MACHINE LEARNING ON IOT MICROCONTROLLERS: A COMPREHENSIVE APPROACH FOR IOT PRIVACY AND SECURITY

Abstract

The advent of Machine Learning (ML) combined with the Internet of Things (IoT) marks a new era in technology. This advancement has opened up possibilities for Even low-end IoT devices, such as microcontrollers, to become intelligent. This study underscores the importance of implementing onboard machine learning on these limited-resource microcontroller devices.Throughout this research, we explore the application of an Intrusion Detection System (IDS) within a Wireless Sensor Network (WSN). WSNs, composed of strategically placed sensor nodes, gather environmental data. Due to their distributed nature, reliance on multi-hop data transmission, and the use of open wireless mediums, WSNs are particularly vulnerable to security breaches, making IDSs crucial for detecting and thwarting security threats. Our proposed approach entails developing an IDS workflow to be integrated into each sensor node in a WSN, enabling the detection of potential security threats. This process involves capturing radio messages via Wireshark, extracting pertinent features from the data, and applying the Random Forest algorithm for anomaly detection. We created a training dataset through simulations in COOJA and evaluated the model using Arduino Nano 33 and esp8266 modules. The study also reviews existing techniques for retraining models on microcontrollers.

Keywords: Machine Learning (ML); Internet of Things (IoT) ; Wireless Sensor Network (WSN) ; Intrusion Detection System (IDS) ; Microcontrollers ; Anomaly detection.

Author

ShekharTyagi
Discipline of Computer Science & Engineering
IIMT College of Engineering
Greater Noida, India.
shekhartyagirocks007@gmail.com.

I. INTRODUCTION

1. **Significance of ML and IoT Fusion:** The fusion of Machine Learning (ML) and the Internet of Things (IoT) marks the dawn of a revolutionary technological epoch. ML advancements have unlocked fresh potential for elementary IoT nodes, such as microcontrollers, to achieve cognitive capabilities. However, standard ML configurations are ill-suited for microcontrollers due to their intense memory and computational requirements. This study underscores the imperative of facilitating on-device machine learning for these resource-restricted microcontroller devices. In this research, we delve into the exemplar of an Intrusion Detection System (IDS) within a Wireless Sensor Network (WSN). WSNs are composed of strategically positioned sensor nodes that gather environmental data. Given their inherent vulnerabilities, such as their dispersed setup, multi-node data relays, and unsecured wireless channels, WSNs are especially prone to security breaches. This necessitates the implementation of IDSs to identify and thwart potential threats.

   Our recommended approach outlines an IDS process that can be integrated into each WSN sensor node to identify any surrounding rogue nodes. This process commences with the accumulation of radio signals via Wireshark, followed by the distillation of pertinent attributes from the amassed data. Subsequently, we employ the Random Forest technique for anomaly detection. Our training dataset was procured through COOJA simulations, and the model was evaluated using the Arduino Nano 33 and esp8266 units. Additionally, we discuss prevalent strategies for updating models on microcontrollers.

2. **Historical Context and Challenges:** Historically, standard machine learning configurations were reserved for high-end computers. These configurations are not feasible for microcontrollers due to their limited memory and computational capabilities. The merger of machine learning with the IoT signifies a transformative technological shift. Countless devices interconnected within the Internet of Things (IoT) landscape generate vast data volumes. This data is the fuel for machine learning, facilitating the extraction of meaningful patterns. By evaluating historical data, Machine Learning (ML) crafts models that anticipate forthcoming behaviors, identifying underlying patterns.

   The rise of ML has illuminated previously untapped opportunities for basic IoT components, like microcontrollers, to evolve in intelligence. Historically, conventional ML was synonymous with high-end computers characterized by rapid CPUs and GPUs, abundant RAM, or leveraging cloud-based solutions for algorithm execution. Such configurations are not feasible for microcontrollers, given their restricted memory and computational capabilities. This investigation accentuates the criticality of integrating machine learning directly onto these limited-capacity microcontroller devices.

   Diving deeper into the research, our attention is drawn to the application of an Intrusion Detection System (IDS) within a Wireless Sensor Network (WSN). WSNs are networks of sensors purposefully positioned to monitor their environment. The inherent design features of WSNs, like decentralized structures, relay-based data transmission, and unprotected wireless communication channels, render them vulnerable to potential security breaches. Hence, the paramount importance of IDSs in identifying and mitigating such threats.

In the approach we advocate for, an IDS procedure is conceptualized, which can be embedded within each WSN sensor, enabling it to recognize potential threats in its vicinity. This procedure involves the initial gathering of radio transmissions via Wireshark, followed by the isolation of significant data attributes, and then deploying the Random Forest technique for spotting irregularities. The application of our recommended IDS procedure is demonstrated through various simulations and practical tests. We also shed light on prevalent approaches to updating models within microcontroller environments. The aspiration is to elevate microcontrollers from mere data gathering instruments to devices with predictive and learning capabilities via on-device ML. In this context, Saha and colleagues [18] introduced a universally relevant procedure for crafting and implementing ML models on devices equivalent to microcontrollers. Their work is recognized as an inaugural step in the TinyML movement, highlighting the possibility of adapting advanced ML models to severely restricted environments. The authors anticipate the advent of next-gen TinyML platforms to address the challenges mentioned, which have thus far received limited attention. Yazici and team [22] undertook an analysis of the Raspberry Pi's proficiency in executing ML algorithms. They experimented with three distinct algorithms that achieved accuracy rates exceeding 80% while ensuring low energy consumption: Support Vector Machine (SVM), Multi-Layer Perceptron, and Random Forests. Prominent libraries recognized for producing streamlined C code encompass sklearn-porter [15], m2cgen [1], emlearn [17], and micromlgen [2]. These tools facilitate the development of ML applications tailored for TinyML hardware, leveraging non-neural network ML strategies like KNN, SVM, and Naive Bayes. In this process, the refined ML model is initially translated to generic C code and subsequently incorporated into a designated .h file. For those keen on implementing tree-based models such as decision trees and random forests, the SRAM-optimized method [21] is available as an efficient alternative

Application in Wireless Sensor Networks: Our study focuses on the application of an Intrusion Detection System (IDS) within a Wireless Sensor Network (WSN). Given the vulnerabilities of WSNs, the implementation of IDSs is crucial for security.

