack, etc.

C. Properties of Middleboxes

Middleboxes are network devices placed strategically in the network path to inspect, monitor, and manipulate traffic passing through them. They can inspect various network-layer and application-layer attributes of the packets, such as source and destination IP addresses, port numbers, packet size, protocol types, and payload content. Middleboxes can be equipped with pattern recognition and anomaly detection algorithms. These algorithms can learn normal traffic behavior and identify deviations from the expected patterns. Since middleboxes can be implemented with rate-limiting algorithms for incoming and outgoing packets, thresholds can be set for acceptable traffic rates, thereby mitigating the impact of DoS attacks by discarding excess traffic.

Previously, researchers have explored and suggested the use of SDN based monitoring of traffic and their literature suggests that using flow-based features can be highly effective in detecting security issues without the need of deep packet inspection. Routers must be able to handle high-bandwidth traffic and thus, a given algorithm must inspect traffic anomaly based on how packets are routed and not what is in its headers. They also must be able to handle packets across multiple protocols such as TCP, UDP, HTTP, etc.

III. Model Assumptions

The model presented in this paper has made various assumptions about connected vehicles and the network architecture.The model is created in order to be able to run on middleboxes so that it can monitor traffic between connected sensors on vehicles and the rest of the Internet. All traffic traverses this middlebox and can be manipulated by it as per needs.

The goal in this paper is to detect DDoS attack from within a network, as a wider and broader network would need real-world traffic data. Each device linked to the middlebox has the capability to transmit both regular network traffic and attack traffic concurrently. Additionally, each device can execute various types of DDoS attacks sequentially, with successive attacks potentially varying in duration. The current model is focused on monitoring and detecting DoS attacks within the connected vehicles' local network itself.

IV. DDoS Attack Detection Framework

This section presents a machine learning based DDoS detection framework for Connected Vehicles as presented in Figure 1.

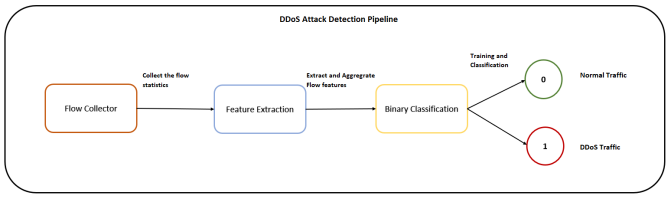

Fig. 1. DDoS Detection Pipeline

A. Dataset creation

In order to collect data for traffic classification, I set up an experimental low-scale IoT network mimicking devices installed within a car. I have connected a Raspberry Pi v3 as a WiFi access point with a GPS module, accelerometer, and a camera to collect benign and malicious IoT traffic. I have connected an Android Phone to collect HTTP and TCP data. The setup design and architecture has been presented in Figure 2.

To collect normal traffic, all connected sensors were manipulated to produce some data along with performing internet browsing for 10 minutes, while recording pcap files using Wireshark. To collect DoS traffic, I installed Kali Linux virtual machine running on a laptop as the DoS source and and a Raspberry Pi 2 as running Apache Web Server as the DoS victim. Two common classes of DoS attack namely TCP SYN flood and UDP flood was simulated using Kali Linux's hping3 utility. Using Goldeneye tool, an HTTP GET flood was also simulated. Available dataset from public IDS datasets CSE-CIC-IDS2018-AWS, CICIDS2017, CIC DoS dataset(2016) were also added to inculcate real-world data into the project. Then all of the data was combined together and randomly shuffled. This created a dataset of 12794627 data points including ddos and benign samples.

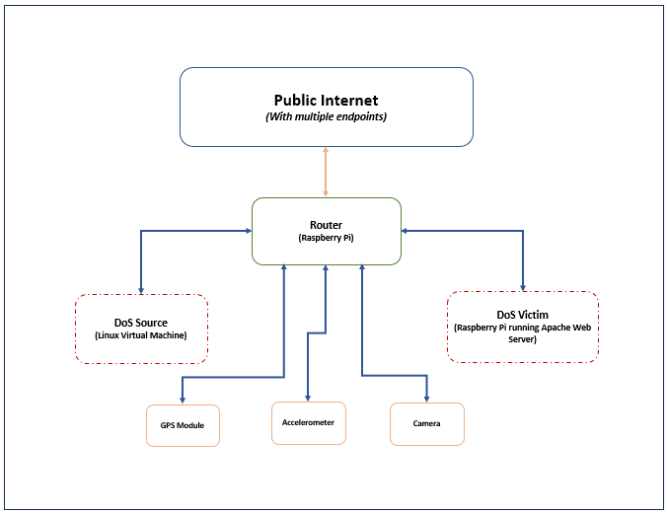

Fig. 2. Experimental Setup

B. Feature Engineering

The important features that are part of the dataset are Flow ID, Timestamp, source IP, destination IP, flow IAT min (minimum inter-arrival time), Source Port, Total Fwd Pkts (total packets forwarded), Fwd Seg Size Min (minimum size of forwarded segments) and others. Following are the observations made with respect to how different features help in distinguishing between normal and attack data:

