PRIVACY PRESERVED ONLINE DDOS ATTACK DETECTION FRAMEWORKS FOR IOT SYSTEMS

.

by

YONAS KIBRET BESHAH

Dissertation Submitted as Partial Fulfilment for the Requirements of

Degree of Doctor of Philosophy in

Cybersecurity
to

College of Technology and Built Environment

School of Information Technology and Engineering (SiTE)

ADDIS ABABA UNIVERSITY

Dec 22, 2025

CERTIFICATION

This is to certify that the dissertation prepared by **Yonas Kibret Beshah** entitled" **Privacy Preserved Online DDoS Attack Detection Frameworks for IoT Systems**" and submitted as a partial fulfilment for the Degree of Doctor of Philosophy complies with the regulations of the University and meets the accepted standards with respect to originality, content and quality.

Signed by Examining Board:

SGC, Chairperson Signature, Date

_____

External Examiner Signature, Date

_____

External Examiner Signature, Date

_____

Internal Examiner Signature, Date

_____

DECLARATION

I hereby declare that this dissertation entitled" **Privacy Preserved Online DDoS Attack Detection Frameworks for IoT Systems**" was prepared by me, with the guidance of my advisors. The work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted, in whole or in part, for any other degree or professional qualification. Parts of this work have been published in Journal of Electronics, Journal of EEE Access and Journal of Scientific Reports.

Author: Signature, Date

Yonas Kibret Beshah

Witnessed by:

Name of student advisor: Signature, Date

Surafel Lemma (PHD)

Name of student co-advisor: Signature, Date

Henock Mulugeta (PHD)

ABSTRACT

Internet of Things (IoT) security is becoming important with the growing popularity of IoT devices and their wide applications. IoT systems are widely used in a variety of sectors, including transportation, utilities, manufacturing, healthcare and home automation. Although IoTs promise to have a significant positive impact on productivity and efficiency, they also pose several privacy and security issues. One of the most destructive attacks on the IoT is Distributed Denial-of-Service (DDoS) attacks. Machine learning-based DDoS attack detection systems have proven effective in detecting and preventing DDoD attacks in IoT systems. However, these DDoS attack-detection systems are batch learning and centralized learning which usually fails to detect zero-day DDoS, and adversarial attacks, and preserve privacy. The dynamicity IoT environment causes concept drift issues that result in performance degradation in detecting DDoS. Despite the rapidly increasing use of federated learning in cyber security domain to address privacy issue, existing methods have limitation in terms of accuracy, convergence speed, and scalability in non-IID (non-independent and identically distributed) condition. Furthermore, the current adversarial defenses are tailored to detect known adversarial attacks by training on predefined attack patterns.

On this dissertation, we first proposed an adaptive online DDoS detection framework to tackle concept drift in streaming data using a novel Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE), that achieves detection accuracies of 99.54% and 99.33% on the IoTID20 and CICIoT2023 datasets, respectively. AUWPAE outperforms other state-of-the-art online adaptive learning methods, such as ARF-ADWIN, ARF-DDM, SRPs-ADWIN, SRPs-DDM, KNN-ADWIN, HTs, LB, and PWPAE. AUWPAE address different type of concept drift issue and detect zero-day attacks. Second, our dissertation introduces a novel Multi-Stage Adversarial Attack Defense (MSAAD) mechanism that combines resilient adversarial purification, diversified classifier ensembles, and a Multi-Armed Bandit selection strategy to mitigate known and unknown adversarial threats in real-time. This defense system substantially improves model robustness, with adversarial detection accuracy rising up to 99.48% across the same datasets. Third, a novel Dynamic Weighted Clustered Federated Learning (FedDWC) framework is developed to enhance detection accuracy and convergence under non-IID conditions by leveraging bi-level optimization and performance-based dynamic weight updates across clustered clients. Theoretical analysis demonstrates fast convergence of the FedDWC framework. Moreover, the experiment demonstrates the clustering capability and scalability of proposed framework for different size and

complexity of IoT devices. FedDWC outperforms conventional FL methods like FedAvg, FedProx, and IFCA, with accuracy gains up to 1.9% on the same above dataset. Collectively, this dissertation contributes a privacy-preserving, robust to adversarial attack and scalable online DDoS attack detection system that advance the state-of-the-art through a synergy of adaptive learning, adversarial resilience, and federated optimization.

Keywords: IoT DDoS attack, privacy preserving, adversarial attack, online learning, federated learning, dynamic weighting, IoT DDoS attack, privacy preserving, concept drift detection and adaptation, AUWPAE, MSAAD and FedDWC

ACKNOWLEDGMENTS

I would like to express my deepest and most sincere gratitude to Surafel Lemma (PHD), my main supervisor, for his essential advice, unwavering support and intellectual mentorship throughout my PhD journey. His perceptive remark, constructive criticism, and unwavering encouragement significantly shaped the direction and quality of this research.

I am equally grateful to my co-supervisor Henock Mulugeta (PHD), for his invaluable comments and advice. His expertise and encouragement played a crucial role in refining this work and overcoming challenges encountered during the research process.

Above all, I want to express my heartfelt appreciation to my family, whose love, patience and sacrifices made this achievement possible. You were my strength when I felt exhausted and my motivation when the journey felt overwhelming. This accomplishment is as much yours as it is mine.

Finally, I thank God for granting me the strength, determination, and wisdom to complete this journey. This dissertation serves as a testament to the collective support, guidance, and love I received along the journey.

Table of Contents

List of Table

List of Figures

List of Abbreviations

AUWPAE	Accuracy Update Weighted Probability Averaging Ensemble
AI	Artificial Intelligence
ANN	Artificial Neural Network
BLSTM	Bidirectional Long Short-Term Memory
BNN	Binarized Neural Network
BPTT	Back Propagation Through Time
C&C	Command and Control
CNN	Convolutional Neural Network
CPS	Cyber Physical System
FedDWC	Dynamic Weighted Clustered Federated Learning
DBM	Deep Boltzmann Machine
DBN	Deep Belief Network
DDoS	Distributed Denial of Service
DL	Deep Learning
DNN	Deep Neural Network
DoS	Denial of Service
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
DT	Decision Tree
FDL	Federated Deep Learning
FL	Federated Learning
FNN	Feedforward Neural Network
FN	False Negative
FP	False Positive
GRU	Gated Recurrent Unit
ICS	Industrial Control System
IoT	Internet of Things
kNN	k-Nearest Neighbor
MLP	Multilayer Perceptron
MSAAD	Multi-Stage Adversarial Attack Defense
ML	Machine Learning
NB	Naive Bayes
non-IID	non-independent and identically distributed
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network

Chapter One

1. Introduction

The Internet of Things (IoT) has grown exponentially in the past ten years by transforming ordinary objects into smart and intelligent ones that can work together to make decisions. The desire for smart applications and devices that can work autonomously without requiring human involvement has been one of the main drivers in this field. The development of efficient applications, enhanced communication protocols, and breakthroughs in embedded systems design, along with end-user demand have all contributed to the acceleration of IoT growth. It is predicted that the number of connected IoT devices used around the world would increase by 12% annually on average from 27 billion in 2017 to 125 billion in 2030[1]. These devices play a pivotal role in various sectors such as healthcare, manufacturing, power distribution, environmental monitoring, and smart cities.

Despite the wide range of applications, IoT remains highly vulnerable to cyber threats due to a lack of built-in security mechanism, its resource-constrained devices and open wireless communication mechanism [2]. Lack of built-in security features similar to those found in traditional systems has been one of the most obvious vulnerabilities of the IoT. This vulnerability results from the fact that the majority of embedded devices lack the computational power necessary to implement sophisticated security procedures and encryption. Manufacturers of IoT devices often have given more emphasis to prototype, production, and packaging without giving security much attention due to increased market competitiveness[3]. Due to their wireless nature, IoT are open, and any new devices can connect to them automatically. This characteristic of IoT allows attackers to access the IoT devices easily, making them susceptible to a variety of attacks, particularly DDoS attack[4]. According to the NETSCOUT 2020 report, there was a 126% rise in DDoS attacks and 31% increase in throughput attacks in the first half of this year. A prominent example is the Mirai botnet attack of 2016, which exploited vulnerable IoT devices such as IP cameras and home routers to orchestrate one of the largest DDoS attacks in history, reaching 1.2 Tbps[5]. It exposed the systemic risks of unsecured IoT deployments. Similarly, IoT DDoS incidents continue to rise, including Mozi and Hajime botnets that underscoring urgent need for robust, adaptive, and real-time defense mechanisms.

Traditional information security measures like encryption, authentication, and access control are not sufficient to defend against DDoS attacks in IoT systems [6]. As a result, an effective DDoS

attack detection solution is required to supplement current security measures. DDoS attack detection monitors any network traffic records generated within IoT systems in order to detect DDoS attacks[7]. Such systems can be developed and implemented at IoT network gateways and alert network managers and prevent DDoS attacks. In this regard, machine learning-based detection systems have become one of the promising solutions. Several machine learning-based DDoS attack detection systems in IoT utilize a centralized approach[8], which transfers network traffic data from all IoT edge nodes to a central cloud server for model training and detection. This strategy does not protect the privacy of network device owners, as it collects data at the IoT edge nodes and transfers it through the network. In addition, these models make use of batch learning, which depends on the availability of complete datasets for training. Batch learning is not the best approach for real-time, streaming IoT environments. These systems are computationally intensive and suffer from performance degradation due to concept drift, where the statistical properties of data change over time. The system also gives incorrect classification results when adversarial attacks are used to deceive the models.

Existing DDoS detection methods rely primarily on batch learning approaches, which are ill-suited to handle the dynamic nature of streaming data. These methods struggle to adapt to rapid changes in data distributions and make them ineffective in detecting various types of concept drift in dynamic IoT environments. The adversarial defense methods are designed for batch learning in DDoS attack detection. This approach is ineffective in resource-constrained, dynamic IoT environments, where concept drift detection and adaptation are critical. Furthermore, the current adversarial defenses are tailored to detect known adversarial attacks by training on predefined attack patterns. Despite the rapidly increasing use of federated learning in cyber security domain to address privacy issue, existing methods like FedAvg, FedProx and IFCA have limitation in terms of accuracy, convergence speed, and scalability in non-IID condition. FedAvg performs uniform averaging without considering the data distribution. This leads to a slow convergence speed and performance degradation under non-IID IoT environments. FedProx partially addresses the convergence issue by introducing a proximal term. However, it still experiences moderate accuracy degradation and limited scalability due to fixed regularization parameters. IFCA achieves high local personalization by clustering clients but incurs high computational and communication overhead as the number of clients and clusters grows.

To address these challenges, this dissertation introduces privacy preserved online DDoS attack detection frameworks for IoT systems. The proposed solution consists of three novel components

that work in synergy to address the combined issues of adversarial robustness, online drift adaptation, and privacy-preserving DDoS attack detection for the IoT. The Multistage Adversarial Attack Defense (MSAAD) model is a multi-stage adversarial defense pipeline beginning with RADP (adversarial attack detector and purifier), proceeding through multiple classifier, and finally adaptive classifier selection by means of Multi-Armed Bandits (MAB) algorithm with Thompson Sampling (MABTSE). In this system, Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) plays two key roles as an independent online drift-adaptive model, and as an adaptive ensemble approach to combine classifier predictions in adversarial settings. FedDWC is the decentralized backbone of the system by securely combining cleaned, performance-ordered updates from edge devices in a privacy-preserving fashion. Clustering and dynamic weighting allow Dynamic Weighted Clustered Federated Learning (FedDWC) to deal with non-IID data and enhance convergence without loss of privacy.

First component, novel AUWPAE framework to detect DDoS attacks using real-time data streaming and address concept drift issue. The AUWPAE ensemble adapts to both gradual and sudden concept drifts using drift detection methods such as ADWIN [9] and DDM [10]. Base leaners Base leaners such as ARF [11], SRPs [12], and KNN [13] are used to construct the ensemble framework. The framework is evaluated on two benchmark IoT datasets: IoTID20 and CICIoT2023 datasets. The results show that the proposed adaptive online DDoS attack detection framework is able to detect DDoS attacks on IoT systems with an accuracy of 99.32% and 99.25% for the two datasets, respectively, demonstrating strong performance under streaming conditions.

Second component, a MSAAD framework to ensure robustness against adversarial attacks. The MSAAD framework comprises three key stages: (1) Resilient Adversarial Detector and Purification (RADP), (2) multiple classifiers, and (3) an ensemble defense method. The RADP defense mechanism detects and purifies adversarial attacks targeting an online DDoS detection system, from both multiple and unknown attacks. It consists of a Resilient Adversarial Detector (RAD) and Purification Modules. The RAD module detects adversarial attacks by leveraging the strengths of Generative Adversarial Networks (GANs) and multisource adversarial perturbations (MSAP). A GAN was used to reduce the gap between the training data and the real-world data. MSAP crafts various adversarial attack methods to create diverse perturbations to train an adversarial attack detector. The pruning method is used to purify adversarial perturbations to enhance the robustness of online learning models by systematically removing detected data points using RAD. The multi-classifier defense method is designed to complicate the replication of the

DDoS attack-detection model by adversarial attackers. These methods incorporate a diverse array of classifiers to introduces greater uncertainty, making it more challenging for adversarial strategies to succeed. The MABTSE is utilized to dynamically select the optimal classifier or ensemble of classifiers for each incoming traffic request. MABTSE has been applied for online DDoS attack detection in conjunction with the AUWPAE ensemble approach [9]. Experimental validation on the same datasets shows that the accuracy of the DDoS detection system improved from 32.38%–60.58% to 99.39%–99.48% for IoTID20 and from 66.60%–86.20% to 99.01%–99.14% for CICIoT2023 in adversarial scenarios.

Third component, FedDWC proposed to address the limitations of centralized ML approaches to preserve user privacy and non-IID data issue. FedDWC integrate model personalization and knowledge sharing by clustering similar clients to learn shared models and formulating bi-level optimization of the learning process across distributed IoT environments. Moreover, the framework dynamically adjusts the weight based on the performance of the local model for each IoT device. This approach preserve data privacy, improves detection accuracy and reduces convergence time in the face of evolving DDoS attacks. We present a theoretical analysis to demonstrate the convergence property of the proposed framework. The experimental results show that the proposed FedDWC framework outperforms other state-of-the-art methods: FedAvg, FedProx, and IFCA in terms of convergence and DDoS attack detection accuracy under non-IID data conditions. In terms of accuracy, FedDWC achieved improvements of 1.9%,1.31%, and 1.01% over FedAvg, FedProx, and IFCA, respectively, when using the IoTID20 non-IID dataset of 10 clusters and 200 IoT devices.

The overarching goal of this research is to develop a distributed, adaptive, and secure DDoS detection system for IoT environments that is resilient to evolving threats, adversarial attacks, and concept drift while ensuring data privacy and efficient learning.

1.1. Statement of the Problem

Most of the existing DDoS attack detection models are batch learning-based machine learning techniques. Batch learning techniques frequently require access to a complete dataset for model training. Learning massive IoT datasets demands a significant amount of time for retraining, computational resources, and memory due to the real-time nature of the environment. Online learning is more appropriate than batch learning techniques for IoT DDoS attack detection given the real-time nature of IoTs [8]. Concept drift is challenge which is caused due to the dynamic

nature of IoT environments. Concept drift makes already trained models less useful in recognizing zero-day attacks. A good detection model must accurately detect and adapt to observed drifts in order to prevent concept drift and maintain high prediction accuracy. Ensuring the robustness of online learning models against adversarial attacks is crucial for maintaining the security and reliability of IoT systems. Without adequate protection, these models can continuously learn from compromised data, leading to gradual degradation in performance and an increased risk of security breaches [12]. The current adversarial defenses detect only known adversarial attacks by training on predefined attack patterns. Moreover, DDoS attack detection in IoT environment poses unique challenge due to privacy concerns in centralized environment and non-IID nature of IoT data. Federated Learning (FL) is a promising collaborative learning approach that address privacy issue [13], [14]. The efficacy of FL is often challenged by the non-IID nature of the data across networked devices[15]. Data in FL inherently exhibit variations in feature space, label distributions, and quantity, which are compounded by emerging divergences such as concept drift and temporal shifts[16]. These challenges degrade the performance in terms of detection accuracy and slow convergence of FL algorithms such as FedAvg[17]. In addition, within the group of IoT devices, low-performing IoT devices negatively affect the global model, as FedAvg averages all local models equally[18].

In particular, in this research, we aim to address the problem of batch learning and concept drift, adversarial defenses strategies that detects only known adversarial attacks and privacy that arises while using a centralizes DDoS detection approaches. In this regard, we have proposed three frameworks, a robust and adaptive online learning Accuracy Update Weighted Probability Averaging Ensemble framework (AUWPAE) for detecting DDoS attacks under real-time streaming conditions, a Multistage Adversarial Attack Defense (MSAAD) framework using Resilient Adversarial Detector and Purification (RADP), multiple classifiers, and an ensemble defense method to protect the online DDoS detection system against adversarial attack and Dynamic Weighted Clustered Federated Learning framework (FedDWC) that incorporates dynamic weighting and client clustering to preserve privacy and enhanced accuracy and convergence under non-IID conditions.

1.2. Research question

This dissertation aims to contribute to ongoing research efforts in developing robust, online, privacy-preserving DDoS attack detection framework in IoT environments. This research is guided by the following questions:

- **RQ1:** How effective are adaptive online learning methods at detecting DDoS attacks accurately in real-time with minimal computational and memory overhead?
- **RQ2:** How can multi-stage defense mechanisms be designed to improve the robustness of online DDoS detection systems against multiple and unknown adversarial attacks?
- **RQ3**: To what extent do dynamic weight averaging and clustered federated learning techniques preserve detection accuracy and ensure fast convergence under non-IID data conditions while maintaining data confidentiality?

1.3. Objective of the Study

The objective of this study is to design a distributed, online DDoS detection framework for IoT environments that preserve user privacy, adapts to the evolving attack patterns and is robust to adversarial attack. The specific objectives are:

- **OB1:** Develop novel adaptive online DDoS detection Accuracy Update Weighted Probability Averaging Ensemble framework (AUWPAE) capable of detecting zero-day attacks using real-time network streaming.
- **OB2:** Develop multi-stage adversarial defense (MSAAD) strategies that address robustness of online DDoS attack detection model against multiple and unknown adversarial attack in IoT network.
- **OB3:** Design and develop privacy- preserving Dynamic Weighted Clustered Federated Learning Framework (FedDWC) with high classification accuracy, low communication cost and fast convergence in non-IID environments.

1.4. Contribution of the Study

This research offers both theoretical and practical contributions towards developing a online, privacy-preserving, and adversarial resilient DDoS detection system for IoT environments. The following key contributions of this dissertation published in three journal articles, namely MDPI, IEEE and Springer.

- A robust and adaptive online learning Accuracy Update Weighted Probability Averaging Ensemble framework (AUWPAE) for detecting DDoS attacks under real-time streaming conditions.

  Y. K. Beshah, S. L. Abebe, and H. M. Melaku, "Drift Adaptive Online DDoS Attack Detection Framework for IoT System," Electronics (Basel), vol. 13, no. 6, p. 1004, Mar. 2024, doi: 10.3390/electronics13061004.

- A comprehensive adversarial defense framework called Multistage Adversarial Attack Defense (MSAAD) framework using Resilient Adversarial Detector and Purification (RADP), multiple classifiers, and an ensemble defense method.

  Y. Kibret Beshah, S. Lemma Abebe, and H. Mulugeta Melaku, "Multi-Stage Adversarial Defense for Online DDoS Attack Detection System in IoT," IEEE Access, vol. 13, pp. 72657–72673, 2025, doi: 10.1109/ACCESS.2025.3560186.

- A novel privacy-preserving Dynamic Weighted Clustered Federated Learning framework (FedDWC) that incorporates dynamic weighting and client clustering for enhanced accuracy and convergence under non-IID conditions.

  Y.K. Beshah, Abebe, S.L. & Melaku, H.M. Dynamic weight clustered federated learning for IoT DDoS attack detection. Sci Rep 15, 34036 (2025). https://doi.org/10.1038/s41598-025-13204-y

Theoretical Contributions

- **CB1:** A novel Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) framework for online DDoS detection, which addresses gradual drift and sudden concept drift issues in a dynamic environment, is proposed.
- **CB2:** A novel Multistage Adversarial Attack Defense (MSAAD) framework is designed to protect online DDOS attack detection systems from multiple and unknown adversarial attacks. An online DDoS attack detection model utilizing a Multi-Armed Bandit (MAB) approach with Thompson Sampling (MLBTSE) that dynamically selects the optimal classifier or ensemble of classifiers for each incoming traffic request.
- **CB3:** A novel Dynamic Weighted Clustered Federated Learning Framework (FedDWC framework that preserve privacy in IoT systems. FedDWC incorporates dynamic weighting and client clustering for enhanced accuracy and convergence under non-IID conditions. We formulated an optimization problem using a bi-level optimization framework to create an efficient solution for distributed learning environments and

performed both a theoretical analysis and extensive experiments across various IoT settings involving DDoS attacks.

The main practical contributions of these research:

- **PC1:** Real-time protection of IoT systems from DDoS attacks, including zero-day DDoS attack and ensure uninterrupted and reliable service delivery for critical applications.
- **PC2:** Increased robustness of production-grade IoT DDoS detection systems against adversarial attacks and make them more secure and resilient.
- **PC3:** Enhanced protection of IoT end-user privacy by eliminating the need to transmit sensitive data across the network.

1.5. Organization of the Thesis

The rest of this thesis is organized as follows. Chapter 2 presents literature review and related work on IoT and IoT architecture, DDoS attack, datasets, challenge of the existing DDoS attack detection and related works. Chapter 3 presents overall proposed solutions. Chapter 4 Drift Adaptive Online DDoS Attack Detection Framework for IoT System. Chapter 5 Multi-Stage Adversarial Defense for online DDoS attack detection system in IoT. Chapter 6 Dynamic Weight Clustered Federated Learning (FedDWC) framework. Chapter 7 presents proposed solutions deployment location in real-world environment. Chapter 8 presents the conclusion and future research direction.

Chapter Two

2. Research methodology

This chapter describes the methodological framework guiding the development and validation of the proposed privacy-preserved online DDoS detection frameworks. The research adopts a hybrid methodological approach that integrating theoretical analysis and experimental validation. The approach ensures that each proposed framework (AUWPAE, MSAAD, and FedDWC) is theoretically sound and empirically validated under dynamic IoT environment. The methodology follows a structured paradigm encompassing research design, data strategy, analytical techniques, and validation protocols, providing a comprehensive blueprint for investigating the research questions while maintaining academic rigor and reproducibility.

2.1. Research philosophy and approach

This research bases on practical research philosophy that prioritizes problem-solving through the integration of theoretical reasoning and empirical validation. The approach applies both deductive and inductive. The deductive approach applies established theories of machine learning, concept drift, adversarial robustness, and federated optimization to design novel frameworks. The inductive approach derives new insights and refinements from experimental observations. This philosophy acknowledges the complexity of IoT security problems that require solutions. The solutions are not only theoretically justified but also empirically proven to work under dynamic, real-world constraints. The cyclical process of theory-informed design followed by experimental validation forms the core of the methodological approach.

2.2. Research design

The research employs a multi-strategy research design combining design science, computational experimentation, and comparative analysis. The design is structured in three coherent components corresponding to each corresponding research objective. First component, involved the theoretical formulation and experimental evaluation of the AUWPAE framework. AUWPAE utilizes streaming data to test its drift detection and adaptation capabilities in order to detect zero-day DDoS attack. Second component, focused on testing MSAD framework against adversarial attack, where theoretical defense mechanisms were subjected to empirical attack

scenarios. Third component, verifies the theoretical development and validation of the FedDWC framework, and convergence analysis with federated simulation experiments.

2.3. Data requirements and sources

The empirical validation required datasets that accurately reflect the heterogeneous and dynamic nature of IoT network traffic. The dataset includes both normal behaviour and sophisticated DDoS attacks. CICIoT2023 and IoTID20 are two publicly available benchmark datasets were selected as primary data source. CICIoT2023 dataset is prominent for its comprehensiveness and generated from real IoT devices. The IoTID20 dataset is recognized for its focused collection of IoT-specific Mirai-botnet DDoS attacks. These datasets were chosen for their empirical richness, containing the necessary variability in attack patterns, traffic features, and temporal dynamics to carefully evaluate model performance, robustness, and generalization. This provides an empirical foundation to closely approximates real-world IoT environments and ensure experimental reproducibility.

2.4. Data collection and synthesis methods

The network traffic data are collected from the static datasets. The research involved significant data synthesis and transformation to be suit for the specific experimental setup. The proposed MSAD framework is tested against adversarial attack, detailed in Chapter 5. The adversarial attacks are generated using state-of-the-art attack algorithms (FGSM, BIM, PGD, C&W) and GANs. FedDWC framework were tested by changing the original datasets algorithmically to create synthetic non-IID data distributions across simulated clients. This synthesis is critical for empirically testing the frameworks under non-IID data conditions, detailed in Chapter 6.

2.5. Sampling and experimental partitioning strategy

A stratified temporal sampling approach was used for online learning experiments to preserve the chronological order and inherent concept drift. The data streams sequentially spited an approximate 70:30 for training and testing. For federated learning simulations a controlled non-IID partitioning strategy was implemented using a Dirichlet distribution. Dirichlet distribution is used to create artificially realistic data skew across simulated IoT devices. This methodological step was essential for empirically investigating the core challenge of non-IID data in federated

environment. It is used also to evaluate fairly FedDWC framework's clustering and weighting mechanisms against established baselines.

2.6. Data preparation and pre-processing

A standardized and replicable pre-processing pipeline was applied to ensure data quality and comparability. This includes cleaning, encoding, normalization, and feature selection. The details for data preparation and pre-processing is presented in subsequent chapters. This pre-processing transforms the raw dataset into a form suitable for both theoretical analysis and experimental computation. This directly impact the empirical results of model training and evaluation.

2.7. Tools, instruments, and computational environment

The proposed online DDoS detection frameworks is designed to detect DDoS attacks targeting IoT systems. To evaluate the performance of the proposed frameworks, we developed a prototype using the Python 3.10 within the Jupyter Notebook environment. The River library [117] is utilized for data stream analytics, and addresses concept drift through machine learning. The Adversarial Robustness Toolbox (ART) [118] employed to simulate adversarial attacks. The experiment was conducted on a machine running equipped with Intel(R) Xeon(R) CPU @2.20GHz with 16 GB of RAM. This technical infrastructure provided the necessary environment for implementing theoretical algorithms and executing empirical tests efficiently and reproducibly.

2.8. Data Analysis and evaluation methods

The analysis employs evaluation strategy encompassing both theoretical analysis and empirical metrics. The convergence of FedDWC is analysed theoretically and regret bounds is derived for AUWPAE convergence analysis. comprehensive set of quantitative metrics are used to analyse empirically. Standard classification metrics like accuracy and F1-Score, efficiency measures like latency and throughput, robustness indicators like attack success rate and robust accuracy, and federated learning-specific measures like communication rounds and convergence speed are used to analyse empirically. This combination provides a holistic assessment of each framework's performance.

2.9. Validity, reliability, and trustworthiness

This research uses several strategies to ensure validity and reliability. Validity comes from measuring the proposed frameworks with different metrics. Internal validity is improved through controlled experiments, ablation studies, and repeated trials. External validity is supported by testing on two different, complex datasets, which increases the generalizability of the findings. Reliability is ensured by thoroughly documenting all procedures, parameters, and software versions, allowing for exact replication. The combination of theoretical proof with empirical validation strengthens the overall trustworthiness of the research outcomes and shows that the proposed frameworks are not only innovative in concept but also effective in practice.

Chapter Three

3. Literature Review and Related Works

3.1. Internet of Things

The Internet of Things (IoT) is a new revolution in internet communication. The term "Internet of Things' was coined in 1999 by British technology pioneer Kevin Ashton, co-founder of the Auto-ID Laboratory at MIT. IoT is made up of a variety of heterogeneous smart objects (also known as things), such as smartphones, and intelligent network infrastructure. It described as any object with computing, storage, and communication capabilities connected via a number of different communication protocols, such as Bluetooth, Wi-Fi, ZigBee, NB-IoT, LoRa, and GSM. The widespread use of IoT technologies makes it easier to enhance many aspects of our lifestyle. IoT-enabled smart appliances, home automation, intelligent energy management, electronic healthcare employing wearable medical devices, etc., revolutionize consumer' lifestyles while offering an improved standard of living style. Intelligent IoT systems are widely used in the industry sector to optimize production and enhance services.

IoT has brought a significant number of technological advances in our daily lives. The major objective of this intelligent technology is to assist objects to connect anytime, anywhere, with everything via a variety of links. The IoT concept is implemented using a variety of technologies and sensors. Radio frequency identification (RFID)[19], near field communication (NFC)[20], wireless sensor networks (WSN)[21], and other communication technologies are utilized to implement the IoT concept. IoT has been implemented in numerous applications. One of the areas where IoT used is healthcare. In healthcare, IoT sensors are used to measure a person's body temperature, blood pressure, and heart rate[22]. Smart home is another application where people use various electrical devices at home, including fans, heaters, air conditioners, microwave ovens, and refrigerators. Animal tracking is another IoT application. An animal's body is equipped with GPS sensors to make easy tracking. In addition to these, there are many IoT applications, including smart retail, smart building, smart agricultural, smart power grid and smart infrastructure management.

IoT has a lot of benefits, but it also has significant drawbacks, including poor administration, energy efficiency, identity management, security, and privacy [11]. The most important concerns with IoT development are security and privacy. In IoT, all devices are connected to the Internet

since they cannot function without it. On the Internet, many hackers take sensitive information from objects in the IoT. Users may suffer significant losses as a result of attackers' exploitation of information in any unlawful manner to suits their needs[23].

3.2. IoT Architectures

There is no universally accepted IoT design standard. Researchers have proposed various architectures to model IoT systems. Some academics claim that the IoT design has three layers, while others state that IoT has four-layer architecture. Other believe that the three-layer architecture cannot satisfy the needs of applications because of improvements in IoT. The five-layer design was introduced as a solution to address security and privacy issue in earlier models.

3.2.1. Three Layer Architecture

The three-layer architecture is relatively simple and aligns with fundamental IoT principle. It was proposed during early days of IoT development[24][25]. As it is shown in Figure 1, the three-layer architecture of IoT, which comprised perception, network, and application layers.

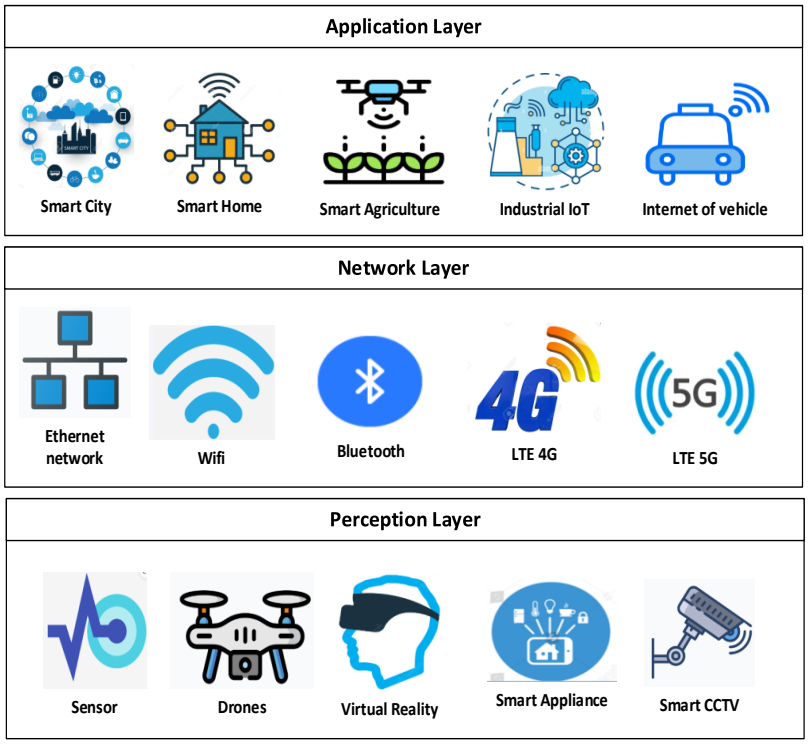

Figure 1: Three-layer Architecture

a) Perception layer

Perception layer is sometimes referred to as a sensor layer. It functions like human eyes, ears, and noses. It collects physical data using sensors such as RFID tags, 2D barcodes, and other sensor types, based on application requirements.

The sensors are chosen according to the needs of the applications. These sensors can collect data of location, changes in the environment, motion, vibration, and other things. Sensors are the major focus of attackers, as a result, sensors are the source of the majority of threats[26][27]. Security risks in perception layer include:

Eavesdropping: an unauthorized real-time attack where an attacker intercepts private communication such as phone conversations, text messages, fax transmissions, or video conferences, to steal data transferred through a network.

Node Capture: a critical threat where an attacker gains complete control over a key node, such as a gateway. The data saved in memory of the node could be leaked when the node is under control [28].

Fake Node: involves adding a false node to the system and inputting false data. It seeks to stop sharing accurate information. In order to destroy the network, an attacker might add a fake node in order to consume the valuable energy of actual nodes.

Replay Attack: is also called a playback attack. It is a sort of attack where a third-party intercept conversation between a sender and a receiver in order to steal the sender's real information. An intrusive party communicates with the victim using the same authenticated information that has already been sent to him by providing evidence of his identity and authenticity. The recipient may see it as a legitimate request and carry out the intruder's instructions because the message is encrypted [29].

Timing Attack: is applied on devices with limited computing power. By analyzing how long it takes the system to reply to various requests, inputs, or cryptographic algorithms, an attacker can find flaws and extract secrets kept from a system[30].

b) Network Layer

Perception layer is referred as a sensor layer. Network layer serves as a bridge between the perception layer and the application layer. It collects data from sensor sends the data to application layer. Wire-based or wireless technology can be used as the transmission medium. Additionally,

it assumes responsibility for establishing a connection between various networks, network devices, and smart things. It is therefore quite vulnerable to attacks. It has significant security challenges with the authenticity and integrity of the data transmitted across the network. Security issues of network layer are:

DDoS \ DoS Attack: A DoS attack attempts to block legitimate users from using IoT or other network resources. In order to prevent some or all authorized users from using the targeted IoT or network resources. It is often performed by flooding them with excessive requests. A denial of service attack is one in which the target IoT is drained by a single computer or system. Because, a IoT may have good resources, it can withstand a denial-of-service attack. To attack such IoT, the intruders use a lot of computers to target a single IoT and exhaust it so that it can no longer serve its users[31]. A distributed denial of service attack is one that employs numerous computer systems to initiate such attacks.

MiTM (Man-in-the-Middle) Attack: A MITM attack occurs when the attacker deceives the sender and receiver into thinking they are speaking directly to one another by secretly intercepting and changing their communication. An attacker can alter messages to meet their needs because they have control over the communication.

Attack on Storage: Users' personal data kept on storage devices or in the cloud. Attackers may target storage devices or cloud in order to change user information for attacker purposes.

Exploit Attack: Any immoral or illegal attack on software or hardware IoT device known as an exploit. It makes use of security vulnerabilities in software, hardware, or operating systems of IoT. It frequently attempts to take control of the system and steals network-stored data[32].

c) Application Layer

Any application that makes use of IoT technology or where IoT has been implemented is referred to as an application layer. Smart cities, smart homes, smart health, animal tracking, etc. are only a few examples of IoT applications. IoT systems have numerous threats and vulnerabilities from both inside and outside, particularly when used to build smart homes. One of the key challenges in implementing robust security in an IoT-based smart home is that the ZigBee-based devices commonly utilized in these environments. The ZigBee-based devices have poor processing capability and little storage[33]. Common security challenges and threats on this layer are:

Site-to-Site Scripting: is an injection attack. A client-side script, such as a java script, can be inserted by an attacker and placed on a reliable website that is accessed by other users. Hacker has the ability to entirely alter the application's contents to suit his purposes and make illegal use of the original data[34].

Attack by malicious code: is a piece of software code that is designed to harm the system and have unintended consequences[35]. Anti-virus software might not be able to stop or manage this kind of attacks. It may either start up automatically or function more like a program that demands user input.

3.2.2. Four Layer Architecture

The most basic architecture includes three layers. The three-layer architecture was unable to meet all IoT standards due to ongoing development in the field. As a result, researchers suggested a four-layer architecture[36]. Similar to the prior architecture, it contains three layers, plus a fourth layer known as the support layer. The functionality of the support layer in relation to security is as follows. Here, we will not discuss the functionality of the three-layer architecture. Since, we already covered section 2.2.1.

a) Support Layer

The security in the IoT architecture is the reason for adding a fourth layer. In a three-layer architecture, data is routed straight to the network layer. Sending data straight to the network layer increases the likelihood of encountering attacks. Information collected from a perception layer and is delivered to a support layer in a four-layer design. The support layer is in charge of two tasks. It verifies that information is sent by legitimate users is secure. The support layer's second duty is to communicate with the network layer. Both wireless and wired connections can be used as a transmission mechanism for data from the support layer to the network layer. This layer may be subject to a number of threats, including DoS attacks, malicious insider attacks, unauthorized access, etc.

3.2.3. Five Layer Architecture

The growth of the Internet of Things is greatly supported by the four-layer design. In the four-layer architecture, there were certain additional security and storage concerns. To make the IoT secure, researchers suggested a five-layer architecture[37], [38]. Similar to earlier architectures, it has three layers: the perception layer, the transport layer, and the application layer. The two

additional layers are: processing layer and business layer. The recently proposed architecture is capable of meeting IoT security requirements.

a) Processing Layer

A middleware layer is another name for the processing layer. It collects data that is transmitted from a transport layer. It processes the data that has been gathered. It is responsible for removing excess information that is meaningless and extracting the relevant data. It also solves the IoT's big data issue. Large amounts of information are received by the layer, which can affect IoT performance. Numerous threats have the potential to restrict IoT performance by affecting the processing layer. Malware attack and resource exhaustion are common attacks.

b) Business Layer

The business layer manages the entire system and refers to an application's intended behavior. Its duties include managing and controlling IoT applications, business models, and profits models. This layer is also in charge of managing user privacy. It can also decide how information can be created, stored, and changed. Vulnerability at this layer enables attackers to abuse an application. The majority of security issues are flaws in an application that arise from a broken or absent security control. Common issues regarding security of business layer are Zero-Day attack and business logic attack.

3.3. DoS/DDoS Attacks

DoS attacks have become one of the most dangerous cyber threat at the moment and pose a serious threat to modern computer networks [36]. DoS attacks aim to make a service, applications, or systems unavailable to legitimate users by overwhelming it with excessive request[38]. A denial of service attack occurs when a target server is drained by a single computer or system. The target server may withstand a denial-of-service attack because a huge server has many resources. To attack such systems, the intruders use a lot of computers to target a single server and exhaust it so that it can no longer serve its users[31]. A distributed denial of service attack is one that employs numerous computer systems to initiate such attacks. DDoS attacks are difficult to detect and prevent using traditional security mechanisms. Their detection and mitigation require intelligent and scalable defense systems that can distinguish between legitimate and malicious traffic.

The objective of a distributed denial-of-service (DDoS) attack is to prevent benign users from accessing a service, machine, or network by flooding the victim with attack traffic. DDoS attacks

uses infected IoT devices to generate attack traffic is known as IoT-enabled DDoS attacks. IoT-enabled DDoS attacks have increased over the past ten years on the Internet, and this trend is expected to continue. In IoT-enabled DDoS attacks, the attacker gains control of a large number of IoT devices, often forming a botnet. These devices are infected with malware and are remotely controlled by a botmaster. The attacker first deploys Command and Control (C&C) servers to initiate the infection and manage the botnet. The C&C servers scan the internet for vulnerable IoT devices and exploit weak passwords and default configurations to gain access.

For instance, the Mirai worm is able to identify potential victims by sending TCP SYN probes to pseudo-random IPv4 addresses on telnet TCP ports 23 and 2323[39]. The C&C server would start a brute-force login phase if an IoT device replied to a probe. It would attempt to establish a telnet connection using prepared username and password pairs from a list of common credentials or default credentials from IoT vendors. Consequently, because of huge number of Internet-connected devices with default settings, this attack becomes successful. The device will start acting like a bot after infection and wait for commands from the C&C server. The initially infected IoT devices will continue to spread the worm by quickly scanning the Internet and executing their own brute-force attacks, eventually establishing a huge botnet that may possibly cover the entire globe.

After the botnet is established, the botmaster can launch a DDoS attack by specifying target IP address. The C&C servers communicate this to the bots, which then flood the victim with traffic. Depending on the attack type, the botmaster may select only a subset of bots to initiate the attack. These bots can generate various types of DDoS traffic such as HTTP GET, POST, and HEAD attacks, as well as TCP SYN flooding, UDP flooding, TCP ACK flooding, and GRE-flooding. Open ports, security vulnerabilities in firmware images, a lack of reliable patching and update mechanisms, weak credentials, and a lack of strong authentication mechanisms are vulnerabilities that lead to IoT-enabled DDoS attacks. Figure 2 shows Distributed Denial of Service Attack.