II. REVIEW OF EXISTING LITERATURE

1. **ML Integration in Microcontrollers:** The integration of ML in microcontrollers represents a shift from being mere data gathering instruments to predictive devices. Techniques like KNN, SVM, and Naive Bayes are being increasingly tailored for these environments. This chapter delves into contemporary endeavors to facilitate ML on IoT microcontrollers with limited resources. We explore various techniques to acquire and process real-time data and evaluate the array of anomaly detection methodologies currently employed.

This section sheds light on contemporary research related to IDS systems. Historically, many investigations relied on the KDD99 [3] dataset. However, the KDD99 dataset is somewhat archaic, lacking consideration for the most recent classifications of cyberattacks.

To enhance the precision in detecting and classifying the four distinct Denial of Service (DoS) attacks—Blackhole, Grayhole, Flooding, and Scheduling—a specialized

dataset tailored for WSN has been developed [5]. This research gives significant weight to the LEACH protocol, one of the frequently adopted hierarchical routing strategies in WSNs. There's a structured approach to source data from the Network Simulator 2 (NS-2) and refine it to yield 23 attributes. This curated dataset is christened WSN-DS. Employing this dataset, an Artificial Neural Network (ANN) was trained to discern and classify various DoS incursions. In scenarios devoid of attacks, the classification precision for the Blackhole, Flooding, Scheduling, and Grayhole attacks stood at 92.8%, 99.4%, 92.2%, 75.6%, and 99.8% respectively.

In another study [11], a testing environment mirroring the IoT landscape was crafted using the Node MCU ESP8266, the DHT11 sensor, and a wireless router. A computer system was deployed to simulate an antagonistic entity carrying out eavesdropping and data tampering operations. Sensors collected metrics related to temperature, humidity, and deadlines, subsequently transmitting them to the ThinkSpeak platform via a wireless gateway. During periods of regular activity, the Node MCU retrieves sensor readings and dispatches them to Think Speak's server, where they're archived under the label "normal data." As the communication ensues between Node MCU and the Think Speak server, the adversary covertly captures and modifies the data. In the malicious phase, ARP Poisoning is harnessed to instigate a Man in the Middle disruption within the network, and the compromised data is tagged as "attack data." Machine learning models, encompassing techniques like Naive Bayes, SVM, decision trees, and Adaboost, are then employed to segregate the data into "normal" and "attack" categories.

2. **Data Acquisition in Real-time:** Historically, many studies relied on outdated datasets. However, new datasets tailored for WSNs have been developed, focusing on modern classifications of cyber-attacks.

III. METHODOLOGY AND TECHNICAL OVERVIEW

1. **Understanding Wireless Sensor Networks:** WSNs are designed for specific environmental monitoring, relaying the acquired data for further analysis. They are characterized by their resource limitations and prioritize efficient networking.

2. **Introduction to 6LoWPAN:** 6LoWPAN is an adaptive protocol designed to optimize IPV6 utilization over energy-efficient wireless sensor networks by introducing an adaptation layer [4]. It employs a myriad of compression techniques to streamline IPV6, originally cumbersome for energy-constrained networks, ensuring its seamless integration in such environments. Recognizing the battery-driven nature of many WSNs, the IEEE 802.15.4 standard emphasizes energy efficiency by introducing variable data rates. Some protocols, particularly IPV6, can be bandwidth-intensive due to their extensive headers. 6LoWPAN mitigates this by enabling header compression and fragmentation, thus effectively reducing the IPV6 overhead [19].

3. **The Utility of WSN:** WSNs find applications in diverse sectors, originating from military needs, which also catalyzed the conceptualization of sensor networks, these networks have found utility in healthcare for patient monitoring, and in commercial sectors for tasks like seismic activity monitoring, wildlife tracking, and weather forecasting.

Industrial applications of WSN encompass asset monitoring, rapid emergency response systems, automated climate control in structures, and more.

4. **Hurdles in WSN Deployment:** WSNs face a slew of challenges, from bandwidth constraints, limited battery life, to restricted processing capabilities [13]. Data transmission and inter-node communication are the primary energy-consuming operations. Mobile sensor nodes face challenges integrating with the internet due to their mobility. Challenges include bandwidth constraints, limited battery life, and security concerns.

5. **Techniques for Data Harvesting and Feature Distillation:** Security remains a paramount concern for WSNs, especially when deployed in vulnerable settings. The wireless nature of these networks makes them susceptible to unauthorized access and manipulation. Moreover, the lack of interoperability across protocols and components can make systems more vulnerable

IV. AN OVERVIEW OF COOJA

COOJA is a Contiki OS-based network simulator that emulates genuine hardware ecosystems, focusing on network behaviors. COOJA, a Contiki OS-based network simulator, facilitates the emulation of genuine hardware ecosystems and predominantly emphasizes network behaviors. With COOJA, one can simulate a WSN without the need for a specific mote. The simulator extends support for numerous standards, including the TR 1100, TI CC2420, and more. For our research, COOJA aids in simulating WSNs, producing both regular and malicious data for model calibration and evaluation.

1. Insights on Wireshark: Wireshark captures packets across network connections, recording vast streams of data interactions. It provides an avenue to visualize entire communication streams and sequences. For our study, Wireshark is pivotal in intercepting radio communications across sensor nodes, storing the data streams as PCAP files.

V. DELVING INTO RANDOM FOREST

Random Forest is an ensemble learning methodology that uses decision trees for classification, providing robust solutions for high-dimensional data categorization. Random Forest, an ensemble learning methodology, harnesses decision trees for classification. It amalgamates multiple decision trees, trained through the Bagging method [40]. Each tree's output determines the overall classification, enhancing the accuracy and mitigating overfitting issues associated with individual trees. Random Forest offers a robust solution, especially for high-dimensional data categorization, with its ability to manage noise and anomalies. It operates as a data-driven, non-parametric classifier, crafting classification rules from the provided data. The generalization capability of the system, alongside the similarity and correlation of decision trees, are pivotal metrics determining the efficiency of random forests.

VI. ADAPTIVE LEARNING TECHNIQUES

To address the dynamic shifts in feature distribution post-deployment, it's essential to regularly refine the model operating on the device [12]. Transmitting models, periodically trained on updated server data to the microcontroller, might not always be feasible due to limited bandwidth and potential privacy concerns. Traditional training methods that rely on GPUs are not suited for microcontrollers, given that their training memory and energy consumption often surpass those required for inference [7]. This has led to the emergence of various federated learning (FL) and on-device training (ODT) systems tailored for microcontrollers.