- **Packet Size**: More than 90% of attack packets are smaller in size than normal traffic data. This indicates that during an attack like TCP SYN flood, the attacker is trying to open as many connections as possible with small packets to maximize the number of connection requests made. Compared to that, normal traffic can range from small to large packets depending on the type of data being sent.
- **Inter-Packet Arrival Time**: Normal traffic has consistent and regular inter-packet intervals and limited burstiness. As opposed to that, attack traffic has minimal or zero inter-packet intervals.
- **Source, Destination IP Addresses**: Destination IP address can be used to study attack traffic effectively. Within a particular time period, a higher count of distinct IP addresses may indicate attack traffic as the experimental setup only has a limited number of endpoints. Within a particular time period, frequent change in distinct IP addresses may also indicate attack traffic. Packets associated with attack are usually in contact with more endpoints compared to normal traffic. It is believed that ML model will be able to leverage these differences.
- **Protocol**: Since the dataset contains TCP and UDP packets primarily, I have one-hot encoded them as either TCP, UDP or HTTP since HTTP GET flood was also conducted. Unnecessary protocols that might add up noise

to the dataset has been classified as a third category of OTHER.

C. Training and Evaluation

The following five models have been applied to the traffic dataset for binary classification and the chosen hyperparameters have been listed alongside.

1. Support Vector Machine: Linear Kernel, C=1.0
2. K-Nearest Neighbours: K=6
3. Decision Tree: max_depth=5
4. Random Forest: max_depth=5, n_estimator=200
5. Multilayer Perceptron Network : 4-layer fully connected neural network with 12 neurons per layer, trained over 100 epochs with 32 batch size. Binary cross entropy loss has been used to measure loss.

All of the above models were implemented using Scikit-learn Python library. Only the Artificial Neural Network was implemented using the Keras library. Training and test sets were 80% and 20% of the complete dataset. The training was further split into 80% and 20% for training and validation purposes.

V. Project Results

In this section, I have listed down the metrics used for measuring performance and presented the results obtained from all the classification models.

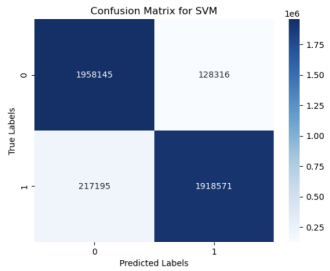

Fig. 3. SVM

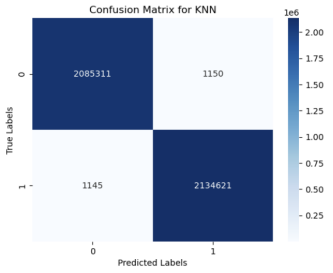

Fig. 4. K-NN

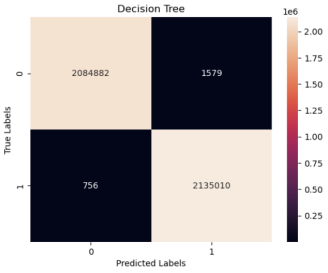

Fig. 5. Decision Tree

Figures 3-7 display the Confusion Matrices for each of the models on test data.

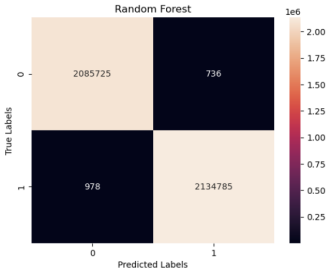

Fig. 6. Random Forest

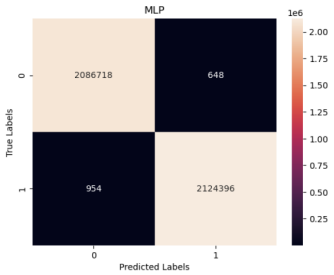

Fig. 7. MLP

Metric	SVM	KNN	MLP	Decision Tree	Random Forest
Accuracy (%)	98.94	99.94	99.97	99.89	99.91
F1-Score (%)	92.77	99.95	98.63	99.45	99.87
Precision (%)	99.21	99.95	98.33	99.68	99.95
Recall (%)	87.27	99.93	98.99	99.34	99.82

TABLE I
Traffic Classification Results

A. Metrics

- **Accuracy**: It is defined as the number of correct predictions over all data samples present in the dataset.

$$Accuracy = \frac{TP + TN}{Total\ Data\ Samples}$$

- **Precision**: It measures the accuracy of predictions made by the model.

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

- **Recall**: It is a performance metric that measures the ability of a model to correctly identify positive instances out of all actual positive instances in the dataset.

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

- **F1-Score**: This computes the harmonic mean of precision and recall.

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

B. Model Performance

Table 1 presents the performance metrics (Accuracy, F1-Score, Precision, and Recall) for five different classification models: SVM, K-Nearest Neighbors (KNN), Multi-Layer Perceptron (MLP), Decision Tree, and Random Forest.

The **SVM model** achieved a high accuracy of 98.94%, indicating that it correctly classified 98.94% of the samples in the dataset. However, the F1-Score, which is the harmonic mean of precision and recall, is relatively lower

at 92.77%. This suggests that the data is not linearly separable and may not perform as well in balancing precision and recall for both classes.