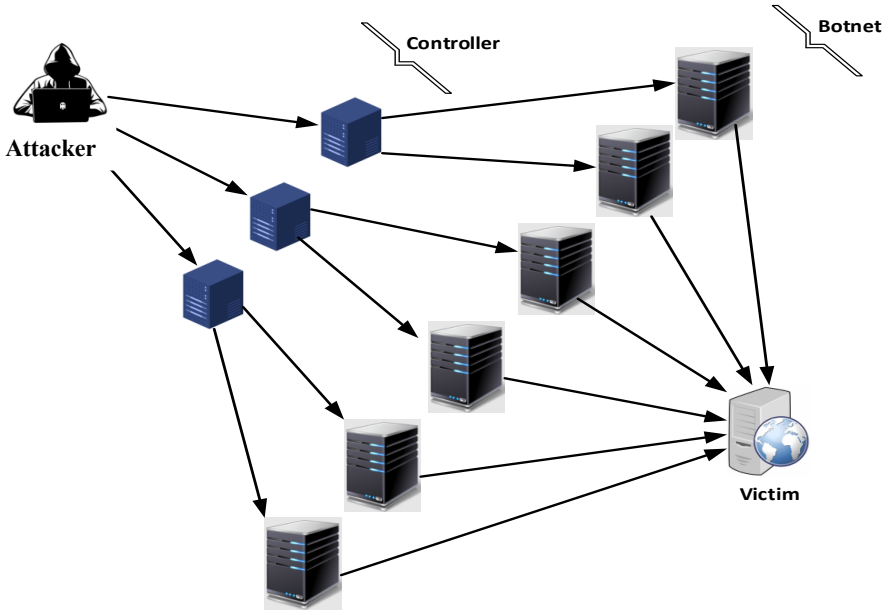

Figure 2: Distributed Deny of Service Attack

3.4. DDoS Detection Techniques

To detect DDoS attack traffic, different methods have been developed. The early methods focused on stochastic analysis to monitor the behavior of network traffic flows and take advantage of network traffic entropy to recognize normal behavior and identify anomalous intrusion events. Recently, utilizing machine learning techniques to identify and categorize malicious traffic has become popular techniques of detecting DDoS attack activity. This section reviews both traditional and advanced intelligent detection techniques.

3.4.1. Traditional Detection Techniques

Traditional detection techniques employ statistical and information theory-based analysis. Such techniques consider a given model as normal state and anything different as attack. The data of any monitored traffic periodically inferred and compared to the predetermined model. Any traffic that does not comply is classified as an attack. The authors of [38] proposed detection strategies that track the ratios of packets sent to and received from the protected network. They assert that for a given protocol, the monitored ratio for legitimate traffic should be lower than a particular threshold. Then, DDoS attack traffic identified suitably. The authors of [32], [40], [41]look at the strong association presented by either attack or legitimate traffic from several angles. They analyze network traffic metrics using correlation coefficients, compare the observed changes, and come to decisions.

Information-based metrics are also frequently employed in DDoS detection. Entropy can be calculated for a variety of parameters with a specified time window, including network flows, source/destination IP addresses, packet counts, source/destination IP ports, etc. To detect the presence of DDoS attack traffic, the changes in the estimated entropy values are evaluated using several information metrics. Difficulty in applying these techniques comes from the need to precisely define a normal profile and distinguish between normal and anomalous behavior. It is simple to imitate the statistical behavior of authorized users if there are enough active hacked machines [88]. Therefore, the basic premise of the offered works will be broken, and the defense strategies are misled. Furthermore, the deployment location of the detection system might have a significant impact on the performance of various detection techniques [43].

3.4.2. Intelligent-Based DDoS Detection

Recent research on utilizing machine-learning approaches for DDoS detection has produced promising outcomes. These methods overcome the limitations of traditional detection methods by being able to intelligently learn the underlying data properties without the need to explicitly identify legitimate and malicious activity. The topic of DDoS detection can conceptually be modeled as a binary classification problem, where monitored traffic is either classed as legitimate or attack traffic. Various classification techniques have been used and put to the test on well-known benchmarks. Researchers have also looked into possible feature sets to enhance detection accuracy while reducing false alarms[42].

Machine learning techniques, has effectively been used to address difficult issues in a variety of industries, including computer vision, social network filtering, video games, machine translation, healthcare, and bioinformatics. However, it has not been widely used in DDoS detection. Several studies use autoencoder, an unsupervised learning technique, to extract non-linear representations from the input data, and then they use a classification strategy to separate malicious traffic from legitimate traffic. Machine learning approaches are classified based on the technique in which the model is trained into four major categories: supervised, unsupervised, semi-supervised and reinforcement learning.

3.5. Generative Adversarial Network

A Generative Adversarial Network (GAN) is a deep learning architecture that competes two neural networks against one another in the context of a zero-sum game. GAN introduced initially

by Ian GoodFellow for estimating generative models[43]. GANs aim to produce new, synthetic data that resembles a well-known data distribution. Generative Adversarial Networks (GANs) are neural networks that are used for unsupervised learning. The Generator creates fake data samples in an effort to deceive the Discriminator. On the other hand, the Discriminator tries to distinguish between the real and fake samples. Both the Generator and the Discriminator are neural networks, and during the training phase, they compete with one another. The procedures are repeated several times making the Generator and Discriminator become better at what they are doing. GANs have diverse application such as producing synthetic data to increase the number of data samples and to resolve under-sampling issues[44]. GAN uses to generate well-designed adversarial attacks. GAN is also using for adversarial training defense by using competition game to regulate the feature selection during the training[45].

3.6. Datasets for DDoS Detection in IoT Network

Although several datasets available for network DDoS detection, may face significate limitation, such as the absence of reliable labels, a lack of attack diversity, and a lack of real-world realism. This section reviews commonly used datasets for training and evaluating machine learning-based DDoS detection models and highlights their constraints in representing actual IoT environments [43].

The University of New South Wales developed UNSW-NB15 in 2015[46], includes nine different types of attacks, such as DoS, Backdoors, and Worms. It consists of 2,218,761 recordings of normal traffic and 321,283 records of attacks. Although the dataset includes a variety of abnormal traffic types, it has fewer samples per attack class. Consequently, models trained on this dataset may overfit to the majority normal class. This reduce their effectiveness in detecting minority (attack) classes. The CIC-IDS2017 dataset, developed by the Canadian Institute for Cybersecurity in 2017[47]. The data was collected over a five-day period and includes both normal and attack traffic, such as SQL Injection, DoS using GoldenEye[48], and DDoS using the Low Orbit Ion Cannon (LOIC) tool[49] to send UDP, TCP, and HTTP requests to the victim server. The developers of the dataset contend based on a study published by McAfee in 2016. The dataset contains the most common and current attacks as of that time. However, IoT devices were not included in CIC-IDS2017 dataset's testbed, limiting its applicability for IoT-specific scenarios

To address this, the CSE-CIC-IDS2018 dataset was released in 2018[50] as an extension. The CSE-CIC-IDS2018 comprises well-known attacks including DDoS using LOIC and DoS using

GoldenEye. The testbed contains more windows machines than the original CIC-IDS2017. Hundreds of them arranged in a LAN on Amazon Web Services and divided into five subnets to represent different business divisions. The dataset includes various attacks, including DoS and DDoS. However, like its predecessor, it lacks realistic representation of IoT behaviour due to the absence of IoT devices in the experimental environment. The University of New South Wales released Bot-IoT in 2019[51], simulates a smart house with five simulated Internet of Things (IoT) devices such as a fridge, a garage door, a weather monitoring system, motion-activated lighting, and a thermostat. It includes attacks like DoS and DDoS that use GoldenEye to target HTTP traffic and Hping3 to target TCP and UDP traffic. Ostinato[52] was used to simulate the normal traffic. Although focused on IoT, the dataset lacks physical IoT hardware, which limits its realism and generalization to real-world deployments.

The Canadian Institute for Cybersecurity developed the CICIoT2023 dataset by 2023 [53]. Its objective was to provide a novel and extensive dataset for research and development in the IoT cybersecurity domain. To generate attack traffic, 33 attacks were executed on 105 IoT devices. Traffic includes both normal and attacked traffic. Attack traffic is classified into seven categories: DDoS, DoS, recon, web-based DDoS, brute-force, spoofing, and mirai. This dataset comprises both the common and current attacks.

The IoTID20 dataset has been utilized to develop DDoS attack-detection methods in several studies [54]. The authors of the IoTID20 dataset developed different binary and multiclass classifiers and evaluated the accuracy using the IoTID20 dataset [55]. The attack traffic for the Mirai botnet was generated separately using a laptop and was later changed to simulate its origin from IoT devices. DoS, scanning, and man-in-the-middle attacks were simulated using the NMap tools. DDoS attack types, such as mirai ACK flooding, mirai brute force, mirai HTTP Flooding, and mirai UDP Flooding, were included.

The University of New South Wales released TON IOT in 2020[56], which includes simulated and physical of Things (IoT) devices. The physical devices include two smartphones, a smart TV, and a smart fridge, while the simulated devices use Node-RED[57]. DDoS and ransomware are two types of attacks that have been conducted against IoT devices. As usual traffic produced by the publishing and subscribing techniques used by the mentioned devices to communicate with private and public MQTT gateways. Although this dataset contains a range of simulated and physical IoT devices as well as various attacks, the normal traffic is still a byproduct of the

activities conducted on the local testbed and is not available to actual users for using the services. The Canadian Institute for Cybersecurity published the CIC IoT dataset in 2022[58]. It analyzes how physical IoT equipment, such as cameras, a lamp, and a coffee maker, behave in various settings that simulate a smart home. The systems were being attacked utilizing DoS employing LOIC based on HTTP, UDP, and TCP protocols, as well as brute force attacks using the Real Time Streaming Protocol of the cameras (RTSP). Although actual IoT devices are included in this dataset, DDoS attacks are not included, and it primarily focused on behavioural study of various IoT devices.

The following comparative table exhibiting critical characteristics, drawbacks, and usage of commonly used datasets in IoT DDoS detection studies. This enhanced analysis justifies our dataset (IoTID20 and CICIoT2023) choice and decides the use and constraints of each dataset in realistic environments. We select IoTID20 and CICIoT2023 because of IoT generated data and concept drift support.

Table 1: Comparative table of commonly used datasets

Dataset	**DDoS Attack**	**IoT-Specific**	**Concept Drift Support**	**Class Imbalance**	**Realism / Network Type**	**Zero-Day Coverage**
IoTID20	Yes	Yes	Yes	High	Simulated lab IoT network	Partial
CICIoT2023	Yes	Yes	Yes	High	Emulated smart home network	No
BoT-IoT	Yes	Yes	No	Very High	Simulated using virtual IoT	No
TON_IoT	Yes	Yes	No	High	Real IoT telemetry/logs	No
UNSW-NB15	Yes	Partially	Yes	High	Emulated enterprise network	Limited

3.7. Challenge on the existing DDoS attack detection

This section presents the key challenges in deploying DDoS attack detection system in IoT system and discuss the limitations of the current DDoS attack detection approaches.

3.7.1. Centralized Learning and non-IID nature of IoT data

The majority of the existing DDoS detection models relay on centralized learning system, where data from distributed devices is sent to a central server for training. Such models may provide high accuracy, they are computationally expensive and present significant privacy concerns [14]. Additionally, there is a delay on classification of IoT data due to requests and responses to central cloud server. As a result, distributed or similar approaches to DDoS detection are becoming more popular. In the modern day IoT networks are rapidly becoming more scalable. As a result, it is challenging to offload large amounts of distributed IoT network traffic data to a remote central cloud server for data processing due to network limitations. The centralized machine learning method demand has a higher memory space for data storage, longer training period, and high communication overhead. Since, it involves sending network traffic features from all participating IoT devices to a central cloud server. The centralized method does not ensure the privacy and security of IoT devices without proper encryption. The network traffic features may contain sensitive information about the owners of the IoT devices. Using a third-party cloud server for machine learning will specifically increase the risk of privacy leakage in IoT systems[59], [60], [61].

Federated learning is a cutting-edge artificial intelligence strategy that aims to protect the privacy of participating nodes without significantly affecting the classification accuracy, perform generalizability of the machine learning models[62], [63]. In Federated Learning, several distributed edge nodes work together to construct a reliable machine learning model for detecting DDoS attacks in IoT critical infrastructure while maintaining data privacy.

Each of the edge nodes trains a local machine learning model using its own private network traffic and delivers model updates back to the central cloud server for model aggregation. The central cloud server performs model parameter aggregation and send back to each participating edge nodes. All of the participating nodes gain experience from each other without revealing any personal information. Hence, the federated learning approach has lower latency, power consumption, and memory requirements as network traffic data does not need to be transmitted to the central cloud server[64]. Figure 3 shows the Federated Learning approach.

The non-IID nature of the data across devices is the most significant challenge in the cooperative learning approach. Data in FL inherently exhibit variations in feature space, label distributions, and quantity, which are compounded by emerging divergences such as concept drift

and temporal shifts[16]. These challenges degrade the performance in terms of detection accuracy and slow convergence of FL algorithms such as FedAvg[17]. In addition, within the group of IoT devices, low-performing IoT devices negatively affect the global model, as FedAvg averages all local models equally[18]. If one local model within a specific cluster performs poorly, it might result in an underperforming global model for that specific group, because FedAvg averages all local models equally.

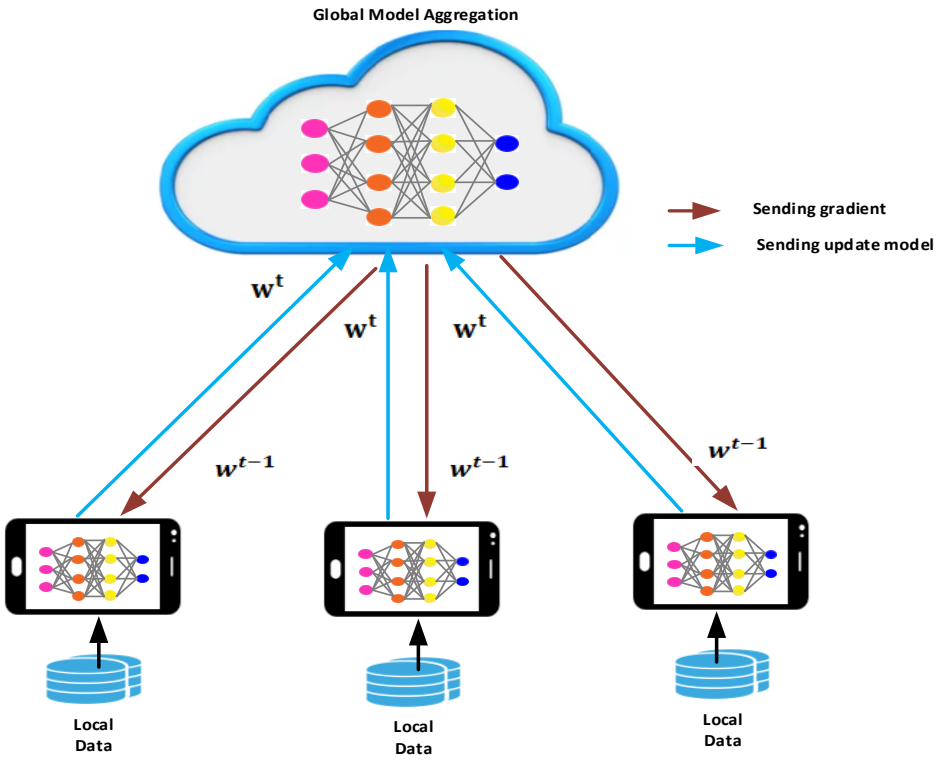

Figure 3: Federated learning

FL algorithms have been extended in various ways to mitigate the effect of non-IID data across IoT devices in large scale environment and to address convergence time limitation. FedProx[65], and IFCA[66] are enhancement over baseline FedAvg to address non-IID nature of IoT data and convergence time limitations. FedProx addresses non-IID nature by introducing a proximal term. Proximal term into the local optimization objective used to penalizing local updates that deviate from the global model. This improves moderate convergence speed in the presence of non-IID data. However, FedProx assume static client participation and uniform influence which leads to

low performance in dynamic IoT environments. IFCA attempts to address non-IID data issue by using clustering clients based on data distribution similarity. Each cluster performs training using a separate model. This approach improves local personalization and outperform FedAvg and FedProx algorithm in terms of accuracy. However, each cluster model does not adapt dynamically during training. This results in slow convergence, high computational and communication overhead when the number of IoT device and cluster increase. This challenge emphasises the need for adaptive and efficient solutions.

The key benefit of federated learning is parties will not exchange their data in order to train a certain model. However, recent research indicates that the need for privacy-preserving techniques to prevent attacks while exchanging parameters and weights during training [67]. A malicious aggregator could deceive the model being trained by changing the received parameters. Even a trustworthy but inquisitive aggregator may launch a reconstruction attack to derive training data from these parameters using a variety of methods.

a) Federated Learning with Non-IID Data

Addressing the challenges related to non-IID data in a federated learning (FL) environment is essential, because it affects the performance of the machine learning models across a network. Zhao et al. [68] investigated the impact of non-IID data distributions on the federated learning performance. The author proposed a data-sharing strategy that utilizes a small subset of globally shared data to mitigate this impact. Wang et al. [69] developed an advanced federated learning framework called FAVOR. It utilizes deep reinforcement learning approaches and convergence on non-IID data. Sattler et al. [70] proposed a Sparse Ternary Compression (STC) method to enhance federated learning in non-IID data. STC reduces communication costs by using eternalization and optimal Golomb encoding for gradient updates and maintaining learning performance efficiently. Duan et al. [71] developed an Astraea framework that utilizes Z-score-based data augmentation and mediator-based multiclient rescheduling to manage data imbalances. The authors utilized a methodology for dynamically balancing the data distribution across clients to improve the performance of the FL model. Wang et al. [72]developed an approach to address the class imbalance challenge in federated learning environments. The authors used a monitoring scheme to mitigate the impact of data imbalance. The authors demonstrated the performance of the proposed approach over the traditional approaches.

b) Clustered Federated Learning Approaches

The Clustered Federated Learning approach is used to address the impact of non-IID data on the performance of federated learning by grouping IoT devices with similar data distribution in the same group. Jeong et al. [73] developed a Cluster-Driven Adaptive Training Approach for Federated Learning (CATA-Fed) to address non-IID data challenges and straggler effects in IoT edge-computing environments. The authors utilized clustering to address non-IID challenges and proportional fair scheduling to optimize resource allocation among clients. Dennis et al. [74] developed the k-FED method for clustered federated learning. An efficient one-shot method was used to facilitate clustering and reduce the communication round requirements in federated learning. The efficacy of k-FED compared with traditional centralized methods and the experimental results showed the effectiveness of the proposed approach in terms of scalability and robustness to node failures. Brigg et al. [75] presented a hierarchical clustered federated learning (FL) approach to address the challenges of non-IID data across distributed networks. The methodology was validated through an empirical analysis for real-world applications. Ghosh et al. [76] proposed an Iterative Federated Clustering Algorithm (IFCA) framework. The framework utilizes a client-clustering approach to improve the overall performance by iteratively estimating cluster memberships and optimizing model parameters. The framework demonstrated significant improvements in handling non-IID data compared with other approaches. Long et al. [77] proposed a federated learning framework that enhances personalization using a multicenter aggregation approach. The framework clusters clients based on data distribution similarity. This helped the proposed framework handle the non-IID data effectively. The framework was evaluated on benchmark datasets and demonstrated better performance against baselines.

c) Convergence Analysis of Federated Learning

A convergence analysis was used to ensure the reliability of the federated learning algorithms. It is helpful to study the convergence of federated learning in non-IID data-distribution environments. This helps to understand the federated learning solution convergence, the rate at which it converges, and the factors that influence convergence. Several FL-based problems have been solved using stochastic gradient descent (SGD)[77]. Hence, convergence analysis is usually based on SGD convergence. Stich et al. [78] and Kahaled et al. [79] performed a convergence analysis using local Stochastic Gradient Descent (SGD) to mitigate the challenges of non-IID and straggler. Both provide theoretical convergence analyses for both identical and heterogeneous data. Wang et al. [80] provided federated optimization algorithms that consider communication

efficiency, data heterogeneity, and privacy requirements. The author provides recommendations on how to formulate and evaluate the convergence and efficiency of federated learning optimization algorithms. They provided a theoretical analysis and practical implementation of the proposed approach. Xing et al. [81] proposed different optimization methods, such as the momentum of federated averaging (FedAvg) and adaptive methods to improve convergence rates and resource utilization. This study contributes to the practical application and theoretical analysis of federated learning optimization. Li et al. [17] analyzed the convergence challenge of FedAvg on non-IID data. The author provides a theoretical and empirical analysis using synthetic data to demonstrate how non-IID affects the performance of FedAvg and hyperparameters, such as local epochs. A theoretical convergence analysis of FedAvg was performed, and the convergence rate of FedAvg $O(\frac{1}{T})$ was derived. In addition, the authors have also analyzed the impact of convergence on the participation of clients and non-IID data.

3.7.2. Dynamic nature of IoT data

The dynamic nature of the incoming traffic makes it challenging to classify real-time data. This is a major issue for machine learning algorithms. The learning algorithms are first trained, but because the dynamic nature of streaming data is always changing, they must be retrained. Learning massive IoT datasets demands retraining a significant amount of time, computational resources, and memory due to the real-time nature of the environment. Retraining the system from scratch is a possibility but a time-consuming operation given the current training set and newly acquired training data. The process of continuously improving an old model with new data known as online learning. Online learning techniques are used to train models using continuously incoming IoT streams in dynamic IoT environments. IoT devices possess limited computing power and storage capacity [82].

This hinders their ability to handle and retain massive amounts of data and complex DDoS attack-detection models. Therefore, it is critical to develop detection models with low spatial and temporal complexities. Online learning models can reduce memory requirements for data storage and adapt new data patterns. Online learning models can often achieve real-time processing and address concept drift issues. Therefore, in dynamic environments, online learning is often more effective than batch learning. The batch learning method trains a learning model on the entire training dataset, in contrast to online learning methods. Online learning continuously updates the learning model with new data samples when it arrives within a short execution time. The online

DDoS attack detection model often encounters concept drift issues when data distribution shift occurs. There are three major types of data distributions over time that cause concept drift: sudden, gradual, and recurring. Sudden drift occurs when rapid, irreversible changes occur over a short period. Gradual drift occurs when the data distribution gradually replaces old data over time. Recurring drift occurs when the previous data distribution occurs again over time.

The majority of IoT devices used in IoT systems have limited computing power and storage [4]. This limited memory capacity hinders their ability to handle and retain massive amounts of data and complex learning models. Therefore, it is crucial to develop analytics models with low computing complexity. Online learning methods that enable real-time analytics are able to fulfill IoT system time and memory requirements. Online learning methods can continuously update the learning model with new data samples when they arrive within a short execution time in contrast to that required under the batch learning method, which frequently trains a learning model on the entire training dataset. IoT data samples are usually produced dynamically in constantly changing IoT environments. The dynamic nature of streaming traffic also makes already trained models less useful in recognizing zero-day attack.

Data analytics frequently experiences concept drift issues in real-world applications due to the change in IoT data distribution over time. Concept drift is caused by three main data distribution changes: recurring, gradual, and sudden [17]. Sudden drift happens when rapid irreversible changes occur in a short period of time. Gradual drift happens when the data distribution gradually replaces the old one over time. Recurring drift happens when the previous data distribution happens again over time.

Concept drift is formally defined as a set of samples, indicated as $S_{0,t} = \{d_0, \dots, d_t\}$ for a time interval of [0, t], where $d_i = (x_i, y_i)$ represents a single observation, x_i is the feature vector, y_i is the label, and $S_{o,t}$ adheres to a specific distribution $f_{o,t}(x, y)$. If $f_{o,t}(x, y) \neq f_{t+1,\infty}(x, y)$, denoted as $\exists_t : p_t(x, y) \neq p_{t+1}(x, y)$, then concept drift happens at timestamp t_{t+1}. Concept drift at time t can also be defined as the change in the joint probability of x and y at time t, expressed as $p_t(x, y) = p_t(x) * p_t(y/x)$. Concept drift will happen under the following three conditions.

- $p_t(x) = p_{t+1}(x)$ while $p_t(y/x) = p_{t+1}(y/x)\ p_t(x)$. This type of drift is known as virtual drift because $p_t(x)$ does not affect the decision boundary.

- $p_t(y/x) \neq p_{t+1}(y/x)$ while $p_t(x) = p_{t+1}(x)$ while $p_t(x)$ remains unchanged. This is considered actual drift because it affects the decision boundary and also leads to a decline in learning accuracy.
- The combination of the first two, $p_t(x) \neq p_{t+1}(x)$ and $p_t(y/x) \neq p_{t+1}(y/x)$.

Concept drift poses significant issues when building machine learning models because it can cause changes in data distribution that result in machine learning model performance gradually declining over time [18]. Hence, advanced online machine learning models need to be developed in order to detect and adapt the concept drift that occurs in IoT data streams.

d) Drift Detection

Drift detection is a crucial element for adaptive machine learning models that can solve concept drift issues. The two primary categories of drift detection methods are performance-based and distribution-based [19].

- ***Performance-Based Methods***

  Performance-based concept drift detection methods are based on changes in the metrics used to evaluate model performance. Common examples of indicators of concept drift are accuracy decline and increase in error rate. If the error rate of a learner gradually decreases or remains constant as more samples are learned, it often indicates constant data distribution without drift. On the other hand, if a learner's error rate drastically increases as more data is processed, it often indicates the presence of concept drift. The two well-known performance-based drift detection methods are the drift detection method (DDM) and early drift detection method (EDDM), which are able to detect concept drift by keeping track of degradations in performance. The DDM is a popular performance-based drift detection method that measures model error rate and standard deviation changes using two predefined thresholds: the warning threshold and the drift threshold [20]. DDM often performs well on data streams with sudden drift, but its reaction time is often too slow for detecting gradual drift. The early drift detection method (EDDM) is an enhanced version of the DDM that detects concept drift using the same concept drift detection mechanism [10]. The EDDM often performs well on data streams with gradual drift. Even though the EDDM frequently performs better than does the DDM, it still falls short for sudden drift. Furthermore, due to its sensitivity to noise, it may mistake noise for drift, resulting in false alarms.

- ***Distribution-Based Methods***

  Distribution-based concept drift detection is based on changes in data distributions. Data distribution changes can be measured using statistical variables such as mean, variance, and information entropy. Adaptive windowing (ADWIN) is one such widely used method that uses adaptive sliding windows to detect concept drift based on the statistical difference between two adjacent sub-windows. The adaptive windowing method uses characteristic values such as the mean and variance, as well as variable-size sliding windows to detect concept drift. The window size is dynamically increased if there is no concept drift and reduced when concept drift is detected [9]. IoT systems with less memory often adopt distribution-based methods since they only need to retain the most recent samples. Windowing methods are often quick and simple to use. However, they could miss certain important historical data.

e) Drift Adaptation

Concept drift must be handled after it is detected by updating the current models using the appropriate drift adaptation methods. Drift adaptation is a procedure used to update a model automatically when a concept drift occurs to enhance performance of model and detect zero-day attacks. The procedures usually fully retrain or alter the learning model using new dataset. The three categories of drift adaptation methods are model retraining, incremental learning and ensemble learning.

- ***Model Retraining***

  One of the more simple and straightforward methods to handle concept drift is model retraining. Offline models are often unable to accurately predict unseen incoming streaming data. This problem can be addressed by retraining the model using the most recent data streams. However, employing this technique can cause unnecessary model retraining or drift adaptation delays. Therefore, it is crucial to use learning models together with an appropriate drift detection method to determine when to retrain the learning model for timely and necessary updates. There are two types of model retraining methods: full retraining and partial retraining. Retraining the learning model using the entire dataset and all available samples is known as full retraining, while partial retraining trains the model on selected parts of the dataset. The window-based method is used to partially retrain a model using the most recent data. This reduces training times but may result in the loss of historical patterns. Hence, selecting the right

window size is crucial. ADWIN is a drift detection method that uses model retraining. ADWIN performs better as it uses a dynamic window to fit new data [9]. The learning model is partially retrained on only the new concept samples in order to save training time.

- ***Incremental Learning Methods***

  Incremental learning has become commonly used in data stream analytics research. Incremental learning involves updating the learning model when each instance is processed. Incremental learning methods often partially update the learning model to fit a new data sample [21]. They do not require a sufficient amount of data prior to the training process due to the incremental learning approaches' capacity to support progressive learning. However, only a few machine learning algorithms such as MLP and multinomial NB support partial updates.

- ***Ensemble Learning Methods***

  Ensemble learning approaches have been developed to provide powerful learners for data stream analytics to enable stronger concept drift adaptation. Ensemble learning combines multiple base learners to tackle the same problem [22]. Ensemble learning base learners can be constructed using different algorithms, and different configurations of hyperparameters configuration. Ensemble learning models are often more generalizable than single models because they combine the outputs of multiple base learners. Reusing existing models in an ensemble is significantly more effective for concept drift adaptation than training new models on data streams with recurrent concept drift [23].

  Block-based ensembles and online ensembles are the two main categories of ensemble techniques used in data stream analytics [24]. Data streams are divided into fixed-size blocks by block-based ensembles, which then train a base learner on each block. The base learners will be evaluated and updated each time a new block arrives. Block-based ensembles react to gradual drifts accurately, but they frequently take longer to respond to sudden drifts. Three common block-based ensembles are accuracy-weighted ensemble (AWE) [25], accuracy-updated ensemble (AUE) [26], and the streaming ensemble algorithm (SEA) [27]. The AUE often performs better among the block-based ensembles [28]. AUE uses non-linear error functions to apply weights to base learners in order to improve performance.

To enhance the learning performance of online ensembles, different incremental learning models such as Hoeffding trees (HTs) are included. Gomes et al. [11] proposed the adaptive random forest (ARF) approach, which makes use of HTs as base learners and ADWIN as a drift detector. The drift detection mechanism replaces underperforming base trees with new trees that better fit. Since random forest is a well-known, effective machine learning algorithm, ARF frequently outperforms many other methods. ARF also includes a powerful resampling method and the flexibility to accommodate various drifts types.

Gomes et al. [12] also proposed streaming random patches (SRPs) for the adaptive ensemble approach. Although its execution time is usually longer, SRPs often exhibit slightly better prediction accuracy than ARF. Leverage bagging (LB) [29] is another online ensemble that constructs base learners using bootstrap samples. Although LB is simple to construct, it usually performs worse than SRPs and ARF. Despite the fact that there are several concept drift adaptation strategies in use today, their efficacy is constrained by slow drift response and poor prediction accuracy.

Incremental learning methods often perform poorly due to their low model complexity and limited drift adaptability [30], whereas block-based ensembles face significant difficulties in determining block size and responding quickly to drift. Online ensembles, like ARF and SRPs, often outperform incremental learning and block-based ensemble methods. However, because of their randomization strategies, they give unstable learning models, adding more unpredictability to DDoS attack detection.

This dissertation aims to detect DDoS attacks in IoT systems while addressing the concept drift issue. In particular, this dissertation proposes adaptive online DDoS attack detection using the ensemble learning method to address the issue of concept drift in IoT system DDoS attack detection.

3.7.3. Adversarial attack

In recent years, researchers have found severe security threat concerns to the existing machine learning. The first noticeable adversary attack for computer vision on machine learning is discovered by Polikar[72]. Due to the great success of machine learning, its vulnerability has attracted a great deal of attention, especially for security-critical applications and autonomous driving. For instance, the backdoor attack manipulates the behavior of the machine learning model

primarily poisoning certain training data at the training stage. This adversarial attack aims to deceive the model by adding malicious perturbations onto the input in the inference stage. They insert small perturbations in the form of carefully crafted input to confuse the model in order to misclassify the object.

The methods used to generate adversarial attacks vary based on the nature of the data and the sector. Below, we describe five adversarial attack generation methods.

a) Fast Gradient Sign Method (FGSM)

The Fast Gradient Sign Method (FGSM) is an easy and efficient method for generating adversarial perturbation [83]. Adversarial perturbations are inputs that have been intentionally crafted to maximize the amount of perturbation to the input data to misclassify machine learning models. After determining the gradient of the loss function with respect to the input data, the FGSM crafts a perturbation in the direction of the sign of the gradient. The amount of modification of the input data is controlled by a hyperparameter called epsilon (ε), which determines the magnitude of perturbation. The FGSM approach must be adapted to operate using online learning algorithms. In online learning, models are updated incrementally as new data points arrive. Therefore, an adversarial attack should be able to perturb data points as they arrive. Given an online learning model f(θ_t, x_t) with parameters θ_t and data point x_t at time t and a loss function $l(\theta_t, x_t, y_t)$ where y_t is the true label at time t, the objective is to create an adversarial example x_t' that maximizes the loss. For each incoming data point x_t, generate an adversarial example x_t' by perturbing x_t in the direction of the gradient of the loss function with respect to x_t:.

$$x_t' = x_t + \mathcal{E} \, . sign(\nabla_{x_t} l(\theta_t, x_t, y_t)) \quad (1)$$

Perturbation $\mathcal{E}$ controls the magnitude of the change in the input, ensuring that the adversarial example remains close to the original input. The constraint is typically applied to the l_∞ norm:

$$||x_t' - x_t||_\infty \leq \mathcal{E}. \quad (2)$$

Compute the gradient of the loss function with respect to the current input x_t:

$$\nabla_{x_t} l(\theta_t, x_t, y_t) \quad (3)$$

b) *Basic Iteration Method (BIM)*

The Basic Iteration Method (BIM) first proposed by Kurakin et al. [84] in 2017. BIM is a basic extension of the FGSM. BIM adopts FGSM in an iterative approach, rather than applying multiple inputs with small step-size perturbations in the direction that maximizes the loss of the model. The intention is to provide an adversarial example that can deceive the model's predictions while still seemingly comparable to the original input. BIM generates adversarial examples by iteratively applying FGSM with a smaller step size. The details of the mathematical formulation of BIM, particularly in the context of online learning, are formulated as follows:

For each iteration i, initialize:

$$x_{adv}^{(0)} = x_t \tag{4}$$

Iterative update

$$\mathrm{x}_{\mathrm{adv,t}}^{(\mathrm{t+1})} = \mathrm{x}_{\mathrm{adv,t}}^{(\mathrm{i})} + \alpha \,.\, \mathrm{sign}(\nabla_{\mathrm{x}_{\mathrm{adv,t}}^{(\mathrm{i})}} \mathcal{L}(\mathrm{x}_{\mathrm{adv,t}}^{(\mathrm{i})}, \mathrm{y_t}; \theta_\mathrm{t})) \tag{5}$$

Where, α is the step size and $(\nabla_{x_{adv,t}^{(i)}} \mathcal{L}(x_{adv,t}^{(i)}, y_t; \theta_t)$ is the gradient of the loss with respect to the adversarial example at iteration i.

Clip perturbation:

$$\mathrm{x}_{\mathrm{adv,t}}^{(\mathrm{i+1})} = \mathrm{clip}(\mathrm{x}_{\mathrm{adv,t}}^{(\mathrm{i+1})}, \mathrm{x_t} - \epsilon, \mathrm{x_t} + \epsilon) \tag{6}$$

where ϵ is the maximum perturbation allowed and $clip(\cdot)$ ensures that the perturbation remains within the range $[-\epsilon,\ +\epsilon]$.

After N iterations, the final adversarial example is:

$$\mathrm{x}_{\mathrm{adv,t}} = \mathrm{x}_{\mathrm{adv,t}}^{(\mathrm{N})} \tag{7}$$

c) *Projected Gradient Descent (PGD)*

Projected Gradient Descent (PGD) is an iterative optimization algorithm[85], [86]. This is designed to minimize the loss function. PGD builds upon the principles of FGSM but introduces an iterative refinement process to overcome some of FGSM's limitations. The PGD operates by iteratively applying small perturbations to the input in a manner that maximizes the loss function

of the model while remaining within a specified perturbation budget. This iterative nature allows the PGD to explore a wider range of perturbations. Mathematically, is represented as

$$\theta_{t+1} = P_c(\theta_t - \alpha_t \nabla f(\theta_t, x_t, y_t)) \quad (8)$$

Where, x_t is the input at iteration t, α_t is the step size, $\nabla f(\theta_t, x_t, y_t))$ is the gradient of the loss function with respect to the input and $P_c(\cdot)$ is projection operator ensuring perturbed input stays within predefined bounds.

d) Carlini and Wagner (C&W)

A C&W attack is a state-of-the-art optimization-based attack[86]. CW is a highly effective adversarial attack method that can be adapted for online learning scenarios to generate minimal perturbations that mislead machine learning models. There are two variants of CW attack: L2 and L∞. The CW objective is to minimize the following:

$$min_{\delta}||\delta||_2 + c.f(x_t + \delta) \quad (9)$$

where $||\delta||_2$ is the L_2norm of the perturbation, and c is a constant balancing the trade-off. The function $f(x_t + \delta)$ defined as:

$$f(x_t + \delta) = max(0, Z(x_t + \delta)_y - \max_{i \neq y} Z(x_t + \delta)_i + \kappa) \quad (10)$$

where $Z(x_t + \delta)$ denotes the logits of the model, $Z(x_t + \delta)_y$ is the logit for the true class, $Z(x_t + \delta)_i$ is the logit for the target class, and κ is a confidence parameter.

e) Generative Adversarial NetworkS (GANs)

A Generative Adversarial Network (GAN) is a deep learning architecture that competes with two neural networks against one another in the context of a zero-sum game. The GAN was initially introduced by Goodfellow to estimate the generative models [43]. GANs aim to produce new synthetic data that resemble well-known data distributions. Generative Adversarial Networks (GANs) are neural networks used for unsupervised learning. The generator created fake data samples to deceive the discriminator. In contrast, the discriminator attempts to distinguish between real and fake samples. Both the generator and discriminator are neural networks, and they compete with each other during the training phase. The procedures are repeated several times, and the generator and discriminator become better at what they are doing. GANs have diverse applications, such as producing synthetic data to increase the number of data samples and resolving under-

sampling issues [44]. A newly emerging area of research focuses on integrating machine-learning technology with privacy-protection technology.

The training process in online learning GANs is formulated as a minimax game, with the objective function $V(G, D)$ expressed as:

$$min_G max_G\, V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\, log\, D(x)] + \mathbb{E}_{z \sim p_z(z)}[\, log(1 - D(G(z)))] \quad (11)$$

In each iteration, the discriminator D aims to maximize its ability to differentiate between real samples $\boldsymbol{x}$ from the current data batch and fake samples G(z) , where $\boldsymbol{z}$ is a noise vector.

$$max_G\, \mathbb{E}_{x \sim p_{data}(x)}[\, log\, D(x)] + \mathbb{E}_{z \sim p_z(z)}[\, log(1 - D(G(z)))] \quad (12)$$

Simultaneously, the generator G attempts to minimize the discriminator's ability to correctly identify its synthetic samples as fake.

$$min_G\, \mathbb{E}_{z \sim p_z(z)}[\, log(1 - D(G(z)))] \quad (13)$$

This iterative adversarial process continues as new data arrives and ensures that both networks improve over time.

3.7.4. Feature Dimensionality Reduction

Machine learning is an effective method for detecting DDoS attacks. Enough amount of network traffic data is required to provide effective classification performance in order to design an effective machine learning method for DDoS attack detection in IoT networks[87]. However, the amount of memory space and network traffic that is needed is usually large. As a result, it is extremely difficult to implement the machine learning method in IoT devices with limited memory. The "curse of dimensionality" might result from processing and analyzing high-dimensional network traffic data. Processing high-dimensional data is difficult and necessitates a lot of computing power and storage[88], [89]. IoT devices do not have enough capacity to store the large amounts of network traffic data required for machine learning. Therefore, end-to-end machine learning-based DDoS attack detection approach is required that will minimize the high dimensionality of big network traffic features and accurately detect complex and recent DDoS attack.

One method for reducing input features is feature selection. In order to create meaningful features or to extract high-level information from the raw features, different feature extraction methods are proposed. Data compression is one method for doing this, which involves reducing

the number of bits used to represent the data. Data compression is carried out via AutoEncoders, Principal Component Analysis and Linear Discriminant Analysis.

3.7.5. Data Imbalance Issues

In the field of machine learning, classification problems are common. We aim to predict the class label by analyzing the input data or predictor, knowing that the target or output variable in a classification problem is a categorical variable by nature. If you have dealt with classification issues before, you must have seen situations where one of the target class labels' numbers of observation is noticeably smaller than those of other class labels. A typical sort of dataset used in real-world classification situations is one with an imbalanced class composition. This type of machine learning problem is notoriously difficult to solve using conventional methods.