1. **Learning Directly on the Device:** Typically, frameworks designed for on-device training segment the learning process into three pivotal phases. Initially, the system must determine when there's a notable shift in the input data (identifying the need to learn). Metrics such as the continuous mean and variance of incoming data or prediction confidence scores can assist in identifying variances in primary feature components. Next, considering the device's constraints and available training samples, the framework must initiate model adjustments (deciding the learning approach). Lastly, for optimizing learning, especially in transfer learning scenarios to avert sudden model degradation, the framework must discern which training samples are most beneficial (selection of learning data). Common techniques include prioritizing samples based on their gradient magnitude, using strategies like weighted replay or importance sampling, or oversampling underrepresented classes [12] [20]. Regrettably, most renowned TinyML platforms don't support on-device model graph modification, rendering them incompatible with these training frameworks. Ensuring that resource boundaries aren't breached, these training systems are typically compatible with networks that offer simple and defined architectural choices. To simplify these models further, additional memory constraints might be integrated into NAS systems.

2. **Distributed Learning via Federation:** Federated Learning amplifies on-device training, adapting it for a distributed setting. Federated Learning (FL) amplifies on-device training, adapting it for a distributed, non-IID setting. In this approach, edge devices refine the shared model parameters locally, dispatch the localized model versions to a central server, and subsequently receive a consistent, amalgamated model – all while ensuring data remains on the edge devices [6] [14]. For FL to be successful, the communication framework connecting the server and edge devices must be both efficient and robust. While methods like FedAvg offer the needed reliability and efficiency, other FL strategies leverage pruning or class probability knowledge distillation [10]. Communication protocols such as bidirectional gRPC and WebSocket, as offered by platforms like Flower [6] and DIoT [16], facilitate synchronous, low-latency communication between the central servers and clients. To be truly versatile, FL frameworks should seamlessly handle devices with varying computational and communication capacities. These frameworks should be adept at identifying resource metrics and task completions. Platforms like Flower [6] integrate a virtual client engine dedicated to task scheduling and resource optimization. Moreover, these systems should also strategize based on the recognized metrics.

VII. ENVISIONED IDS FRAMEWORK

In our advanced approach, we have proposed integrating an IDS mechanism within each sensor node. This ensures that every node remains vigilant, identifying potential threats in its vicinity, and promptly alerting the central system upon intrusion detection. This IDS mechanism is essentially a series of algorithms meticulously designed to operate efficiently on resource-limited microcontrollers

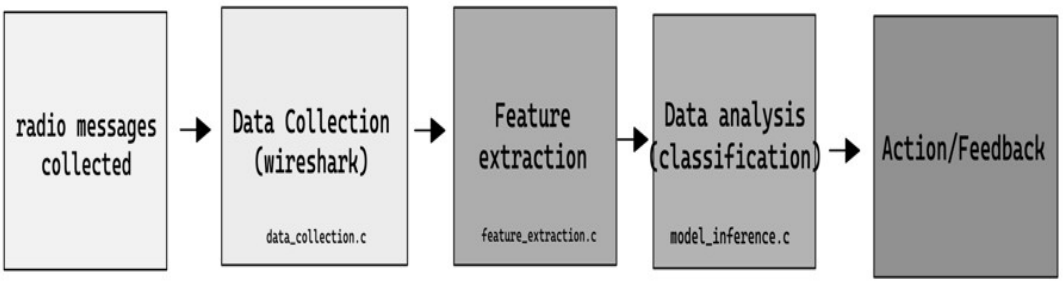


Figure 1: IDS System Workflow

We start by capturing packets from the incoming network stream directed to the host node, storing them as a PCAP file within a short time frame. This PCAP file is then processed by a script that examines the data and extracts relevant information. This consolidated data is subsequently input into the pre-trained ML model, which generates an anomaly score for every data instance. Should a breach be detected within the network, an alert is relayed to the control center based on this score. The entire process is illustrated in Figures 1, 2, and 3.

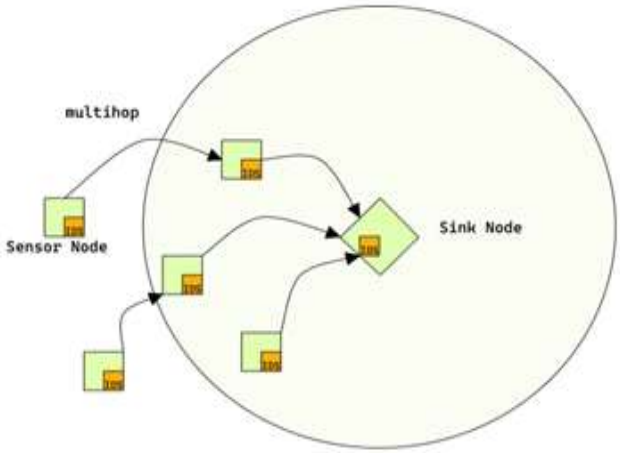


Figure 2: WSN without Malicious node

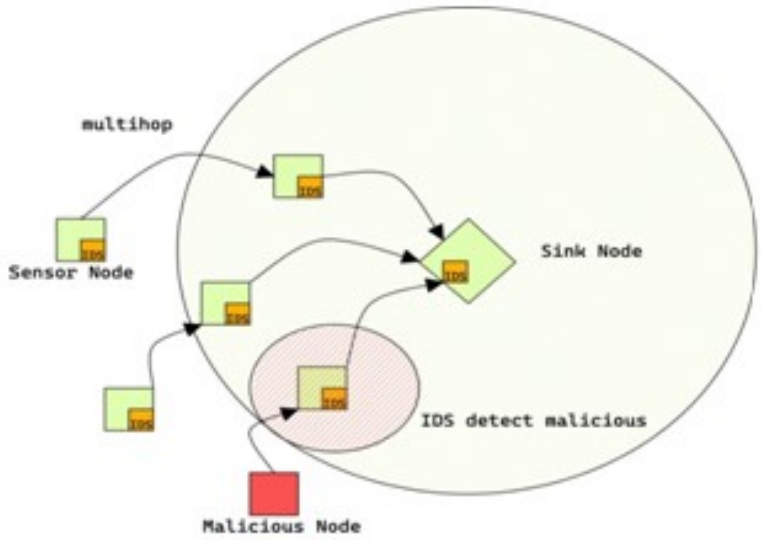


Figure 3: WSN with Malicious node

We first simulate anormal condition and collect the data.Fig.4 shows as emulation of WSN containing one sink and twenty sensor nodes.