The **KNN model** achieved an outstanding accuracy of 99.94%, indicating a high percentage of correct classifications. The F1-Score is also very high at 99.95%, suggesting a well-balanced trade-off between precision and recall. This model shows excellent performance across all metrics. This indicates that the two different classes clustered well in feature-space.

The **MLP network** achieved the highest accuracy of 99.97%, indicating its ability to classify data accurately. However, the F1-Score is relatively lower at 98.63%, suggesting some imbalance between precision and recall. Despite this, the precision and recall values are still relatively high.

The **Decision Tree** model demonstrated high accuracy, precision, and recall, with F1-Score at 99.45%. This suggests that the model performs well in classifying the data with a good balance between precision and recall.

The **Random Forest** model achieved high accuracy, precision, and recall, with an impressive F1-Score of 99.87%. This indicates excellent overall performance, and the model seems to be well-suited for the given classification task.

VI. Future Work and Discussions

This paper focuses on creating a low-scale setup for a general IoT network, which can reflect connected vehicles and its sensors. It proves that simple classification algorithms and low-dimensional features can effectively distinguish normal network traffic from DDoS attack traffic. This result motivates follow-up research by creating a more real-world setting.

First, the study and setup can be replicated on larger datasets obtained in collaboration with research institutions or companies. This dataset can comprise more devices that are installed in the ECUs in vehicles across multiple other types of networks. Considering other types of traffic and its additional features will help in identifying more attack network patterns.

Conducting further experiments by exploring additional features and employing more advanced machine learning techniques beyond the ones mentioned in this paper would be of interest to me. I strongly believe that deep learning holds significant potential for anomaly detection in IoT and Connected Vehicle networks, especially in identifying attacks that are more intricate than DoS floods. My aspiration is that this research will encourage further endeavors to develop specialized network protection methods tailored specifically for IoT devices.

Another topic of study could be the risk mitigation methods undertaken one DDoS attacks are detected. Simply removing the device or halting the network connectivity may not be feasible, especially when concerned with automotive networks. Notifying the driver of the vehicle is an alternative, yet not the best safety-assuring feature.

VII. Conclusion

In this study, the effectiveness of packet-level machine learning DoS detection in accurately distinguishing normal and DoS attack traffic in the context of connected vehicles was demonstrated, albeit with a low-scale IoT network. To ensure real-time classification and middlebox deployment feasibility, I created a constrained feature set, reducing computational overhead. The feature selection was based on the assumption that network traffic patterns in connected vehicles differ from those of well-studied non-IoT networked devices.

I have evaluated five different ML classifiers using a dataset of normal and DoS attack traffic collected from an experimental connected vehicle network. Notably, all five algorithms achieved test set accuracies around 99%. These preliminary findings strongly suggest the need for further exploration of machine learning anomaly detection techniques to safeguard networks from insecure connected vehicle devices.

Acknowledgment

This study and demonstration was conducted as part of CS 656 - Computer Networks Course at the University of Waterloo and I would like to thank Professor Mohammad Salahuddin for guiding me through this topic. He has provided me with resources that helped me in this study and has taught me the basics of networking thoroughly.

References

[1] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018.

[2] Goldeneye code repository. [Online]. Available: https://github.com/jseidl/GoldenEye, 2017

[3] hping3 package description. [Online]. Available: http://tools.kali.org/information-gathering/hping3 (2017)

[4] Scikit learn: Machine learning in python. [Online]. Available: http://scikit-learn.org/stable/

[5] F. Chollet et al., "Keras," https://github.com/fchollet/keras, 2015.

[6] .Z. He, T. Zhang, and R. B. Lee, "Machine Learning Based DDoS Attack Detection from Source Side in Cloud," in Proceedings of the 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), pp. 114–120, New York, NY, USA, June 2017.

[7] P. K. Singh, S. Kumar Jha, S. K. Nandi and S. Nandi, "ML-Based Approach to Detect DDoS Attack in V2I Communication Under SDN Architecture," TENCON 2018 - 2018 IEEE Region 10 Conference, Jeju, Korea (South), 2018, pp. 0144-0149, doi: 10.1109/TENCON.2018.8650452.

[8] Khan, Zadid et al. "In-Vehicle False Information Attack Detection and Mitigation Framework using Machine Learning and Software Defined Networking." ArXiv abs/1906.10203 (2019): n. pag.

[9] A. Shiravi, H. Shiravi, M. Tavallaee, A.A. Ghorbani, Toward developing a systematic approach to generate benchmark datasets for intrusion detection, Comput

[10] A. Talpur and M. Gurusamy, "Machine Learning for Security in Vehicular Networks: A Comprehensive Survey," in IEEE Communications Surveys Tutorials, vol. 24, no. 1, pp. 346-379, Firstquarter 2022, doi: 10.1109/COMST.2021.3129079.

[11] E. Eskin, W. Lee, and W. Stolfo, "Modeling system call for intrusion detection using dynamic window sizes," 2001.