Network traffic data exhibits a large tail distribution [49], similar to the majority of real-life classification tasks[90]. For instance, in the case of botnets, samples of network traffic produced will be much higher than those produced by DDoS on actual IoT devices. The number of samples in a class divided by the total number of samples in the remaining classes is the class imbalance ratio for that given class. If the class imbalance ratio is more than 10:1[91], training data is said to be severely unbalanced in deep learning approaches. Modern machine learning approaches may not perform as well as they should because of the class imbalance problem, which causes the models to be biased in favor of the classes with the majority of samples[92].

3.7.6. Model Hyperparameter Selection

The classification performance of the local models at the nodes depends on the choice of the right set of models hyperparameters, which vary for different application contexts. These hyperparameters include the number of hidden layers and their respective hidden units, the learning rate, the hidden layers' activation functions, the batch size, the optimizer, and the number of epochs. However, most the hyperparameters of the models in previous works randomly selected with little or no justification.

3.8. Related work

3.8.1. DDoS attack detection in IoT

The development of efficient and effective DDoS attack detection techniques in IoT systems has received research attention in the past decade [8]. Special focus has been given to

implementing DDoS detection based on network traffic analysis. Machine learning-based DDoS detection techniques, in particular, have been hailed as promising for making inferences about DDoS attacks. Chen et al. [7] proposed a machine learning-based multi-layer IoT DDoS attack detection framework that includes IoT devices, IoT gateways, software-defined network (SDN) switches, and cloud servers. The author constructed eight smart poles equipped with different sensors on campus networks and collected sensor data as datasets over wired and wireless networks. The experimental findings demonstrated that the multi-layer DDoS detection systems have an accuracy above 97% for different datasets. Additionally, the SDN controller can efficiently block malicious devices.

Attota et al. [14] proposed an ensemble multi-view federated method for IoT intrusion detection. The authors addressed the limitations of centralized deployment by using edge computing paradigms for maintaining data privacy. Three artificial neural network (ANN) models were trained using the bidirectional traffic flow, unidirectional traffic flow, and packet of network traffic features. To select the optimum set of network traffic features for ANN model training, the grey wolf optimization (GWO) technique was used. This reduced the amount of memory space needed to store the training data by reducing the dimensionality of the network traffic features. ANN models' outputs are fed into a random forest (RF) model to predict attacks. The hyperparameters of the models, however, are not disclosed. An evaluation of the proposed approach was conducted using the MQTT dataset, which lacked samples of IoT DDoS attacks.

Nguyen et al. [15] demonstrated the vulnerability of federated learning-based intrusion detection systems to backdoor attack. Backdoor attack is a type of poisoning attack in which the attacker corrupts specific input to a model in order to make incorrect predictions. The anomaly detection system used the gated recurrent unit (GRU), which is a type of recurrent neural network (RNN) for detecting the anomaly behavior of IoT devices. The experiment results showed the effectiveness of backdoor attacks in circumventing state-of-the-art defenses against federated learning poisoning. The attacker performed white box poisoning on the training data by stealthily injecting malicious traffic into the benign training dataset. As a result, the model incorrectly classified malicious traffic as benign.

Ullah et al. [6] developed anomaly-based intrusion detection models for IoT networks using convolutional neural network models. The proposed convolution neural network (CNN) model was implemented using 1D, 2D, and 3D network architectures. CNN and transfer learning were

employed as deep learning models for multi-class and binary classification on BoT-IoT, IoT network intrusion, MQTT-IoT-IDS2020, and IoT-23 intrusion detection datasets.

Cheng et al. [16] proposed the federated transfer learning approach for intrusion detection in mobile edge computing. The federated learning approach uses transfer learning to speed up model training, lower computational costs, boost communication effectiveness, and enhance classification performance. A CNN model architecture was utilized for binary classification that included three convolutional layers, two max-pooling layers, a batch normalization layer, a dropout layer, and two dense layers. The NSL-KDD dataset was used to train the model in the source domain, and the UNSW-NB15 dataset was utilized to finish the training in the target domain in order to evaluate the performance of the method.

Boonchai et al. [1] facilitate multiclass classification capability for DDoS attack detection using DNN models. In this research, two models are implemented, namely the CNN-AE architecture and simple DNN. The DNN architecture consists of six dense layers. The CNN-AE architecture consists of convolutional, max pooling, and up-sampling layers constructed via an AE technique. The CIC-DDoS2019 dataset was used to evaluate the proposed models. These models show good accuracy, with rates of up to 87% and 91.9% for the respective models.

Zainudin et al. [2] provide the CNN and LSTM hybrid mode for DDoS attack classification. The author utilized the extreme gradient boosting (XGBoost) feature selection technique in order to determine the top 10 relevant features. The proposed model was evaluated using CIC-DDoS2019 dataset, which includes DDoS attack and benign data. The experiment results show an accuracy of 99.5%.

Kumar et al. [3] developed an LSTM-based DDoS attack detection model using the CICDDoS2019 dataset. The proposed model was performed using binary classification to distinguish between DDoS attacks and benign traffic. The experimental results show that the proposed model achieved an accuracy of 98.6%.

Table 2: Related work summery for DDoS attack detection

Reference	Year	Model	Drift Detection	Learning Method	Dataset
Zainudin et al. [7]	2020	FL	✗	Batch learning	ToN_IoT
Attota et al. [14]	2021	ANN and RF	✗	Batch learning	MQTT dataset
Nguyen et al. [15]	2020	GRU	✗	Batch learning	IoTDs

Ullah et al. [6]	2021	CNN	✗	Batch learning	BoT-IoT, IoT MQTT-IoT-IDS2020, and IoT-23
Cheng et al. [16]	2022	CNN	✗	Batch learning	NSL-KDD and UNSW-NB15
Boonchai et al. [17]	2023	CNN-AE	✗	Batch learning	CIC-DDoS2019
Zainudin et al. [18]	2022	CNN and LSTM	✗	Batch learning	CIC-DDoS2019
Kumar et al. [19]	2023	LSTM	✗	Batch learning	CIC-DDoS2019
Proposed Solution		**Ensemble**	✓	**Online learning**	**IOTID20 and CICIoT2023**

The recent research papers on machine learning-based DDoS attack detection for IoT devices are summarized in Table 1. From the Table, we can see that none of the above works considered the dynamicity of the IoT environment, which could result in concept drift. The learning methods proposed in the above works are batch learning detection methods.

3.8.2. Adversarial Attack and Defense in IoT system

In this section, we explore the most recent literature on adversarial attacks and defense in the IoT network context. Specifically, we focus our review on recent research concerning adversarial attacks and defense against DDoS attacks. This review provides an overview of the models, techniques, and tools utilized for adversarial attacks and defense, as well as the datasets used to evaluate these systems.

Zhou et al. [93] proposed a state-of-the-art method for generating adversarial attacks, called hierarchical adversarial attacks (HAA). The objective of this method is to operate within a specified budget limit and construct a sophisticated level-aware black-box attack against a GNN-based IDS in IoT networks. The authors evaluated their HAA method using the UNSW-SOSR2019 dataset. The experimental results show that adversarial attacks based on the HAA method reduce the classification precision of both GNN models by more than 30%. Kotak et al. [94] proposed a new method to generate a real-time adversarial example using heatmaps from Grad-CAM++ and Class Activation Mapping (CAM). The authors explored the vulnerability of fully connected neural networks (FCN), convolutional neural networks (CNN), and Global Average Pooling (GAP) using the IoT Trace dataset. Surprisingly, adversarial examples can be transferred to different model architectures. The GAP model behaves differently, suggesting that it may be a robust model for adversarial examples.

Yumlembam et al. [82] proposed a new approach called VGAE-MalGAN, which can effectively add nodes and edges to an existing API graph and dynamically generate an adversarial network against GNN-based malware detection. The authors developed robust Android malware detection using VGAE-MalGAN adversarial training and achieved a high malware detection accuracy. Ibitoye et al. [95] demonstrated the effects of adversarial samples on a deep learning-based Intrusion Detection System (IDS) using the BoT-IoT dataset. They conducted a performance comparison of two neural networks: a feedforward neural network (FNN) and a Self-normalizing Neural Network (SNN), employing metrics such as accuracy, precision, and recall. The authors used the Fast Gradient Sign Method (FGSM), the Basic Iteration Method (BIM), and Projected Gradient Descent to craft adversarial samples. The experimental results show that the self-normalizing feature enhance the SNN more resilient to gradient-based adversarial samples, while the FNN outperforms the SNN in intrusion detection within IoT networks.

Papadopoulos et al. [96] evaluated traditional machine learning and deep learning models using Bot-IoT datasets. The attacker generates adversarial examples employing the Fast Gradient Sign Method (FGSM) to evade detection measures against binary and multiclass Artificial Neural Networks (ANNs). Additionally, the author analyzed and conducted various experiments to demonstrate how the attacker could circumvent the detection model with significant probability, highlighting the robustness of the detection model against adversarial examples. Since the attacker possesses knowledge of datasets and models, this scenario is classified as white-box attack. Experimental results indicate that an attacker can manipulate or circumvent detection with a significant probability. Benaddi et al. [97] proposed an adversarial training method to improve the effectiveness of a hybrid CNNLSTM-based Intrusion Detection System (IDS) using Conditional Generative Adversarial Networks(C-GAN). The authors integrated C-GAN into the training pipeline to address classes with small sample sizes and to mitigate data imbalance in the BoT-IoT dataset. Initially, the BoT-IoT dataset was utilized to train the IDS model, and certain low-performing classes were identified. These classes were then used to train the C-GAN, and the C-GAN generator was used to retrain the IDS model, which enhanced the performance of the identified classes.

Benaddi et al. [98] employed a similar approach to improve the robustness and efficacy of an IDS in the IIoT. The study suggests employing Deep Reinforcement Learning (DRL) in conjunction with GAN to improve the performance of an IDS. Experiments were conducted on the Distributed Smart Space Orchestration System (DS2OS) dataset, and it was found that the

proposed DRL-GAN model performs better than the standard DRL in identifying anomalies in imbalanced datasets related to the IIoT. However, during the training phase, the proposed approach required a significant amount of computational resources. Jiang et al. [99] proposed a novel defense framework for an IDS against adversarial attacks on an IoT network. This framework is known as a feature grouping and multi-model fusion detector (FGMD). This framework integrates different models, each of which processes distinct portions of input data or features to better withstand the effects of adversarial attacks. The authors used two publicly available datasets: MedBIoT and IoTID to validate the proposed model using adversarial examples. They utilized DT, LSTM, and RNN as the baseline models for comparison. The experimental results demonstrated the effectiveness of FGMD against adversarial attacks. FGMD outperform the baseline models in terms of detection rate.

Prasad et al.[100] proposed the Two-Tier Optimization Strategy for Robust Adversarial Attack Mitigation (TTOS-RAAM) model for IoT intrusion detection system. This hybrid framework combining Conditional Variational Autoencoders (CVAE) for adversarial detection and ICAVO for parameter tuning to address adversarial attacks. The attack type involved adversarial samples targeting IoT network traffic. Defense methods included feature optimization with CGWO and parameter tuning with ICAVO. The model achieved 99.91% accuracy on the RT-IoT2022 dataset; However, it lacked adaptive attack prevention mechanisms. Ramaiah et al.[101] proposed an ensemble classifier utilizing the Optimal Correlation Strategy (OCS) for feature selection in malware detection systems. While the defense method employed a robust feature selection approach it didn't incorporate defense against adversarial attack. The model demonstrated strong performance on the MTA-KDD'19 malware datasets, but its adaptability to zero-day attacks was limited.

Sánchez Sánchez et al.[102] employed an LSTM-CNN architecture based on hardware performance behavior for individual device identification. They conducted adversarial attack using deep learning evasion attacks. The adversarial defense strategies included adversarial training and the model distillation method to improve robustness. The model's robustness increased from an attack success ratio of 0.88 to 0.96. The experiment performed using LwHBench dataset. Ma et al.[103] proposed a method for detecting Low-rate Denial of Service (LDoS) attacks in Artificial Intelligence of Things (AIoT) environments. The authors introduced a novel cooperative defense scheme against hybrid LDoS attacks. They utilized programmable utilized the plane of Software-Defined Networking (SDN) to synchronize real-time and high-precision defenses. Experimental

results of LDoS demonstrated that their method detects more accurately and mitigates effectively than the traditional method.

Table 3: Related work summery for adversarial attack detection and defense

Author & year	Model	Attack technique	Defence	Learning method	Dataset
[93](2022)	GNN	HAA	✗	Batch learning	UNSW-SOSR2019
[94](2023)	CNN, GAP and CNF	Grad-CAM++ nad CAM	✗	Batch learning	IoT Trace dataset
[82](2023)	GNN	VGAE-MalGAN	✗	Batch learning	CICMaldroid and Drebin dataset
[95](2020)	FNN & SNN	FGSM & BIM	✗	Batch learning	BoT-IoT dataset
[96](2021)	ANN and SVM	FGSM	✗	Batch learning	Bot-IoT datasets
[97](2022)	DRL and GAN	✗	✗	Batch learning	DS2OS dataset
[99](2022)	DT, LSTM, and RNN	FGMD	✗	Batch learning	MedBIoT and IoTID
[100](2025)	TTOS-RAAM	✗	CVAE	Batch learning	RT-IoT2022 dataset
[101](2024)	RF, KNN, XGB and Voting	✗	✗	Batch learning	MTA-KDD'19
[102](2023)	LSTM-CNN	FGSM, BIM, MIM, PGD, JSMA, Boundary Attack, C&W	✓	Batch learning	LwHBench dataset
.[103](2024)	CNN-RF	✗	✗	Batch learning	
Proposed Solution	**ARF-ADWIN, ARF-DDM, HT, KNN-ADWIN, SRP-DDM, SRP-ADWIN, and PWPAE**	**FGSM, BIM, MIM, PGD, C&W and GAN**	Multi-layer	**Online learning**	**IOTID20 and CICIoT2023**

The reviewed research papers primarily focus on batch learning methods for DDoS detection, neglecting the dynamic nature of IoT environments. Consequently, these approaches fail to address concept drift, which occurs when data distributions change over time. None of the examined papers propose an adversarial defense for online DDoS detection systems. Furthermore, they are unable to manage multiple and unknown adversarial attacks in real-time environments. Our proposed

framework protects multiple and unknown adversarial against online DDoS attack detection system.

3.8.3. Federated Learning in Cybersecurity

Federated learning addresses the cybersecurity privacy challenge by moving the computation to the edge, rather than centralizing the data to perform the learning process. Hence, no data are shared with the central server for cyber-attack detection. Below, we discuss studies that utilized federated learning for DDoS attack detection.

Doriguzzi-Corin et al. [104] proposed an FLAD approach that utilized dynamic computational resource allocation based on attack difficulty profiles for DDoS attack detection. The authors used a recent DDoS dataset to compare FLAD with other traditional FL algorithms in terms of the convergence speed and model accuracy. The experimental results showed that FLAD outperformed traditional FL algorithms. Pourahm et al.[105] introduced an outlier exposure (OE)-enabled cross-silo federated learning framework (FedOE) for DDoS attack detection at the network edges. FedOE enhances detection capabilities by utilizing outlier exposure methods.

Yin et al. [106] provided a multidomain federated learning approach for DDoS attack detection. This approach ensures data privacy and defends against poisoning attacks. It utilizes blockchain and integrates multidomain DDoS attack detection to enhance detection capabilities. Tian et al.. [107] provided a lightweight residual network framework for DDoS attack detection using federated learning. The framework minimizes computational overhead and maintains high detection accuracy. Popoola et al. [108] provided a zero-day botnet attack detection using Federated Deep Learning (FDL). It also addresses data-privacy issues. This method utilizes a Deep Neural Network (DNN) for IoT botnet attack detection. Chaudhuri et al. [109] provided a Dynamic Weighted Federated Averaging (DW-FedAvg) method for Android malware detection to improve detection performance. This method applies weighted client updates based on the local model performance to improve traditional federated learning averaging. The experimental results showed that DW-FedAvg performed better on four benchmark datasets than the standard FedAvg.

Ragab at el. [110]proposed privacy preserving cyber threat detection using AAIFLF-PPCD framework in IoT environment. The author used an integration of feature selection method using Harris Hawk Optimization, a Stacked Auto-Encoder for classification and Walrus Optimization for hyperparameter tuning. The experiment result shows an accuracy of 99.47 that outperform baseline models cyber threat detection. Torre et al. [111]developed federated learning-based

Intrusion detection system using 1D CCN-based framework. The privacy of the proposed framework enhanced using that integrate feature selection method using Harris Hawk Optimization. The author used CICIoT2023, CICIoMT2024 and EdgeIIoT dataset. The experiment result demonstrated an accuracy of 97.31 while maintaining privacy. Alsaleh et al.[112] proposed clustered based FL intrusion detection system that assig a cluster head to each cluster to act on behalf of FL clients. This reduced communication overhead. The proposed models combine three different techniques such as LSTM, BiLSTM, and WGA using the CICIoT2023 dataset. The experiment result show BiLSTM achieves better performance under resource-constrained IoT devices. Olanrewaju-George et al. [113]proposed an intrusion detection system that utilize federated learning with unsupervised autoencoder (AE) and supervised deep neural network (DNN) model. The author used randomized search for hyperparameter optimization and N-BaIoT dataset to evaluate the proposed model. The experiment result shows FL-trained with AE model outperformed DNN mode. Moreover, the intrusion detection system is effective in preserving privacy in IoT environment.

Despite the rapidly increasing use of federated learning in cyber security domain, existing methods such as FedAvg, FedProx and IFCA have limitation in terms of accuracy, convergence speed, and scalability in non-IID condition. FedAvg performs uniform averaging without considering the data distribution. This leads to a slow convergence speed and performance degradation under non-IID IoT environments. FedProx partially addresses convergence issues by introducing a proximal term. However, it still experiences moderate accuracy degradation and limited scalability due to fixed regularization parameters. IFCA achieves high local personalization by clustering clients but incurs high computational and communication overhead as the number of clients and clusters grows. however, it lacks a dynamic weighting mechanism leading to moderate scalability and accuracy loss in highly non-IID environments. These limitations emphasise the need for a framework that incorporate clustered federated learning and dynamic weighted averaging component that address non-IID data, preserve privacy, and maintain high detection accuracy in real-world IoT cybersecurity scenarios.

Table 4:Related work summery for federated learning in cyber security

Author & year	Sector	Approach	Averaging weight	Non-IID
(Doriguzzi-Corin et al., 2023)	Cyber security	FLAD approach	Static	✗
(Pourahm et al.,2023)	Cyber security	FedOE	Static	✗
(Yin et al., 2022)	Cyber security	FL	Static	✗
(Popoola et al,2022)	Cyber security	DFL	Static	✗
(Chaudhuri et al. ,2022)	Cyber security	DW-FedAvg	Dynamic weighting	✗
(Ragab at el.,2025)	Cyber security	AAIFLF-PPCD	Static	
(Alsaleh et al.,2025)	Cyber security		Static	Clustering
(X. Li et al., 2019)	Image	FedAvg	Static	✗
(T. Li et al, 2018)	Image	FedProx	Static	✗
(A. Ghosh et al.,2020)	Image	IFCA	Static	Clustering
Proposed	**Cyber security**	**DWCFL**	**Dynamic weighting**	**Clustering**

As we can see from Table 3, none of the previous papers integrates clustered federated learning and dynamic weighted averaging. The integration of clustered federated learning and dynamic weighted averaging for clustered clients in the context of cybersecurity has not been explored in non-IID environment. Our contribution mainly lies in leveraging the strengths of both approaches to handle non-IID and enhance the performance of federated learning and preserve privacy. Clustered federated learning groups similar IoT devices based on data distribution, allowing more personalized and relevant model updates within each cluster. This approach effectively handles non-IID data issues by clustering similar IoT devices to enhance the learning process. Moreover, the dynamic weighting adjusts the weight of each client's model update based on local performance to mitigate the impact of noisy or less significant data to ensure. We have seen also inadequate theoretical convergence analysis and extensive empirical validations across various benchmark datasets to demonstrate the performance improvement and convergence speed while preserving IoT data privacy.

3.8.4. Summary of related works

The reviewed literature in section DDoS detection, adversarial attack defense, and federated learning in IoT systems highlights significant progress and research gaps. Several machine learning DDoS attack detection approaches such as CNN, LSTM and Hybrid models have demonstrated high detection accuracy using benchmark datasets. These DDoS detection models are batch learning. These batch learning models fail to detect and adapt concept drift in dynamic IoT environments. Although several adversarial attack generation and defence strategies such as hierarchical attacks, adversarial training, and GAN-based methods, none of these strategies are designed to operate under online data streaming. Moreover, they are robust to known adversarial attacks. This limit practical applicability in real-time IoT environments. Federated learning has shown promising privacy-preserving solution for distributed DDoS attack detection. However, the existing methods such as FedAvg, FedProx, and IFCA struggle in high accuracy in non-IID conditions, lack dynamic client weighting, or incur high communication and computational overhead. Despite the potential benefits of integrating clustered federated learning with dynamic weighted averaging. There is no prior work that integrate these strategies for improving detection accuracy, scalability, and convergence speed in non-IID IoT environments. This dissertation aims to bridge this gap by proposing online, robust to adversarial attack, and privacy-aware DDoS attack detection framework suitable for real-world IoT deployments.

Chapter Four

4. Proposed solutions

In the landscape of Internet of Things (IoT) security, the widespread occurrence of zero-day attacks, concept drift, and adversarial manipulation necessitates an end-to-end and robust detection framework. To counter such dynamic threats, for this dissertation, three expert solutions: AUWPAE, MSAAD, and FedDWC are proposed to be a cohesive, end-to-end pipeline for secure, privacy-preserving, and adaptive Distributed Denial-of-Service (DDoS) detection. Each framework has a unique but complementary function within the system: AUWPAE is a stand-alone online learner and a reusable ensemble technique; MSAAD incorporates adversarial detection and cleansing techniques over; and FedDWC offers scalable, federated learning over IoT edge devices. This chapter demonstrates the organization of the architecture and the coordination of the operations of these components, and an illustration of how their coordination allows strong detection under real-world constraints such as adversarial inputs, non-IID data, and resource constraints. The remaining sections explain the integration of the frameworks layer by layer, their specific tasks, and how coordination among them provides greater resilience, lower communication overhead, and system-level adaptability.

4.1. Proposed architecture and integration components

The Multi-Stage Adversarial Attack Defense (MSAAD) framework has three defense layers. First layer, the Resilient Adversarial Detection and Purification (RADP) which employs a pre-trained model using GAN and Multi-Source Adversarial Perturbation Generator (MSAPG). RADP detects and purifies adversarial perturbations from IoT data stream prior to downstream processing. The second layer introduces a collection classifier, which are intended to enhance robustness by making it hard for an adversary to generalize or evade the detection. The third layer is MBATSE framework which dynamically select the best-performing classifiers for each input data. AUWPAE utilized as adaptive ensemble strategy that aggregating the results of the chosen base learners according to performance in real-time. AUWPAE fulfils a dual function in the architecture: First, as an independent drift-aware online detection model. Second, as ensemble method incorporated within the third layer of MSAAD for increased DDoS attack protection.

FedDWC serves as the core of the architecture by enabling distributed learning. At the client side, all edge nodes execute the complete MSAAD pipeline within AUWPAE is integrated as a

key component. Adversarial tested and adaptively aggregated clean model updates are regularly uploaded to FedDWC for secure aggregation. FedDWC clients are dynamically clustered based on data distribution. Each cluster performs intra-cluster aggregation, followed by global aggregation at the cloud level. This design enhances personalization while preserving scalability and privacy. Furthermore, FedDWC provides high-level performance feedback to fine-tune AUWPAE's ensemble weights and MSAAD's classifier selection and purification thresholds, thereby completing the optimization loop.

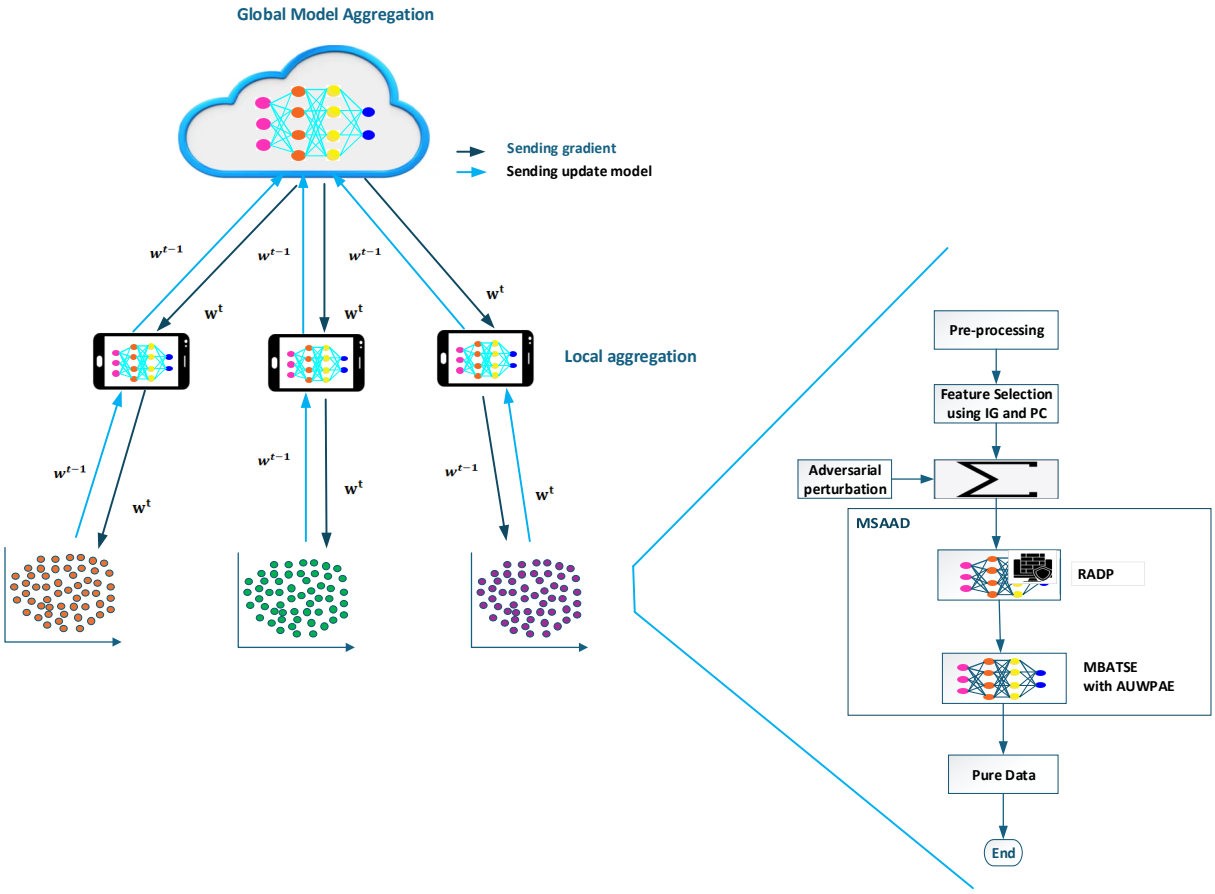

Figure 4: Proposed solutions high level design

4.2. Process Flow

In the proposed architecture process flow as follows:

- real-time IoT data streams pass through a pipeline beginning with the pre-processing module, which normalizes and structures the raw inputs.

- These pre-processed inputs are then forwarded to the MSAAD framework. Within MSAAD, the RADP module powered by MSAPG and GAN module detect and purifies known and unknown adversarial perturbations. If this purification succeeds, clean data proceeds to the multi classifier.
- If RADP is overwhelmed or partially fails (e.g., due to high perturbation budgets or novel attack strategies), the system still forwards the possibly perturbed inputs to the ensemble detection layer, where AUWPAE dynamically re-weights classifiers to maintain robust detection even under degraded conditions.
- The predictions and model weights generated by AUWPAE either in standalone online mode or embedded within MSAAD are then transmitted to FedDWC system. At the client side, all edge nodes execute the complete MSAAD pipeline within AUWPAE is integrated as a key component. FedDWC clients are dynamically clustered based on data distribution. Each cluster performs intra-cluster aggregation, followed by global aggregation at the cloud level

4.3. Key Algorithms

The integrated solution is built upon three foundational algorithms that enable its core capabilities. The Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) framework, an adaptive online learning approach specifically designed for real-time DDoS detection in IoT systems. The full methodological details in Chapter 4. The Multistage Adversarial Attack Defense (MSAAD) framework, designed to improve the robustness of online detection systems against both known and unknown adversarial attacks. While its MABTSE component selects optimal classifiers via Thompson sampling, as fully elaborated in Chapter 5. The FedDWC algorithm ensures scalable and private model aggregation by dynamically clustering clients based on data similarity and applying dynamic weighted averaging within clusters. The complete algorithmic formulation and convergence analysis provided in Chapter 6.

4.4. Novelty and Contributions

The novelty of this research lies in the synergistic integration of three originally designed frameworks that collectively address the intertwined challenges of concept drift issue of online DDoS attack detection, adversarial attacks defence against online DDoS attack detection, and data heterogeneity in IoT DDoS attack detection. The AUWPAE framework introduce an online DDoS attacks detection under real-time streaming conditions, detailed in Chapter 4. The MSAAD

framework provide novel multi-stage defense pipeline combining Resilient Adversarial Detector and Purification (RADP), multiple classifiers, and an ensemble defense method as presented in Chapter 5. The FedDWC framework innovates federated learning for IoT through dynamic clustering and performance-weighted aggregation, enhancing personalization and convergence under non-IID data, as developed in Chapter 6. Collectively, their integration establishes a novel end-to-end paradigm for adaptive, adversarial robustness and privacy-preserving DDoS attack detection.

4.5. Scope and Limitations

Although AUWPAE enhances detection accuracy of online system, the ensemble aspect of it affords reasonable computational overhead, which could be a concern for deployment on ultra-low-power IoT devices. MSAAD layered defense introduces latency and has moderate memory requirements, which is difficult for highly resource-constrained IoT deployments. FedDWC expects honest clients provide honest local model updates. The system does not automatically protect against Byzantine attacks like model poisoning or backdoor injection. Moreover, the system also relies on the assumption that clients have adequate connectivity and stability to participate in regular local training and aggregation cycles. Performance and privacy assurances can be lost in cases of unstable bandwidth, frequent packet loss, or hacked clients. On this research, we planned to address DDoS attack focussing on the concept drift and non-IID data distribution in IoT environment, privacy protection from the conventional federated learning viewpoint and adversarial attack detection. Hence, energy consumption analysis, testing using live IoT, privacy using cryptographic or other preserving mechanisms and defend against less overt attacks, such as 'clean-label poisoning' or 'model inversion' attacks are out of scope.

4.6. Expected Outcomes

The proposed system is designed to achieve measurable performance gains validated across subsequent experimental chapters. Detection performance is expected to exceed 99% accuracy with robust recovery from adversarial degradation, as quantitatively demonstrated in Chapter 5. Latency metrics for AUWPAE and MSAAD components are designed to meet real-time IoT thresholds below 3 ms per sample, with empirical analysis provided in Chapters 4 and 5. Federated efficiency through FedDWC is anticipated to show accelerated convergence and enhanced stability with non-IID data and client dropouts, supported by scalability experiments in Chapter 6. These combined outcomes position the framework as a comprehensive solution for next-generation adaptive IoT security.

Chapter Five

5. Drift Adaptive Online DDoS Attack Detection Framework for IoT System

The dynamic nature of IoT environments needs DDoS detection solution that operate in real time with minimal resource overhead. Traditional batch learning approaches fails to meet this requirement due to the need a complete dataset for retraining model. This create delays, and high computational costs. In addition to, the presence of concept drift such as both gradual, sudden and incremental decline the performance static models over time and make in effective in detecting zero-day attacks. This chapter introduces the Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) framework, an adaptive online learning approach specifically designed for real-time DDoS detection in IoT systems. AUWPAE addresses Research Question 1 (RQ1) and fulfils Objective (OB1) by enabling continuous learning from streaming data ensuring timely detection of both known and zero-day DDoS attacks. AUWPAE achieves high detection accuracy with minimal memory and processing requirements, making it suitable for deployment on resource-constrained IoT devices.

The proposed online DDoS attack detection framework using online data stream analytics in IoT systems has three main steps: pre-processing IoT data, DDoS detection using a base model, and DDoS detection using online ensemble methods. The K-means clustering approach is used to acquire a more representative subset of the incoming IoT data streams. In order to standardize sample data distribution and scale every feature, a Z-score and min–max normalization is used. This step is discussed in Section 3.2. To handle concept drift adaptation and perform DDoS attack detection, four base learners, ARF-ADWIN, ARF-DDM, SRPs-DDM, and KNN-ADWIN, are developed. The development of the base learners is presented in Section 3.3. These base learners are made to adapt to the changing data distribution and detect DDoS attack in the data stream. Finally, the proposed AUWPAE approach, which is based on the ensemble model, is discussed in Section 3.4. AUWPAE combines the perdition of base models using their real-time mean square error rates. Figure 5 shows the proposed drift adaptive online DDoS attack detection framework for IoT systems.

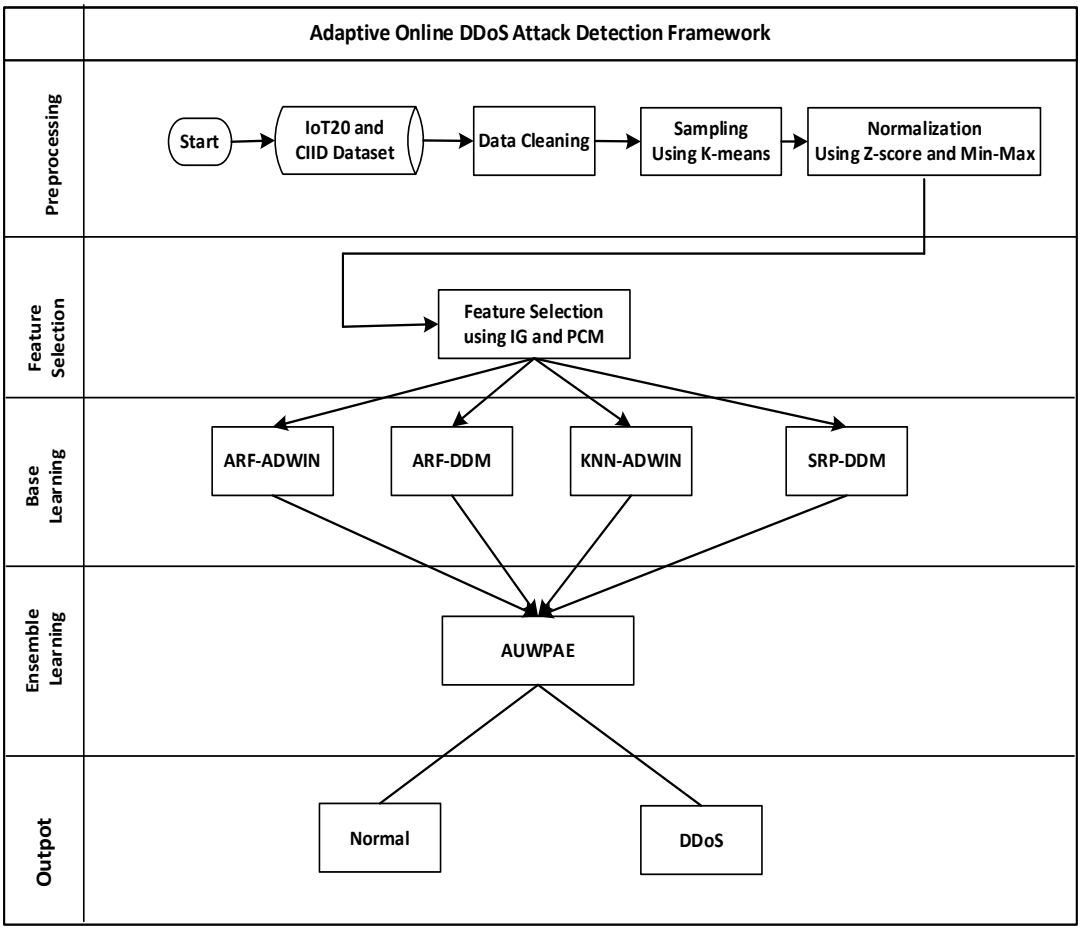

Figure 5: Drift adaptive online DDoS detection framework.

5.1. Data Pre-processing

Data pre-processing is a basic step in all machine learning applications including DDoS attack detection. The performance and accuracy of the detection method can be significantly impacted by the representation, size, and quality of the incoming data. Particularly, selecting a dataset with high dimensionality and a large number of duplicate and irrelevant features will affect training. To address these issues, in the data pre-processing phase, we used data cleaning, data encoding, data normalization, and feature selection techniques.

5.1.1. Data Cleaning

The selected dataset network contains a number of different distinct dataset files that are merged into a single file. The merged single file is referred to as a combined dataset. IoT data values are generated in real-world applications as words or strings. Real-world datasets frequently have missing values due to data accessibility issues or challenges in gathering the required data. Missing values such as blank spaces, NaNs, and erroneous data types could be represented using

null values. Most machine learning models, however, cannot deal with missing values directly or are negatively impacted during learning. The goal of data cleaning is to fill the gaps left by missing data with acceptable values [31]. A number of basic cleaning techniques are used to fill missing variables. In this research, we replaced the missing values with the means of each column.

5.1.2. Data Sampling

The selection of highly representative data samples requires the use of an efficient data sampling method. The proposed framework uses the K-means based cluster sampling method to obtain a representative subset. Clustering algorithms have important roles in unsupervised models in grouping data samples based on their similarities. One of the common clustering methods is K-means, which divides an unlabeled dataset into K clusters based on the degree of similarity between data points.

5.1.3. Data Encoding

In real world IoT data, values are generated as words or strings to make them human-readable. Data encoding involves the conversion of string information into numerical features that machine learning models are able to understand and process. Label, one-hot, and target encoding are examples of frequently used encoding methods. In this research, we used target encoding to replace categorical values with the means of the target variables.

5.1.4. Data Normalization

There are different attribute values in the combined dataset, and some features have values that are spread out over a large range while other values are spread out over a small range. This variation could affect the performance of the models. To solve this issue, we used normalization techniques to scale the feature values in the same range. Two common normalization methods for data analytics are obtaining the Z-score and min–max normalization. The min–max normalization approach transforms the original data linearly to achieve a balance of comparative values between the data before and after processing. The min–max normalization method scales a feature's values to a range between 0 and 1. The min–max scaling approach is shown in Equation (14) [32]:

$$X_{new} = \frac{X - Min(X)}{\mathrm{Max}(X) - Min(X)} \tag{14}$$

where Xnew is the result of normalization and X is the value to be normalized; Min and Max are the minimum and maximum values in each feature.

The Z-score normalization method is based on the mean and standard deviation of the data. This method is helpful if the minimum and maximum values of the data are unknown. Equation (15) shows the formula for Z-score normalization [33]:

$$X_{new} = \frac{X - \mu}{\delta} \tag{15}$$

where Xnew is the result of normalization, X is the value to be normalized, μ = population mean, and σ = standard deviation. Min–max normalization can keep outliers in datasets; hence, it is more appropriate for DDoS attack detection models. On the other hand, Z-score normalization is robust against outliers; hence, it usually works well for data analytics problems related to non-outliers. As a result, the proposed framework chooses both the Z-score and min–max normalization to handle this issue.

5.1.5. Feature Selection

Feature selection is used to choose a subset of the original feature set in order to increase the performance and speed of machine learning models. The best-performing minimal feature set can be identified by evaluating all feature combinations. However, with a high number of features, optimization approaches can be used to examine the feature search space and determine the optimal feature set. Information gain (IG) and Pearson correlation methods are used to remove irrelevant and duplicate features, respectively.

5.2. Concept Drift

The majority of IoT devices used in IoT systems have limited computing power and storage [4]. This limited memory capacity hinders their ability to handle and retain massive amounts of data and complex learning models. Therefore, it is crucial to develop analytics models with low computing complexity. Online learning methods that enable real-time analytics are able to fulfill IoT system time and memory requirements. Online learning methods can continuously update the learning model with new data samples when they arrive within a short execution time in contrast

to that required under the batch learning method, which frequently trains a learning model on the entire training dataset. IoT data samples are usually produced dynamically in constantly changing IoT environments. The dynamic nature of streaming traffic also makes already trained models less useful in recognizing zero-day attack.

Data analytics frequently experiences concept drift issues in real-world applications due to the change in IoT data distribution over time. Concept drift is caused by three main data distribution changes: recurring, gradual, and sudden [17]. Sudden drift happens when rapid irreversible changes occur in a short period of time. Gradual drift happens when the data distribution gradually replaces the old one over time. Recurring drift happens when the previous data distribution happens again over time.

Concept drift is formally defined as a set of samples, indicated as $S_{0,t} = \{d_0, \dots, d_t\}$ for a time interval of [0, t], where $d_i = (x_i, y_i)$ represents a single observation, x_i is the feature vector, y_i is the label, and $S_{o,t}$ adheres to a specific distribution $f_{o,t}(x, y)$. If $f_{o,t}(x, y) \neq f_{t+1,\infty}(x, y)$, denoted as $\exists_t : p_t(x, y) \neq p_{t+1}(x, y)$, then concept drift happens at timestamp t_{t+1}. Concept drift at time t can also be defined as the change in the joint probability of x and y at time t, expressed as $p_t(x, y) = p_t(x) * p_t(y/x)$. Concept drift will happen under the following three conditions.