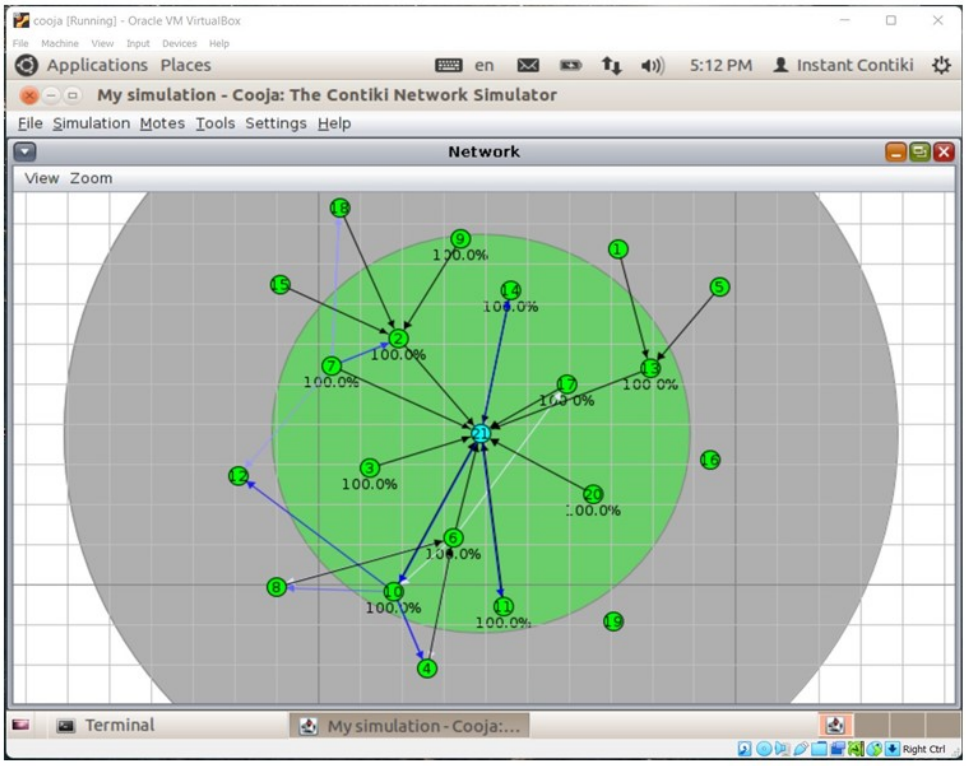


Figure 4: COOJA:1 Sink node(cyan)+20Sensor node(green)

1. **WSN with Malicious node:** Next, we simulate a WSN with one malicious node and collect the data for same time interval. Fig. 5 shows a simulation of WSN containing one sink, twenty normal sensor nodes and one malicious node.

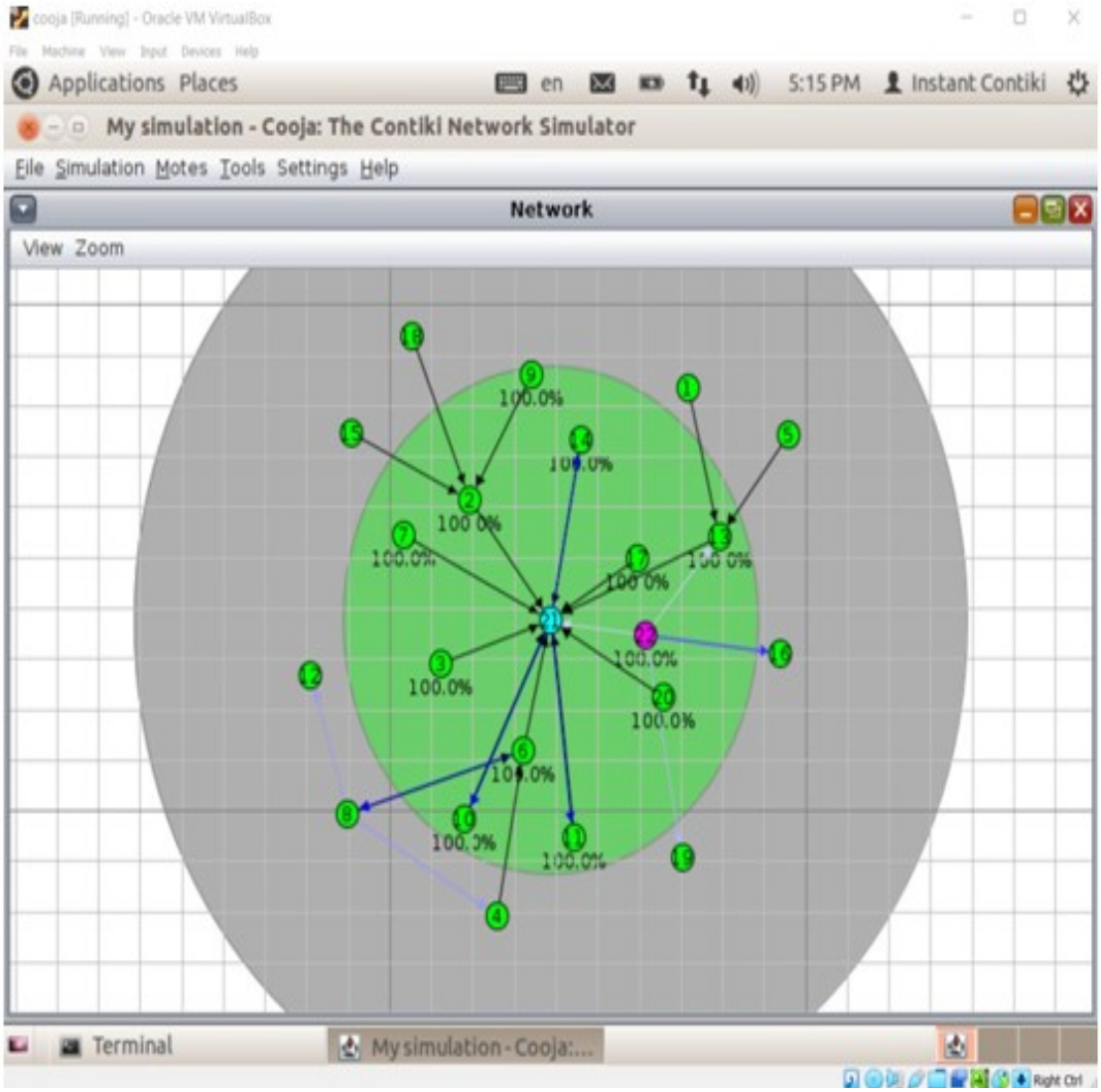


Figure 5: COOJA: 1 Sink node(cyan)+20Sensornode(green)+1MaliciousNode(purple)

2. **Radio Messages:** In the COOJA console, we can collect the payload of radio messages (Fig.6) in the WSN and save the packet capture (PCAP)file.