- $p_t(x) = p_{t+1}(x)$ while $p_t(y/x) = p_{t+1}(y/x)$ $p_t(x)$. This type of drift is known as virtual drift because $p_t(x)$ does not affect the decision boundary.
- $p_t(y/x) \neq p_{t+1}(y/x)$ while $p_t(x) = p_{t+1}(x)$ while $p_t(x)$ remains unchanged. This is considered actual drift because it affects the decision boundary and also leads to a decline in learning accuracy.
- The combination of the first two, $p_t(x) \neq p_{t+1}(x)$ and $p_t(y/x) \neq p_{t+1}(y/x)$.

Concept drift poses significant issues when building machine learning models because it can cause changes in data distribution that result in machine learning model performance gradually declining over time [18]. Hence, advanced online machine learning models need to be developed in order to detect and adapt the concept drift that occurs in IoT data streams.

5.3. Drift Detection

Drift detection is a crucial element for adaptive machine learning models that can solve concept drift issues. The two primary categories of drift detection methods are performance-based and distribution-based [19].

5.3.1. Performance-Based Methods

Performance-based concept drift detection methods are based on changes in the metrics used to evaluate model performance. Common examples of indicators of concept drift are accuracy decline and increase in error rate. If the error rate of a learner gradually decreases or remains constant as more samples are learned, it often indicates constant data distribution without drift. On the other hand, if a learner's error rate drastically increases as more data is processed, it often indicates the presence of concept drift. The two well-known performance-based drift detection methods are the drift detection method (DDM) and early drift detection method (EDDM), which are able to detect concept drift by keeping track of degradations in performance. The DDM is a popular performance-based drift detection method that measures model error rate and standard deviation changes using two predefined thresholds: the warning threshold and the drift threshold [20]. DDM often performs well on data streams with sudden drift, but its reaction time is often too slow for detecting gradual drift. The early drift detection method (EDDM) is an enhanced version of the DDM that detects concept drift using the same concept drift detection mechanism [10]. The EDDM often performs well on data streams with gradual drift. Even though the EDDM frequently performs better than does the DDM, it still falls short for sudden drift. Furthermore, due to its sensitivity to noise, it may mistake noise for drift, resulting in false alarms.

5.3.2. Distribution-Based Methods

Distribution-based concept drift detection is based on changes in data distributions. Data distribution changes can be measured using statistical variables such as mean, variance, and information entropy. Adaptive windowing (ADWIN) is one such widely used method that uses adaptive sliding windows to detect concept drift based on the statistical difference between two adjacent sub-windows. The adaptive windowing method uses characteristic values such as the mean and variance, as well as variable-size sliding windows to detect concept drift. The window size is dynamically increased if there is no concept drift and reduced when concept drift is detected [9]. IoT systems with less memory often adopt distribution-based methods since they only need to retain the most recent samples. Windowing methods are often quick and simple to use. However, they could miss certain important historical data.

5.4. Drift Adaptation

Concept drift must be handled after it is detected by updating the current models using the appropriate drift adaptation methods. Drift adaptation is a procedure used to update a model automatically when a concept drift occurs to enhance performance of model and detect zero-day attacks. The procedures usually fully retrain or alter the learning model using new dataset. The three categories of drift adaptation methods are model retraining, incremental learning and ensemble learning.

5.4.1. Model Retraining

One of the more simple and straightforward methods to handle concept drift is model retraining. Offline models are often unable to accurately predict unseen incoming streaming data. This problem can be addressed by retraining the model using the most recent data streams. However, employing this technique can cause unnecessary model retraining or drift adaptation delays. Therefore, it is crucial to use learning models together with an appropriate drift detection method to determine when to retrain the learning model for timely and necessary updates.

There are two types of model retraining methods: full retraining and partial retraining. Retraining the learning model using the entire dataset and all available samples is known as full retraining, while partial retraining trains the model on selected parts of the dataset. The window-based method is used to partially retrain a model using the most recent data. This reduces training times but may result in the loss of historical patterns. Hence, selecting the right window size is crucial. ADWIN is a drift detection method that uses model retraining. ADWIN performs better

as it uses a dynamic window to fit new data [9]. The learning model is partially retrained on only the new concept samples in order to save training time.

5.4.2. Incremental Learning Methods

Incremental learning has become commonly used in data stream analytics research. Incremental learning involves updating the learning model when each instance is processed. Incremental learning methods often partially update the learning model to fit a new data sample [21]. They do not require a sufficient amount of data prior to the training process due to the incremental learning approaches' capacity to support progressive learning. However, only a few machine learning algorithms such as MLP and multinomial NB support partial updates.

5.4.3. Ensemble Learning Methods

Ensemble learning approaches have been developed to provide powerful learners for data stream analytics to enable stronger concept drift adaptation. Ensemble learning combines multiple base learners to tackle the same problem [22]. Ensemble learning base learners can be constructed using different algorithms, and different configurations of hyperparameters configuration. Ensemble learning models are often more generalizable than single models because they combine the outputs of multiple base learners. Reusing existing models in an ensemble is significantly more effective for concept drift adaptation than training new models on data streams with recurrent concept drift [23].

Block-based ensembles and online ensembles are the two main categories of ensemble techniques used in data stream analytics [24]. Data streams are divided into fixed-size blocks by block-based ensembles, which then train a base learner on each block. The base learners will be evaluated and updated each time a new block arrives. Block-based ensembles react to gradual drifts accurately, but they frequently take longer to respond to sudden drifts. Three common block-based ensembles are accuracy-weighted ensemble (AWE) [25], accuracy-updated ensemble (AUE) [26], and the streaming ensemble algorithm (SEA) [27]. The AUE often performs better among the block-based ensembles [28]. AUE uses non-linear error functions to apply weights to base learners in order to improve performance.

To enhance the learning performance of online ensembles, different incremental learning models such as Hoeffding trees (HTs) are included. Gomes et al. [11] proposed the adaptive random forest (ARF) approach, which makes use of HTs as base learners and ADWIN as a drift

detector. The drift detection mechanism replaces underperforming base trees with new trees that better fit. Since random forest is a well-known, effective machine learning algorithm, ARF frequently outperforms many other methods. ARF also includes a powerful resampling method and the flexibility to accommodate various drifts types.

Gomes et al. [12] also proposed streaming random patches (SRPs) for the adaptive ensemble approach. Although its execution time is usually longer, SRPs often exhibit slightly better prediction accuracy than ARF. Leverage bagging (LB) [29] is another online ensemble that constructs base learners using bootstrap samples. Although LB is simple to construct, it usually performs worse than SRPs and ARF. Despite the fact that there are several concept drift adaptation strategies in use today, their efficacy is constrained by slow drift response and poor prediction accuracy.

Incremental learning methods often perform poorly due to their low model complexity and limited drift adaptability [30], whereas block-based ensembles face significant difficulties in determining block size and responding quickly to drift. Online ensembles, like ARF and SRPs, often outperform incremental learning and block-based ensemble methods. However, because of their randomization strategies, they give unstable learning models, adding more unpredictability to DDoS attack detection.

In line with the previous research, this research aims to detect DDoS attacks in IoT systems while addressing the concept drift issue. In particular, this paper proposes adaptive online DDoS attack detection using the ensemble learning method to address the issue of concept drift in IoT system DDoS attack detection.

5.5. Drift Adaptation Base Model Selection

The proposed framework consists of four base models that are used to construct the online ensemble model (see Figure 5). The framework aims to balance learning performance and efficiency since it is designed for online IoT systems. Most of the existing data stream classification algorithms specialize in only one type of drift. Some classifiers are best suited for gradual drift while others are suited for sudden drifts. Our research aims to develop a data stream classifier that reacts to both types of concept drift. In this research, the two commonly used drift detection methods, ADWIN and DDM, are used together to detect concept drift. ADWIN's sliding window can be expanded to a large-size window to detect long-term changes. This makes ADWIN an ideal match for data streams with gradual drift. DDM often works well on data streams with sudden drift, but its reaction time is often too slow for detecting gradual drift. The framework does not immediately discard older models when drift is detected; instead, it assigns weights to each learner based on recent predictive accuracy and drift-awareness. This allows previously relevant models to retain reduced influence and recover if the underlying data distribution returns, thus supporting robustness in the face of recurrent concept drift. Additionally, to mitigate false positives caused by noise, the framework employs a probabilistic averaging mechanism with decline functions and adaptive thresholds, which smooth short-term fluctuations. This balance ensures that the ensemble remains both responsive and stable under varied drift conditions.

Online ensembles can improve learning performance using various incremental learning models like Hoeffding trees (HTs). The adaptive random forest (ARF) method uses HTs as base learners and ADWIN as a default drift detector [11]. The drift detection mechanism replaces underperforming base trees with new trees that fit the new concept. Since the random forest method is an effective technique, ARF often outperforms a variety of other methods. ARF makes the best use of resampling, and it is adjusted to a variety of drift types.

Streaming random patches (SRPs) is an alternative adaptive ensemble method for streaming data analytics [12]. SRPs uses a combination of online bagging and random subspace algorithms for predictions. It uses a similar approach to the one above, except SRPs employs a global subspace randomization mechanism and ARF's local subspace randomization. A flexible technique for increasing the diversity of base learners is global subspace randomization. Although SRPs' execution time is longer than ARF's, its prediction accuracy is frequently slightly higher.

ARF and SRPs are advanced drift adaptation approaches that have shown better performance through experimental research than other drift adaptation approaches have [11,12]. ARF is an advanced ensemble model that builds HTs for drift adaptation using local subspace randomization and that uses a drift detector for drift detection. ARF has demonstrated good performance and excellent execution time compared to those of others in solving data stream analytics problems [11]. As a result, ADWIN and DDM drift detectors are selected as base models for the proposed ensemble framework. The ensemble uses two ARF models with different drift detectors to preserve high accuracy.

The other two models are then selected among seven online learning methods (LB, SRPs, OPA, PWPAE, SRPs-ADWIN, SRPs-DDM, and KNN-ADWIN) by considering execution time and performance. LB [34], SRPs, and SRPs-ADWIN can effectively solve concept drift but they are not as effective as ARF. Although HTs' computational costs are low, this method is not selected due to performance limitations. The other two models, OPA and PWPAE [30], are not selected because of their high computational complexity. Hence, ARF-ADWIN, ARF-DDM, SRPs-DDM, and KNN-ADWIN are selected for our proposed online learning framework. Below is a summary of the justifications for selecting these models.

ADWIN performs better for detecting gradual drift whereas DDM performs effectively for detecting sudden drift. Combining these two makes it possible to detect sudden and gradual concept drifts.

Experimental results demonstrate that ARF-ADWIN and ARF-DDM perform better at handling concept drift than do other drift adaptation approaches. Since ARF-ADWIN and ARF-DDM are extensions of the adaptive random forest (ARF) method, the proposed framework inherits the ability to adapt to concept drift and improve overall system performance.

The selected models, ARF-ADWIN, ARF-DDM, SRPs-DDM, and KNN-ADWIN, result in better performance while maintaining a shorter execution time. Hence, they are well suited for online DDoS attack detection and concept drift adaptation.

5.6. Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE)

In this research, a novel Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) is proposed for combining the base learners for IoT data stream analytics. AUWPAE makes use of the advantages of the previous approaches while mitigating their limitations.

AUWPAE provides dynamic weights to base learners in accordance with their real-time performance. The AUWPAE approach utilizes the weighted average approach. The prediction probability of each base learner is assigned a weight and multiplied by the prediction probability of that base learner. The multiplication results are summed to determine the final prediction probability. The prediction class of the proposed model will be the final one with the highest average probability [21]. Given a data stream, $D = \{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\}$, consisting of c different target classes, $y \in 1, \ldots c$. The predicted target category, $\hat{y}$, for each data input, x, can be expressed mathematically using the following Equation (16) [21]:

$$\hat{y} = \underset{i \in (1, \ldots, c)}{arg\ max} \frac{\sum_{j=1}^{k} w_j p_j(y_j = i \,|L_j,\ x)}{b} \tag{16}$$

where L_j stands for the jth base learner model, $p_j(y = i\,|L_j, x)$ defines the probability of predicting a class value, i, on a data sample, x, using the jth base learner, L_j; b is the number of base learner models, which in our case is b = 4; and w_j denotes the weight of each base learner model, L_j. The weight of base learner can be quantitatively assessed using the accuracy-updated ensemble method [28]. For every data stream, the weight of the base learner model, w_j, is calculated by estimating the mean square error rate on data stream $D = (x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$, as shown in Equations (17)–(19):

$$MSE_{ij} = \frac{1}{|x_i|} \sum_{\{x,y\} \in x_i} (1 - f_y^j(x))^2 \tag{17}$$

$$MSE_r = \sum_{y} p(y)(1 - p(y))^2 \tag{18}$$

$$w_{ij} = \frac{1}{MSE_r + MSE_{ij} + \in} \tag{19}$$

where the function $f_y^j(x)$ represents the probability given by the base learner model, L_j, that X is an instance of class y. The accuracy-updated ensemble algorithm considers probabilities for all classes instead of making predictions for a single class. MSE_{ij} evaluates the prediction error of the base learner model, L_j, on datum x_i. MSE_r is the mean square error of a randomly predicted base learner model and is used as a reference point for the current class distribution. To avoid issues with division by zero, a very little positive value, $\in$, is also added to the equation. Equation (20) is used as weighting formula to combine the accuracy of the base learner model with the current class

distribution. In addition, assigning ensemble members new weights for each datum, x_i, a candidate base learner model, L_j, is built from instances in the most recent data. L_j is a "perfect" base learner model, since it is trained on the most recent data. Equation (20) is used to determine its weight.

$$w_{ij} = \frac{1}{MSE_r + \in} \tag{20}$$

The weight of the candidate base learner model, L_j, does not take into consideration the prediction error for base learner L_j in comparison with the function used to weight the members of the current ensemble. This approach is based on the assumption that the most recent data reflect the distribution of the present and the near-future data. L_j is considered to be the best base learner model available, because it is trained using the most recent data. The proposed Accuracy Update Weighted Probability Average Ensemble Algorithm (Algorithm 1) is shown below.

The computational complexity of the AUWPAE model is primarily determined by the complexity of the selected based models; the AUWPAE method itself has a linear computational complexity of O(pwb), where p is probability of predicting the base learner, b is the number of base learner models, which in our case is b = 4, and, w denotes the weight of each base learner model. Most of the performance comparisons are conducted with the base learners used in the proposed framework. AUWPAE adds a weighting algorithm that is linear in complexity on top of the base learners. Hence, the proposed framework should have nearly the same complexity as the base learner, which can be further improved by replacing lower complexity base learners

Algorithm 1: *Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE).*
Input: *X_train, y_train, X_test, y_test*
Output: *t, m (time steps and accuracy metrics)*
function *AUWPAE (X train, y train, X test, y test)*

Initialize a, t, e, eps=0.0001
Initialize base learners $j = 4$
for each *instance* (x_i, y_i) ***in*** X_train, *y_train* ***do****. // learn base learners using training set*
    $L_j \leftarrow (x_i, y_i)$
end for
for each *instance* (x_i, y_i) ***in*** X_test, ***in*** y_test ***do****. // predict using test set*
    $y\_pred_j \leftarrow bl\ (x_i)$ *//predict target*
    $y\_prob_j \leftarrow blProbPred\ (x_i)$ *//target prediction probability*
    $bl.update(x_i, y_i)$ *//update base leaner.*
end for
for *j = 1 to 4* ***do***
    $e_i \leftarrow MSE_i(y_i, y\_pred_i)$*//calculate real-time MSE for each base learner*

13 $W \leftarrow 1/(e_j + eps)$ *//calculate weight*
14 $y\_prob_{j0} \leftarrow y\_prob_j$ *//probability for class 0*
15 $y\_prob_{j1} \leftarrow y\_prob_j$ *//probability for class 1*
16 $y\_prob_0 \leftarrow y\_prob_j \sum_{j=1}^{4} W_j * yprob_{j0}$ *//average weighted probability for class 0*
17 $y\_prob_1 \leftarrow y\_prob_j \sum_{j=1}^{4} W_j * yprob_{j1}$ *//average weighted probability for class 0*
18 ***end for***
19 ***If*** $y\_prob_0 > y\_prob_1$
20 . $y\_pred \leftarrow 0$
21 ***else*** $y\_prob_0 < y\_prob_1$
22 $y\_pred \leftarrow 1$
23 *calculate accuracy, precision, recall, and F1-score;*
24 ***return*** *t, a*
25 ***end function***
26 *Initialize a, t, e, eps=0.0001*

The proposed Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) makes use the advantages of the previous approaches while mitigating their limitations. We developed AUWPAE to address both sudden and gradual drifts, and enhance the detection and adaptability capabilities of current methods. Further, this study aims to proposed a novel ensemble framework that successfully achieves a trade-off between execution time and predictive accuracy.

5.7. Convergence Analysis of the AUWPAE.

Performance theoretical validation our proposed AUWPAE model employ regret analysis from online learning theory. Regret analysis quantifies the cumulative difference between the loss incurred by the ensemble model and the best-performing base learner selected in hindsight. The regret bound for AUWPAE is $R_T \leq \sqrt{2T \log(n)}$ where, T is the number of time steps and n is the number of base learners. We conclude that AUWPAE gets sublinear regret, and its average regret goes to zero as $T \rightarrow \infty$. The result demonstrates that AUWPAE converges to the optimal individual learner's performance, guaranteeing stable learning even under streaming scenarios. The proof relies on assumptions like convexity and boundedness of loss function, and emphasizes the theoretical stability of AUWPAE even in concept drift. The detailed proof for the regret bounds and convergence analysis for AUWPAE as follows:

Assumptions

1. **Convex Loss Functions**: The loss functions $\mathcal{L}_t$ are convex.

2. **Bounded Losses**: The individual losses are bounded, i.e., for all t and all models f in the hypothesis class F, there exists a constant U such that $l(f(x_t), y_t) \leq U$
3. **Number of Base Learners**: The ensemble consists of n base learners

Theorem 1: Regret Bound

Let $\mathcal{L}_t$be the loss of the ensemble model at time t on instance $(x_t , y_t)$and let $\mathcal{L}_t^*$ be the loss of the best possible model in hindsight. The cumulative regret R_T after T rounds is defined as:

$$R_T = \sum_{t=1}^{T} \mathcal{L}_t - \sum_{t=1}^{T} \mathcal{L}_t^* \tag{21}$$

The cumulative regret R_T is bounded by:

$$R_T = \sqrt{2nT \log(n)} \tag{22}$$

Assume the ensemble model is initialized with n base learners. Each learner j is associated with an initial weight $w_i(1)$ such that the weights sum to 1:

$w_j(1) = \frac{1}{n}$ for all j

The loss of the ensemble at time t is the weighted average of the losses of the base learners:

$$\mathcal{L}_t = \sum_{j=1}^{n} w_j(t) l_j (x_t, y_t) \tag{23}$$

The loss of the best possible model in hindsight at time t is

$$\mathcal{L}_t^* = \min_{f \in F} l(f(x_t), y_t)$$

The cumulative regret R_T is

$$R_T = \sum_{t=1}^{T} \mathcal{L}_t - \sum_{t=1}^{T} \mathcal{L}_t^* \tag{24}$$

The weight update rule for the Hedge algorithm is:

$$w_j(t+1) = w_j(t)\, exp\, (-\eta l_j(x_j, y_j)) \tag{25}$$

where η, is the learning rate

Normalize the weights so that they sum to 1:

$$\overline{w_j}(t+1) = \frac{w_j(t+1)}{\sum_{j=1}^{n} w_j(t+1)} \tag{26}$$

Define the potential function:

$$\theta = \sum_{j=1}^{n} w_j\,(t) \tag{27}$$

analysis using potential function

Initially, $\theta = 1$ because the weights are initialized to sum to 1.

The update rule ensures that the potential function evolves as:

$$\theta_{t+1} = \sum_{j=1}^{n} w_j(t)\, exp\,(-\eta l_j(x_j, y_j)) \tag{28}$$

To apply Jensen's inequality, For a convex function f, Jensen's inequality states that for any probability distribution $\{p_j\}$ and values $\{x_j\}$:

$f(\sum_{j=1}^{n} p_j\, x_j) \leq \sum_{j=1}^{n} p_j \mathrm{f}(x_j)$

In our context, we can normalize the weights to form a probability distribution:

$\frac{w_j(t)}{\theta_t}$, where $\sum_{j=1}^{n} \frac{w_j(t)}{\theta_t} = 1$

Now, apply Jensen's inequality to the exponential function:

$$exp(-\eta \sum_{j=1}^{n} \frac{w_j(t)}{\theta_t}\, l_i(x_t, y_t)) \leq \sum_{j=1}^{n} \frac{w_j(t)}{\theta_t}\, exp(-\eta l_j(x_t, y_t)) \tag{29}$$

Applying jensen's inequality to the potential function

Multiply both sides of the inequality by θ_t:

$$\theta_t\, exp(-\eta \sum_{j=1}^{n} \frac{w_j(t)}{\theta_t}\, l_i(x_t, y_t)) \leq \sum_{i=1}^{n} w_j(t)\, exp(-\eta l_j(x_t, y_t)) \tag{30}$$

Notice that the right-hand side of the inequality is θ_{t+1}

$$\theta_t\, exp(-\eta \sum_{j=1}^{n} \frac{w_j(t)}{\theta_t}\, l_j(x_t, y_t)) \leq \theta_{t+1} \tag{31}$$

By rearranging the inequality, we obtain:

$$\theta_{t+1} \leq \theta_t\, exp(-\eta \sum_{j=1}^{n} \frac{w_j(t)}{\theta_t}\, l_j(x_t, y_t)) \tag{32}$$

The concept of the "Best Expert's Loss" refers to the performance of the best-performing expert (or base learner) in hindsight, after observing the entire sequence of data points. In the context of online learning and the Hedge algorithm, it helps to measure the effectiveness of the algorithm by comparing its cumulative loss to that of the best expert.

Lower Bound on Potential Function

On the other hand, by using the best expert's loss, we have:

$$\theta_{T+1} \geq \max_t w_i(1)\, exp(-\eta \sum_{j=1}^{n} \frac{w_j(t)}{\theta_t}\, l_j(x_t, y_t)) \geq \frac{1}{n}\, exp(-\eta \sum_{t=1}^{T} \mathcal{L}_t^*) \quad (33)$$

Combining Inequalities

Combining the upper and lower bounds, we get:

$$exp(-\eta \sum_{j=1}^{n} \mathcal{L}_t \geq \frac{1}{n}\, exp(-\eta \sum_{t=1}^{T} \mathcal{L}_t^*) \quad (34)$$

taking the logarithm on both sides and rearranging terms:

$$-\eta \sum_{j=1}^{n} \mathcal{L}_t \geq \log\left(\frac{1}{n}\right) - \eta \sum_{t=1}^{T} \mathcal{L}_t^* \quad (35)$$

Simplifying, we obtain the bound on the cumulative regret:

$$\sum_{t=1}^{T} \mathcal{L}_t^* - \sum_{j=1}^{n} \mathcal{L}_t \geq \frac{\log n}{\eta} \quad (36)$$

Choosing the optimal learning rate η to minimize this bound, we set:

$$\eta = \sqrt{\frac{\log n}{2T}} \quad (37)$$

Substituting this value of η into the bound, we get:

$$R_T \leq \sqrt{2T \log(n)} \quad (38)$$

The detailed steps show that the cumulative regret R_T for ensemble methods of proposed is bounded by $\sqrt{2T \log(n)}$. This bound indicates that the average regret $\frac{R_T}{T}$ converges to zero as T increases. This demonstrates the ensemble's performance approaches the performance of the best possible model over time. The convergence rate $\sqrt{\frac{\log(n)}{T}}$. quantifies how quickly this convergence occurs.

5.8. Methodology and Experiments

5.8.1. Dataset

To evaluate the proposed online DDoS detection framework, we used two security datasets: CICIoT2023 and IoTID20. In 2023, the Canadian Institute for Cybersecurity developed CICIoT2023 dataset [35]. To generate the attack traffic, 33 attacks are executed on 105 IoT devices as targets. The traffic includes normal and attack traffic from DDoS, DoS, Recon, and Web-based DDoS attack, brute force, spoofing, and Mirai. The dataset contains the most common and current attacks as of this time. However, for this experiment, we use DDoS and normal traffic. CICIoT2023 is a new realistic IoT dataset that was created by generating real IoT device traffic data from both legitimate and malicious IoT devices that include different DDoS attacks.

The IoTID20 dataset has been used for developing DDoS attack detection in several studies [36]. The authors of the IoTID20 dataset [37] performed binary and multiclass classification, and reported the accuracy scored for different classifier methods. DDoS attack types included in the IoTID20 dataset are: Mirai-ACK flooding, Mirai brute force, Mirai-HTTP flooding, and Mirai-UDP flooding. The IoTID20 dataset is a relatively new dataset that considers IoT devices while containing DDoS attacks, and several recent works have used it to develop an IDS [38]. IoTID20 focuses on IoT security and provides a wide range of attack and normal samples from various IoT devices. The following Table 4 shows representative attributes for CICIoT2023 and IoTID20 datasets.

Table 5: Representative attributes used for CICIoT2023 and IoTID20 dataset

Family	CICIoT2023 (examples)	IoTID20 (examples)
Identification / headers	flow_duration, protocol, src_port, dst_port	flow_duration, protocol, src_port, dst_port
Volume	fwd_pkt_cnt, bwd_pkt_cnt, fwd_bytes, bwd_bytes, avg_pkt_len	tot_pkts, tot_bytes, fwd_bytes, bwd_bytes, avg_pkt_len
Timing	fwd_iat_mean/max/min/std, bwd_iat_*, flow_pkts_per_s, flow_bytes_per_s	iat_mean/max/min/std, pkts_per_s, bytes_per_s
TCP flags	counts/indicators for SYN, ACK, FIN, RST, PSH, URG (when applicable)	counts/indicators for SYN, ACK, FIN, RST
Ratios & asymmetry	fwd_bwd_pkt_ratio, fwd_bwd_byte_ratio, header/payload ratios	fwd_bwd_pkt_ratio, fwd_bwd_byte_ratio
Error/retransmission (if present)	fwd_retrans, bwd_retrans, out-of-order flags	—

5.8.2. Performance Metrics

The proposed framework is analyzed from a variety of perspectives to provide a comprehensive view of the experiment. The four-performance metrics, accuracy, precision, recall, and F1-score, are the primary metrics we used to evaluate the performance of the proposed framework. Latency and throughput are also measured to evaluate the learning and detection speed.

In this case, latency refers to the delay in the response time to a specific input. It represents the typical processing time needed by our model to analyze and classify each sample. On the contrary, throughput is a measure of how many data the system can process in a given amount of time. In our context, it corresponds to the number of samples our model can analyze and classify in a given unit of time. Low latency and high throughput are the two essential performance criteria for machine learning and data analytics models. A balance between prediction accuracy and latency should be maintained by efficient learning models in order to achieve real-time analytics.

5.8.3. Experiment Environment

The proposed online DDoS detection framework aims to detect DDoS attacks on IoT systems. To observe the performance of the proposed approach, we developed a prototype using Python 3.10 programming language in the Jupyter Notebook environment. The River library [39] was used for data stream analytics and addressing concept drift through machine learning. To support reproducibility and adversarial robustness evaluation, we used Python 3.10 and explicitly documented all critical dependencies, including River 0.15.0, Scikit-learn 1.2.2, and TensorFlow 2.11. A fixed random seed (42) was used across all experiments to ensure consistency. The experiment was conducted on a machine running Intel(R) Xeon(R) CPU @2.20 GHz and with 16 GB of RAM.

5.9. Result and discussion

The datasets used during the experiments are IOTID20 and CICIoT2023. The attack patterns on both datasets have changed overtime, resulting in three and seven concept drifts on the IOTID20 and CICIoT2023 datasets, respectively. The changes and in particular the occurrence of the concept drifts in the datasets show the dynamic nature of IoT streaming traffic. The concept drifts are shown in Figures 6 and 7 using black arrows. All other concept drifts shown for both the IOTID20 dataset in Figure 6 and CICIoT2023 dataset in Figure 7 are sudden drifts, except for the third and the last drifts shown for the CICIoT2023 dataset in Figure 7. The third and the last drifts shown for the CICIoT2023 dataset in Figure 10 are gradual drifts. The performance of the proposed

model, which is AUWPAE, is compared with that of other state-of-the-art online adaptive learning methods, such as ARF-ADWIN, ARF-DDM, SRPs-ADWIN, SRPs-DDM, KNN-ADWIN, HTs, LB, and PWPAE, using two datasets: IOTID20 and CICIoT2023. The experimental results are presented in Figures 6 and 7, and Tables 5 and 6.

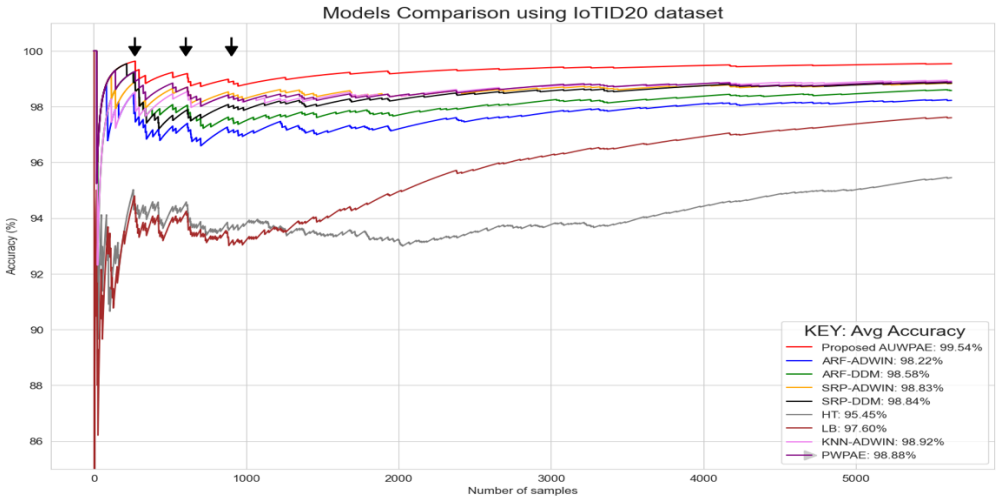

Figure 6: Model comparison using IoTID20 dataset. Arrows indicate concept drifts.

Table 6: Model comparison using IoTID20 dataset.

Models Comparison Using IoTID20 Dataset						
	Accuracy	**Precision**	**Recall**	**F1-Score**	**Latency (ms)**	**Throughput (Sample Per Second)**
ARF-ADWIN	98.22	98.60	99.53	99.06	0.24	4166
ARF-DDM	98.58	98.80	99.7	99.25	0.26	3846
SRPs-ADWIN	98.83	98.94	99.83	99.38	**2.79**	**358**
SRPs-DDM	98.84	98.94	99.85	99.39	**2.90**	**345**
HTs	95.45	95.91	99.42	97.63	0.11	9090
LB	97.60	98.22	99.25	98.72	1.56	641
KNN-ADWIN	99.13	99.10	99.75	99.42	1.15	869
PWPAE	98.88	98.86	99.96	99.41	**2.79**	**358**
AUWPAE	**99.54**	**99.51**	**99.99**	**99.76**	**2.73**	**365**

Figure 6 and Table 5 show performance comparisons of the proposed model, which is AUWPAE, with other previously proposed models, i.e., ARF-ADWIN, ARF-DDM, SRPs-ADWIN, SRPs-DDM, HTs, LB, KNN-ADWIN, and PWPAE, using the IOTID20 dataset. The performance metrics used for comparison were accuracy, precision, recall, F1-score, latency, and

throughput. From the experimental results, in terms of accuracy, the proposed model achieved an average accuracy of 99.54%. This result shows that AUWPAE outperforms the other online adaptive learning methods. The reason why the proposed model performed better than the others is mainly attributed to the weighting algorithm used for ensemble. The two selected base models, SRPs-DDM and KNN-ADWIN, achieved better accuracy. The accuracies of SRPs-DDM and KNN-ADWIN are 98.84% and 99.13%, respectively, while using the IoTID20 dataset. This is because the proposed model is an extension of the AUE, and the weighting of classifiers is performed using non-linear error function, which contributes to performance enhancement. In terms of precision, recall, and the F1-score, the experimental results show that the proposed model achieved 99.51%, 99.99%, and 99.76%, respectively. These results show that the proposed approach has relatively accurate and precise DDoS attack detection capability than the other solutions and is robust to concept drifts.

In terms of latency, the proposed model uses an average of 2.73 ms to classify a given sample as attack and normal while using IoTID20 dataset (see Figure 6 and Table 5). The other online adaptive models HTs, ARF-ADWIN, ARF-DDM, KNN-ADWIN, and LB performed better than the proposed approach, i.e., AUWPAE, in terms of processing time. AUWPAE is better than only PWPAE, SRPs-ADWIN and SRPs-DDM, respectively. The reason why the proposed model took more time to process and give results is mainly attributed to the ensemble algorithm. However, on the proposed ensemble model, ARF-ADWIN and ARP-ADWIN have positive contribution on latency. Since, the two models ARF-ADWIN and ARF-DDM achieved excellent latency compared to the other models which are 0.24 ms and 0.26 ms using the IoTID20 dataset, respectively. ARF-ADWIN is the second fastest approaches in the experiment while KNN-ADWIN is the third.

For this experimental setup, the last performance metrics used were throughput. The experimental result shows that the average throughput achieved by the proposed model (AUWPAE) is 365 samples/second. This result would mean that AUWPAE is better than PWPAE, SRPs-DDM, and SRPs-ADWIN. ARF-DDM, LB, HTs, and KNN-ADWIN performed better than the proposed model. The relatively high throughput values inversely correlate with the latency of the methods. Their accuracy and F-score values, however, is low when compared to the proposed approach. This shows that even though the proposed system is slower to classify data samples in microseconds, it is able to detect DDoS attacks better than the other methods in the presence of concept drift.

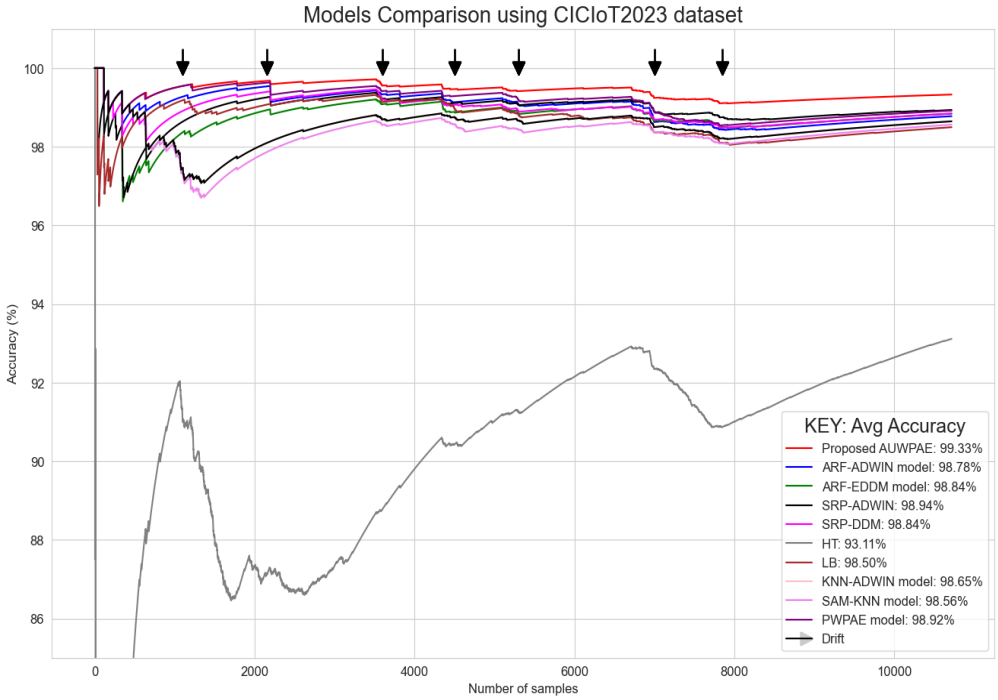

Figure 7: Models comparison using CICIDS2017 dataset. Arrows indicate concept drifts.

Figure 7 and Table 6 show the experimental results of the proposed model and the other methods considered in this research while using CICIoT2023 dataset. In terms of accuracy, the proposed model, i.e., AUWPAE, has achieved 99.33% accuracy, which is the highest when compared to the other methods in this experiment. Moreover, in terms of precision, recall, and F1-score performance metrics, AUWPAE has achieved 98.88%, 96.53%, and 97.98%, respectively, which are better than the other methods. These results indicate that the proposed approach is able to detect DDoS attacks more accurately than the other methods in the presence of concept drifts.

Table 7: Model comparison using CICIoT2023 dataset.

Models Comparison Using CICIoT2023 Dataset						
	Accuracy	**Precision**	**Recall**	**F1-Score**	**Latency (ms)**	**Throughput (Sample Per Second)**
ARF-ADWIN	98.78	97.37	94.58	95.95	0.26	3723
ARF-EDDM	98.83	98.16	94.21	96.15	**1.29**	**773**
SRPs-ADWIN	98.94	97.75	95.25	96.48	**2.32**	**430**
SRPs-DDM	98.83	97.03	95.37	96.19	**2.41**	**413**
HTs	93.11	75.89	80.69	78.22	0.25	3997
LB	98.50	95.44	95.81	95.62	0.12	7692
KNN-ADWIN	98.65	94.53	96.77	95.64	0.13	7182
PWPAE	98.92	97.75	95.13	96.14	**1.38**	**721**
AUWPAE	**99.33**	**99.88**	**96.53**	**97.78**	**1.29**	**773**

The latency of AUWPAE is 1.29 ms. AUWPAE is relatively slow when it is compared with HTs, ARF-ADWIN, KNN-ADWIN, and LB, and fast when compared with PWPAE, SRPs-DDM, and SRPs-ADWIN. LB is the fastest method but has low accuracy and a low F1-score when compared with AUWPAE. The higher latency value of AUWPAE is mainly attributed to the ensemble algorithm. However, this has given it the advantage to detect DDoS attacks accurately. This result is also consistent with the IOTID20 dataset.

The throughput of the proposed model, the AUWPAE, achieved 773 samples/second, making it better than ARF-EDDM, SRPs-ADWIN, SRPs-DDM, and PWPAE. However, it performed worse than ARF-ADWIN, HTs, and KNN-ADWIN did.

The AUWPAE showed better accuracy and F1-score values for both datasets while taking relatively more time (in ms) to process the data. This indicates that AUWPAE is able to correctly classify normal and DDoS attacks accurately while there are three or more concept drifts in a short period of time. In real-world contexts, however, concept drifts would happen less frequently. Hence, the AUWPAE is expected to perform better in a real-world scenario. Even though AUWPAE is relatively slow, we believe that the accuracy of the classification plays an important role in making a decision to protect an IoT system. Moreover, the latency of AUWPAE is less than the latency requirement of IoT production applications [40]. The target application areas are cyber-physical systems, smart grid monitoring, and traffic control in smart cities. The current 2.73 ms/sample latency performance for AUWPAE is comfortably within acceptable values dictated by NIST SP 800-183 and relevant research on real-time IoT infrastructure.

5.10. Summery

IoT DDoS attack detection solutions are usually developed to protect IoT systems from DDoS attacks using IoT data stream analytics. However, IoT data are usually dynamic and could have concept drifts. This paper provides a novel Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) approach to detect concept drift and perform zero-day DDoS detection. We evaluated the proposed model using the IoTID20 and CICIoT2023 datasets with benign and DDoS traffic data that had concept drifts. The results show that AUWPAE achieved better accuracies of 99.54% and 99.33% for the respective datasets when compared with those of the other eight models. This result indicates that the proposed adaptive online DDoS attack detection framework, which uses AUWPAE is able to detect DDoS attacks in the presence of concept drifts. In this

paper, we also presented the IoT DDoS attack detection solution deployment framework for IoT systems.

Chapter Six

6. Multi-Stage Adversarial Defense for online DDoS attack detection system in IoT

Online learning systems become increasingly vulnerable to adversarial attacks that subtly manipulate input data to degrade model performance. Most of the current adversarial defense mechanisms are batch learning and trained on known attack patterns, and fail to generalize to unseen adversarial attacks. This chapter proposes the Multistage Adversarial Attack Defense (MSAAD) framework, designed to improve the robustness of online detection systems against both known and unknown adversarial attacks. Addressing Research Question 2 (RQ2) and fulfils Objective OB2. MSAAD integrates a Resilient Adversarial Detector and Purification (RADP) module, multiple base classifiers, and an adaptive ensemble based on Multi-Armed Bandits with Thompson Sampling (MLBTSE). This layered defense ensures sustained model accuracy even under adversarial conditions, enhancing the trustworthiness and resilience of IoT DDoS detection systems

The proposed framework is a comprehensive solution designed to protect online DDoS attacks detection systems against adversarial attacks in IoT environments. It integrates multiple components to achieve a robust online DDoS detection system. The framework consists two main components: pre-processing and the Multi-Stage Adversarial Attack Defense (MSAAD) framework. The MSAAD framework is designed to protect IoT systems from DDoS attacks through a combination of Resilient Adversarial Detection and Purification (RADP), multiple classifiers, and ensemble defense strategies. These components interact seamlessly to ensure real-time, robust performance against multiple and unknown adversarial attacks.