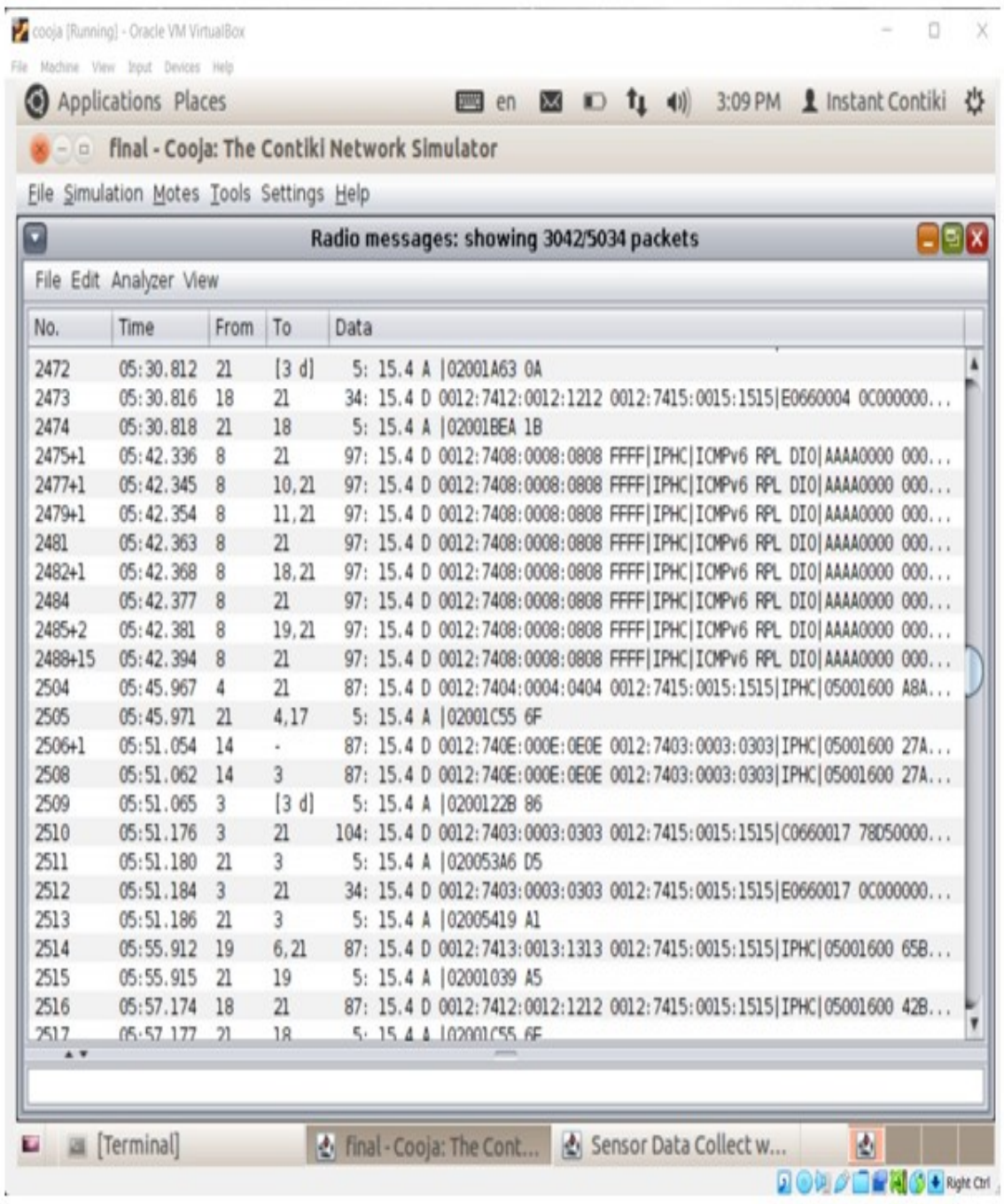


Figure 6: COOJA: Radio messages being collected in COOJA tool

Feature Extraction: Wireshark is a network protocol analyzer, or an application that captures packets from a network connection. It continuously listens to a network connection and then captures whole streams of communication, sometimes tens of thousands of packets at a time. It also allows you to visualize entire conversations and

network streams. Fig. 7 shows a visualization of part of our WSN streams in COOJA environment.