1. Pre-Processing

Pre-processing ensures clean, well-represented, and balanced IoT data streams for optimal model DDoS attack detection accuracy. This step includes encoding, cleaning, sampling, balancing, normalization, and feature selection of data. These techniques prepare IoT data streams for online DDoS attack detection.

2. Multi-Stage Adversarial Defense (MSAAD)

The proposed MSAAD composed of three layers of defense:

1) RADP (First Defense Layer)

The Resilient Adversarial Detection and Purification (RADP) designed to detects and purifies adversarial attacks against online DDoS attack detection systems addressing from multiple and unknown adversarial attacks. The RADP consists two main components: Resilient Adversarial Detector (RAD) and purification modules. The RAD module detects adversarial attacks by leveraging the strengths of Generative Adversarial Networks (GANs) and multisource adversarial perturbations (MSAP). A GAN are employed to reduce the gap between the training data and the real-world data. MSAP crafts a variety of adversarial attack methods to create diverse perturbations to train adversarial attack detectors. The purification module is used to purify adversarial perturbations to enhance the robustness of online learning models by systematically removing the detected data points using RAD.

2) Multiple Classifiers (Second Defense Layer)
   This layer consists of multiple classifiers designed to defend against adversarial attacks. Utilizing multiple classifiers significantly makes it harder for attackers to exploit the system. Each classifier is optimized for different traffic patterns, ensuring high accuracy and robustness against sophisticated attack strategies.
3) Ensemble Defense (Third Defense Layer)
   The third layer of defense is the Multi-Armed Bandit with Thompson Sampling (MABTSE), which operates by balancing exploration (trying different online learning models) and exploitation (using the best-known online learning model) to dynamically select an optimal classifier or ensemble for each data point. For the ensemble, we employed the Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) approach to combine the base learners (classifiers) for IoT data stream analytics [9].

6.1. Data pre-processing

Data pre-processing is essential to ensure that incoming data are clean, well-represented, and balanced for optimal model performance. The performance of the DDoS attack detection method can be significantly affected by the representation, size, and quality of incoming data. In particular, selecting a dataset with high dimensionality and a large number of duplicate and irrelevant features can adversely impact the training process. To address these issues, we employed data cleaning, encoding, normalization, and feature selection techniques.

a) Encoding

IoT data values were generated as words and strings to enhance readability. Data encoding involves converting string information into numerical features that can be understood and processed by online learning models. One-hot encoding and target encoding are examples of frequently used data encoding methods. Hot encoding is a common method used to represent each category as a binary vector. However, in an online environment, incremental one-hot encoding is used to dynamically add new categories to the encoding matrix as data arrive. In this study, we utilized incremental one-hot encoding to handle the dynamic nature of the IoT data.

b) Data Cleaning

IoT streams often experience missing values due to data accessibility issues. These gaps are typically addressed using incremental imputation techniques, such as mean or median imputation or last observation carried forward (LOCF), as data arrives. In an IoT environment, data distribution can change over time. Therefore, the cleaning process must be adaptive, and its parameters should be continuously updated to reflect the current state of the incoming data stream. In this study, we employed the missing values using mean or median imputation techniques.

c) Sampling

The selection of highly representative data samples necessitates an efficient data-sampling method. In online learning, sampling is conducted using incremental K-means and minibatch K-means. In incremental K-means clustering, data points are assigned to the closest cluster centroid. The centroid is updated incrementally when new data arrives. This approach enables the online DDoS detection model to dynamically maintain its cluster, without requiring access to the entire dataset. Mini-batch K-means is other method used to process small batches of data simultaneously. This makes it more scalable and efficient for an online DDoS attack detection model. The proposed framework employs a mini-batch K-mean-based cluster sampling method to obtain highly representative data.

d) Data Balancing

The sampling method is effective in handling class imbalances in machine learning. The Synthetic Minority Oversampling Technique (SMOTE) generates synthetic samples by adding existing minority class samples to create a balanced dataset. We use the online SMOTE sampling method to generate synthetic samples when new minority

class instances arrive. This helps to prevent our framework from becoming biased towards the majority class. This leads to improved performance of the proposed framework.

e) Data normalisation

Data normalization is a key pre-processing step in online learning to ensure that the feature values are within the same range. In online learning, data arrive incrementally; therefore, normalization must be performed dynamically. Two common normalization methods used on data analytics are the Z-score and min-max normalization. The min-max normalization approach transforms the original data linearly to achieve a balance of comparative values between the data before and after processing. The min-max normalization method scales the values of a feature to a range between 0 and 1. The Min-Max scaling approach is shown in Equation 39. [115]

$$X_{new} = \frac{X - Min(X)}{\mathrm{Max}(X) - Min(X)} \tag{39}$$

Where X_{new} is the result of the normalization, X is the value to be normalized, and Min and Max are the minimum and maximum values of each feature, respectively.

The Z-score normalization method was based on the mean and standard deviation of the data. This method is useful if the minimum and maximum data values are unknown. Equation 40 shows the formula for Z-score normalization [116].

$$X_{new} = \frac{X - \mu}{\delta} \tag{40}$$

where $X_{new}$is the result of normalization, X is the value to be normalized, μ is the population mean, and σ is the standard deviation. min-max normalization retains outliers in the datasets, making it appropriate for DDoS attack detection models. Therefore, the proposed framework uses min-max normalization to address this requirement.

f) Feature selection

Feature selection is employed to improve the online learning model performance by reducing noise and focusing on the most relevant features. The optimal minimal feature set can be determined by evaluating all possible feature combinations. However, when dealing with the large number of features, optimization approaches can be utilized to explore the feature search space and determine the optimal feature set. Information

Gain (IG) and Pearson Correlation methods were utilized to remove irrelevant and duplicate features, respectively. Information Gain (IG) is used to remove irrelevant features by measuring the amount of information that contributes to predicting the target variable. Pearson Correlation helps identify and remove duplicate features by evaluating the linear relationships between pairs of features. Consequently, the proposed framework selects both Information Gain and Pearson Correlation to identify the best-performing minimal feature set.

6.2. Multi-Stage Adversarial Attack Defence (MSAAD) Framework

The proposed MSAAD framework operates through three key stages: Resilient Adversarial Detection and Purification (RADP), multiple classifiers, and ensemble defense. The RADP defense method detects and purifies adversarial attacks against an online DDoS attack-detection system, addressing multiple and unknown adversarial attacks. RADP consists of Resilient Adversarial Detection (RAD) and purification modules. RAD is trained using a generative adversarial network (GAN), that mimic crafted adversarial samples and multi-source adversarial perturbation generator (MSAPG) that generates adversarial perturbation, as shown in Figure 8. The GAN leverage its strengths to reduce the gap between the training data and real-world data. This makes the online DDoS detector more aware of multiple and unknown adversarial attacks. MSAPG generates various adversarial attack methods to create diverse perturbations to train an adversarial attack detector. It includes adversarial training by crafting perturbations using PGD, FGSM, C&C, and BIM methods. This ensures that the adversarial detector is trained on diverse adversarial perturbations to make it more robust and resilient to a broad spectrum of attacks. By combining the GAN and MSAPG submodules, the proposed MSAAD framework significantly enhances the detection capability of the RAD. The Pruning method is used to purify adversarial perturbations to enhance the robustness of online learning models by systematically removing detected data points using RAD. MSAPG is utilized for creating well-known adversarial attacks with different perturbation budgets (ϵ) for enabling robustness to different types of attacks, and the GAN sub-module is utilized for emulating unknown, zero-day attacks for enabling generalization against unknown attacks. The multi-level training procedure enhances RADP's capacity to detect and purify adversarial perturbations before they can affect downstream processing.

The purification module employs a pruning method to remove detected adversarial perturbations. However, to prevent the model from losing valuable information or becoming over-sensitive, the purification process is governed by a threshold-based stopping criterion. The core

idea is to not simply remove any sample flagged by the detector but to quantify the degree of perturbation. We compute a confidence score for the RAD module's detection. Samples with a confidence score above a predefined threshold τ are only considered for purification. Furthermore, for samples near the decision boundary, a distance-to-boundary metric is used. The purification 'walk back' is halted if the sample's distance to the original model's decision boundary exceeds a safe margin δ, ensuring that clearly legitimate samples are not altered. This dual-threshold approach: detection confidence (τ) and boundary distance (δ) ensures the purification selective and conservative. This approach focuses on high-confidence adversarial samples and avoids excessive purification that could otherwise remove benign or novel-but-legitimate traffic, thus maintaining a balance between robustness and generalizability. The details of the MSAPG framework are shown in Figure 9.

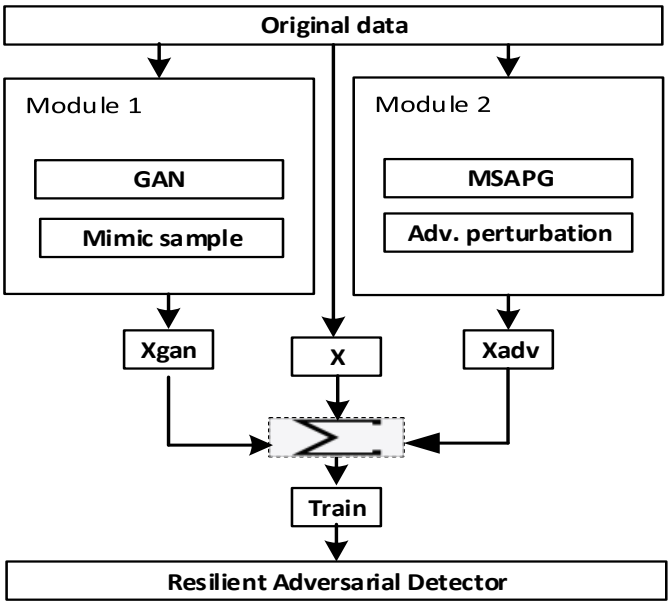

Figure 8: Training Resilient Adversarial Detection model through MSAPG and GAN

The second stage of MSAAD employs a multiple-classifier approach to further protect the DDoS attack detection system against adversarial attacks. The concept behind utilizing multiple classifiers as a defense instead of using a single classifier is to increase the complexity of attackers when attempting to replicate the DDoS attack detection model. If an attacker manages one classifier to bypass it, it is difficult to fool and bypass multiple classifiers. An attacker cannot predict the classifier responsible for classifying a given stream of traffic. This is because they

cannot predict the specific model responsible for classifying a given request. This enhances the overall resilience of DDoS attack detection systems to sophisticated adversarial attacks.

The third stage of MSAAD employs the Multi-Armed Bandits (MAB) algorithm with Thompson Sampling (MABTSE) to dynamically select the optimal classifier or ensemble of classifiers for each incoming traffic request. For the ensemble, the Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) approach is employed to combine the base learners (classifiers) for IoT data stream analytics. This allows for seamless integration and enhances adaptability. It maintains the performance of conventional DDoS attack detection systems in non-adversarial attack scenarios by selecting the most appropriate classifiers for each request type. The MAB algorithm was specifically designed to account for the diversity of classifiers within the system. For example, if the SRP-ADWIN classifier demonstrates a high probability of being accurate, whereas the SRP-DDM and KNN-ADWIN classifiers have lower probabilities, the MAB algorithm will prioritize the SRP-ADWIN for that particular request. This process optimizes the selection of classifiers and enhances the overall accuracy of the DDoS attack-detection system. Moreover, the MAB algorithm continually updates the probability estimates for each classifier based on their recent performance, and allows the system to adjust traffic patterns and stay up-to-date with emerging types of attacks.

The MAB algorithm with Thompson Sampling dynamically selects an optimal classifier or an ensemble of classifiers for each incoming traffic request. The DDoS attack detection system balances exploration and exploitation by selecting classifiers that have achieved recent success but are adaptable to multiple and unknown types of attacks. In some cases, multiple classifiers exhibit similarly high probabilities of success. To handle this situation, an ensemble method was applied to combine the outputs of the classifiers. In this study, we used the Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) approach to combine the base learners for IoT data-stream analytics. Details of this algorithm are presented in Algorithm 1. AUWPAE provides dynamic weights to base learners in real time. The AUWPAE method uses a weighted average approach. The prediction probability of each base learner is assigned a weight and multiplied by the prediction probability of that base learner. The multiplication results are summed to determine the final prediction probability. AUWPAE mitigates the limitations of other online ensemble models. It addresses sudden and gradual drift issues and enhances the DDoS attack detection. ARF-ADWIN, ARF-DDM, SRP-ADWIN, SRP-DDM, and KNN-ADWIN were used for ensemble-based learners.

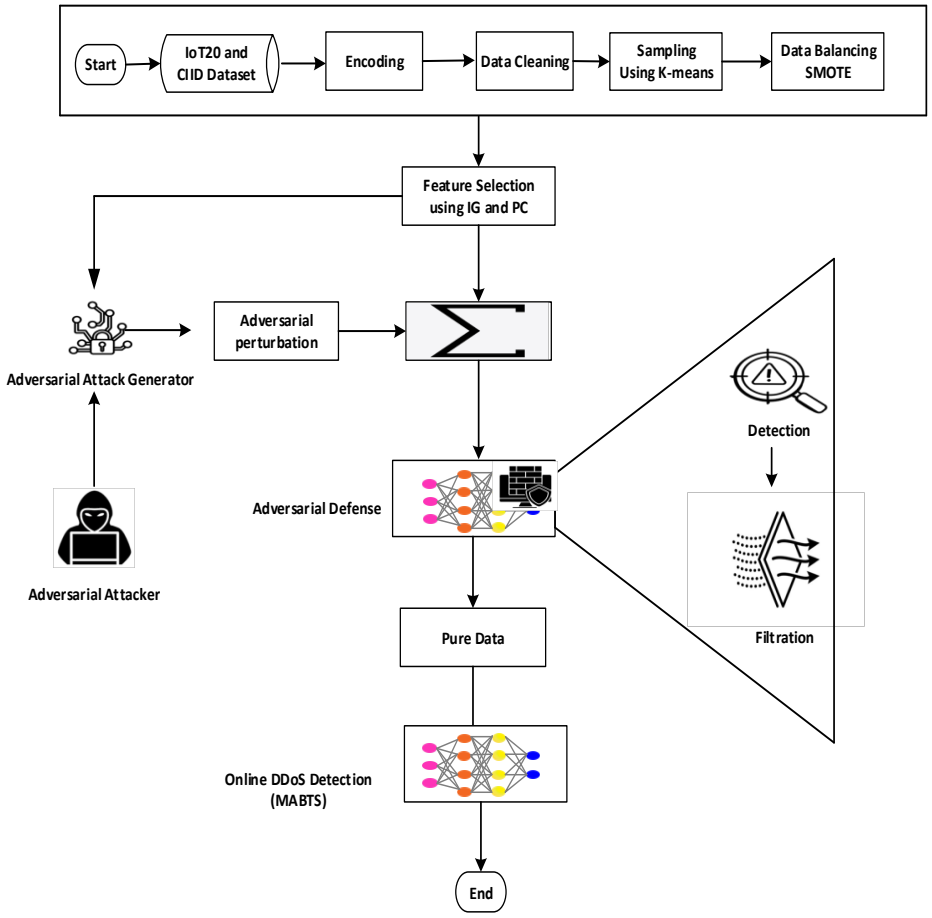

Figure 9: Overview of MSAAD framework

Algorithm 1, MAB with Thompson Sampling and AUWPAE (MBATSE)

Input: *X_train, y_train, X_test, y_test*

Output: *t, m (time steps and accuracy metrics)*

function *TS-WPAE (X_train, y_train, X_test, y_test)*

Initialize a, t, e, eps=0.0001
Initialize base learners $j = 5$
Thompson Sampling parameters α*=1,* β*=1*
for each *instance* (x_i , y_i) ***in*** X_train*, y_train* ***do****. // learn base learners using training set*
    $L_j \leftarrow (x_i, y_i)$
end for
for each *instance* (x_i , y_i) ***in*** X_test*,* ***in*** y_test ***do****. // predict using test set*
    $\theta_j \leftarrow sample(\alpha_j , \beta_j)$
    $theta\_value\_top\_three\_baselearner(\theta_j)$
    $y\_pred_j \leftarrow bl\ (x_i)$ *//predict target*
    $y\_prob_j \leftarrow blProbPred\ (x_i)$ *//target prediction probability*
    $bl.update(x_i, y_i)$ *//update base leaner.*
end for

15 **for** *j = 1 to 3* **do**
16 $e_i \leftarrow MSE_i(y_i, y\_pred_i)$ *// calculate real-time MSE for each base learner*
17 $W \leftarrow 1/(e_j + eps)$ *//calculate weight*
18 $y\_prob_{j0} \leftarrow y\_prob_j$ *// probability for class 0*
19 $y\_prob_{j1} \leftarrow y\_prob_j$ *//probability for class 1*
20 $y\_prob_0 \leftarrow y\_prob_j \sum_{j=1}^{4} W_j * yprob_{j0}$ *// average weighted probability for class 0*
21 $y\_prob_1 \leftarrow y\_prob_j \sum_{j=1}^{4} W_j * yprob_{j1}$ *// average weighted probability for class 0*
22 **end for**
23 **If** $y\_prob_0 > y\_prob_1$
24 . $y\_pred \leftarrow 0$
25 **else** $y\_prob_0 < y\_prob_1$
26 $y\_pred \leftarrow 1$
27 *calculate accuracy, precision, recall, and F1-score;*
28 **return** *t, a*
29 **end function**

6.3. Experiments

6.3.1. Dataset

To evaluate the proposed online DDoS detection framework, we used two security datasets: CICIoT2023 and IoTID20. In 2023, the Canadian Institute for Cybersecurity developed CICIoT2023 dataset [35]. To generate the attack traffic, 33 attacks are executed on 105 IoT devices as targets. The traffic includes normal and attack traffic from DDoS, DoS, Recon, and Web-based DDoS attack, brute force, spoofing, and Mirai. The dataset contains the most common and current attacks as of this time. However, for this experiment, we use DDoS and normal traffic. CICIoT2023 is a new realistic IoT dataset that was created by generating real IoT device traffic data from both legitimate and malicious IoT devices that include different DDoS attacks.

The IoTID20 dataset has been used for developing DDoS attack detection in several studies [36]. The authors of the IoTID20 dataset [37] performed binary and multiclass classification, and reported the accuracy scored for different classifier methods. DDoS attack types included in the IoTID20 dataset are: Mirai-ACK flooding, Mirai brute force, Mirai-HTTP flooding, and Mirai-UDP flooding. The IoTID20 dataset is a relatively new dataset that considers IoT devices while containing DDoS attacks, and several recent works have used it to develop an IDS [38]. IoTID20 focuses on IoT security and provides a wide range of attack and normal samples from various IoT devices.

6.3.2. Experiment Environment

The proposed online DDoS detection framework is designed to detect DDoS attacks targeting IoT systems. To evaluate the performance of the proposed approach, we developed a prototype using the Python 3.10 within the Jupyter Notebook environment. The River library [117] is utilized for data stream analytics, and addresses concept drift through machine learning. The Adversarial Robustness Toolbox (ART) [118] employed to simulate adversarial attacks. The experiment was conducted on a machine running equipped with Intel(R) Xeon(R) CPU @2.20GHz with 16 GB of RAM.

6.3.3. Experiment setup

In this experiment, we utilized state-of-the-art online adaptive learning models as baseline to evaluate the performance of the proposed MABTSE model. ARF-ADWIN, ARF-DDM, SRPs-ADWIN, SRPs-DDM, KNN-ADWIN, HTs, LB, and PWPAE are the baseline models. We utilized four common types of adversarial attacks including PGD, FGSM, C&C, and BIM to generate adversarial perturbations to evaluate MABTSE framework. We have done hyperparameter tuning carefully to optimize the performance of the proposed online learning model. The ARF tuned with 3 trees and utilized ADWIN and DDM drift detectors to handle concept drift effectively. The KNN model utilizes 5 neighbours and ADWIN concept drift detector using windows size of 100 samples. The SRP and LB ensembles were configured with three base learners. The OPA algorithm was set with an aggressiveness parameter of 1.0. The SAM-KNN and EFDT were fine-tuned to handle non-stationary environments by leverage incremental updates. The generation of adversarial attack perturbation conducted using default parameters, specifically epsilon values of 0.1 to simulate realistic attack scenarios. These hyperparameter configurations were applied in the experiment to achieve optimal performance multiple and adversarial scenarios.

We evaluate the framework against adversarial attacks using three scenarios. In the first scenario, we evaluated the proposed framework as a baseline without adversarial attacks or the RADP defense method. In the second scenario, we conduct adversarial attacks using FGSM, BIG, PGD, and C&W, against the MBATSE framework. The third scenario was designed to evaluate the proposed MSAAD framework, which is mainly utilizes RADP, to detect and purify detected adversarial attacks. This demonstrate the necessity and effectiveness of this defense framework in

mitigating adversarial attacks and maintaining the reliability of online DDoS attack detection in IoT systems. The following procedures were used to evaluate the proposed defense framework:

a) First, we evaluate the proposed online DDoS detection model, MABTSE, using the original dataset. This serves as the baseline without adversarial attack.
b) Generate adversarial perturbation using Generative Adversarial Networks (GANs), and multisource adversarial perturbation generator (MSAPG) submodules and combined with the original training dataset. The method simulates a disrupted input dataset for adversarial attacks.
c) The performance of the online DDoS attack-detection model for adversarial samples generated in Procedure 2 was evaluated. This demonstrates the performance of the model under adversarial attack.
d) Develop an adversarial sample detection model utilizing the same online model. This model is trained using a combination of the original training dataset and adversarial samples to effectively distinguish between adversarial samples and the original dataset.
e) The detected adversarial perturbation was purified from the disrupted input dataset by filtering the detected adversarial perturbations.
f) Evaluate the performance of the model to demonstrate the effectiveness of MSAAD framework.

The performance of the proposed online DDoS detection model (MABTSE) and MSAAD defense framework was evaluated by comparing their accuracy, precision, recall, and F1-score, learning time, latency, CI and p-value, as described in Section 5. The evaluation utilized CICIoT2023 and IoTID20 datasets. The evaluation results of the online DDoS attack-detection model are presented in three stages for comparison: original, under attack, and after recovery. These stages are described in detail in Section 4.2, procedure 1,3 and 6. The experiment began with the evaluation of online DDoS attack detection with the original dataset. Following this, the online DDoS attack detection model was evaluated under PGD, FGSM, C&C, and BIM attacks. Subsequently, the Resilient Adversarial Detection and Purification (RADP) method was utilized to detect and purify the detected adversarial attack. Finally, we retrained and evaluated the performance of the online DDoS attack-detection model after recovery. The effects of adversarial attacks against online DDoS attack detection, utilizing the original datasets, during and after the defense mechanism are presented in Table 6 and Figure 11 and 12 based on IoTID20 and CICIoT2023 datasets.

A critical consideration when generating adversarial perturbation for DDoS attack detection is the preservation of the malicious payload. Unlike image classification, where any perturbation can alter the label, a perturbed DDoS attack sample must remain functionally a DDoS attack at the network protocol level to be a realistic threat. The adversarial perturbation should evade the DDoS attack detector without changing the fundamental nature of the traffic from malicious to benign.

Therefore, when employing perturbation methods (FGSM, BIM, PGD, and C&W) to the IoTID20 and CICIoT2023 datasets, semantic integrity of the DDoS attack should maintain. This involved identifying and protecting features that are fundamental to the attack's definition:

- Protocol and port features: features such as protocol, src_port, and dst_port was kept immutable. For example, a Mirai UDP Flood attack in the IoTID20 dataset must remain a UDP packet directed at a victim's port. Changing the protocol to TCP or the destination port to a non-targeted service would fundamentally alter the attack.
- TCP flag indicators: in attacks from the CICIoT2023 dataset that exploit specific flag patterns (e.g., SYN floods), the raw TCP flag counts (for example: SYN_flag_count, ACK_flag_count) were considered critical and were excluded from perturbation to preserve the flood's character.
- Attack-specific volume baselines: while volume-based features such as fwd_pkt_cnt, tot_pkts, and tot_bytes are prime targets for evasion, their perturbations were bounded. We ensured that the perturbed values did not fall below a threshold that would be realistic for a high-rate DDoS flow, thus preserving the "flooding" nature of the attack.

In contrast, features more related to the statistical distribution of the traffic over time such as flow_duration, fwd_iat_mean/max/min/std, bwd_iat, and avg_pkt_len, were considered more permissible for perturbation. Since, attacker could manipulate timing and sizings without negating the attack's malicious intent. This approach ensures that the adversarial examples are not just statistical outliers but represent functional DDoS traffic that has been subtly altered to evade our specific DDoS attack detector.

6.3.4. Performance metrics

The proposed MSAAD framework was analysed from various perspectives to provide a holistic view of the experiment. Since the proposed framework utilized an online DDoS detection model,

it reduces memory requirements for data storage and learn new data patterns by processing one data sample at a time. The framework's performance was evaluated using the accuracy, precision, recall, and F1-score metrics. These metrics are commonly employed in machine learning research to provide a well-rounded view of model performance. Due to the inherent imbalance in intrusion detection datasets, relying solely on individual performance metrics such as accuracy, precision, or recall can lead to limited understanding of model performance. Therefore, we considered the accuracy, precision, recall, and F1-scores collectively to ensure a comprehensive comparison of the machine learning models and to avoid skewed evaluation results. Additionally, we include learning time, latency, confidence interval and p-value for the accuracy metrics.

The learning time and latency are utilized to evaluate the proposed framework in real-time scenario. The learning time of the proposed online model, along with others, are analyzed to compare the learning efficiency. This analysis provides insight into the processing time and efficiency requirements of network systems. latency refers the delay in the response time to a specific input stream. It is defined as the processing time required our model to analyze and classify each sample. Low latency an essential performance requirement for online learning to analyze streaming data. Efficient learning model should balance between prediction accuracy and latency to achieve real-time analytics. In the context of network DDoS attack detection and prevention solutions, it is crucial to strike a balance between effectiveness and efficiency should to ensure optimal performance while minimizing computational overhead.

6.4. Result and discussion

Adversarial attacks exploit the vulnerabilities of online learning models by generating adversarial perturbations that can deceive or manipulate models to make incorrect predictions [118]. Such attacks pose a significant threat to IoT systems and their reliabilities. This experiment aims to demonstrate the detrimental effect of adversarial attacks on the proposed online learning model. Additionally, the proposed defense strategies designed to mitigate these attacks and enhance the performance of DDoS attack detection in IoT systems.

6.4.1. MABTSE model using original dataset

The first result demonstrates the performance of online DDoS attack detection model, known as MABTSE, utilizing the original dataset. This model serves as the baseline for comparison. The performance of the proposed online DDoS attack detection model was compared with other state-

of-the-art online adaptive learning methods, including ARF-ADWIN, ARF-DDM, KNN-ADWIN, SRP-DDM, SRP-ADWIN, HT, and PWPAE. The experiment employed IOTID20 and CICIoT2023 datasets. The attack patterns within both datasets evolved over time, leading to concept drifts of three and seven on the IOTID20 and CICIoT2023 datasets, respectively. These concept drifts are illustrated in Figures 10 and 11 indicated by black arrows. The experimental results are presented in Figures 10 and 11, and Tables 7 and 8.

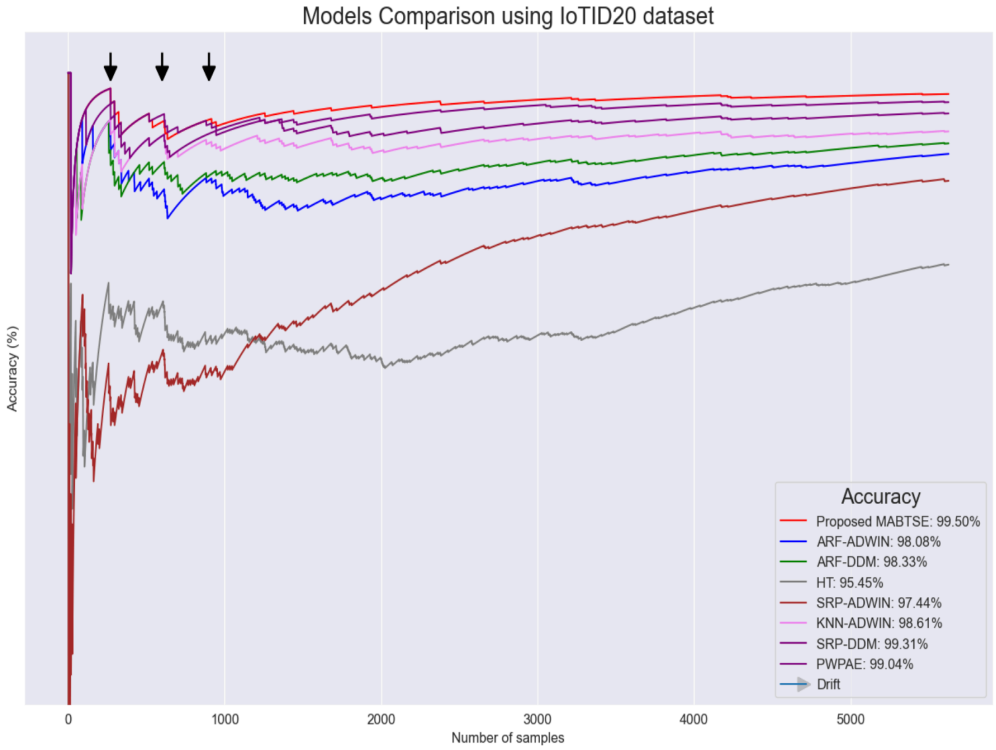

Figure 10: Model comparison using IoTID20 dataset.

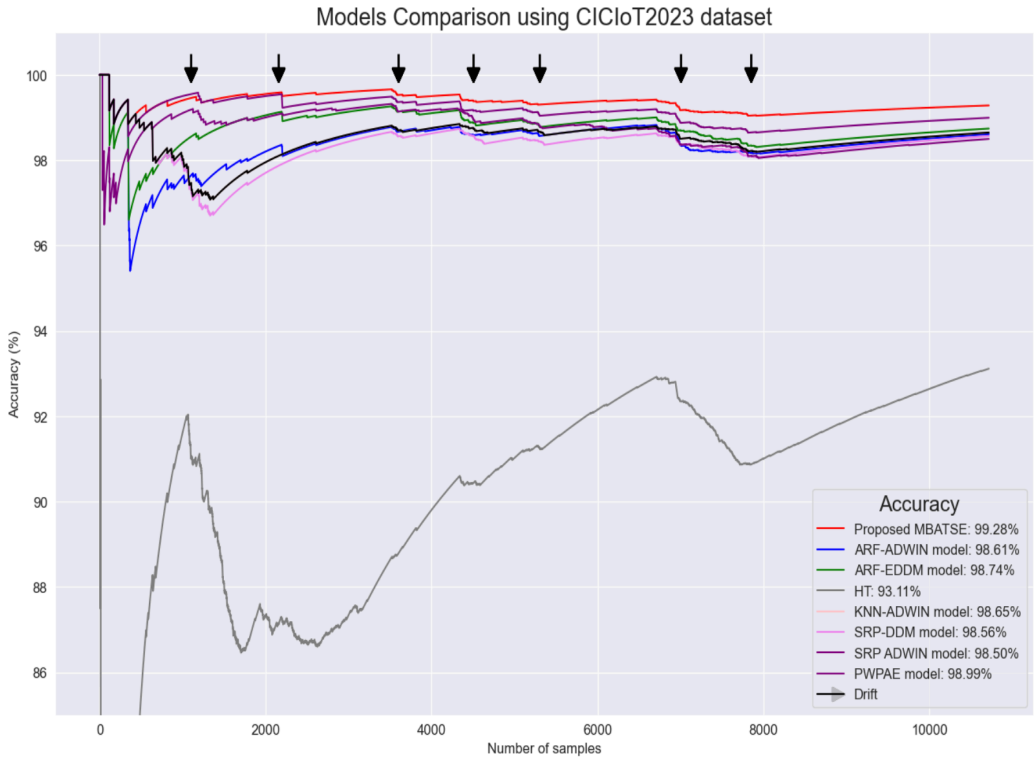

Figure 11: Models comparison using CICIDS2017 dataset.

Table 8: Proposed online DDoS attack detection model comparison using the IoTID20

	Models comparison using IoTID20 dataset					
	A	P	R	F1	L	T
ARF-ADWIN	98.08	98.38	99.06	98.99	0.23	4348
ARF-DDM	98.33	98.63	99.03	99.51	0.28	3571
SRP-ADWIN	97.44	98.00	99.30	98.65	**2.85**	**351**
SRP-DDM	99.31	99.40	99.87	99.63	**2.94**	**440**
HT	95.45	95.91	99.42	97.63	0.11	9091
KNN-ADWIN	98.61	98.83	99.70	99.27	1.24	806
PWPAE	99.04	99.05	99.94	99.63	**2.80**	**357**
MABTSE	**99.50**	**99.47**	**100**	**99.74**	**2.75**	**364**

Table 9: Proposed online DDoS attack detection model comparison using CICIoT2023 dataset

	Models comparison using CICIoT2023 dataset					
	A	P	R	F1	L	T
ARF-ADWIN	98.60	97.58	93.32	95.36	0.25	4001
ARF-DDM	98.74	97.30	94.39	95.83	1.20	833

SRP-ADWIN	98.50	94.80	95.43	95.11	**2.75**	**364**
SRP-DDM	98.56	93.97	96.83	95.38	**2.41**	**415**
HT	97.60	98.22	99.25	98.72	0.19	5263
KNN-ADWIN	98.65	94.53	96.77	95.64	0.16	6250
PWPAE	98.99	97.94	95.43	96.67	**1.45**	**690**
MABTSE	**99.28**	**99.11**	**99.10**	**99.76**	**1.39**	**719**

Legend: A=Accuracy, P= Precision, R= Recall, F= F1-score, T= Model learning time, L= throughput

Figure 10 and Table 7 show the performance comparison of the proposed MABTSE model with other online adaptive models, including ARF-ADWIN, ARF-DDM, SRP-ADWIN, SRP-DDM, HT, KNN-ADWIN, and PWPAE, utilizing the IOTID20 dataset. The performance metrics employed for this comparison were accuracy, precision, recall, F1-score, and learning time, latency, CI and P-value. The experimental results indicate that the proposed model achieved an average accuracy of 99.50%. This result demonstrates that MABTSE outperforms other online adaptive learning methods. The reason why the proposed model outperforms the others is mainly attributed to the Multi-Armed Bandits (MAB) algorithm with the Thompson Sampling method. It selects the dynamically optimal model and the weighting algorithm used for the ensemble. The proposed MABTSE model considers SRP-DDM and KNN-ADWIN as the base models because they achieve better accuracy. The accuracies of the SRP-DDM and KNN-ADWIN were 99.31% and 98.61%, respectively. In terms of precision, recall, and F1 score, the experimental results demonstrate that the proposed model achieved 99.47%, 100%, and 99.74%, respectively. These results demonstrate that the proposed approach has a relatively accurate and precise DDoS attack detection capability compared to other solutions and is robust to concept drifts. In terms of model training time, the proposed model averaged 2.75 millisecond to train the learning model (Table 4). In comparison, the other online adaptive models HT, ARF-ADWIN, ARF-DDM, KNN-ADWIN, and LB, required less training time than the proposed MABTSE approach. The MABTSE outperformed PWPAE, SRP-ADWIN, and SRP-DDM. This is because MABTSE's ability to dynamically selects the optimal classifier or ensemble of classifiers for each incoming traffic request. Additionally, the AUWPAE approach is employed to combine the base learners (classifiers) for the IoT stream analytics. This strategy facilitates seamless integration and enhances adaptability. Furthermore, it sustains the performance of conventional DDoS attack

detection systems in non-adversarial attack scenarios by selecting the most appropriate classifiers for each request type.

The proposed online MABTSE model, achieved 719 samples per second throughput, making it better than SRP-DDM, SRPs-ADWIN, and PWPAE for CICIoT2023 dataset. However, its performance was lower than ARF-ADWIN, HTs, and KNN-ADWIN. This finding is also consistent with the results from the IOTID20 dataset.

Figure 11 and Table 8 presents the performance comparison of the proposed MABTSE model against other online adaptive models including ARF-ADWIN, ARF-DDM, SRP-ADWIN, SRP-DDM, HT, KNN-ADWIN, and PWPAE utilizing the CICIoT2023 dataset. The experimental results indicate that the proposed model achieved an average accuracy of 99.28%, surpassing the performance of other methods. Furthermore, in terms of the precision, recall, and F1-score performance metrics, MABTSE achieved 98.88%, 96.53%, and 97.98%, respectively. These results demonstrate that MABTSE outperforms other online adaptive learning methods in the presence of concept drifts.

We calculated the confidence interval and p-value for the accuracy metrics and proposed MABTSE model and other baseline models including ARF-ADWIN, ARF-DDM, HT, KNN-ADWIN, SRP-DDM, SRP-ADWIN, and PWPAE. The results demonstrate superior performance of the proposed model, supported by statistical significance. The MABTSE model, utilizing IoTID20 dataset achieved a mean accuracy of 99.50%, with confidence interval of [99.44%, 99.56%] and p-value of < 0.0001. These result shows significantly outperform the other baseline models. Similarly, MABTSE model achieved a mean accuracy of 99.28%, a confidence interval of [99.21%, 99.35%], and a p-value of < 0.0001 utilizing CICIoT2023 dataset. These finding further highlight the statistical significance of the proposed model in comparison to the baseline models.

6.4.2. MABTSE under attack

The performance of the proposed MABTSE online DDoS detection model decreased significantly under the influence of perturbations generated by the PGD, FGSM, C&C, and BIM methods utilizing IOTID20 and CICIoT2023 datasets. The accuracy of the MABTSE DDoS detection model declined to a range of 38.92% to 67.12% for IOTID20 dataset and to rage of 13.08% to 32.38% for CICIoT2023 datasets, respectively. As shown in Tables 6 and 7, among the

four adversarial attacks, C&C was the most effective on both the IOTID20 and CICIoT2023 datasets. Furthermore, C&C attacks caused a significant increase in the model learning time, which is 0.7s and 0.5s for the IOTID20 and CICIoT2023 datasets respectively. This is because the C&C algorithm generates sophisticated and dynamic perturbation patterns, leading to increased complexity. Consequently, the presence of C&C attacks significantly increases the model learning time, because the MABTSE DDoS detection model requires additional processing.

6.4.3. Recovered

After the implementation of the Resilient Adversarial Detection and Purification (RADP) method to detect and purify adversarial attacks against an online DDoS attack detection system, the system's performance improved significantly. The accuracy, precision, recall, and F1-scores exhibit significant enhancements, closely resemble to those of the original MABTSE DDoS attack-detection model. The performance of the MABTSE DDoS attack detection model enhanced from a ranged of 99.39% to 99.48% for IOTID20 dataset and from a range of 99.39% to 99.48% for CICIoT2023 datasets, respectively. This demonstrates the effectiveness of the MSAAD framework in identifying and differentiating adversarial samples from original streaming data samples.

Table 10: The effect of adversarial and effect of defense mechanism using IOTID20 dataset

		Models comparison using IoTID20 dataset				
		A	P	R	F1	T
Original		99.50	99.47	100	99.74	0.7
Under attack	FGSM	60.58	80.80	4.22	5.78	0.6
	BIG	40.33	38.91	0.37	3.55	0.4
	PGD	38.44	33.94	0.16	2.56	0.4
	C&W	**32.38**	**26.91**	**0.06**	**1.13**	**0.7**
Recovered	**FGSM**	**99.48**	**99.37**	**99.99**	**99.67**	**0.7**
	BIG	99.40	99.41	99.89	99.64	0.3
	PGD	99.39	99.40	99.99	99.71	0.3
	C&W	99.43	99.33	99.90	99.69	0.8

Table 11: The effect of adversarial attack and effect of defense mechanism using CICIOT2023 dataset

		Models comparison using CICIoT2023 dataset				
		A	P	R	F1	T
Original		99.28	99.11	99.10	99.76	0.2
Under attack	FGSM	86.20	96.19	31.11	45.33	0.2
	BIG	84.68	96.10	46.74	49.87	0.3
	PGD	77.33	33.32	2.22	5.48	0.3
	C&W	**66.60**	**23.22**	**1.98**	**1.72**	**0.5**
Recovered	FGSM	99.26	99.01	99.09	99.67	0.2
	BIG	99.23	99.14	99.05	99.33	0.3
	PGD	99.22	99.11	99.11	99.71	0.3
	C&W	**99.27**	**99.14**	**99.03**	**99.35**	**0.4**

Legend: A=Accuracy, P= Precision, R= Recall, F= F1-score, T= Model learning time

Figure 12 and 13 illustrate the proposed MABTSE model for adversarial attacks, including FGSM, BIG, PGD, and C&W. These experiments demonstrate the vulnerability of the proposed MABTSE online DDoS attack detection models when evaluated using the IoTID20 and CICIoT2023 datasets. Notably, PGD and C&W exhibit particularly strong performance as accuracy declines. The IoTID20 dataset appeared to be more sensitive to FGSM and BIG attacks, showing sharper accuracy declines compared with the CICIoT2023 dataset, which can be attributed to the inherent characteristics of data. However, after implementing the recovery mechanisms (i.e., RADP), the proposed MABTSE model restore performance to level nearly equivalent to those of original dataset (without adversarial attack). This result demonstrates the robustness of the proposed defence strategy.

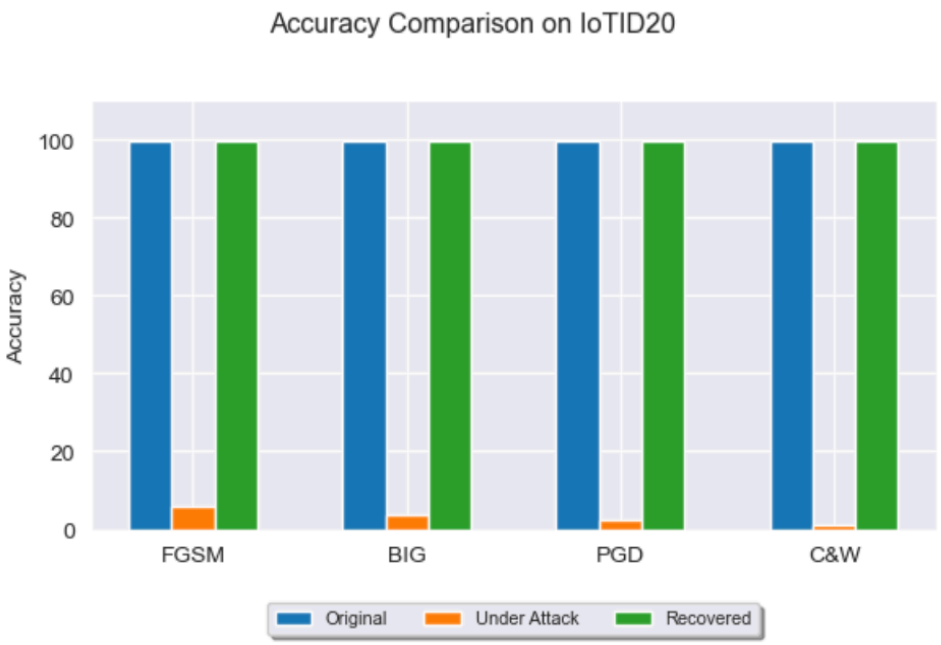

Figure 12: performance change using IoTID20 dataset

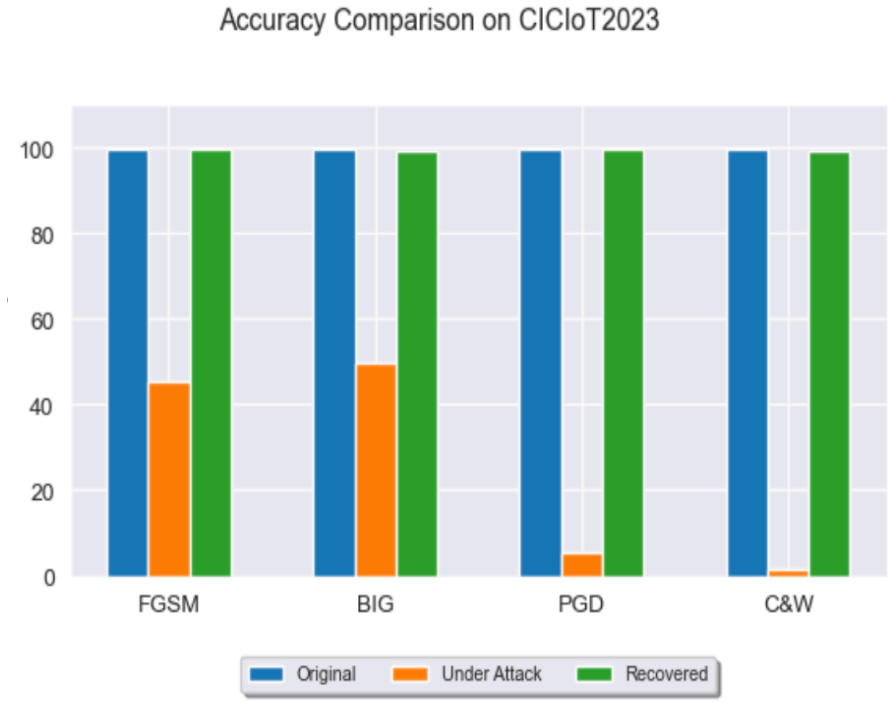

Figure 13: performance change using CICIoT2023 dataset

6.5. Summery

Adversarial attacks pose a significant challenge to the detection of Distributed Denial of Service (DDoS) attack in IoT systems. These attacks can exploit the vulnerability of online DDoS attack detection solutions, thereby degrade their accuracy. The current adversarial attack detection systems primarily focus on batch learning and are not equipped to handle multiple and unknown adversarial attacks. To address this challenge, this study provides a novel MSAAD framework that can protect online DDoS attack detection solutions against adversarial attacks. The proposed framework comprises of three defense layers: the Resilient Adversarial Detector and Purification (RADP), which detect and purify adversarial attacks against online DDoS attack detection systems for multiple and unknown adversarial attacks; A multiple-classifier approach, which makes it more challenging for an attacker to replicate the DDoS attack detection model; and MABTSE , which select dynamically classifiers or ensembles using the the Multi-Armed Bandits (MAB) algorithm with Thompson Sampling for each incoming traffic request. The frameworks maintain the performance of conventional DDoS attack detection systems in non-adversarial attack scenarios by selecting the most appropriate classifiers for each request type. The experiment results demonstrate the effectiveness of the proposed MABTSE model. The MABTSE achieved an average accuracy of 99.50% and 99.28% on IOTID20 and CICIoT2023 datasets respectively. The accuracy of the DDoS detection model declined significantly under adversarial attacks. however, it is successfully restored to range of 99.39% to 99.48% for IOTID20 dataset and range of 99.01% to 99.14% for CICIoT2023 datasets respectively in adversarial attack scenario. This demonstrates the robustness of MSAAD framework in dynamic IoT environment

Chapter Seven

7. Dynamic Weight Clustered Federated Learning (FedDWC) framework

The centralized learning approach of DDoS detection raises significant privacy concerns and poses scalability challenges, especially in IoT non-IID data distributions. Federated Learning (FL) offers a decentralized solution but suffers from slow convergence and degraded accuracy due to non-IID data and equal model aggregation schemes. This chapter presents the Dynamic Weighted Clustered Federated Learning (FedDWC) framework, which addresses Research Question 3 (RQ3) and fulfils Objective OB3. FedDWC enhances FL performance by introducing dynamic weighting and clustering IoT devices with similar data distributions. The result is a privacy-preserving detection model that maintains high classification accuracy, reduces communication overhead, and fast convergence under real-world, non-IID IoT conditions.

7.1. Problem Formulation

Distributed Denial of Service (DDoS) attacks are a serious challenge to cyber security, and require rapid and effective detection methods. The basis of our approach is the requirement to handle non-IID data distributions across IoT devices. IoT devices that participate in Clustered Federated Learning (CFL) capture distinct traffic patterns indicative of potential DDoS attacks. We considered a distributed learning setting with one central server and N IoT devices. Each IoT device corresponds to a user in a Federated Learning framework. Each IoT device has a distinctive dataset $D_1, D_2, D_3, \dots D_N$ which represents its network activity. IoT devices are partitioned into $c_1, c_2, c_3, \dots c_C$ disjoint clusters. We assume that each IoT device dataset contains non-IId data points $\xi_1, \xi_2, \xi_3, \dots \xi_K$ drawn from D_i. The data points consist of a pair of features and responses, denoted by $\xi_k = (x_k, y_k)$. The central server and IoT devices can communicate with each other using predefined protocols for T communication rounds. The primary goal is to protect each client's data privacy (i.e., no data should be shared with the central server and between clients) and autonomy, while minimizing a global objective function that accurately detects DDoS attack patterns.

Table 12: Table of notations

FL components	
N	Number of IoT device in the FL system
$D_i, \lvert D_i \rvert$	The dataset and its size on IoT device i

ξ_k	Data points consist of pair of features and response
$l(\phi_c, D_i)$	loss function that measures the performance of the cluster centroid ϕ_c on the local data D_i
$\nu_i(t)$	The learning rate for Client i in Iteration t
T	Number of local update steps
Clustering components	
C	Number of clusters
ϕ_i	Centroid or the average model parameters for cluster C
$s_{i,c} \in \mathbb{R}^{n*c}$	Binary assignment variable indicating whether client i is assigned to cluster C, where $s_{i,c} = 1$ if $i \in C$ else $s_{i,c} = 0$
$\|\|\omega_i - \phi_i\|\|_2^2$	This represents the squared Euclidean distance between the model parameters of IoT device i and the centroid of cluster ϕ_c
β_i	The importance weight of IoT device i in Cluster c and $\sum_{i \in k} \beta_i = 1$

In this study, a bi-level approach for optimizing federated learning systems was used by adopting and formulating Multi-Center Federated Learning[77]. Our goal is to optimize a collection of cluster-specific models that jointly enhance the overall learning performance of a distributed set of IoT devices, each containing data with a different data distribution. Moreover, we aim to improve DDoS attack detection accuracy by addressing data privacy and scalability challenges.

The bi-level optimization approach consists of two interconnected optimization problems. An upper-level problem aims to optimize a global objective function by learning a set of cluster-specific models that are effectively combined to enhance the detection of DDoS attacks across the network. This enhanced the overall performance and robustness of the federated model. The upper-level optimization problem focuses on minimizing the overall loss throughout the IoT devices to assign clients to their respective clusters at a lower level. This is a common approach in clustered federated learning, where IoT devices are grouped according to their similarity and each cluster maintains its own global model to personalize the learning process. The weighted average of the model parameters ω_i of the clients within cluster k at the tth iteration after the maximization step in an expectation-maximization (EM) algorithm in the context of federated learning is represented by ϕ_c.

$$\underset{\{\phi_c\}}{\text{minimize}} \; R = \frac{1}{N} \sum_{c=1}^{C} \sum_{i=1}^{N} s_{i,c} \, l(\phi_c, D_i) \tag{41}$$

In this context, the loss function l in this context is formulated to balance the accuracy of local DDoS detection against the need to generalize well to the broader network context. A lower-level problem is critical for optimal assignment of a client to a specific cluster. Client assignment to a cluster is a critical step to ensure that clients with similar data distributions are assigned within the same cluster. Client assignment to a cluster is performed in a way that minimizes the distance between the client and the centroid of the cluster. The efficacy of the upper-level problem is directly impacted by the lower-level optimization because clients within the cluster have similar data distributions. This can result in more accurate and efficient learning for each cluster-specific model.

$$\text{Subject to } s_{i,k} = \underset{s_{i,k}}{\operatorname{argmin}} \frac{1}{N} \sum_{c=1}^{C} \sum_{i=1}^{N} s_{i,k} \, || \, \omega_i - \phi_c ||_2^2 \tag{42}$$

where $\boldsymbol{\phi_c} = \frac{\sum_{i \in c} \boldsymbol{\omega_i}}{\sum_{i \in c} \boldsymbol{s_{i,c}}}$ is the centroid of the cluster $\boldsymbol{c}$.

7.2. FedDWC Framework

The proposed FedDWC framework handles the non-IID issue and maximizes the performance of DDoS attack detection by using clustered federated learning and dynamic weights. The dynamic weight within cluster federated learning is an important indicator for clustering to be consistent with the loss function in federated learning. Hence, we design a general form of the objective function for the clustered FL problem by considering dynamic weighted clustering within the bi-level optimization problem.

In this section, we present the theoretical framework and steps of the proposed FedDWC algorithm. This is crucial for bridging the gap between the conceptual formulation of the problem and its implementation. Figure 14 shows how the FedDWC operates and the details of the algorithm in steps. As stated in Section 1, the motivation for using dynamic weighted clustered federated learning is to address the challenges posed by non-IID data across clients. The core concept is to cluster IoT devices based on their data-distribution similarity and perform federated learning within each cluster to build tailored models.

The dynamic weight was adjusted based on the performance of the local model for each IoT device. The global server is first initialized by assuming that every local model performs equally; hence, it is essential. The weights assigned to each local model were then assembled into a global priority index matrix using a global server. Subsequently, the weights were automatically modified

according to the performance of the local models. This implies that better-performing models have a greater influence on the global model.

The proposed framework introduces a global cumulative objective function that incorporates dynamic weighted clustering. The global cumulative objective is described by two key components: the lower-level objective and upper-level objective. The lower-level objective is associated with the clustering technique, whereas the upper-level objective is associated with the averaging weights within the cluster. Liu et al.[18] and Long et al.[77] were the bases of our proposed framework. The FedDWC algorithm iterates through the assignment and local update steps until a convergence criterion is achieved. The assignment step optimization problem focuses on cluster assignments. The local update step determines the best set of model parameters for each cluster in a federated network. Convergence is assessed based on the stability of the cluster assignments and minimization of global or cluster-specific loss functions. The upper-level objective minimizes the global loss across all clients and clusters, modified by the importance weights of the clients. The upper-level objective in Equation 43, demonstrates the primary goal of federated learning to learn a global model that performs well across distributed datasets. Upper-level objective (Dynamic weighting) optimizes global accuracy by dynamically adjusting the weights of clients based on their individual performance, increasing contributions from higher-performing models. For instance, clients consistently providing accurate models receive higher weights, significantly boosting global model accuracy.

$$\underset{\{\phi_c\}}{minimize} \quad R = \frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{k}\sum_{i=1}^{n} s_{i,c}\beta_i \mathrm{L}\left(\phi_c, D_i\right) \tag{43}$$

The lower-level objective (Clustering) in Equation 44 aims to reduce the total weighted distance between the client models and the cluster centroids. This objective ensures that clients are assigned to clusters in a manner that reflects the similarity of data distribution and promotes more effective and specialized learning within each cluster.

$$subject\ to\ s_{i,c}, \{\phi_c\} = \underset{s_{i,c},\{\theta_c\}}{argmin} \quad S : \frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{k=1}^{c}\sum_{i=1}^{n} s_{i,c}\beta_i ||\omega_i - \phi_c||_2^2 \tag{44}$$

Traditional federated learning algorithms such as FedAvg equally average local model updates, regardless of their individual performance, resulting in slower convergence and lower accuracy in non-IID scenarios. Our dynamic weighting mechanism addresses this by adaptively adjusting

weights(β_i) based on local model performance. Dynamic weight β_i is an important weight for each IoT device within a cluster. The initial weight of client i in the first round is $\beta_{i,1} = \frac{1}{N}$. After the first round, the weight of each client $\beta_{i,t}$ in the cluster is determined using Equation 45:

$$\beta_{i,t} = \begin{cases} \beta_{i,t-1} + \beta_{i,t-1} * \xi, & if\ AccC_i > AccP \\ \beta_{i,t-1} - \beta_{i,t-1} * \xi, & if\ AccC_i < AccP \\ \beta_{i,t-1} & otherwise \end{cases} \quad (45)$$

$AccC_i$ $and\ AccP$ represent the current round accuracy for client i and the accuracy from the previous round for all clients, respectively, and ξ is the incentive for the client within the cluster for reward or penalty. Normalizing the weight is required after adjusting the weights based on the performance, that is, the accuracy of each client.

$$\beta_{i,t} = \frac{\beta_{i,t}}{\sum_{j=1}^{N} \beta_{j,t}} \quad (46)$$

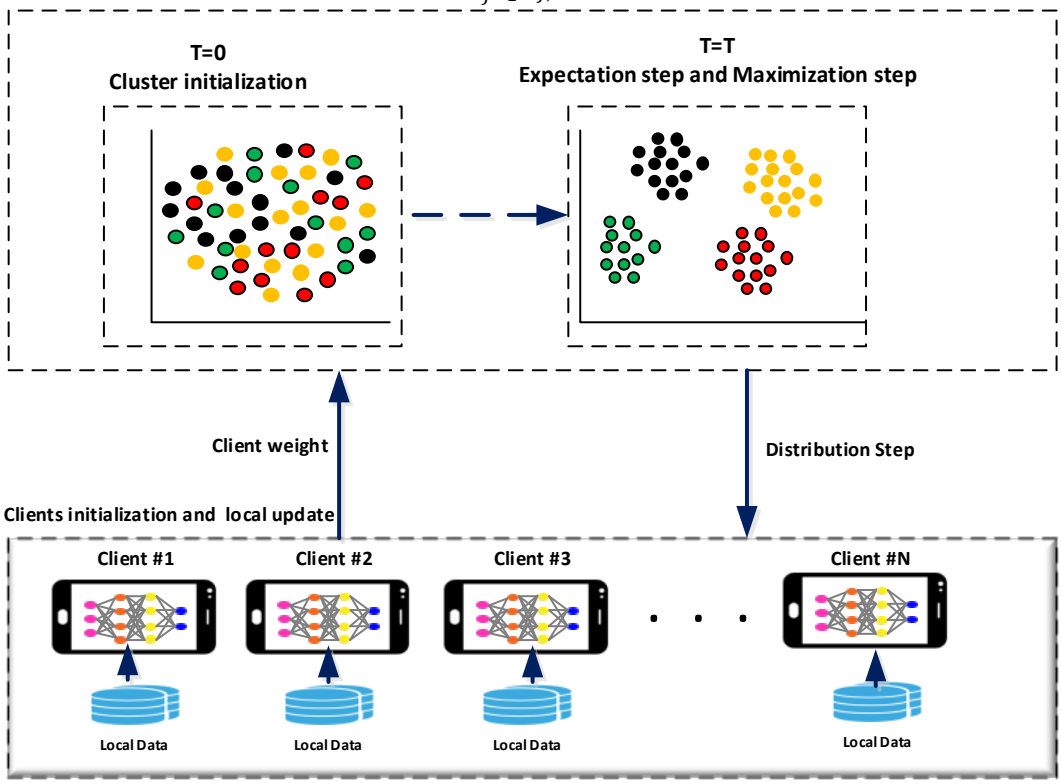

Figure 14: Dynamic Weight Clustered Federated Learning (FedDWC) framework

Algorithm 3:
Key steps in the Algorithm.

Input: Local datasets D = D_1, D_2, D_3 ... D_N
Initialized cluster centroid ϕ_1, ϕ_2, ϕ_3 ... ϕ_C
Initialized local network weight $\omega_1, \omega_2, \omega_3$ ... ω_N
Number of clusters C
Number of clients per cluster $clients\_per\_cluster$
Number of communication rounds T

Procedure:

1. **Standardize features**
   X_scaled ← **standardize_Features**(X)
2. **Reduce dimensionality**
   X_pca ← **apply_PCA** (X_scaled, min (30, X_scaled. shape))
3. **Initialized clusters**
   indices_clusters ← **Initialize_Clusters** (K, alpha_cluster, len(X_scaled))
4. **Generate and distribute non-iid data to cluster**
   clusters_data, clusters_labels ← **generatet_distribute_Data_To_Clusters** (X_pca, y, indices_clusters, K)
5. **Distribute non-iid data to client and initialize model**
   clients, cluster_models ← **initialize_Clients_and_Models** (clusters_data, clusters_labels, K, clients_per_cluster) // Distribute non-iid data to each client with in cluster and initialize model
6. **FOR t = 1 TO T DO**

   Expectation Step
   assignments ← **expectation_Step** (client_models, cluster_centroids)// Clients are assigned to clusters based on the distance's minimization between model parameters and the centroids of clusters.

   $$c = argmin_c \, ||\omega_i - \phi_c \, ||_2^2$$

   Maximization Step
   maximization_Step (client_models, client_assignments, num_clusters) // Each cluster's model parameters are updated for better suit the assignment of client to clusters

   dynamic_weighting (client_weight, client_accuracy**)**

   $$S = \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{k=1}^{c} \sum_{i=1}^{n} s_{i,c} \beta_i ||\omega_i - \phi_c||_2^2$$

   Distribution Step
   distribute_Models (cluster_centroids, client_assignments) // updated model of cluster distributed back to the clients.

   Local update and evaluate
   local_update and_evaluate_clients (clients_data) // minimizing the client's local loss function using few iterations of a local SGD

   $$minimize\{\phi_c\} \quad R = \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{c=1}^{k} \sum_{i=1}^{n} s_{i,c} \beta_i L(\phi_c, D_i)$$

7. **END**
8. **visualize_TSNE** (clients)

Dynamic Weighting

input: $\beta_{i,t} = \frac{I}{N_c}$

penalty/reward ξ

1. **dynamic_weighting (**client_weight, client_accuracy**)**

$$if\ AccC_i > AccP_i\ \beta_{i,t-1} + \beta_{i,t-1} * \xi$$
$$elseif \quad if\ AccC_i > AccP_i\ \beta_{i,t-1} + \beta_{i,t-1} * \xi$$
$$else\ \beta_{i,t-1}$$

$$\beta_{i,t} = \frac{\beta_{i,t}}{\sum_{j=1}^{N_c} \beta_{j,t}}$$

In $Algorithm\ 1$, the cluster centroids $\phi_1, \phi_2, \phi_3 \ldots \phi_C$ are initialized randomly. The algorithm then enters a loop in which each iteration consists of expectation, maximization, distribution and local update steps to refine these centroids based on the data distributed across different IoT devices.

The algorithm dynamically modifies the weights of IoT devices in federated learning based on their accuracy in comparison with the global accuracy of the previous round. Each IoT device starts with an equal share of influence, which is then adjusted based on whether its accuracy exceeds, equals, or is less accurate than the global accuracy from the previous round. If an IoT device's performance improves compared to the global one, its weight is increased (i.e., reward) by a predetermined ξ incentive; conversely, if performance declines, the weight is reduced (i.e., penalized). These adjustments are performed after each round of communication to ensure that better-performing models have a greater influence on cluster-wise averaging. The weights of all clients were normalized to maintain a consistent total influence across clusters. This cycle was repeated for all rounds of communication.

In the expectation step, each IoT device was assigned to a cluster. The assignment of IoT devices to a cluster is determined by the minimum distance between the IoT device's local model parameter and the centroid of the cluster. The distance measure is considered a similarity measure for each IoT device in a cluster. The minimization is weighted by coefficient β_i, which adjusts the importance or influence of each IoT device's data.

The maximization step updates each cluster centroid by averaging the contributions of all the IoT devices assigned to the cluster. This averaging aims to minimize the global function. The

global function is a weighted sum of the distances between each IoT device's model and its corresponding cluster centroid and then normalized by the sum of all weights. After the maximization step, the distribution step involves sending the updated cluster centroid back to the IoT device for local update.

In the local update step, each IoT device performs a series of gradient descent steps to improve its local model using its own data. The processes of expectation, maximization, distribution, and local update steps in each communication round continue until convergence is achieved. When the convergence requirements are met, the model parameters are properly optimized across all clusters and IoT devices. The optimized cluster centroid and vector indicating each IoT device's cluster assignment were included in the output of the algorithm. This structured approach allows efficient and scalable model training across a distributed network by clustering similar IoT device models and updating the corresponding centroids. This leads to higher performance and generalizable machine learning models in a federated learning environment.

7.3. Convergence Analysis

The convergence of the optimization problem includes an assignment step and a local update step. The assignment step optimization problem is focused on cluster assignments $s_{i,k}$ which are dependent on other optimization processes that minimize the individual objective function losses. The main objective assignment step is to find the optimal model parameters ϕ_c for each cluster that minimize the total weighted loss across all clients so that Equation 47 will not increase. Each client contributes a different weight β_i based on its data distribution. This allows for personalized models that are suited to clusters within a federated network.

$$subject\ to\ s_{i,k}, \{\phi_c\} = \underset{s_{i,k},\{\phi_c\}}{argmin} \quad s : \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{k=1}^{c} \sum_{i=1}^{n} s_{i,c} \beta_i ||\omega_i - \phi_c||_2^2 \tag{47}$$

Equation 47 shows the need to minimize S, which is the weighted sum of the squared Euclidean distances between each client's parameters ω_i and the cluster centres ϕ_c. The weights β_i are normalized by the sum of all β_i values.

The local update step determines the best set of model parameters for each cluster in a federated network. The local update step uses a gradient descent algorithm and a properly selected learning rate to minimize the local objective. These models were applied to their respective clients' data to minimize the weighted sum of their individual losses. This encourages both individual model accuracy and collaboration among models to achieve a minimized global objective as follows:

$$\underset{\{\phi_c\}}{minimize} \quad R = \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{c=1}^{k} \sum_{i=1}^{n} s_{i,c} \beta_i \mathrm{L}(\phi_c, D_i) \tag{48}$$

1) Convergence Analysis of expectation step (S)

The following definitions and assumptions are used to analyze the convergence of the optimization problem of FedDWC:

Assumption 1: Unbiased Gradient Estimator

The unbiased gradient estimator condition states that the expected value of the stochastic gradient $\nabla l(\omega_i, \phi)$ is an unbiased estimator of the local gradient or the true gradient of the loss function for each client. This can be mathematically represented as:

$$\mathrm{E}_{\xi_i \sim D_i}[\nabla \mathrm{l}(\omega_i, \xi)] = \nabla \mathrm{l}(\omega_i). \tag{49}$$

Assumption 2: Bounded Gradients

The bounded gradient condition ensures that the expectation of the L2 norm of the stochastic gradients $\nabla l(\omega_i, \phi)$ is bound by a constant U. This can be mathematically represented as:

$$E_{\xi_{i \sim D_i}}[\| \nabla l(\omega_i, \xi) \|_2^2] \leq U \tag{50}$$

This assumption is important for guaranteeing the stability of the optimization process. It prevents the optimization steps from becoming too large; otherwise, the learning process will diverge or oscillate.

Theorem 1: Clustering problem convergence.

Within the confines of Assumption 1, it is posited that for any arbitrary communication round (t), the proposed framework will converge if the learning rate, $\nu_i^{(t)}$, adheres to $\nu_i^{(t)} \leq \frac{||\omega_i^{(t)} - \phi_c||}{TU}$, then, S is assured to converge.

The theorem provides a criterion for determining the learning rate in relation to the current state of clustering or how close the model parameters are to the cluster centroids. The clustering algorithm is guaranteed to reach a stable solution if the learning rate is scaled suitably. This is vital for guaranteeing the usability and efficacy of the proposed algorithms, which use client clustering

to address data heterogeneity. For personalized and effective federated learning, the algorithm must be able to consistently assign clients to clusters according to the properties of their data, which is demonstrated by the convergence of S under these conditions.

2) Convergence Analysis of local update (R)

Definition 1. Clusterability, in the context of federated learning, measures the similarity of gradients or data distributions in different clients within the same cluster. A lower value of B indicates higher clusterability, indicating the client's data distribution similarity.

$$\frac{\left|\left|\phi_c - \nabla l_i(\omega_i, D_i)\right|\right|}{\phi_c} \leq B \tag{51}$$

The entire fraction is constrained to be less than or equal to some bound B.

Application

This clusterability measure ensures that, within each cluster, clients are not too divergent in terms of the directions and magnitudes of their updates. This can be critical for ensuring stable and efficient convergence in federated learning systems, where data and computational resources are distributed, and heterogeneity can often be a challenge.

Assumption 3. The property of convex functions, where the function at any point y lies below the line that is tangent to the function at point. The actual value of the loss function at y is not greater than the value of the loss at x plus the linear approximation of the change in loss from x to y. Then we will have:

$$l(y) \leq l(x) + \langle \nabla l(x), y - x \rangle \tag{52}$$

Assumption 4. Properties of smooth Lipschitz. It provides an upper bound on the value of function $l(y)$ based on its value and gradient at another point x, as well as the curvature of the function. Each loss function l is ψ-smooth, and we have

$$l(y) \leq l(x) + \langle \nabla l(x), y - x \rangle + \frac{\psi}{2}\left|\left|y - x\right|\right|_2^2 \tag{53}$$

Assumption 5. Properties of bounded gradient variance. The variance of the stochastic gradient $\nabla l(\omega i, \xi)$ is bounded by σ^2,

$$E_{\theta_i \sim Di}[\left|\left|\nabla l(\omega i, \theta) = \nabla l(\omega i)\right|\right|]_2^2 \tag{54}$$

$$E\left[\left|\left|\nabla l(\omega i, \theta)\right|\right|_2^2 - \left|\left|\nabla l(\omega_i)\right|\right|\right]_2^2 \leq \sigma^2 \tag{55}$$

It is also applied for L.

Theorem 2: Local update Convergence of Dynamic Clustered Federated Learning (FedDWC)

Given the fulfilment of Assumptions 1, 3, 4, and 5.

$$\nu_{(t,k)} < \min\{\frac{\| \omega_i^t - \phi_c||_2}{\mathrm{KU}}, \frac{2}{\psi}\left(\frac{E\left[\left|\left|\nabla \mathrm{l}\left(\phi_c^{(t,M,k)}\right)\right|\right|_2^2\right] - KU^2}{E\left[\left|\left|\nabla \mathrm{l}\left(\phi_c^{(t,M,k)}\right)\right|\right|_2^2\right] + \sigma^2}\right)\}$$

Then R is assured to converge.

Theorem 3: Convergence rate of Dynamic Clustered Federated Learning (FedDWC)

Given the fulfilment of Assumptions 1, 5, 6, and 7, the convergence of FedDWC is given by

$\mathcal{L} = R^{(0,L)} - R^{(t,L)}$ where $R^{(0,L)}$ is the initial loss and $R^{(t,L)}$ is the optimal loss.

$$T \geq \frac{\mathcal{L}}{\aleph} \tag{56}$$

$$\frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\nu_k^2}{2} - \nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2 + \nu_k KU^2 + \frac{\psi\nu_k^2}{2}\sigma^2 \leq \aleph \tag{57}$$

Convergence Analysis proof

We can prove the convergence of our Proposed FedDWC framework in two steps: assignment or expectation, and local update. Each stage consists of four steps: expectation, maximization, distribution, and local update.

1. Theorem 1 Proof:

Lemma 1. Expectation Step improvement

During the expectation phase of communication round $t+1$, the current model parameters ω and cluster centers ϕ. If the cluster assignment variable $s_{i,k}$, for client i is set to 1 for cluster c it minimizes the squared norm difference $||\omega_i - \phi_c||_2^{2^2}$ $_{b^2}$.:

$$c = argmin_c ||\omega_i - \phi_c||_2^2 \tag{58}$$

then we can prove that:

$$S^{(t+1,E)} \leq S^{(t,L)} \tag{59}$$

Proof: If $s_{i,c}^{t+1} = 1$, the right cluster c identified for client i that minimizes $\| \omega_i - \Omega_c^{(t,L)} \|$ which is the shortest Euclidean distance from the centroid of each cluster $\phi_1, \phi_2 \ldots\ldots \phi_c$ to client model parameter ω_i:

$$\beta_i \| \omega_i - \phi_c^{((t+1,E))} \|_2^2 \leq \beta_i \| \omega_i - \phi_c^{((t,L))} \|_2^2 \tag{60}$$

The aggregated loss after the expectation step is less than or equal to the aggregated loss before the expectation step.

$$\sum_1^n \beta_i \| \omega_i - \phi_c^{((t+1,E))} \|_2^2 \leq \sum_1^n \beta_i \| \omega_i - \phi_c^{((t,L))} \|_2^2 \tag{61}$$

Then, prove that:

$$F^{t+1,E} \leq F^{(t,L)} \tag{62}$$

Lemma 2. Maximization Step improvement

During the Maximization Step of communication round $t+1$, fixing the current model parameters ω and r. The cluster centroid update for cluster c at the maximization step is defined as:

$$\phi_c^{(t,M)} = \frac{\sum_{i \in c} \beta_i \omega_i}{\sum_{j \in c} \beta_i}$$

Then, prove that:

$$S^{(t,M)} \leq S^{(t,E)} \tag{63}$$

Proof: The square of The Expectation Step loss of arbitrary client i in cluster c is given as

$$\beta_i \| \omega_i - \phi_c^{((t,E))} \|_2^2 \tag{64}$$

Expanded to consider the distance to the new centroid after the Maximization step $\phi_c^{((t,M))}$

$$\beta_i \parallel \omega_i - \phi_c^{((t,E))} \parallel_2^2 = \beta_i \parallel \omega_i - \phi_c^{((t,M))} + \phi_c^{(t,M)} + \phi_c^{(t,E)} \parallel_2^2 \quad (65)$$

Using algebraic identity applying the distributive property of dot products.

$$\beta_i \parallel \omega_i - \phi_c^{(t,E)} \parallel_2^2 = \phi_i \parallel \omega_i - \phi_c^{((t,M))} \parallel_2^2 + \beta_i \parallel \phi_c^{(t,M)} + \phi_k^{(t,E)} \parallel_2^2$$

$$+2\beta_i \langle \omega_i - \sum_{i \in c} \frac{\beta_i}{\sum_{j \in c} \beta_i} \omega_i , \sum_{i \in c} \frac{\beta_i}{\sum_{j \in c} \beta_i} \omega_i - \phi_c^{(t,E)} \rangle \quad (66)$$

Aggregate the weighted distances for all clients assigned to cluster k

$$\sum_{i \in c} \beta_i \parallel \omega_i - \phi_c^{(t,E)} \parallel_2^2 = \sum_{i \in c} \backslash \beta_i \parallel \omega_i - \phi_c^{((t,M))} \parallel_2^2 + \sum_{i \in c} \beta_i \parallel \phi_c^{(t,M)} + \phi_c^{(t,E)} \parallel_2^2 \quad (67)$$

$$+ \sum_{i \in c} \left(2\beta_i \langle \omega_i - \phi_c^{((t,M))} , \phi_c^{((t,M))} - \phi_c^{(t,E)} \rangle \right)$$

$$\sum_{i \in c} \beta_i \parallel \omega_i - \phi_c^{(t,E)} \parallel_2^2 = \sum_{i \in c} \lambda_i \parallel \omega_i - \phi_c^{((t,M))} \parallel_2^2 + \sum_{i \in c} \beta_i \parallel \phi_c^{(t,M)} + \phi_c^{(t,E)} \parallel_2^2$$

$$+ \sum_{i \in c} \left(2\beta_i \langle \omega_i - \sum_{i \in c} \frac{\beta_i}{\sum_{j \in c} \beta_i} \omega_i , \sum_{i \in c} \frac{\beta_i}{\sum_{j \in c} \beta_i} \omega_i - \beta_c^{(t,E)} \rangle \right) \quad (68)$$

The centroid is essentially the balance point of all client weights. Hence, the combined effect of approaches zero:

$$\sum_{i \in c} \beta_i \parallel \omega_i - \phi_c^{(t,E)} \parallel_2^2 = \sum_{i \in c} \beta_i \parallel \omega_i - \phi_c^{((t,M))} \parallel_2^2 + \sum_{i \in c} \beta_i \parallel \beta_c^{(t,M)} + \phi_c^{(t,E)} \parallel_2^2 \quad (69)$$

Perform aggregation over all clusters k, from 1 to K

$$\sum_{c=1}^{C} \left(\sum_{i \in c} \beta_i \parallel \omega_i - \phi_c^{(t,E)} \parallel_2^2 \right) = \sum_{c=1}^{C} \left(\sum_{i \in c} \beta_i \parallel \omega_i - \phi_c^{((t,M))} \parallel_2^2 + \sum_{i \in c} \beta_i \parallel \phi_c^{(t,M)} + \phi_c^{(t,E)} \parallel_2^2 \right) \quad (70)$$

Multiply using. $\frac{1}{\sum_{j=1}^{n} \beta_j}$

$$S^{(t,M)} - S^{(t,E)} = -\frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i \parallel \phi_c^{(t,M)} + \phi_c^{(t,E)}||_2^2 \leq 0. \quad (71)$$

Lemma 3. Distribution Step Improvement

During the Distribution Step of communication round $t+1$, $\omega_{i\in c} = \phi_c$, the centroid ϕ and s. The local update of step T is defined as

$$\omega_i^1 = \omega_i^0 - \eta_i^{(t)} * \nabla l_i(\omega_i, D_i), \ldots$$

$$\omega_i^{(n+1)} = \phi_c - \nu_i^{(t)}\nabla l_i\left(\omega_i^0, D_i\right), -- \ \ldots -- \ \eta_i^{(t)}\nabla l_i\left(\omega_i^{k-1}, D_i\right) \quad (72)$$

The update rule machine learns the training algorithms to find a set of parameters that minimize the loss function.

$$\omega_i^{(n+1)} = \omega_c - \nu_i^{(t)}\sum_{k=1}^{k-1}\nabla l_i\left(\omega_i^q, D_i\right) \quad (73)$$

If $\nu_i^{(t)} \leq \frac{\|\omega_i^t - \phi_c\|_2}{KU}$, Then, prove that:

$$S^{(t,L)} \leq S^{(t,M)} \quad (74)$$

Proof:

First, we proof learning rate $\nu_i^{(t)} \leq \frac{\|\omega_i^t - \phi_c\|_2}{KU}$ and then $F^{(t,L)} \leq F^{(t,M)}$

Proof the learning:

Based on the update rule

$$\omega_i^{(n+1)} = \phi_c - \nu_i^{(t)}\nabla l_i\left(\omega_i^0, \theta_i\right), -- \ \ldots -- \ \nu_i^{(t)}\nabla l_i\left(\omega_i^{K-1}, \theta_i\right) \quad (75)$$

Take the norm squared of both sides and subtract centroid of cluster c

$$\| \omega_i^{n+1} - \phi_c||_2 = ||\phi_c - \nu_i^{(t)} \nabla l_i(\omega_i^0, \theta_i) - \quad \dots - \nu_i^{(t)} \nabla l_i(\omega_i^{K-1}, \theta_i) - \phi_c ||_2 \tag{76}$$

$$= \nu_i^{(t)} || \nabla l_i(\omega_i^0, \theta_i) + \quad \dots + \nabla l_i(\omega_i^{K-1}, \theta_i) ||_2$$

So,

$$\| \omega_i^{n+1} - \phi_k||_2^2 = \left(\nu_i^{(t)}\right)^2 || \nabla l_i(\omega_i^0, \theta_i) + \quad \dots + \nabla l_i(\omega_i^{K-1}, \theta_i) ||_2^2 \tag{77}$$

Using assumption

$$E_{\theta_{i\sim D_i}}[\| \nabla l(\omega_i, \theta) \|_2^2] \leq U$$

$$\| \omega_i^{n+1} - \phi_c||_2^2 \leq \left(\nu_i^{(t)} KU\right)^2 \tag{78}$$

$$\leq \| \omega_i^t - \phi_c||_2^2 \tag{79}$$

$$\nu_i^{(t)} \leq \frac{\| \omega_i^t - \phi_c||_2}{KU} \tag{80}$$

Considering the three Lemma, we can get:

$$S^{(t+1,L)} \leq S^{(t,L)} \tag{81}$$

2. **Theorem 2:** Local update Convergence of Dynamic Clustered Federated Learning (FedDWC)

Given the fulfilment of Assumptions 1, 5, 6, and 7.

$$\nu_{(t,k)} < \min\{\frac{\| \omega_i^t - \phi_c||_2}{KU}, \frac{2}{\psi}\left(\frac{E\left[\left|\left|\nabla l\left(\phi_c^{(t,M,k)}\right)\right|\right|_2^2\right] - KU^2}{E\left[\left|\left|\nabla l\left(\phi_c^{(t,M,k)}\right)\right|\right|_2^2\right] + \sigma^2}\right)\}$$

Then R is assured to converge.