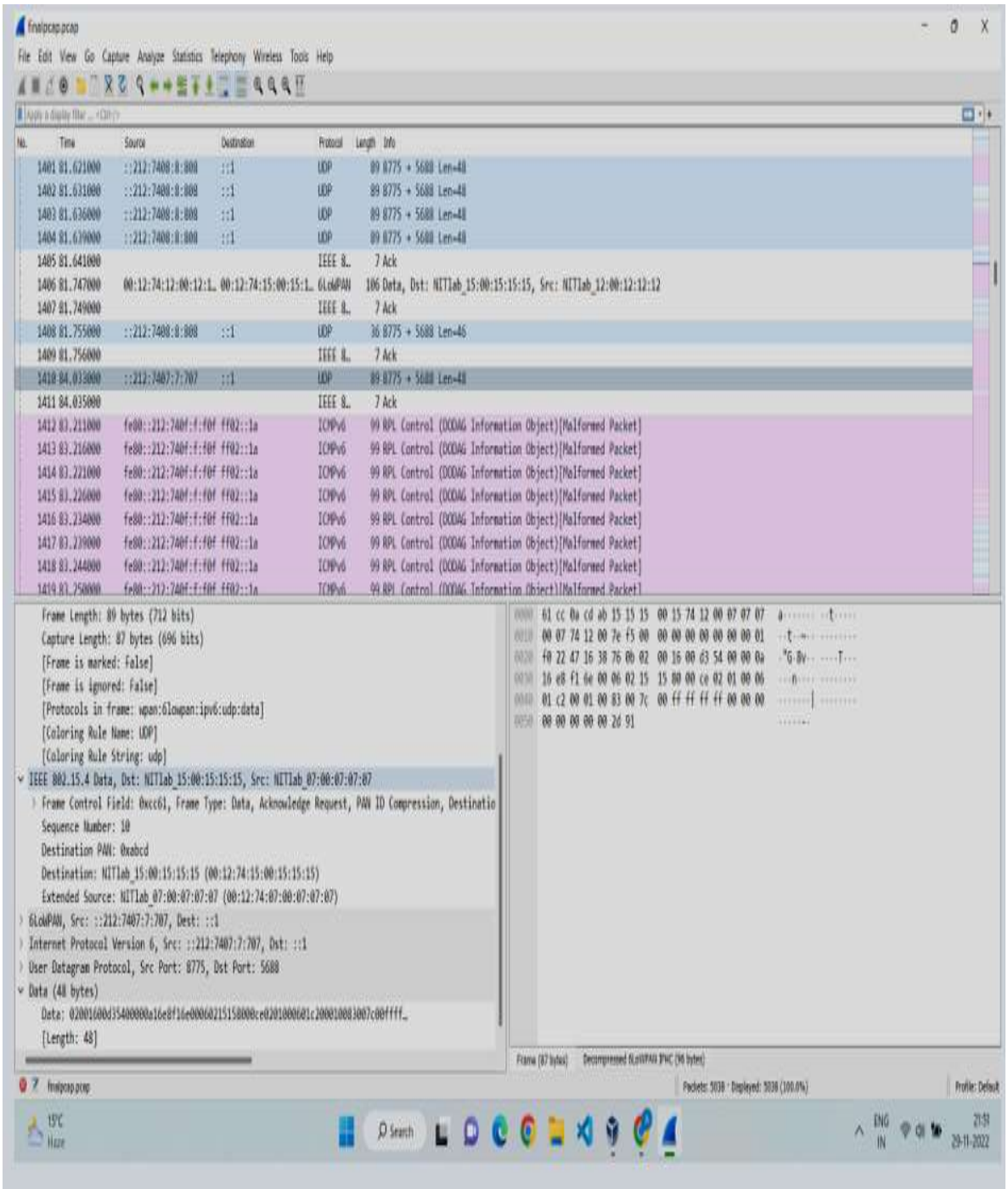
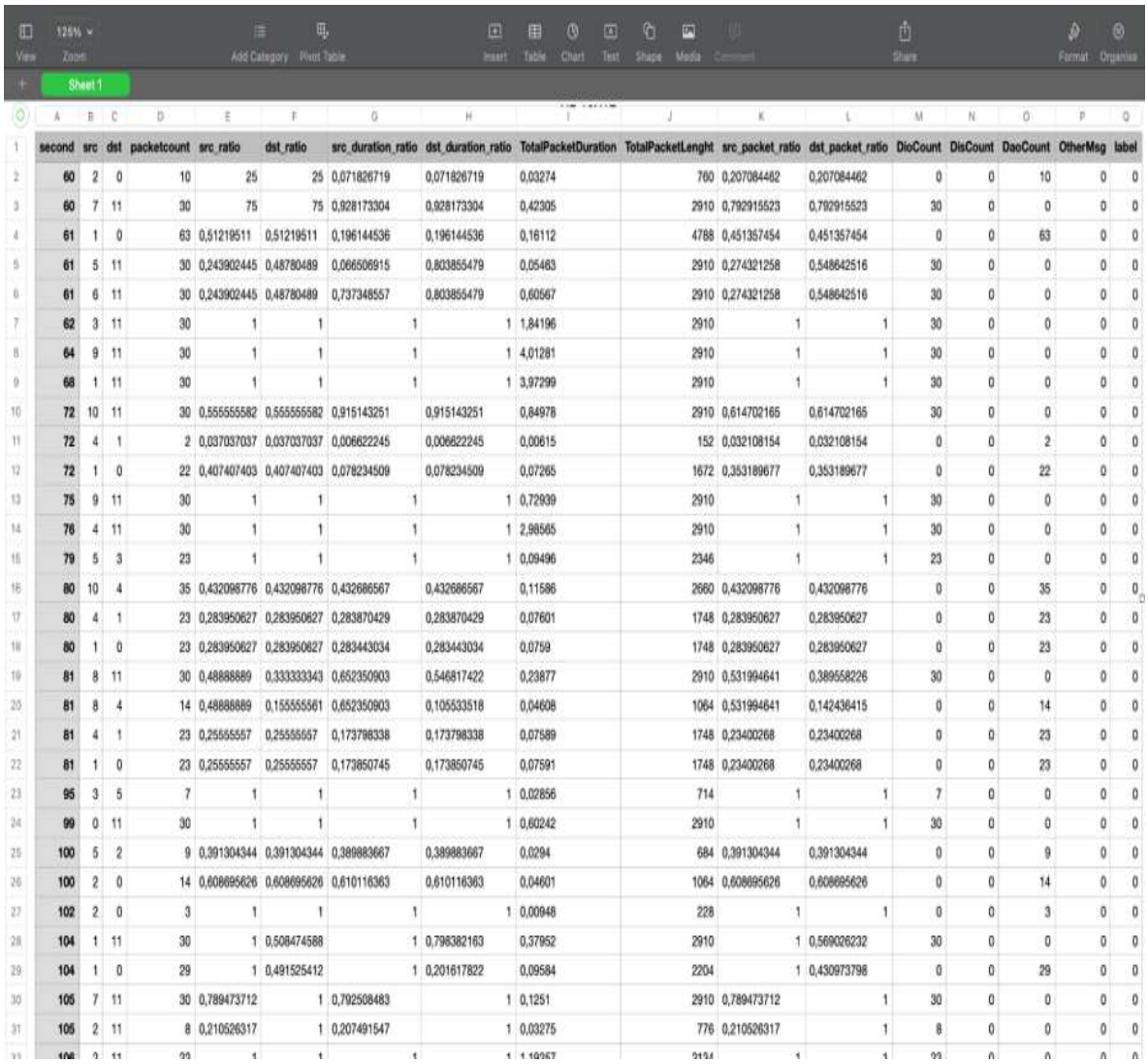