Lemma 4. Expectation step to Maximization step

Given the conditions of Assumptions 1 and 5, from the expectation step to the maximization step in in an arbitrary communication round expressed as:

$$R^M \leq R^E + \mathrm{BU}^2 \tag{82}$$

Proof:

$$R^M - R^E = \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{c=1}^{k} \sum_{i \in c} \beta_i (l(\phi_k^M, D_i) - l(\omega_i, D_i)) \tag{83}$$

Given:

$$\phi_c^M = \sum_{d \in C} \frac{\beta_d}{\sum_{z \in c} \beta_z} \omega_d$$

Proof:

For arbitrary Client i, using Gradient Descent:

R^M: loss function evaluated after the maximization step

$$R^M = \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{c=1}^{C} \sum_{i \in c} \beta_i l(\phi_c^M, D_i)$$

R^E: The loss function evaluated after the expectation step.

$$R^E = \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{c=1}^{C} \sum_{i \in c} \beta_i l(\omega_i, D_i)$$

$$R^M - R^E = \frac{1}{\sum_{j=1}^{n} \beta_j} \sum_{c=1}^{C} \sum_{i \in c} \beta_i (l(\sum_{d \in C} \frac{\beta_d}{\sum_{z \in c} \beta_z} \omega_d , D_i) - l(\omega_i, D_i)) \tag{84}$$

Based on assumption 5 and Equation 66: $(y) \leq l(x) + \langle \nabla l(x), y - x \rangle$ for arbitrary cluster, we have:

$$\sum_{c=1}^{C} \sum_{i \in c} \beta_i (l(\sum_{d \in c} \frac{\beta_d}{\sum_{z \in c} \beta_z} \omega_d , D_i) - l(\omega_i, D_i)) \tag{85}$$

$$\leq \sum_{i \in c} \beta_i (\langle \nabla l \left(\omega_i^M, D_i\right), \sum_{d \in c} \frac{\beta_d}{\sum_{z \in k} \beta_z} \omega_d , \omega_i) \rangle) \tag{86}$$

Based on Cauchy Schwarz:

$$\leq \sum_{i \in c} \beta_i \left|\left|\nabla l \left(\omega_i^M, D_i\right)\right|\right|_2 \cdot \left|\left|\sum_{d \in c} \frac{\beta_d}{\sum_{z \in c} \beta_z} \omega_d - \omega_i\right|\right|_2 \tag{87}$$

$$\leq \sum_{i \in c} \beta_i \left|\left|\nabla l \left(\omega_i^M, D_i\right)\right|\right|_2 \cdot \left|\left|\phi_c^M - \omega_i\right|\right|_2 \tag{88}$$

Based on Assumption 6.1:

$$\leq \sum_{i\in c} \beta_i U \left|\left|\phi_c^M - \omega_{\mathrm{i}}\right|\right|_2 \tag{89}$$

Based on the update rule: $\omega_i^{(n+1)} = \phi_c - \nu_i^{(t)} \nabla l_i\left(\omega_i^0, D_i\right), -- \ \ldots \ -- \ \nu_i^{(t)} \nabla l_i\left(\omega_i^{\mathrm{K}-1}, D_i\right)$

$$\leq \sum_{\mathrm{i}\in \mathrm{c}} \beta_{\mathrm{i}} \, \nu \, \mathrm{KU}^2 \tag{90}$$

$$\leq \nu \, \mathrm{KU}^2 \tag{91}$$

$$\leq \mathrm{BU}^2 \tag{92}$$

Then

$$R^M \leq R^E + BU^2 \text{ or } R^M \leq R^E + \nu \, \mathrm{K}U^2 \tag{93}$$

Lemma 5. Maximization step to Local update

Given the conditions of Assumptions 6 and 7, from the expectation step to the maximization step in in an arbitrary communication round, we have

$$E[R^L] \ - \ R^M \ \leq \frac{1}{\sum_{j=1}^n \beta_j} \sum_{c=1}^{C} \sum\nolimits_{i\in c} \beta_i \sum_{k=1}^{K-1} \left(\frac{\psi \, \nu_k^2}{2} - \nu_k \, E[|| \, \nabla \mathrm{l} \, \left(\phi_c^{(M,k)}\right)||_2^2 \ + \frac{\psi \, \nu_k^2}{2} \sigma^2 \right. \tag{94}$$

Proof:

$$\mathrm{R}^{\mathrm{L}} \ - \ \mathrm{R}^{\mathrm{M}} \ = \frac{1}{\sum_{\mathrm{j}=1}^{\mathrm{n}} \beta_{\mathrm{j}}} \sum_{\mathrm{c}=1}^{\mathrm{C}} \ \sum_{\mathrm{i}\in \mathrm{c}} \beta_{\mathrm{i}} (\mathrm{l}(\phi_{\mathrm{c}}^{\mathrm{l}}, \mathrm{D_i}) - \mathrm{l}(\phi_{\mathrm{c}}^{\mathrm{M}}, \mathrm{D_i})) \tag{95}$$

Gradient Descent for arbitrary Client i

$$l(\Omega_k^l, D_i) - l(\Omega_k^M, D_i) \ = \ \sum_{k=1}^{K-1} l((\phi_c^{M.k+1}, D_i) - l(\phi_c^{(M,k)}, D_i)) \tag{96}$$

Based on Assumption 6.6. Lipschitz Smooth:$l(y) \leq l(x) + \ \langle \nabla \mathrm{l(x)}, \mathrm{y} - \mathrm{x} \rangle + \frac{\psi}{2} \left|\left|y - x\right|\right|_2^2$ then

$$l(\phi_c^{M,k+1}) - l(\phi_c^{M,k}) \ \leq \langle \nabla \mathrm{l}(\phi_c^{M,k}), \ \phi_c^{M,k+1} - \phi_c^{M,k} \rangle \ + \frac{\psi}{2} || \ \phi_c^{M,k+1} - \phi_c^{M,k}||_2^2 \tag{97}$$

Based on Gradient Descent Update: $\phi_c^{M,k+1} = \phi_c^{M,k} - \ \nu \nabla \mathrm{l}(\phi_c^{M,k}, \theta_i^k)$

$$\leq \langle \nabla \mathrm{l}\left(\phi_c^{M,k}\right),\ -\nu \nabla \mathrm{l}(\phi_c^{M,k}, \theta_i^k)\ \rangle\ + \frac{\psi \nu^2}{2} ||\ \nabla \mathrm{l}(\phi_c^{M,k}, \theta_i^k)||_2^2 \tag{98}$$

Taking the expectation over the random batch θ, we get:

$$E[\langle \nabla l\left(\phi_c^{M,k}\right),\ -\nu \nabla l(\phi_c^{M,k}, \theta_i^k)\ \rangle \tag{99}$$

Since the inner product of a vector with itself is just the norm squared

$$-\ \nu E[||\ \nabla l(\phi_c^{M,k}, \theta_i^k)||^2] \tag{100}$$

Combining these two terms and applying the expectation:

$$E[l(\phi_c^{M,k+1}) - l(\phi_c^{M,k})] \leq -\ \nu E[||\ \nabla \mathrm{l}(\phi_c^{M,k})||^2] + \frac{\psi \nu^2}{2} E[||\ \nabla \mathrm{l}(\phi_c^{M,k}, \theta_i^k)||_2^2] \tag{101}$$

Appling the variance σ^2 of the stochastic gradient: $E\left[\left|||\nabla l(\omega_i, \theta)||_2^2 - ||\nabla l(\omega_i)||\right|_2^2 \leq \sigma^2\right.$

$$E[l(\phi_c^{M,k+1}) - l(\phi_c^{M,k})] \leq -\ \nu E[||\ \nabla \mathrm{l}(\phi_c^{M,k})||_2^2] + \frac{\psi \nu^2}{2} (\sigma^2 + \mathrm{E}||\nabla \mathrm{l}(\phi_c^{M,k})||_2^2) \tag{102}$$

$$E\left[l\left(\phi_c^{M,k+1}\right) - l\left(\phi_c^{M,k}\right)\right] \leq \frac{\psi \nu^2}{2} \sigma^2 + (\frac{\psi \nu^2}{2} - \nu)\ ||\ \nabla \mathrm{l}(\phi_c^{M,k})||_2^2 \tag{104}$$

Applying a telescoping sum over all iterations

$$E[l(\phi_c^l, D_i)] - l(\phi_c^k, D_i)\ =\ \sum_{k=1}^{K-1} E[l((\phi_c^{M.k+1}, D_i)] - l(\phi_c^{(M,k)}, D_i)) \tag{105}$$

then:

$$E[l(\phi_c^l, D_i)] - l(\phi_c^M, D_i)\ =\ \sum_{k=1}^{K-1} \frac{\psi \nu_k^2}{2} \sigma^2 + (\frac{\psi \nu_k^2}{2} - \nu_k)\ ||\ \nabla \mathrm{l}(\phi_c^{M,k})||_2^2 \tag{106}$$

Finally, we will get:

$$E[R^L]\ -\ R^M\ \leq \frac{1}{\sum_{j=1}^n \beta_j} \sum_{c=1}^{C} \sum\nolimits_{i \in c} \beta_i \sum_{k=1}^{K-1} \frac{\psi \nu_k^2}{2} \sigma^2 + (\frac{\psi \nu_k^2}{2} - \nu_k)\ ||\ \nabla \mathrm{l}(\phi_c^{M,k})||_2^2 \tag{107}$$

Now, we can proof Theorem 6.8 as follows

Proof: From local updates in round $t-1$ to the expectation step in round t, there is no change in the value of the loss function. Hence,

$$R^{(t-1,L)} = R^{(t,E)} \tag{108}$$

Based on then according to Lemma 4 and 5:

Considering $R^M \leq R^E + \nu \mathrm{KU}^2$ and

$$E[R^L] - R^M \leq \frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}\frac{\psi\nu_k^2}{2}\sigma^2 + (\frac{\psi\nu_k^2}{2} - \nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2 \tag{109}$$

$$E[R^{(t,L)}] - R^{(t-1,L)} \leq \nu_k KU^2 + \frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}\frac{\psi\nu_k^2}{2}\sigma^2 + (\frac{\psi\nu_k^2}{2} - \nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2 \tag{110}$$

$$0 \leq \frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\,\nu_{(t,k)}^2}{2} - \nu_{(t,k)})\,E[||\,\nabla l\left(\phi_c^{(t,M,k)}\right)||_2^2 + \frac{\psi\,\nu_{(t,k)}^2}{2}\sigma^2 + \nu_{(t,k)}KU^2) \tag{111}$$

$$0 \leq (\frac{\psi\,\nu_{(t,k)}^2}{2} - \nu_{(t,k)})\,E[||\,\nabla l\left(\phi_c^{(t,M,k)}\right)||_2^2 + \frac{\psi\,\nu_{(t,k)}^2}{2}\sigma^2 + \nu_{(t,k)}KU^2) \tag{112}$$

$$0 \leq \frac{\psi\,\nu_{(t,k)}^2}{2}E[||\,\nabla l\left(\phi_c^{(t,M,k)}\right)||_2^2 - \nu_{(t,k)}\,E[||\,\nabla l\left(\phi_c^{(t,M,k)}\right)||_2^2 + \frac{\psi\,\nu_{(t,k)}^2}{2}\sigma^2 + \nu_{(t,k)}KU^2 \tag{112}$$

$$E[||\,\nabla l\left(\phi_c^{(t,M,k)}\right)||_2^2 - BU^2 \leq \frac{\psi\,\nu_{(t,k)}}{2}E[||\,\nabla l\left(\phi_c^{(t,M,k)}\right)||_2^2 + \frac{\psi\,\nu_{(t,k)}}{2}\sigma^2 \tag{113}$$

So that:

$$\nu_{(t,k)} < \min\{\frac{\|\omega_i^t - \phi_c\|_2}{\mathrm{KU}}, \frac{2}{\psi}\left(\frac{E\left[\left\|\nabla l\left(\phi_c^{(t,M,k)}\right)\right\|_2^2\right] - KU^2}{E\left[\left\|\nabla l\left(\phi_c^{(t,M,k)}\right)\right\|_2^2\right] + \sigma^2}\right)\} \tag{114}$$

Therefore, from the above equation, we can see that the convergence of the loss function F and loss function R decreases monotonically; hence, we can conclude that FedDWC converges.

Theorem 3: Convergence rate of Dynamic Clustered Federated Learning (FedDWC)

Given the fulfilment of Assumptions 1, 5, 6, and 7, the convergence of FedDWC is given as follows:

$\mathcal{L} = R^{(0,L)} - R^{(t,L)}$ where $R^{(0,L)}$ is the initial loss and $R^{(t,L)}$ is the optimal loss.

$$T \geq \frac{\mathcal{L}}{\aleph} \tag{115}$$

$$\frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\nu_k^2}{2}-\nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2 + \nu_k KU^2 + \frac{\psi\nu_k^2}{2}\sigma^2 \leq \aleph \tag{116}$$

The above equation shows that the number of communication rounds needed to achieve high performance is inversely proportional to $\aleph$. Hence, to achieve higher performance (or smaller loss), more communication rounds T are required. The convergence rate, FedDWC, becomes $T = O\left(\frac{1}{T}\right)$, which is linear.

Proof:

The expected loss function reduction per communication round bounded by the following equation:

$$\mathcal{L} = R^{(0,L)} - R^{(t,L)} \tag{117}$$

where $R^{(0,L)}$ initial loss and $R^{(t,L)}$ optimal loss

$$E[R^{(0,L)}] - R^{(t,L)} \geq \nu_k KU^2 + \frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}\frac{\psi\nu_k^2}{2}\sigma^2 + (\frac{\psi\nu_k^2}{2} - \nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2 \tag{118}$$

$$E[R^{(0,L)}] - R^{(t,L)} \geq \frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\nu_k^2}{2}-\nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2 + \nu_k KU^2 + \frac{\psi\nu_k^2}{2}\sigma^2 \tag{119}$$

Summing the above equation over T communication round

$$\sum_{t=0}^{T-1} E[R^{(0,L)}] - R^{(t,L)}$$

$$\geq T(\frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\nu_k^2}{2}-\nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2\ +\nu_k KU^2+\frac{\psi\nu_k^2}{2}\sigma^2). \quad (120)$$

$$\mathcal{L}\geq\ T(\frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\nu_k^2}{2}-\nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2\ +\nu_k KU^2+\frac{\psi\nu_k^2}{2}\sigma^2) \quad (121)$$

$$T\ \geq\ \frac{\mathcal{L}}{\frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\nu_k^2}{2}-\nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2+\nu_k KU^2+\frac{\psi\nu_k^2}{2}\sigma^2} \quad (122)$$

if

$$\frac{1}{\sum_{j=1}^{n}\beta_j}\sum_{c=1}^{C}\sum_{i\in c}\beta_i\sum_{k=1}^{K-1}(\frac{\psi\nu_k^2}{2}-\nu_k)\,||\,\nabla l(\phi_c^{M,k})||_2^2\ +\nu_k KU^2+\frac{\psi\nu_k^2}{2}\sigma^2\ \leq\ \aleph \quad (123)$$

Then

$$T\ \geq\ \frac{\mathcal{L}}{\aleph} \quad (124)$$

Hence, we can conclude that the convergence rate is proportional to $1/T$ which indicates linear convergence. The above equation shows that the number of communication rounds is inversely proportional to ℵ. Hence, to achieve higher performance, more communication rounds T are needed. The convergence rate was $T = O\left(\frac{1}{T}\right)$

7.4. Experiments

7.4.1. Data Pre-processing and Simulation

Data pre-processing is a basic step in all federated learning applications, including DDoS attack detection. The performance of any detection method is significantly affected by the representation, size, and quality of a dataset. A dataset with high dimensionality and a large number of duplicate and irrelevant features affects the training and performance of the proposed framework. To address these issues, in the data pre-processing phase, we used data cleaning, data encoding, data normalization, and feature selection techniques. We implemented Dirichlet distribution for real-world data distribution simulations across IoT devices. The Dirichlet distribution is a family of

continuous multivariate probability distributions parameterized by a vector of positive reals. A Dirichlet distribution is used in Bayesian statistics, machine learning, and data analysis to model the probabilities of multiple categories. Dirichlet distribution is useful for simulating dynamic data distribution and is particularly useful for creating realistic simulations of data distribution across various IoT devices to mimic real-world scenarios of data heterogeneity.

Our research defines the Dirichlet distribution simulation parameter alpha (α) for our experiment. Parameter α of the Dirichlet distribution controls the concentration of the distribution. A smaller α leads to higher data heterogeneity among IoT devices, meaning that the data distribution is skewed and imbalanced. A larger value of α tends to generate more evenly distributed data across clients. For our experiment, we used α= 10 for intra-cluster to have evenly distributed data across clients and α= 0.1 for inter-cluster to have more skewed and imbalanced data distribution. This is particularly useful in cybersecurity applications within IoT networks, where nodes may not only have different quantities of data but also see vastly different types of data, which is crucial for developing robust DDoS detection systems.

7.4.2. Experiment Environment

The proposed online DDoS detection framework aims to detect DDoS attacks on the IoT systems. To observe the performance of the proposed approach, we developed a prototype using Python 3.10 programming language in the Jupyter Notebook environment. To support reproducibility and adversarial robustness evaluation, we used Python 3.10 and explicitly documented all critical dependencies, including River 0.15.0, Scikit-learn 1.2.2, and TensorFlow 2.11. For adversarial attack generation, we integrated the Adversarial Robustness Toolbox (ART) with a custom-trained GAN to simulate zero-day perturbations. A fixed random seed (42) was used across all experiments to ensure consistency. The experiment was conducted on a machine running an Intel(R) Xeon(R) CPU @2.20GHz and 16 GB of RAM.

7.4.3. Model Architecture and Baseline

The proposed framework uses a lightweight Convolutional Neural Network (CNN) architecture that is suitable for IoT device constraints on computation and memory usage. The proposed framework was optimized using a variant of stochastic gradient descent (SGD) tailored to federated settings. The SGD facilitates efficient convergence in highly skewed data distributions. FedDWC is benchmarked against federated learning algorithms such as FedAvg[6], FedProx[37], and IFCA[38], which underscores its advantages in handling non-IID data

effectively. We train FedAvg, FedProx, and IFCA T times. We evaluated the performance in terms of accuracy using client-wise and cluster-wise generated non-IID data.

7.5. Result and discussion

In this section, we present a detailed analysis of the experimental results obtained by evaluating the proposed FedDWC framework against three state-of-the-art methods, namely, FedAvg, FedProx, and IFCA. The primary objective of the experiments is to analyze the detection accuracy, convergence, size and complexity, and clustering in the context of DDoS attack detection under non-IID and IID data conditions. The evaluation was performed using two datasets, IoTID20 and CICIoT2023, for a comprehensive analysis.

7.5.1. DDoS Attack Detection

In this section, we compare the DDoS attack detection accuracy of the proposed FedDWC framework with those of other state-of-the-art methods: FedAvg, FedProx, and IFCA. The experiment was conducted using non-IID and IID datasets to evaluate DDoS attack detection accuracy. As shown in Figures 15 and 16 and Table 12, FedDWC achieved a DDoS attack-detection accuracy of 99.11% using the IoTID20 non-IID dataset with five clusters and 100 IoT devices. The experimental results showed that FedDWC outperformed FedAvg, FedProx, and IFCA, with accuracies of 96.47%, 97.93%, and 98.14%, respectively. Similarly, FedDWC achieved a DDoS attack-detection accuracy of 98.56% using the CICIoT2023 non-IID dataset with five clusters for 100 IoT devices. When the number of clusters to ten and 200 IoT devices, FedDWC achieved an improved DDoS attack detection accuracy of 99.21% using the IoTID20 non-IID dataset and 99.10% for the CICIoT2023 non-IID dataset as it shown in Table 8.

In the case of the IID datasets, FedDWC achieved 98.74% accuracy, which is relatively better than FedAvg, FedProx, and IFCA, which have accuracies of 98.11%, 98.20%, and 98.23%, respectively, for the IoTID20 IID dataset with five clusters and 100 IoT devices. When the number of clusters to ten and 200 IoT devices, FedDWC achieved an improved DDoS attack detection accuracy of 99.09% using the IoTID20 IID dataset and 99.16% for the CICIoT2023 IID dataset. The performance of FedDWC demonstrates its ability to effectively handle non-IID data through dynamic weighting and clustering. Clustering clients with similar data distributions and applying dynamic weighting to each IoT device within a cluster improves the performance of the proposed framework. FedProx's accuracy (98.20%) and IFCA's accuracy (97.90%) are better than FedAvg's accuracy of 97.31% when using the IoTID20 non-IID dataset with ten clusters and 200

IoT devices. This is mainly because they are built to handle challenges related to non-IID data. FedProx handles non-IID data by optimizing proximal terms. IFCA uses clustering to enhance the performance of federated learning in non-IID environments. However, FedDWC uses a combination of dynamic weighting and clustering to provide superior results for handling non-IIDs.

Our study evaluates the clustered federated learning and dynamic weighting components of FedDWC framework. The original proposed FedDWC framework integrates both dynamic weighting and clustering mechanisms. The experiment is conducted using the IoTID20 non-IID dataset with 10 clusters and 200 IoT devices. It achieved the highest accuracy of 99.21%. While removing the dynamic weighting component from FedDWC, the accuracy decreased to 98.25%. Similarly, while removing the clustering mechanism, the accuracy dropped to 97.87%, which is less by about 1.34% when compared to the original FedDWC. These results clearly indicate that both dynamic weighting and clustering contribute to the accuracy of FedDWC framework. In addition, we compared FedDWC to the state-of-the-art methods under identical conditions. In this comparison, FedAvg, FedProx and IVCA achieved an accuracy of 97.31%, 97.90%, and 98.20%, respectively. Hence, FedDWC framework demonstrates better performance as compared to the existing methods and also emphasizes the important roles of its integrated dynamic weighting and clustering components.

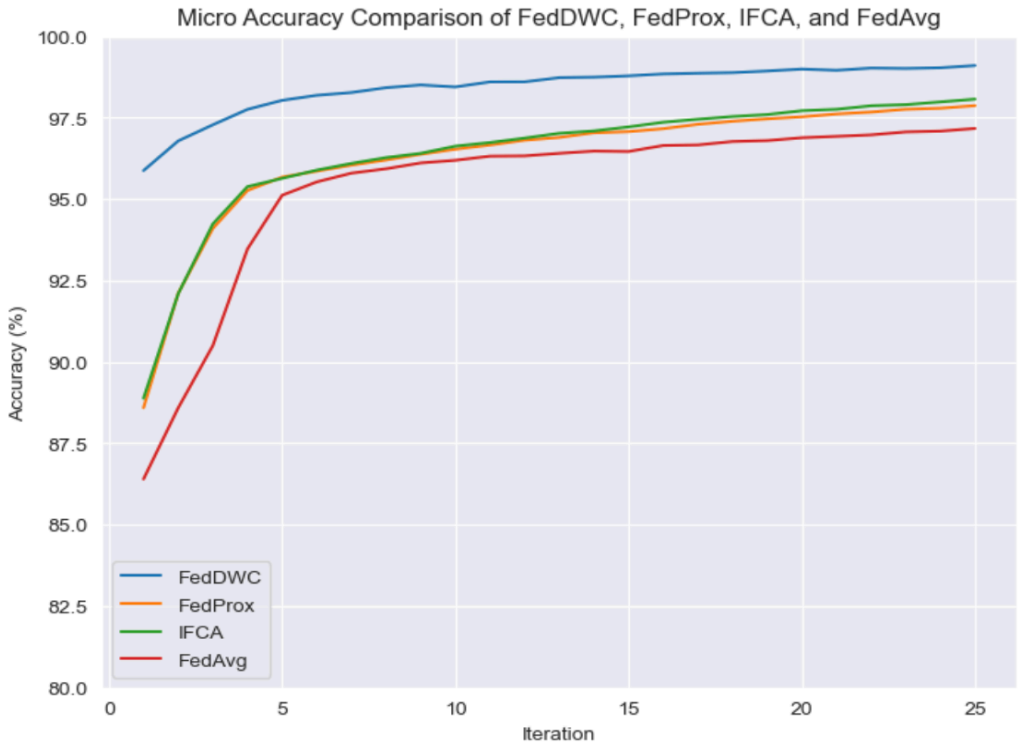

Figure 15:Accuracy comparison using non-IID CICIoT2023 dataset

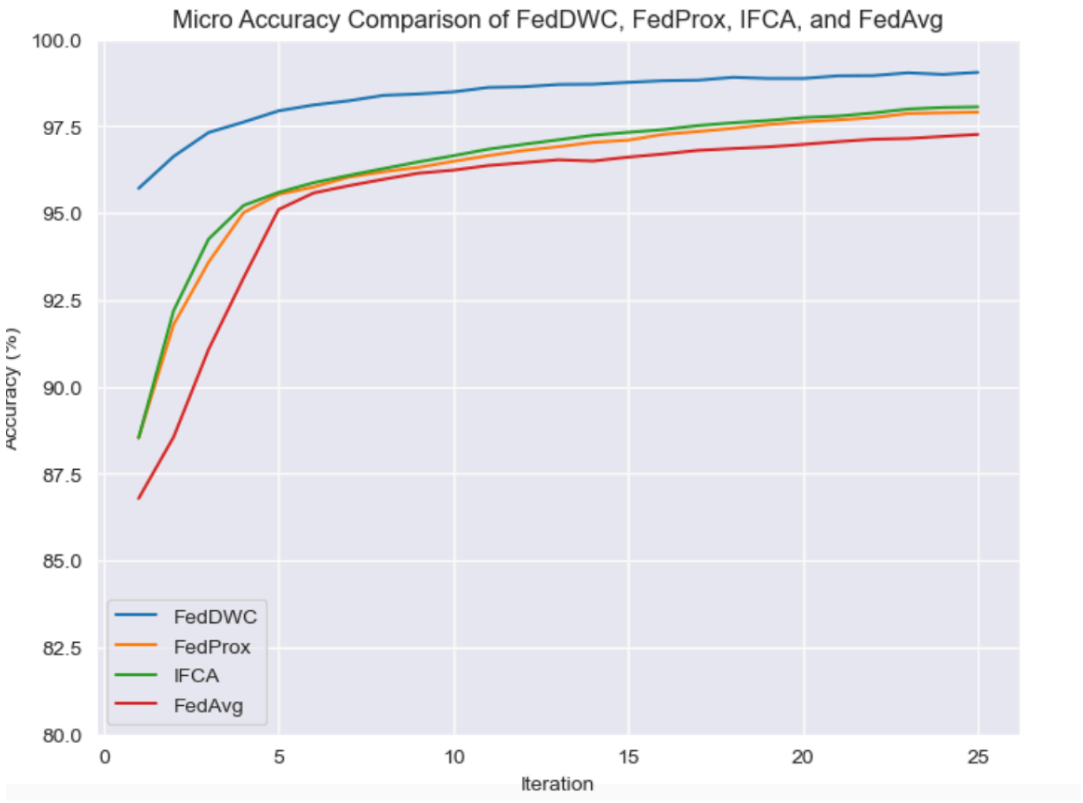

Figure 16: Accuracy comparison using non-IID IoTID20 dataset

Table 13: Performance comparison using non-IID and IID dataset

		IID		**Non-IID**	
		CICIoT2023	**IoTID20**	**CICIoT2023**	**IoTID20**
Cluster = 5, 100 IoT device	**Method**	Accuracy	Accuracy	Accuracy	Accuracy
	FedAvg	98.00%	98.11%	96.94%	96.47%
	FedProx	97.99%	98.20%	97.92%	97.93%
	FICA	98.25%	98.23%	98.15%	98.14%
	FedDWC	**98.61%**	**98.74%**	**98.56%**	**99.11%**
Cluster = 10, 200 IoT device	**FedAvg**	98.13%	98.09%	97.33%	97.31%
	FedProx	98.85%	98.76%	97.90%	97.90%
	FICA	98.78%	98.98%	98.02%	98.20%
	FedDWC	**99.16%**	**99.09%**	**99.10%**	**99.21%**

7.5.2. Size and Complexity

The experiments also examined the impact of the number of clients and clusters on the accuracy and complexity. The experimental results demonstrate that the accuracy of the proposed

framework increases when the number of clients and clusters increases. FedDWC showed an improvement from 99.56% to 99.10% for the CICIoT2023 dataset, where the cluster number and IoT devices were increased from 5 to 10 and 100 to 200, respectively. Our research performed additional scalability experiments by increasing the number of IoT devices to 500 and 1000. The experiment result demonstrates the accuracy obtained using IoTID20 dataset under non-IID conditions are 99.31% and 99.41% for 500 and 1000 IoT devices respectively. This improvement could be attributed to the broader and more diverse dataset represented by the increased number of IoT devices and the model's generalization capability. The scalability experiments validate the capability of FedDWC to leverage large-scale federated learning scenarios effectively. This demonstrated the scalability of the proposed framework. Scalability is important when large-scale IoT device participation is required. The time complexity of the FedDWC is linear, as indicated by the convergence rate analysis, $T = O\left(\frac{1}{T}\right)$. This convergence rate ensures the manageability of the computational overhead. Moreover, this strengthens the practicality and efficiency of the proposed FedDWC framework.

7.5.3. Clustering Analysis

Client assignment to a cluster is a critical process for ensuring the data distribution of the client's similarity within the cluster. The efficacy of the overall performance was directly affected by the client-to-cluster assignment. Clients within the cluster had similar data distributions. Figure 21 and 22 show the cosine similarities of the clusters. The client assignment to the cluster is performed in a way that minimizes the distance between the client and the centroid of the cluster. The effectiveness of the proposed FedDWC framework was verified using t-SNE visualization. This visualization shows how clients are clustered into two dimensional vectors. Figures 17 and 18 show the clustering results generated by the FedDWC framework using the CICIoT2023 and IoTID20 datasets, respectively. We can easily see intra-cluster and inter-cluster distributions, and can also easily distinguish each cluster from the others. The experimental results show that more clients will be concentrated on specific clusters, owing to the non-IID algorithm that generates non-IID data for each client. Figures 19 and 20 show the clustering results for the clusters in the second communication round and 25 communication rounds. Once the convergence stage was reached, the distance from the client to the centroid did not change. The client concentrates on specific clusters, and the distance between the client and centroid of the cluster is minimized. This implies that the cluster structure will be the same in any communication round after convergence.

The scalability of FedDWC was tested by incrementally increasing the number of IoT devices in the cluster and the number of clusters to evaluate how well the learning process is scaled under heightened stress and data diversity. In both scenarios, the performance of the proposed framework was not affected.

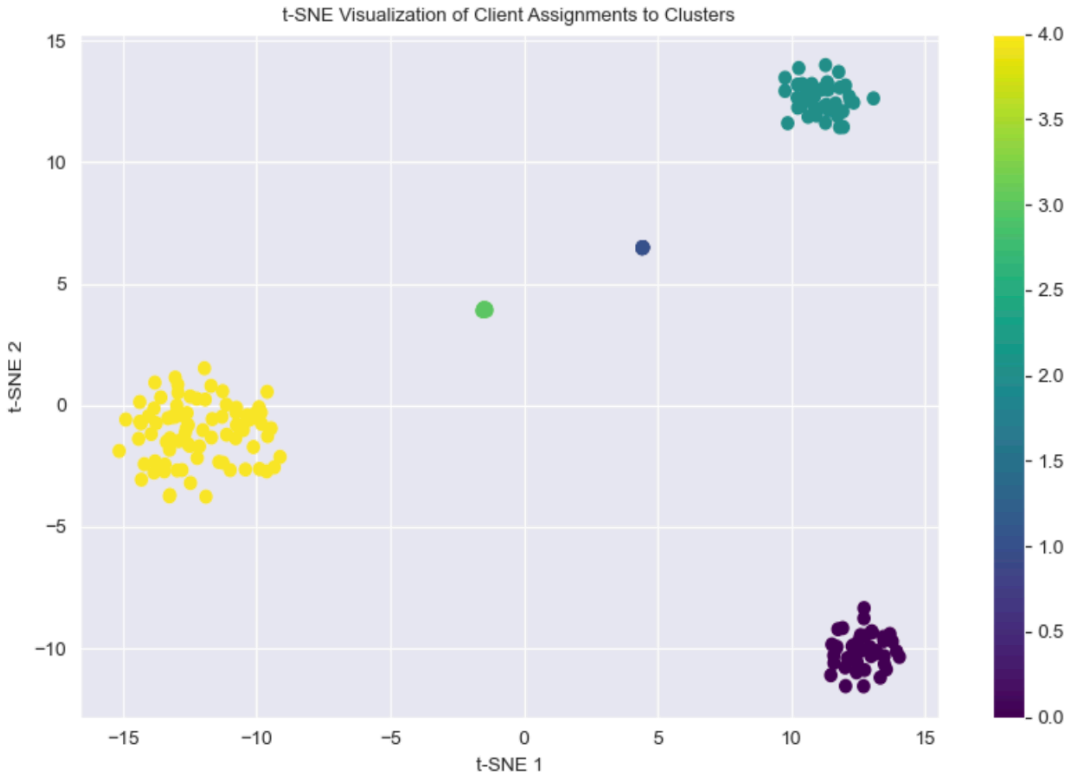

Figure 17: t-SNE visualization of Client Assignment to Cluster using CICIoT2023

Figure 18: t-SNE visualization of Client Assignment to Cluster using IoTID20

Figure 19: t-SNE visualization of Client Assignment to Cluster using CICIoT2023

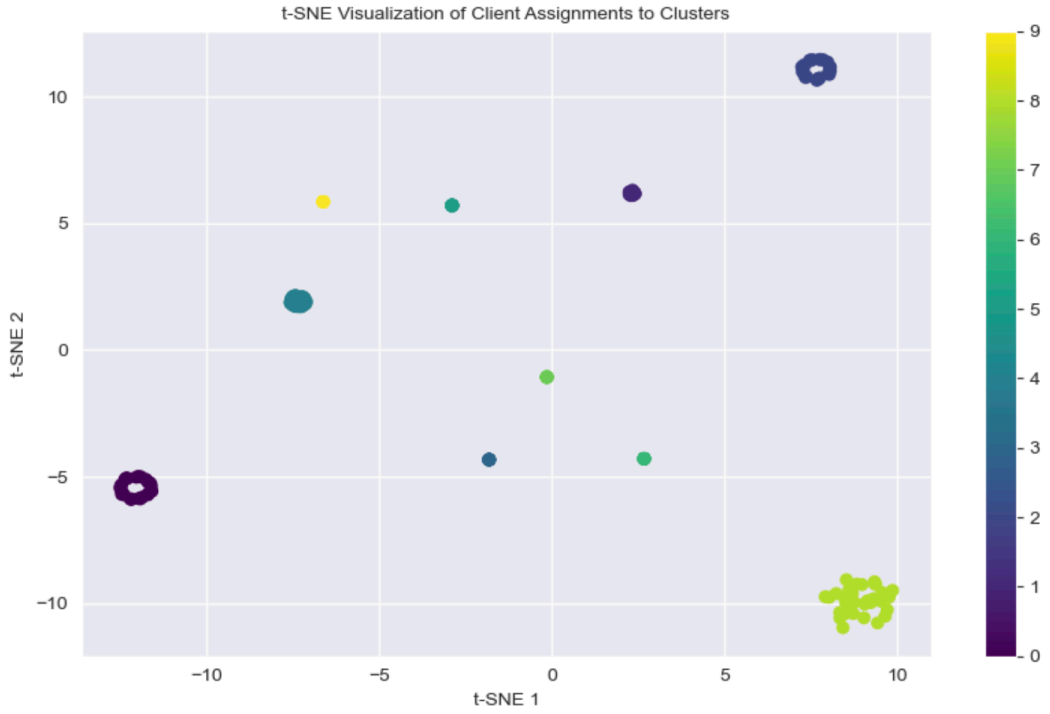

Figure 20: t-SNE visualization of Client Assignment to Cluster using CICIoT2023

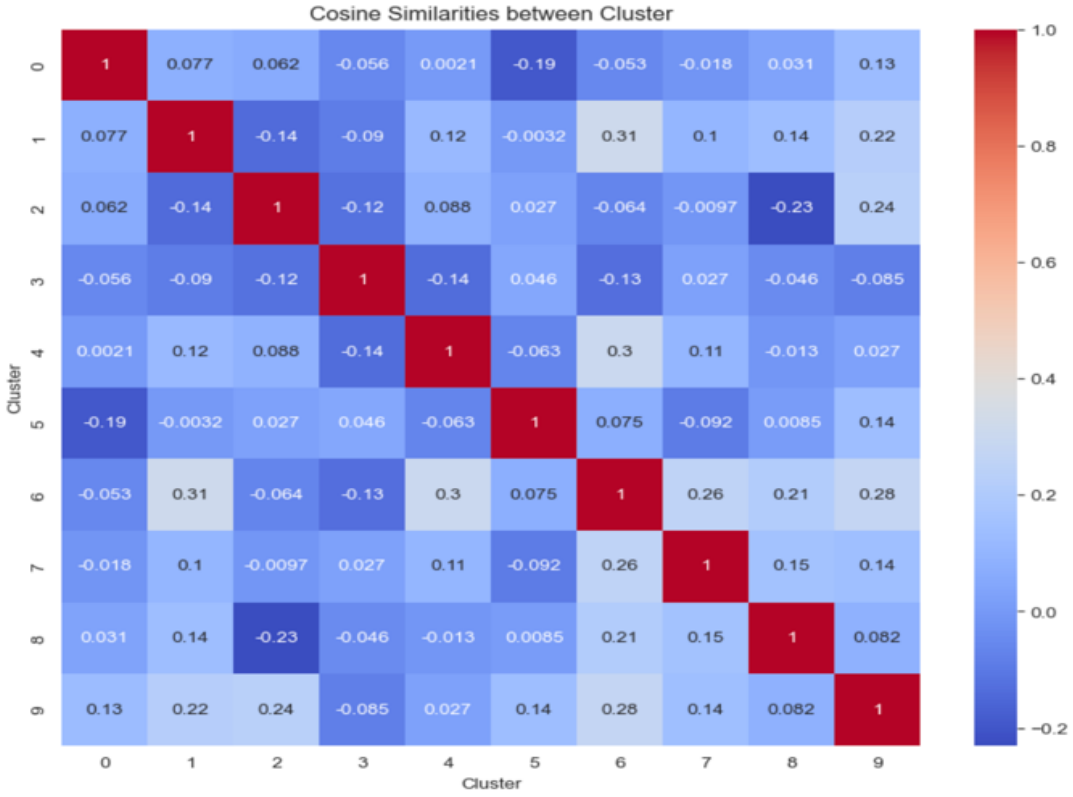

Figure 21: Cosine similarity of 10 clusters on IoTID20 dataset

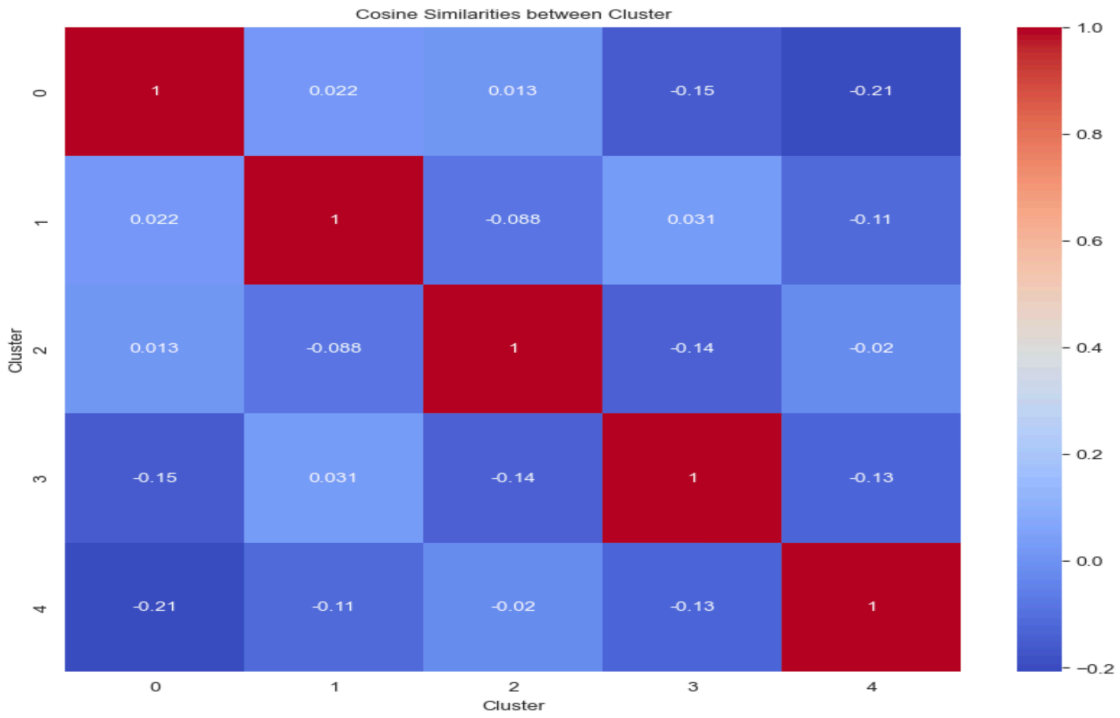

Figure 22: cosine similarity of cluster 5 on CICIoT2023 dataset

7.5.4. Convergence

The convergence analysis shows that FedDWC achieves significantly faster convergence than the other methods. As shown in figure 15 and 16, for the experiments using a non-IID dataset divided into 10 clusters of high variances $\alpha = 0.1$ for inter-cluster and low variances ($\alpha = 10$ for intra-cluster non-IID CICIoT2023 dataset), FedDWC starts to converge at the first communication rounds. FedAvg requires approximately five communication rounds to converge. FedProx and IFCA required approximately four rounds of communication. The significant convergence performance of the FedDWC is supported by the theoretical convergence analysis detailed in the Annex of this paper. This analysis underlines the importance of selecting an appropriate learning rate, as stated in Theorems 1. Carefully selecting the learning rate based on equation in Theorems 1 can significantly enhance the convergence and convergence rate of FedDWC.

The other factors contributing to the rapid convergence of the FedDWC are the dynamic weighting and effective clustering methods used in the proposed framework. These methods ensure that each communication round provides significant progress towards global model optimization. Hence, the total number of rounds required for convergence was reduced. The

efficiency of FedDWC in communication rounds not only accelerates the training process, but also minimizes the communication overhead. This created a highly efficient federated learning approach. This efficiency is valuable in scenarios in which communication resources are limited.

7.6. Summery

The FedDWC framework combines the benefits of clustered federated learning and dynamic weighting approaches to non-IID data issues, and enhances the performance of the DDoS attack detection system. Clustering IoT devices with similar data distributions allows model personalization within each cluster. This helps improve the performance of the global model. The dynamic weighting approach ensures that the IoT contributions are considered to mitigate the impact of noisy or less significant IoT devices. The experimental results show that FedDWC outperforms other state-of-the-art federated learning methods such as FedAvg, FedProx, and IFCA, in handling non-IID data. FedDWC achieved an accuracy of 99.21% using the IoTID20 non-IID dataset with 10 clusters. FedAvg, FedProx, and IFCA achieved an accuracy of 97.31%, 97.90%, and 98.20%, respectively. The theoretical convergence analysis and empirical validations using two datasets demonstrate that FedDWC framework has better convergence, performance in detecting DDoS attacks, and provides a privacy-preserving solution for federated learning in non-IID environments. The proposed FedDWC framework introduces robustness through dynamic weighting and cluster-aware aggregation. However, adversarial attacks on federated learning system such as model poisoning and backdoor attacks are open research challenges. In future work, we plan to implement adversarial defence mechanisms against FedDWC to improve resilience.

Chapter Eight

8. Proposed Solution Deployment Location

This section examines the potential deployment and expected performance of the proposed framework. IoT endpoints are commonly used for IoT data stream collection, IoT edge servers are used for preliminary for DDoS attack detection analytics, and IoT cloud servers are used for comprehensive DDoS attack detection. The proposed AUWPAE and MSAAD framework could be deployed in the IoT cloud, edge servers for IoT DDoS attack detection, and the control server located at the edge layer. The first option is deploying it in IoT edge devices, which provide quick data processing to reduce the size for long-distance data transfer but typically have limited computational capacity. Edge computing allows the detection of local DDoS attacks on edge servers or control servers at the edge layer. Control servers can perform preliminary and fundamental IoT DDoS attack detection jobs locally, including data pre-processing and feature selection. Deploying high-performance computing equipment at the IoT edge layer will significantly minimize latency. Hence, the proposed deployment location considers the DDoS attack detection solution at the IoT edge using high-performance computing equipment.

The other option could be deploying it in IoT cloud servers for DDoS attack detection. IoT cloud servers often include several cloud machines with high computational power and resources, allowing them to use cloud computing to carry out complex DDoS attack detection activities. However, deploying the proposed framework on the cloud server poses a high risk to the privacy of the data. Additionally, there is a delay in the classification of IoT data due to requests and responses to the central cloud server. Figures 23 and 24 show the proposed framework's deployment location at edge servers and in the IoT cloud, respectively

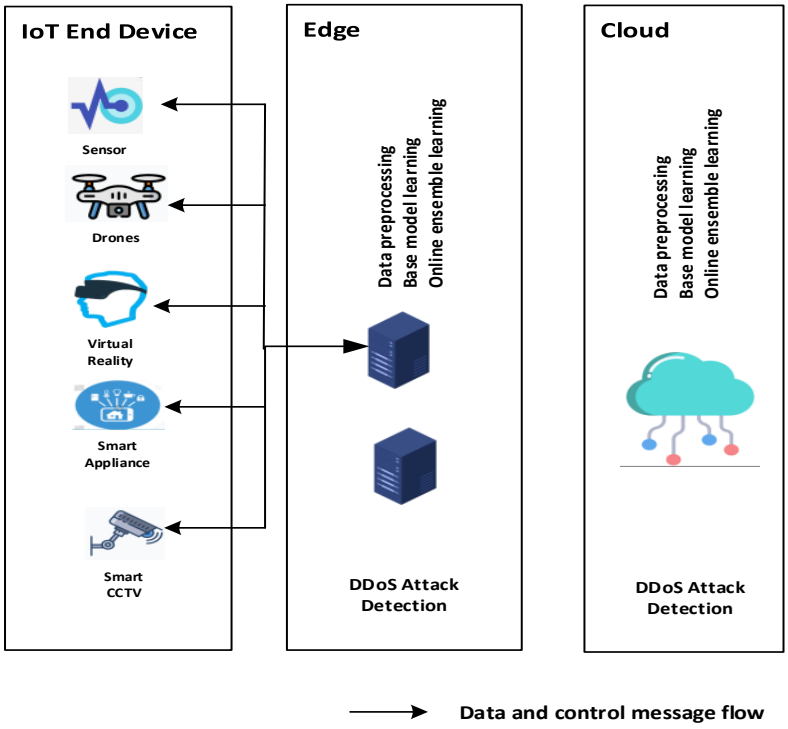

Figure 23: Proposed framework deployment location at IoT Edge Server.

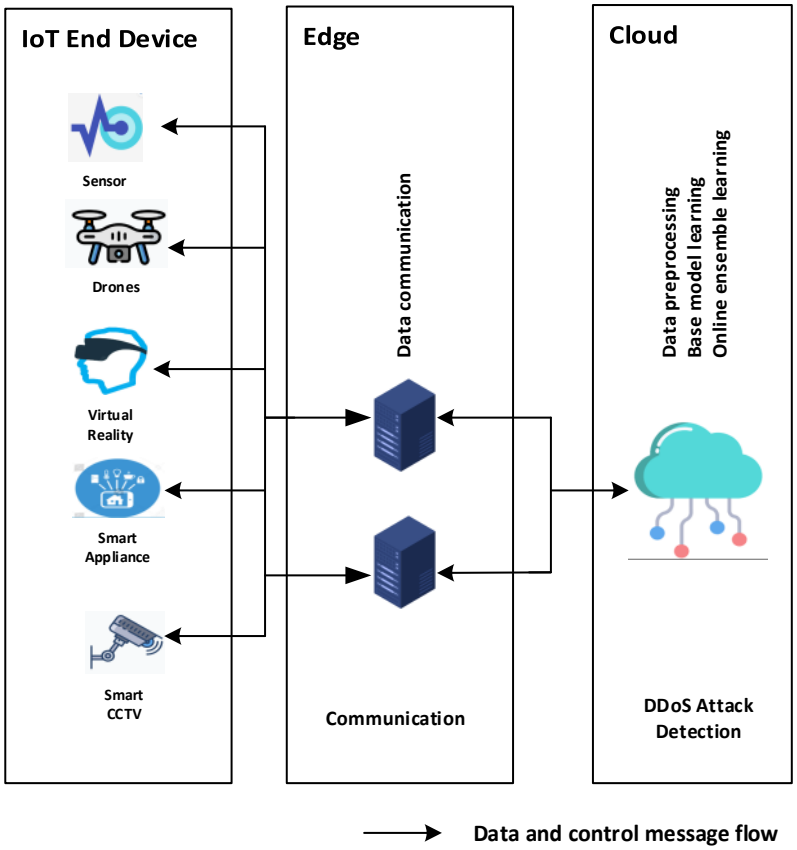

Figure 24: Proposed framework's deployment location in IoT cloud.

The proposed FedDWC framework is designed to be deployed in real-world IoT environments by leveraging the layered structure of IoT infrastructures. The edge computing architecture FedDWC ensures localized processing, communication efficiency, and model personalization. The edge devices such as sensors, routers, and embedded nodes perform initial training using their local traffic data. This edge devices then transmit model update to the gateway node or local aggregator which coordinates cluster-level training and aggregation. The gateway or local aggregator of FedDWC performs clustering of devices based on data distribution similarity. This clustering can be executed periodically or adaptively using clustering metrics to ensuring low computational burden on the gateway. Once the cluster is created, the local model is aggregated using dynamic weighting mechanism. This helps to limit the impact of low-performing or noisy devices and improves resilience in non-IID environments. To address communication overhead, FedDWC adopts a hierarchical federated learning structure. In this architecture:

- Intra-cluster communication occurs between devices and their assigned gateway. The aggregation is performed here more frequently but within smaller groups.
- Inter-cluster or global aggregation is performed less frequently between gateways and the central server. This significantly reduces communication rounds and bandwidth between gateway and central server compared to traditional FL architectures.

This two-tier structure reduces the volume of data exchanged across the network and also supports scalability by enabling parallel processing across clusters and minimizing bottlenecks. The framework preserve privacy because data never leaves the local device to central server. This is an essential requirement in real-world deployments where privacy is a key concern. In conclusion, the FedDWC framework theoretically sound and also practically feasible. Its deployment on edge and gateway nodes ensures low-latency, privacy-aware, and scalable DDoS detection.

Chapter Nine

9. Conclusion and Future work

IoT DDoS attack detection solutions are usually developed to protect IoT systems from DDoS attacks using IoT data stream analytics. However, IoT data are usually dynamic and could have concept drifts. However, these DDoS attack-detection systems use batch learning and centralized approach that makes unable to detect zero-day DDoS attacks in real-time. The dynamicity of non-IID IoT data causes concept drift issues that result in performance degradation in detecting DDoS. Centralized approach of detecting DDoS attacks transfers network data from all IoT edge nodes to a central cloud server for model training. This approach exposes privacy of network devices owner's data as it collects from edge nodes to central server. Moreover, adversarial attacks pose a significant challenge to the detection of Distributed Denial of Service (DDoS) attack in IoT systems. These attacks can exploit the vulnerability of online DDoS attack detection solutions, thereby degrade their accuracy.

To address the above challenge, this dissertation provides: first, novel Accuracy Update Weighted Probability Averaging Ensemble (AUWPAE) approach to detect concept drift and perform zero-day DDoS detection. We evaluated the proposed model using the IoTID20 and CICIoT2023 datasets with benign and DDoS traffic data that had concept drifts. The results show that AUWPAE achieved better accuracies of 99.54% and 99.33% for the respective datasets when compared with those of the other eight models. This result indicates that the proposed adaptive online DDoS attack detection framework, which uses AUWPAE is able to detect DDoS attacks in the presence of concept drifts.

Second, novel MSAAD framework that can protect online DDoS attack detection solutions against adversarial attacks. The proposed framework comprises of three defense layers: the Resilient Adversarial Detector and Purification (RADP), which detect and purify adversarial attacks against online DDoS attack detection systems for multiple and unknown adversarial attacks; A multiple-classifier approach, which makes it more challenging for an attacker to replicate the DDoS attack detection model; and MABTSE , which select dynamically classifiers or ensembles using the the Multi-Armed Bandits (MAB) algorithm with Thompson Sampling for each incoming traffic request. The frameworks maintain the performance of conventional DDoS attack detection systems in non-adversarial attack scenarios by selecting the most appropriate classifiers for each request type. The experiment results demonstrate the effectiveness of the

proposed MABTSE model. The MABTSE achieved an average accuracy of 99.50% and 99.28% on IOTID20 and CICIoT2023 datasets respectively. The accuracy of the DDoS detection model declined significantly under adversarial attacks. however, it is successfully restored to range of 99.39% to 99.48% for IOTID20 dataset and range of 99.01% to 99.14% for CICIoT2023 datasets respectively in adversarial attack scenario. This demonstrates the robustness of MSAAD framework in dynamic IoT environment

Third, The FedDWC framework combines the benefits of clustered federated learning and dynamic weighting approaches to non-IID data issues, and enhances the performance of the DDoS attack detection system. Clustering IoT devices with similar data distributions allows model personalization within each cluster. This helps improve the performance of the global model. The dynamic weighting approach ensures that the IoT contributions are considered to mitigate the impact of noisy or less significant IoT devices. The experimental results show that FedDWC outperforms other state-of-the-art federated learning methods such as FedAvg, FedProx, and IFCA, in handling non-IID data. FedDWC achieved an accuracy of 99.21% using the IoTID20 non-IID dataset with 10 clusters. FedAvg, FedProx, and IFCA achieved an accuracy of 97.31%, 97.90%, and 98.20%, respectively. The theoretical convergence analysis and empirical validations using two datasets demonstrate that FedDWC framework has better convergence, performance in detecting DDoS attacks, and provides a privacy-preserving solution for federated learning in non-IID environments. The proposed FedDWC framework introduces robustness through dynamic weighting and cluster-aware aggregation. However, adversarial attacks on federated learning system such as model poisoning and backdoor attacks are open research challenges. In future work, we plan to implement adversarial defence mechanisms against FedDWC to improve resilience.

Finally, this dissertation provides the IoT DDoS attack detection solution deployment framework for IoT systems. Despite the results presented above, this research acknowledges potential limitations of our current study. The scalability of the framework in large-scale IoT networks, characterized by diverse traffic patterns and device heterogeneity, requires further exploration. In future work, we plan to explore reinforcement learning-based adaptive attackers to stress-test under real-time, dynamic threat scenarios. We will also address adversarial attacks in federated learning systems, such as model poisoning and backdoor attacks. In addition, we will consider Byzantine-resilient aggregation to handle malicious participants. We will validate our framework on live IoT testbeds to assess performance in realistic environments. Finally, we aim

to empirically verify and generalize the theoretical assumptions, including Lipschitz smoothness, while extending complexity analysis for large-scale federated systems.

10. Reference

[1] Y. Otoum, D. Liu, and A. Nayak, "DL-IDS: a deep learning–based intrusion detection framework for securing IoT," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, 2022, doi: 10.1002/ett.3803.

[2] J. Singh, T. Pasquier, J. Bacon, H. Ko, and D. Eyers, "Twenty Security Considerations for Cloud-Supported Internet of Things," *IEEE Internet Things J*, vol. 3, no. 3, pp. 269–284, 2016, doi: 10.1109/JIOT.2015.2460333.

[3] E. Schiller, A. Aidoo, J. Fuhrer, J. Stahl, M. Ziörjen, and B. Stiller, "Landscape of IoT security," May 01, 2022, *Elsevier Ireland Ltd*. doi: 10.1016/j.cosrev.2022.100467.

[4] F. Hussain, S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, and G. A. Shah, "IoT DoS and DDoS Attack Detection using ResNet," in *Proceedings - 2020 23rd IEEE International Multi-Topic Conference, INMIC 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020. doi: 10.1109/INMIC50486.2020.9318216.

[5] A. Affinito, S. Zinno, G. Stanco, A. Botta, and G. Ventre, "The evolution of Mirai botnet scans over a six-year period," *Journal of Information Security and Applications*, vol. 79, p. 103629, Dec. 2023, doi: 10.1016/j.jisa.2023.103629.

[6] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, 2013, doi: 10.1016/j.adhoc.2013.04.014.

[7] P. J. Beslin Pajila and E. Golden Julie, "Detection of DDoS Attack Using SDN in IoT: A Survey," *Lecture Notes on Data Engineering and Communications Technologies*, vol. 33, no. April, pp. 438–452, 2020, doi: 10.1007/978-3-030-28364-3_44.

[8] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke, and L. Shu, "Federated Deep Learning for Cyber Security in the Internet of Things: Concepts, Applications, and Experimental Analysis," *IEEE Access*, vol. 9, no. Ml, pp. 138509–138542, 2021, doi: 10.1109/ACCESS.2021.3118642.

[9] Y. K. Beshah, S. L. Abebe, and H. M. Melaku, "Drift Adaptive Online DDoS Attack Detection Framework for IoT System," *Electronics (Basel)*, vol. 13, no. 6, p. 1004, Mar. 2024, doi: 10.3390/electronics13061004.

[10] A. A. Alahmadi *et al.*, "DDoS Attack Detection in IoT-Based Networks Using Machine Learning Models: A Survey and Research Directions," Jul. 01, 2023, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/electronics12143103.

[11] Z. 'Ruijie *et al.*, "Novel Intrusion Detection Method Based on Lightweight Neural Network for Internet of Things," *IEEE Internet of Things Journal* , vol. 9, no. 12, pp. 9960–9972, Aug. 2021.

[12] S. Taheri, A. Khormali, M. Salem, and J. S. Yuan, "Developing a robust defensive system against adversarial examples using generative adversarial networks," *Big Data and Cognitive Computing*, vol. 4, no. 2, pp. 1–15, 2020, doi: 10.3390/bdcc4020011.

[13] K. Bonawitz *et al.*, "Towards Federated Learning at Scale: System Design," *MLSYS*, Feb. 2019, [Online]. Available: https://mlsys.org/Conferences/2019/doc/2019/

[14] O. Aouedi, K. Piamrat, G. Muller, and K. Singh, "Federated Semisupervised Learning for Attack Detection in Industrial Internet of Things," *IEEE Trans Industr Inform*, vol. 19, no. 1, pp. 286–295, Jan. 2023, doi: 10.1109/TII.2022.3156642.

[15] H. Wang, L. Muñoz-González, D. Eklund, and S. Raza, "Non-IID data re-balancing at IoT edge with peer-to-peer federated learning for anomaly detection," in *WiSec 2021 - Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile*

Networks, Association for Computing Machinery, Inc, Jun. 2021, pp. 153–163. doi: 10.1145/3448300.3467827.
[16] M. F. Criado, F. E. Casado, R. Iglesias, C. V. Regueiro, and S. Barro, "Non-IID data and Continual Learning processes in Federated Learning: A long road ahead," *Information Fusion*, vol. 88, pp. 263–280, Dec. 2022, doi: 10.1016/j.inffus.2022.07.024.
[17] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the Convergence of FedAvg on Non-IID Data," Jul. 2019, [Online]. **preprint** arXiv, Available: http://arxiv.org/abs/1907.02189
[18] J. Liu, J. Wu, J. Chen, M. Hu, Y. Zhou, and D. Wu, "FedDWA: Personalized Federated Learning with Dynamic Weight Adjustment," in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, California: International Joint Conferences on Artificial Intelligence Organization, Aug. 2023, pp. 3993–4001. doi: 10.24963/ijcai.2023/444.
[19] D. Zhang, L. T. Yang, M. Chen, S. Zhao, and S. Member, "Real-Time Locating Systems Using Active RFID for Internet of Things," pp. 1–10, 2014.
[20] K. Curran, A. Millar, and C. Mc Garvey, "Near Field Communication," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 2, no. 3, 2012, doi: 10.11591/ijece.v2i3.234.
[21] A. Whitmore, A. Agarwal, and L. Da Xu, "The Internet of Things—A survey of topics and trends," *Information Systems Frontiers*, vol. 17, no. 2, pp. 261–274, 2015, doi: 10.1007/s10796-014-9489-2.
[22] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K. S. Kwak, "The internet of things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015, doi: 10.1109/ACCESS.2015.2437951.
[23] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: perspectives and challenges," *Wireless Networks*, vol. 20, no. 8, pp. 2481–2501, 2014, doi: 10.1007/s11276-014-0761-7.
[24] I. Mashal, O. Alsaryrah, T. Y. Chung, C. Z. Yang, W. H. Kuo, and D. P. Agrawal, "Choices for interaction with things on Internet and underlying issues," *Ad Hoc Networks*, vol. 28, pp. 68–90, 2015, doi: 10.1016/j.adhoc.2014.12.006.
[25] O. Said and M. Masud, "Towards internet of things: Survey and future vision," *International Journal of Computer Networks*, vol. 5, no. 1, pp. 1–17, 2013, [Online]. Available: http://www.cscjournals.org/csc/manuscript/Journals/IJCN/volume5/Issue1/IJCN-265.pdf
[26] J. Veijalainen, D. Kozlov, and Y. Ali, "Security and Privacy Threats in IoT Architectures," no. May 2014, 2013, doi: 10.4108/icst.bodynets.2012.250550.
[27] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the internet of things: A review," *Proceedings - 2012 International Conference on Computer Science and Electronics Engineering, ICCSEE 2012*, vol. 3, no. March 2015, pp. 648–651, 2012, doi: 10.1109/ICCSEE.2012.373.
[28] R. Bhatt, P. Maheshwary, P. Shukla, P. Shukla, M. Shrivastava, and S. Changlani, "Implementation of Fruit Fly Optimization Algorithm (FFOA) to escalate the attacking efficiency of node capture attack in Wireless Sensor Networks (WSN)," *Comput Commun*, vol. 149, no. September, pp. 134–145, 2020, doi: 10.1016/j.comcom.2019.09.007.
[29] A. Thakare and Y. G. Kim, "Secure and efficient authentication scheme in iot environments," *Applied Sciences (Switzerland)*, vol. 11, no. 3, pp. 1–27, 2021, doi: 10.3390/app11031260.

[30] S. Takarabt *et al.*, “Cache-timing attacks still threaten IoT devices,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11445 LNCS, no. January 2020, pp. 13–30, 2019, doi: 10.1007/978-3-030-16458-4_2.
[31] K. Saurabh, T. Kumar, U. Singh, O. P. Vyas, and R. Khondoker, “NFDLM: A Lightweight Network Flow based Deep Learning Model for DDoS Attack Detection in IoT Domains,” *2022 IEEE World AI IoT Congress, AIIoT 2022*, no. Dl, pp. 736–742, 2022, doi: 10.1109/AIIoT54504.2022.9817297.
[32] Y. Liu, J. Wang, J. Li, S. Niu, and H. Song, “Machine Learning for the Detection and Identification of Internet of Things Devices: A Survey,” *IEEE Internet Things J*, vol. 9, no. 1, pp. 298–320, 2022, doi: 10.1109/JIOT.2021.3099028.
[33] B. Ali and A. I. Awad, “Cyber and physical security vulnerability assessment for IoT-based smart homes,” *Sensors (Switzerland)*, vol. 18, no. 3, pp. 1–17, 2018, doi: 10.3390/s18030817.
[34] G. E. Rodríguez, J. G. Torres, P. Flores, and D. E. Benavides, “Cross-site scripting (XSS) attacks and mitigation: A survey,” *Computer Networks*, vol. 166, no. November, 2020, doi: 10.1016/j.comnet.2019.106960.
[35] H. Alasmary *et al.*, “Analyzing and Detecting Emerging Internet of Things Malware: A Graph-Based Approach,” *IEEE Internet Things J*, vol. 6, no. 5, pp. 8977–8988, 2019, doi: 10.1109/JIOT.2019.2925929.
[36] D. G. Darwish and E. Square, “Improved Layered Architecture for Internet of Things,” *International Journal of Computing Academic Research*, vol. 4, no. 4, pp. 214–223, 2015, [Online]. Available: http://www.meacse.org/ijcar
[37] P. Sethi and S. R. Sarangi, “Internet of Things: Architectures, Protocols, and Applications,” *Journal of Electrical and Computer Engineering*, vol. 2017, 2017, doi: 10.1155/2017/9324035.
[38] S. Madakam, R. Ramaswamy, and S. Tripathi, “Internet of Things (IoT): A Literature Review,” *Journal of Computer and Communications*, vol. 03, no. 05, pp. 164–173, 2015, doi: 10.4236/jcc.2015.35021.
[39] Manos Antonakakis *et al.*, “Understanding the Mirai Botnet,” *USENIX Association*, pp. 1–19, 2017.
[40] I. Yaqoob *et al.*, “Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges,” *IEEE Wirel Commun*, vol. 24, no. 3, pp. 10–16, 2017, doi: 10.1109/MWC.2017.1600421.
[41] T. T. Huong *et al.*, “LocKedge: Low-Complexity Cyberattack Detection in IoT Edge Computing,” *IEEE Access*, vol. 9, pp. 29696–29710, 2021, doi: 10.1109/ACCESS.2021.3058528.
[42] O. Osanaiye, H. Cai, K. K. R. Choo, A. Dehghantanha, Z. Xu, and M. Dlodlo, “Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing,” *EURASIP J Wirel Commun Netw*, vol. 2016, no. 1, 2016, doi: 10.1186/s13638-016-0623-3.
[43] I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” *Commun ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020, doi: 10.1145/3422622.
[44] A. Antoniou, A. Storkey, and H. Edwards, “Data Augmentation Generative Adversarial Networks,” Nov. 2017, **preprint arXvi**, [Online]. Available: http://arxiv.org/abs/1711.04340

[45] G. Liu, I. Khalil, and A. Khreishah, “GanDef: A GAN based Adversarial Training Defense for Neural Network Classifier,” 2019, pp. 19–32. doi: 10.1007/978-3-030-22312-0_2.
[46] N. Moustafa and J. Slay, “UNSW-NB15 : A Comprehensive Data set for Network Intrusion Detection systems,” no. December, 2015, doi: 10.1109/MilCIS.2015.7348942.
[47] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” no. Cic, pp. 108–116, 2018, doi: 10.5220/0006639801080116.
[48] “GoldenEye HTTP DoS Test Tool.” [Online]. Available: https://github.com/jseidl/GoldenEye
[49] “A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018),” 2022, [Online]. Available: https://registry.opendata.aws/cse-cic-ids2018/
[50] Canadian Institute for Cybersecurity, “CSE-CIC-IDS2018,” 2018. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2018.html
[51] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, “Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset,” *Future Generation Computer Systems*, vol. 100, no. November, pp. 779–796, 2019, doi: 10.1016/j.future.2019.05.041.
[52] “Ostinato.” [Online]. Available: https://ostinato.org/
[53] Canadian Institute for Cybersecurity, “CIC IoT Dataset 2023,” 2023.
[54] P. Maniriho, E. Niyigaba, Z. Bizimana, V. Twiringiyimana, L. J. Mahoro, and T. Ahmad, “Anomaly-based Intrusion Detection Approach for IoT Networks Using Machine Learning,” in *CENIM 2020 - Proceeding: International Conference on Computer Engineering, Network, and Intelligent Multimedia 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 303–308. doi: 10.1109/CENIM51130.2020.9297958.
[55] I. Ullah and Q. H. Mahmoud, “A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer, 2020, pp. 508–520. doi: 10.1007/978-3-030-47358-7_52.
[56] A. Alsaedi, N. Moustafa, Z. Tari, and A. Mahmood, “TON _ IoT Telemetry Dataset : A New Generation Dataset of IoT and IIoT for Data-Driven Intrusion Detection Systems,” no. September, 2020, doi: 10.1109/ACCESS.2020.3022862.
[57] “Node-RED.” [Online]. Available: https://nodered.org/
[58] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, “Internet of Things intrusion Detection: Centralized, On-Device, or Federated Learning?,” *IEEE Netw*, vol. 34, no. 6, pp. 310–317, 2020, doi: 10.1109/MNET.011.2000286.
[59] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A. R. Sadeghi, “DÏoT: A federated self-learning anomaly detection system for IoT,” *Proc Int Conf Distrib Comput Syst*, vol. 2019-July, no. Icdcs, pp. 756–767, 2019, doi: 10.1109/ICDCS.2019.00080.
[60] X. Wang *et al.*, “Toward Accurate Anomaly Detection in Industrial Internet of Things Using Hierarchical Federated Learning,” *IEEE Internet Things J*, vol. 9, no. 10, pp. 7110–7119, 2022, doi: 10.1109/JIOT.2021.3074382.
[61] S. Kim, H. Cai, C. Hua, P. Gu, W. Xu, and J. Park, “Collaborative Anomaly Detection for Internet of Things based on Federated Learning,” *2020 IEEE/CIC International Conference on Communications in China, ICCC 2020*, no. Iccc, pp. 623–628, 2020, doi: 10.1109/ICCC49849.2020.9238913.

[62] O. A. Wahab, A. Mourad, H. Otrok, and T. Taleb, “Federated Machine Learning: Survey, Multi-Level Classification, Desirable Criteria and Future Directions in Communication and Networking Systems,” *IEEE Communications Surveys and Tutorials*, vol. 23, no. 2, pp. 1342–1397, 2021, doi: 10.1109/COMST.2021.3058573.
[63] W. Y. B. Lim *et al.*, “Federated Learning in Mobile Edge Networks: A Comprehensive Survey,” *IEEE Communications Surveys and Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020, doi: 10.1109/COMST.2020.2986024.
[64] K. Yang, T. Jiang, Y. Shi, and Z. Ding, “Federated learning via over-the-air computation,” *IEEE Trans Wirel Commun*, vol. 19, no. 3, pp. 2022–2035, 2020, doi: 10.1109/TWC.2019.2961673.
[65] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated Optimization in Heterogeneous Networks,” Dec. 2018, **preprint arXiv**, [Online]. Available: http://arxiv.org/abs/1812.06127
[66] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, “An Efficient Framework for Clustered Federated Learning.” [Online]. Available: https://github.com/jichan3751/ifca.
[67] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, “A survey on security and privacy of federated learning,” *Future Generation Computer Systems*, vol. 115, pp. 619–640, Feb. 2021, doi: 10.1016/j.future.2020.10.007.
[68] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated Learning with Non-IID Data,” Jun. 2018, **preprint arXiv**, doi: 10.48550/arXiv.1806.00582.
[69] H. Wang, Z. Kaplan, D. Niu, and B. Li, “Optimizing Federated Learning on Non-IID Data with Reinforcement Learning.” [Online]. Available: https://github.com/iqua/flsim.
[70] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, “Robust and Communication-Efficient Federated Learning from Non-IID Data,” *IEEE Trans Neural Netw Learn Syst*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020, doi: 10.1109/TNNLS.2019.2944481.
[71] M. Duan, D. Liu, X. Chen, R. Liu, Y. Tan, and L. Liang, “Self-Balancing Federated Learning with Global Imbalanced Data in Mobile Systems,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 1, pp. 59–71, Jan. 2021, doi: 10.1109/TPDS.2020.3009406.
[72] L. Wang, S. Xu, X. Wang, and Q. Zhu, “Addressing Class Imbalance in Federated Learning,” 2021. [Online]. Available: www.aaai.org
[73] Y. Jeong and T. Kim, “A Cluster-Driven Adaptive Training Approach for Federated Learning,” *Sensors*, vol. 22, no. 18, Sep. 2022, doi: 10.3390/s22187061.
[74] D. K. Dennis, T. Li, and V. Smith, “Heterogeneity for the Win: One-Shot Federated Clustering,” *MLR*, Feb. 2021, [Online]. Available: http://proceedings.mlr.press/
[75] C. Briggs, Z. Fan, and P. Andras, “Federated learning with hierarchical clustering of local updates to improve training on non-IID data,” Apr. 2020, **preprint arXiv**, [Online]. Available: http://arxiv.org/abs/2004.11791
[76] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, “An Efficient Framework for Clustered Federated Learning,” *NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems,* Jun. 2020, [Online]. Available: https://dl.acm.org/doi/proceedings/10.5555/3495724
[77] G. Long *et al.*, “Multi-Center Federated Learning: Clients Clustering for Better Personalization,” May 2020, doi: 10.1007/s11280-022-01046-x.
[78] S. U. Stich, “Local SGD Converges Fast and Communicates Little,” *MLR*, May 2018, [Online]. Available: http://proceedings.mlr.press/

[79] A. Khaled, K. Mishchenko, and P. Richtárik, “Tighter Theory for Local SGD on Identical and Heterogeneous Data,” *MLR*, Sep. 2019, [Online]. Available: http://proceedings.mlr.press/
[80] J. Wang *et al.*, “A Field Guide to Federated Optimization,” Jul. 2021, **preprint arXiv**, [Online]. Available: http://arxiv.org/abs/2107.06917
[81] P. Xing, S. Lu, L. Wu, and H. Yu, “BiG-Fed: Bilevel Optimization Enhanced Graph-Aided Federated Learning.”
[82] R. Yumlembam, B. Issac, S. M. Jacob, and L. Yang, “IoT-Based Android Malware Detection Using Graph Neural Network With Adversarial Defense,” *IEEE Internet Things J*, vol. 10, no. 10, pp. 8432–8444, May 2023, doi: 10.1109/JIOT.2022.3188583.
[83] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” Dec. 2014, **preprint arXiv**, [Online]. Available: http://arxiv.org/abs/1412.6572
[84] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” *5th International Conference on Learning Representations, ICLR 2017 - Workshop Track Proceedings*, no. c, pp. 1–14, 2019.
[85] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards Deep Learning Models Resistant to Adversarial Attacks,” Jun. 2017, **preprint arXiv**, [Online]. Available: http://arxiv.org/abs/1706.06083
[86] N. Carlini and D. Wagner, “Towards Evaluating the Robustness of Neural Networks,” *Proc IEEE Symp Secur Priv*, pp. 39–57, 2017, doi: 10.1109/SP.2017.49.
[87] Y. Wang *et al.*, “Distributed Learning for Automatic Modulation Classification in Edge Devices,” *IEEE Wireless Communications Letters*, vol. 9, no. 12, pp. 2177–2181, 2020, doi: 10.1109/LWC.2020.3016822.
[88] F. Luo, B. Du, L. Zhang, L. Zhang, and D. Tao, “Feature learning using spatial-spectral hypergraph discriminant analysis for hyperspectral image,” *IEEE Trans Cybern*, vol. 49, no. 7, pp. 2406–2419, 2019, doi: 10.1109/TCYB.2018.2810806.
[89] J. Peng, W. Sun, and Q. Du, “Self-paced joint sparse representation for the classification of hyperspectral images,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 2, pp. 1183–1194, 2019, doi: 10.1109/TGRS.2018.2865102.
[90] C. Gui, “Analysis of imbalanced data set problem: The case of churn prediction for telecommunication,” *Artif Intell Res*, vol. 6, no. 2, p. 93, 2017, doi: 10.5430/air.v6n2p93.
[91] A. Fernández, S. García, M. J. del Jesus, and F. Herrera, “A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets,” *Fuzzy Sets Syst*, vol. 159, no. 18, pp. 2378–2398, 2008, doi: 10.1016/j.fss.2007.12.023.
[92] E. Lin, Q. Chen, and X. Qi, “Deep reinforcement learning for imbalanced classification,” *Applied Intelligence*, vol. 50, no. 8, pp. 2488–2502, 2020, doi: 10.1007/s10489-020-01637-z.
[93] X. Zhou, W. Liang, W. Li, K. Yan, S. Shimizu, and K. I.-K. Wang, “Hierarchical Adversarial Attacks Against Graph-Neural-Network-Based IoT Network Intrusion Detection System,” *IEEE Internet Things J*, vol. 9, no. 12, pp. 9310–9319, Jun. 2022, doi: 10.1109/JIOT.2021.3130434.
[94] J. Kotak and Y. Elovici, “Adversarial Attacks Against IoT Identification Systems,” *IEEE Internet Things J*, vol. 10, no. 9, pp. 7868–7883, May 2023, doi: 10.1109/JIOT.2022.3229906.
[95] O. Ibitoye, O. Shafiq, and A. Matrawy, “Analyzing adversarial attacks against deep learning for intrusion detection in IoT networks,” *2019 IEEE Global Communications*

Conference, GLOBECOM 2019 - Proceedings, 2019, doi: 10.1109/GLOBECOM38437.2019.9014337.
[96] P. Papadopoulos, O. Thornewill von Essen, N. Pitropakis, C. Chrysoulas, A. Mylonas, and W. J. Buchanan, "Launching Adversarial Attacks against Network Intrusion Detection Systems for IoT," *Journal of Cybersecurity and Privacy*, vol. 1, no. 2, pp. 252–273, 2021, doi: 10.3390/jcp1020014.
[97] H. Benaddi, M. Jouhari, K. Ibrahimi, A. Benslimane, and E. M. Amhoud, "Adversarial Attacks Against IoT Networks using Conditional GAN based Learning," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, IEEE, Dec. 2022, pp. 2788–2793. doi: 10.1109/GLOBECOM48099.2022.10000726.
[98] H. Benaddi, M. Jouhari, K. Ibrahimi, J. Ben Othman, and E. M. Amhoud, "Anomaly Detection in Industrial IoT Using Distributional Reinforcement Learning and Generative Adversarial Networks," *Sensors*, vol. 22, no. 21, Nov. 2022, doi: 10.3390/s22218085.
[99] H. Jiang, J. Lin, and H. Kang, "FGMD: A robust detector against adversarial attacks in the IoT network," *Future Generation Computer Systems*, vol. 132, pp. 194–210, Jul. 2022, doi: 10.1016/j.future.2022.02.019.
[100] K. S. Prasad *et al.*, "A two-tier optimization strategy for feature selection in robust adversarial attack mitigation on internet of things network security," *Sci Rep*, vol. 15, no. 1, p. 2235, Jan. 2025, doi: 10.1038/s41598-025-85878-3.
[101] M. Ramaiah, V. Chandrasekaran, P. Adla, A. Vasudevan, M. F. A. Hunitie, and S. I. S. Mohammad, "Optimal feature selection based on OCS for improved malware detection in IoT networks using an ensemble classifier," *International Journal of Data and Network Science*, vol. 8, no. 4, pp. 2129–2140, Sep. 2024, doi: 10.5267/j.ijdns.2024.6.018.
[102] P. M. Sánchez Sánchez, A. Huertas Celdrán, G. Bovet, and G. Martínez Pérez, "Adversarial attacks and defenses on ML- and hardware-based IoT device fingerprinting and identification," *Future Generation Computer Systems*, vol. 152, pp. 30–42, Mar. 2024, doi: 10.1016/j.future.2023.10.011.
[103] J. Ma, W. Su, Y. Li, Y. Yuan, and Z. Zhang, "Synchronizing real-time and high-precision LDoS defense of learning model-based in AIoT with programmable data plane, SDN," *Journal of Network and Computer Applications*, vol. 229, p. 103916, Sep. 2024, doi: 10.1016/j.jnca.2024.103916.
[104] R. Doriguzzi-Corin and D. Siracusa, "FLAD: Adaptive Federated Learning for DDoS attack detection," *Comput Secur*, vol. 137, Feb. 2024, doi: 10.1016/j.cose.2023.103597.
[105] V. Pourahmadi, H. A. Alameddine, M. A. Salahuddin, and R. Boutaba, "Spotting Anomalies at the Edge: Outlier Exposure-Based Cross-Silo Federated Learning for DDoS Detection," *IEEE Trans Dependable Secure Comput*, vol. 20, no. 5, pp. 4002–4015, Sep. 2023, doi: 10.1109/TDSC.2022.3224896.
[106] Z. Yin, K. Li, and H. Bi, "Trusted Multi-Domain DDoS Detection Based on Federated Learning," *Sensors*, vol. 22, no. 20, Oct. 2022, doi: 10.3390/s22207753.
[107] Q. Tian, C. Guang, C. Wenchao, and W. Si, "A lightweight residual networks framework for DDoS attack classification based on federated learning," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2021*, Institute of Electrical and Electronics Engineers Inc., May 2021. doi: 10.1109/INFOCOMWKSHPS51825.2021.9484622.
[108] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT-Edge Devices," *IEEE Internet Things J*, vol. 9, no. 5, pp. 3930–3944, Mar. 2022, doi: 10.1109/JIOT.2021.3100755.

[109] A. Chaudhuri, A. Nandi, and B. Pradhan, “A Dynamic Weighted Federated Learning for Android Malware Classification,” Nov. 2022, doi: 10.1007/978-981-19-9858-4_13.
[110] M. Ragab *et al.*, “Advanced artificial intelligence with federated learning framework for privacy-preserving cyberthreat detection in IoT-assisted sustainable smart cities,” *Sci Rep*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-88843-2.
[111] D. Torre, A. Chennamaneni, J. Jo, G. Vyas, and B. Sabrsula, “Toward Enhancing Privacy Preservation of a Federated Learning CNN Intrusion Detection System in IoT: Method and Empirical Study,” *ACM Transactions on Software Engineering and Methodology*, vol. 34, no. 2, pp. 1–48, Feb. 2025, doi: 10.1145/3695998.
[112] S. Alsaleh, M. E. B. Menai, and S. Al-Ahmadi, “A Heterogeneity-Aware Semi-Decentralized Model for a Lightweight Intrusion Detection System for IoT Networks Based on Federated Learning and BiLSTM,” *Sensors*, vol. 25, no. 4, p. 1039, Feb. 2025, doi: 10.3390/s25041039.
[113] B. Olanrewaju-George and B. Pranggono, “Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models,” *Cyber Security and Applications*, vol. 3, p. 100068, Dec. 2025, doi: 10.1016/j.csa.2024.100068.
[114] M. Data and M. Aritsugi, “T-DFNN: An Incremental Learning Algorithm for Intrusion Detection Systems,” *IEEE Access*, vol. 9, pp. 154156–154171, 2021, doi: 10.1109/ACCESS.2021.3127985.
[115] Y. Kayode Saheed, A. Idris Abiodun, S. Misra, M. Kristiansen Holone, and R. Colomo-Palacios, “A machine learning-based intrusion detection for detecting internet of things network attacks,” *Alexandria Engineering Journal*, vol. 61, no. 12, pp. 9395–9409, Dec. 2022, doi: 10.1016/j.aej.2022.02.063.
[116] H. Wang, Q. Wei, and Y. Xie, “A Novel Method for Network Intrusion Detection,” *Sci Program*, vol. 2022, 2022, doi: 10.1155/2022/1357182.
[117] “River ,” Aug. 2023.
[118] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, “Adversarial Attacks and Defences: A Survey,” *CAAI Trans Intell Technol*, vol. 6, no. 1, pp. 25–45, Mar. 2021, doi: 10.1049/cit2.12028.