Figure7: Wire shark: Network streams collected from COOJA as PCAP file

Finally, the PCAP data can be converted to standby doing some operations we can extract about 23 features from it as shown in Fig.8



second	src	dst	packetcount	src_ratio	dst_ratio	src_duration_ratio	dst_duration_ratio	TotalPacketDuration	TotalPacketLenght	src_packet_ratio	dst_packet_ratio	DioCount	DisCount	DaoCount	OtherMsg	label
60	2	0	10	25	25	0,071826719	0,071826719	0,03274	760	0,207084462	0,207084462	0	0	10	0	0
60	7	11	30	75	75	0,928173304	0,928173304	0,42305	2910	0,792915523	0,792915523	30	0	0	0	0
61	1	0	63	0,51219511	0,51219511	0,196144536	0,196144536	0,16112	4788	0,451357454	0,451357454	0	0	63	0	0
61	5	11	30	0,243902445	0,48780489	0,066506915	0,803855479	0,05463	2910	0,274321258	0,548642516	30	0	0	0	0
61	6	11	30	0,243902445	0,48780489	0,737348557	0,803855479	0,60567	2910	0,274321258	0,548642516	30	0	0	0	0
62	3	11	30	1	1	1	1	1,84196	2910	1	1	30	0	0	0	0
64	9	11	30	1	1	1	1	4,01281	2910	1	1	30	0	0	0	0
68	1	11	30	1	1	1	1	3,97299	2910	1	1	30	0	0	0	0
72	10	11	30	0,555555582	0,555555582	0,915143251	0,915143251	0,84978	2910	0,614702165	0,614702165	30	0	0	0	0
72	4	1	2	0,037037037	0,037037037	0,006622245	0,006622245	0,00615	152	0,032108154	0,032108154	0	0	2	0	0
72	1	0	22	0,407407403	0,407407403	0,078234509	0,078234509	0,07265	1672	0,353189677	0,353189677	0	0	22	0	0
75	9	11	30	1	1	1	1	0,72939	2910	1	1	30	0	0	0	0
76	4	11	30	1	1	1	1	2,98565	2910	1	1	30	0	0	0	0
79	5	3	23	1	1	1	1	0,09496	2346	1	1	23	0	0	0	0
80	10	4	35	0,432098776	0,432098776	0,432686567	0,432686567	0,11586	2660	0,432098776	0,432098776	0	0	35	0	0
80	4	1	23	0,283950627	0,283950627	0,283870429	0,283870429	0,07601	1748	0,283950627	0,283950627	0	0	23	0	0
80	1	0	23	0,283950627	0,283950627	0,283443034	0,283443034	0,0759	1748	0,283950627	0,283950627	0	0	23	0	0
81	8	11	30	0,48888889	0,333333343	0,652350903	0,546817422	0,23877	2910	0,531994641	0,389558226	30	0	0	0	0
81	8	4	14	0,48888889	0,155555561	0,652350903	0,105533518	0,04608	1064	0,531994641	0,142436415	0	0	14	0	0
81	4	1	23	0,25555557	0,25555557	0,173798338	0,173798338	0,07589	1748	0,23400268	0,23400268	0	0	23	0	0
81	1	0	23	0,25555557	0,25555557	0,173850745	0,173850745	0,07591	1748	0,23400268	0,23400268	0	0	23	0	0
95	3	5	7	1	1	1	1	0,02856	714	1	1	7	0	0	0	0
99	0	11	30	1	1	1	1	0,60242	2910	1	1	30	0	0	0	0
100	5	2	9	0,391304344	0,391304344	0,389883667	0,389883667	0,0294	684	0,391304344	0,391304344	0	0	9	0	0
100	2	0	14	0,608695626	0,608695626	0,610116363	0,610116363	0,04601	1064	0,608695626	0,608695626	0	0	14	0	0
102	2	0	3	1	1	1	1	0,00948	228	1	1	0	0	3	0	0
104	1	11	30	1	0,508474588	1	0,798382163	0,37952	2910	1	0,569026232	30	0	0	0	0
104	1	0	29	1	0,491525412	1	0,201617822	0,09584	2204	1	0,430973798	0	0	29	0	0
105	7	11	30	0,789473712	1	0,792508483	1	0,1251	2910	0,789473712	1	30	0	0	0	0
105	2	11	8	0,210526317	1	0,207491547	1	0,03275	776	0,210526317	1	8	0	0	0	0

Figure 8: Extracted Features from PCAP File

VIII. MODEL SELECTION

We choose a group of lightweight ML models fit for anomaly detection and use the extracted data to train and test on these different algorithms. In our project, following model shave been chosen:

- Gaussian Naïve Bayes
- Decision Tree
- Random Forest Classifier
- Support Vector Machine
- Logistic Regression
- Gradient Boosting Classifier
- Artificial Neural Network

Among these model, Random Forest Classifier gives the best result.

Hence, we move forward with the same.

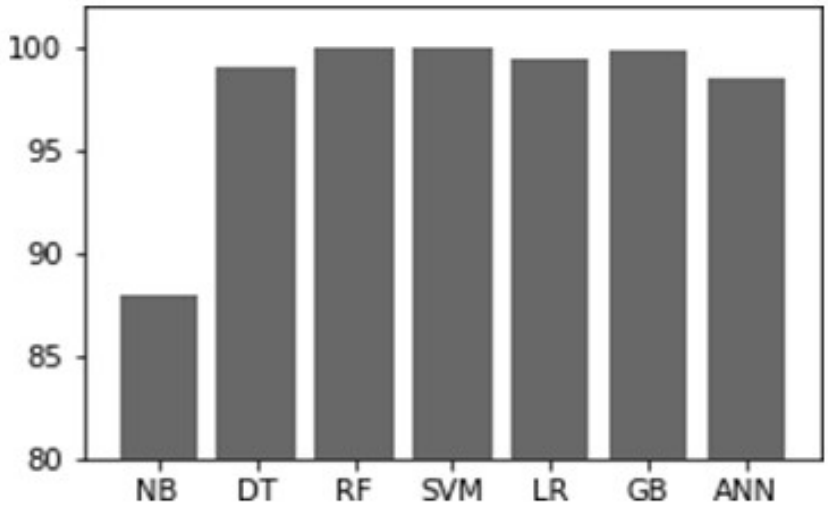


Figure 9: Training Accuracy

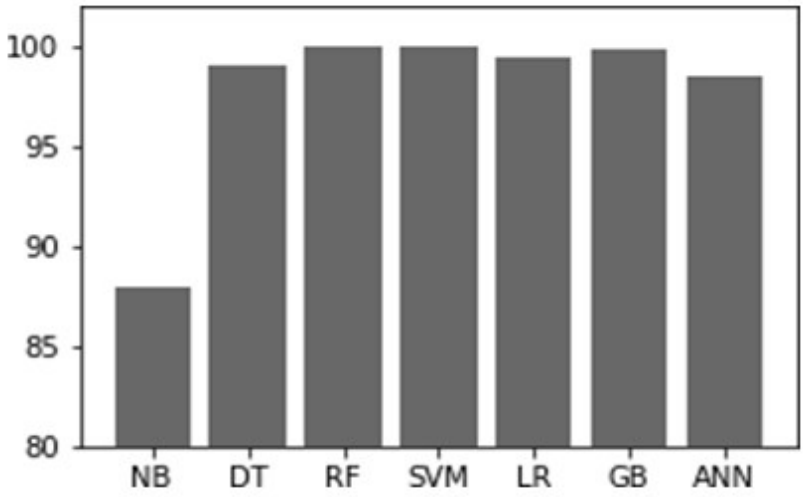


Figure10: Testing Accuracy

IX. FINAL REFLECTIONS AND PROSPECTIVE DEVELOPMENTS

1. **Final Thoughts:** The amalgamation of IoT with ML is pivotal in today's technological landscape. Our research presented a methodology for an intrusion detection system for WSNs, achieving a remarkable accuracy rate of 99.9% in anomaly detection. This chapter delves into the conclusions drawn from our research findings and provides a glimpse into the potential avenues for future work in this domain.

   The amalgamation of IoT with ML stands as a pivotal advancement in today's technological landscape. Given the inherent limitations like minimal battery life and processing capabilities of microcontrollers, incorporating ML within them remains a subject of intense exploration. In our research, we presented a methodology to devise an intrusion detection system for wireless sensor networks, leveraging proven technologies

at each juncture. Our proposed solution, upon rigorous testing, exhibited promising results. By harnessing 23 distinct features extracted from network stream data, we implemented a Random Forest classifier, achieving a remarkable accuracy rate of 99.9% in anomaly detection.

We perceive our efforts as an introductory phase in the realm of TinyML, with ample room for further advancements. Even with the current constraints

2. **Looking Ahead:** The field of TinyML is burgeoning, with the potential for further advancements and optimizations. The ongoing developments in on-device and federated learning open avenues for more innovations. , there's a vast scope to refine and optimize ML models tailored for such environments. As technology progresses, we foresee the evolution of frameworks capable of supporting intricate models on future microcontrollers, even with their inherent resource limitations. Additionally, the importance of real-time adaptive learning for these ML models cannot be overstated. Given that these microcontrollers might be deployed in remote or hard-to-reach areas, the prospect of continuously updating them with new data becomes a challenge due to retraining constraints. As the domain is still in its nascent stages, the ongoing developments in on-device and federated learning beckon researchers to innovate and devise novel strategies and frameworks.

REFERENCE

[1] m2cgen. Code-generation for various ml models into native code, 2020.
[2] micromlgen. MicroML is an attempt to bring Machine Learning algorithms to microcontrollers.
[3] Kdd 99 data set, Accessed Feb 14, 2018.
[4] Charu C Aggarwal, Naveen Ashish, and Amit Sheth. The internet of things: A survey from the data-centric perspective. In *Managing and mining sensor data*, pages 383–428.Springer, 2013.
[5] Iman Almomani, Bassam Al-Kasasbeh, and AL Mousa. Akhras. wsnds: A dataset for intrusion detection systems in wireless sensor networks. *Journal of Sensors*, 2016.
[6] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, Pedro PBde Gusmão, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.
[7] Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. Tinytl: Reduce memory, not parameters for efficient on-device learning. *Advances in Neural Information ProcessingSystems*, 33:11285–11297, 2020.
[8] Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt. Contiki-a lightweight and flexible operating system for tiny networked sensors. In *29th annual IEEE international conference on local computer networks*, pages 455–462. IEEE, 2004.
[9] Arfah A Hasbollah, Sharifah HS Ariffin, and M Ismi A Hamini. Performance analysis for 6lowpan ieee 802.15. 4 with ipv6 network. In *TENCON 2009-2009 IEEE Region Conference*, pages 1–5. IEEE, 2009.
[10] Ahmed Imteaj and M Hadi Amini. Fedparl: Client activity and resource-oriented lightweight federated learning model for resource-constrained heterogeneous iot environment. *Frontiers in Communications and Networks*, page 10, 2021.
[11] KVVNL Sai Kiran, RN Kamakshi Devisetty, N Pavan Kalyan, K Mukundini, and R Karthi. Building a intrusion detection system for iot environment using machine learning techniques. *Procedia Computer Science*, 171:2372–2379, 2020.
[12] Seulki Lee and Shahriar Nirjon. Learning in the wild: When, how, and what to learn for on-device dataset adaptation. In *Proceedings of the 2nd International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things*, pages34–40, 2020.
[13] Mamoona Majid, Shaista Habib, Abdul Rehman Javed, Muhammad Rizwan, Gautam Srivastava, Thippa Reddy Gadekallu, and Jerry Chun-Wei Lin. Applications of wireless
sensor networks and internet of things frameworks in the industry revolution 4.0: A systematic literature review. *Sensors*, 22(6):2087, 2022.

[14] Akhil Mathur, Daniel J Beutel, Pedro Porto Buarque de Gusmao, Javier Fernandez-Marques, Taner Topal, Xinchi Qiu, Titouan Parcollet, Yan Gao, and Nicholas D Lane. On-device federated learning with flower. *arXiv preprint arXiv:2104.03042*, 2021.

[15] Darius Morawiec. sklearn-porter. Transpile trained scikit-learn estimators to C, Java, JavaScript and others.

[16] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, N Asokan,and Ahmad-Reza Sadeghi. Dïot: A federated self-learning anomaly detection system for iot. In *2019 IEEE 39th International conference on distributed computing systems (ICDCS)*, pages 756–767. IEEE, 2019.

[17] Jon Nordby. emlearn: Machine Learning inference engine for Microcontrollers and Embedded Devices, March 2019.

[18] Swapnil Sayan Saha, Sandeep Singh Sandha, and Mani Srivastava. Machine learning formicrocontroller-class hardware–a review. *arXiv preprint arXiv:2205.14550*, 2022.

[19] Zach Shelby and Carsten Bormann. *6LoWPAN: The wireless embedded Internet*. John Wiley & Sons, 2011.

[20] Bharath Sudharsan, John G Breslin, and Muhammad Intizar Ali. Imbal-ol: Online machine learning from imbalanced data streams in real-world IoT. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 4974–4978. IEEE, 2021.

[21] Bharath Sudharsan, Pankesh Patel, John G Breslin, and Muhammad Intizar Ali. Ultra-fastmachine learning classifier execution on iot devices without sram consumption. In *2021IEEE International conference on pervasive computing and communications workshops and other affiliated events (PerCom Workshops)*, pages 316–319. IEEE, 2021.

[22] Mahmut Taha Yazici, Shadi Basurra, and Mohamed Medhat Gaber. Edge machine learning: Enabling smart internet of things applications. *Big data and cognitive computing*, 2(3):26, 2018.

